北京邮电大学

硕士学位论文

IPTV机顶盒软件系统的设计与实施

姓名：沈昕

申请学位级别：硕士

专业：通信与信息系统

指导教师：王雷

20070319

IPTV机顶盒软件系统的设计与实施

摘 要

IPTV这种在互联网时代更为灵活的传媒方式对于未来电信运营商、内容运营商、IT服务提供商的增值都有着巨大的影响。IPTV具有非常鲜明的特点，可以实现媒体提供者和媒体消费者的实质性互动，它能根据用户的选择配置多种多媒体服务功能，包括数字电视节目、可视IP电话、DVD/VCD播放、互联网浏览、收发电子邮件以及多种在线信息咨询、娱乐、教育及商务功能。IPTV服务或技术是一种全新的集中式电信和广播服务或技术。它能通过具有QOS控制的有线和无线网等宽带集中IP网络对包括视频、音频、数据和其他各种应用在内的各种多媒体内容进行管理、控制和安全传输，它能够将这些应用从业务平台发送到具有STB模块或相似设备的电视，PDA，移动电话或移动电视终端为消费者提供服务。

本文在分析了IPTV机顶盒的基本定义及其当前国内国外市场现状和发展状况后，总结了IPTV机顶盒的业务要求。通过研究IPTV机顶盒的业务需求以及相关文档，对IPTV机顶盒的软硬件环境进行了选取。在硬件方面选取了在目前市场上系统成本最优化,成本最具竞争力的ST-7100高清 H264/VC1 单片解决方案，以提供最基本的业务支持；在软件方面，IPTV机顶盒作为客户端产品，除了需要具有良好的硬件平台外还需要软件系统的配合才能够实现IPTV业务功能，在综合考虑

了各种需求后选定了与Linux相似、以核心为基础的、完全内存保护、多任务多进程的操作系统——STLinux作为嵌入式实时操作系统。

在选定IPTV机顶盒的软硬件软件系统后，结合IPTV机顶盒的功能要求，在充分考虑软件结构的健壮性、可移植性以及可扩展性的情况下。进行了软件系统的结构设计和功能实现定义，并对设计方案进行了比较详细的描述。在IPTV软件操作系统结构设计的实施过程中，主要结合了嵌入式浏览器Dillo，在对Dillo的解决方案进行分析后，将两者结合对IPTV软件系统进行了实现。其中，对操作系统的界面现实以及配置功能进行了详细的说明，其他部分的实现方法也进行了简要的说明。

最后，针对实现后的IPTV软件系统设计了一套测试方案，并进行了有针对性的测试。

关键字： IPTV、机顶盒、Dillo

THE DESIGN AND REALIZATION OF IPTV SET-TOP BOX'S SOFTWARE SYSTEM

ABSTRACT

IPTV, a more flexible communication method, has its huge influences on the increment of the ISP, ICP and IT service carrier in the future. IPTV has the extremely bright characteristic, may realize the media tender and the media consumer's substantive interaction, it can act according to the user the choice disposition many kinds of multimedia services function, including digital television program, visible IP telephone, DVD/CD broadcast, Internet browsing, receiving and dispatching email as well as many kinds of on-line information consultant, entertainment, education and commercial function.An IPTV service (or technology) is the new convergence service (or technology) of the telecommunication and broadcasting through QoS controlled Broadband Convergence IP Network including wire and wireless for the managed, controlled and secured delivery of a considerable number of multimedia contents such as Video, Audio, data and applications processed by platform to a customer via Television, PDA, Cellular, and Mobile TV terminal with STB module or similar device.

In this article, after analyzing the basic definitionand of IPTV set-top

box and its present situation home and aboroad, we summarized the IPTV set-top box's service requirements. By studying the service demands as well as thecorrelation documents of IPTV set-top box, we decided the software and hardware environment. For the hardware, we choose ST-7100 supporting H264/VC1 monolithic solution, which is the most optimizated, the most competitived chip system in the present market, to provid the basic functionality support; For the software, as a customer end product, IPTV requires a real-time operation system as well as hardware support to realize the required service functionalities. After considering all the demands, we decided to use STLinux, which uses Linux core with complete memory protection and multi-tasks, multi-threads, to be IPTV set-top box's real-time operating system.

Since we have selected the the hardware and RTOS of IPTV set-top box and know the functionalities of IPTV STB, we should design the software system architecture in the light of robustness, portability and scalability. In view of the above requirements for the structural design and realization of software systems, we give a more detailed description of the software system. During the implementation process of IPTV software system, we mainly finish the job with the embedded browser -- Dillo. After the analysis Dillo's solution, a combination of Dillo and the designed architecture becomes the implementation of IPTV set-top box software system. In the article, we take the the reality of GUI and

Configuration parts as examples to decribe the implementation. For other parts, we just give a brief note.

Finally, we design the the test plan of the implemented IPTV software system , and test the whole system.

KEY WORDS: IPTV, set-top box, dillo

独创性（或创新性）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，论文中不包含其它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它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：_____ 日期：_____

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：_____ 日期：_____

导师签名：_____ 日期：_____

第一章 综 述

1.1 课题背景介绍

随着信息通信技术，尤其是 IP 技术的不断发展以及国家信息化战略大力的推进，我国的互联网技术得到了极大发展。互联网上的多媒体业务和应用，如 IP 电话、视频会议、可视电话、音视频下载、电子商务、电子政务、互动游戏、视频点播、网络电视、旅游咨询、数字图书馆、数字博物馆、搜索引擎等都得到了蓬勃发展。随着这些 IP 多媒体业务的发展和繁荣，正在改变着经济增长方式，进一步满足了人民群众对信息技术、网络和服务的需要，成为电信网向下一代网络(NGN)发展的战略目标的重要组成部分。

在互联网及其业务和应用的发展过程中，出现了网络电视(或称 IPTV)这种融合了传统电信和传统广电业务的新型业务。所谓的 IPTV 是指交互式网络电视，是一种使用宽带 IP 网向用户提供影视节目在线观看的崭新业务。IPTV 与传统的有线电视相比有着根本性的改变，它的接收终端不再仅局限于电视，而可以是任何连接到 IPTV 机顶盒的显示终端或是具有 IPTV 机顶盒模块的数字电视

IPTV 不同于传统的模拟式有线电视和正在兴起的数字电视。因为，无论是模拟电视还是数字电视都无法拜托广播电视频分制、定时、单向广播等固有特点。而数字电视相对于模拟电视的许多技术革新，也仅仅只是信号形式的改变，并没有触及媒体内容的传播方式。IPTV 也不同于数据广播，数据广播通过设置一定的菜单供用户挑选，可以实现用户与中心简单的互动，但不能实现真正意义上的多种交互式服务，详细内容请查看参考文献[12]中的说明。真正的 IPTV 是利用宽带网络以及有线电视网的基础设施，以连接到 IPTV 机顶盒的显示终端或具有 IPTV 机顶盒模块的数字电视作为主要终端，通过互联网络协议来提供包括电视节目在内的多种数字媒体服务。IPTV 具有非常鲜明的特点，能够真正实现媒体提供者和媒体消费者的实质性互动，它能根据用户的选择配置多种多媒体服务功能，包括数字电视节目、可视 IP 电话、DVD/VCD 播放、互联网浏览、收发电子邮件以及多种在线信息咨询、娱乐、教育及商务功能，为消费者提供更灵活，更丰富的业务内容。经营 IPTV 业务的前提是拥有视频资源，作为最大网络资源提供者的电信运营商可以寻求与内容提供商的合作，并通过建立合理的合作模式提供丰富多彩的 IPTV 运营内容，而不必像广播电视那样仅仅只由指定的电视台提供节目资源。这一服务形式能够满足宽带用户对宽带应用和业务的需求，同时也能够为运营商带来潜在的丰厚的利润。可以说 IPTV 是运营商的宽带接入发展到一定阶段时，挖掘新的用户增长方式和业务量增长方式的一个好的途径，相信 IPTV 将成为拉动宽带网络发展的一项杀手级应用。

当今世界 IPTV 发展较快的国家是美国与日本。在美国，IPTV 是由电视运营商推出的，目前已经实现了三重业务整合的目标(数据、语音、多媒的结合，即 Triple Play 具体业务内容请查看参考文献[11]中的说明)。IPTV 业务的推出对只向

客户提供语音与数据服务的电信公司而言将会构成巨大的威胁，越来越多的电信公司认识到 IPTV 将是宽带未来发展的杀手级应用，它们正在加强其 DSL 基础设施建设并希望通过采用能高效利用带宽的先进视频编解码器，以增加其 IP 视频与 IP 语音服务

自从 2003 年 9 月，上海文广传媒集团的"东方宽频"推出网络电视业务开始，已经有很多公司尝试运营 IPTV。2005 年 5 月，广电总局将中国第一张 IPTV 牌照发放给了上海文广集团。中国电信等固网运营商也早已经开始基于 IPTV 的项目试点，并开始召集软件平台厂商、IP 机顶盒厂商和 SP、CP 等内容提供商探讨进军 IPTV 领域的可操作性。中国电信集团也正在组建"IPTV 联盟"，统一网络电视企业标准等问题。2004 年中国电信及中国网通与上海文广合作，在广东、上海、辽宁等地进行了商业化运作。同年，中国网通也在黑龙江、辽宁开通 IPTV 的商用系统。2005 年初，长虹与中国电信签订 IPTV 战略合作备忘录，明确了双方在 IPTV 领域的合作意向，展开从 IPTV 终端到渠道的全方位合作。2005 年末，中国电信与中国网通纷纷开始集中发力，开展 IPTV 相关的设备测试、网络建设招投标、业务试点与商业试运行等一系列的活动，中国电信启动了 5 省市 17 个本地网的 IPTV 试点，中国网通也宣布将 IPTV 试点城市从 1 个扩展到 20 个，这使得中国 IPTV 产业呈现出星火燎原之势。

1.2 IPTV 机顶盒简介

1.2.1 IPTV 的定义

首先，我们需要对 IPTV 有一个比较准确的定义，以便在后面能够更容易理解 IPTV 机顶盒软件系统。由于不同的行业、组织或知识背景的人，根据自身的情况对 IPTV 的含意存在理解上的差异性，使得业界到目前也没法对 IPTV 做一个公认的定义，对 IPTV 的业务形态(如直播、时移和点播)做出一个权威的规定。在此，我们以 ITU-T IPTV FG（IPTV ForceGroup）2006 年举行的 IPTV 相关讨论会上提出的定义为准，具体内容请查看参考文献[1]：

An IPTV service (or technology) is the new convergence service (or technology) of the telecommunication and broadcasting through QoS controlled Broadband Convergence IP Network including wire and wireless for the managed, controlled and secured delivery of a considerable number of multimedia contents such as Video, Audio, data and applications processed by platform to a customer via Television, PDA, Cellular, and Mobile TV terminal with STB module or similar device.

IPTV 服务或技术是一种全新的集中式电信和广播服务或技术。它能通过具有服务质量(QoS)控制的有线和无线网等宽带集中 IP 网络对包括视频、音频、数据和其他各种应用在内的各种多媒体内容进行管理、控制和安全传输，它能够

将这些应用从业务平台发送到具有 STB 模块或相似设备的电视、PDA、移动电话或移动电视终端为消费者提供服务。

根据上述定义，IPTV 所能够提供的总体业务构成应该如下图所示，业务结构定义请查看参考文献[19]、[20]、[21]、[24]、[25]、[26]：

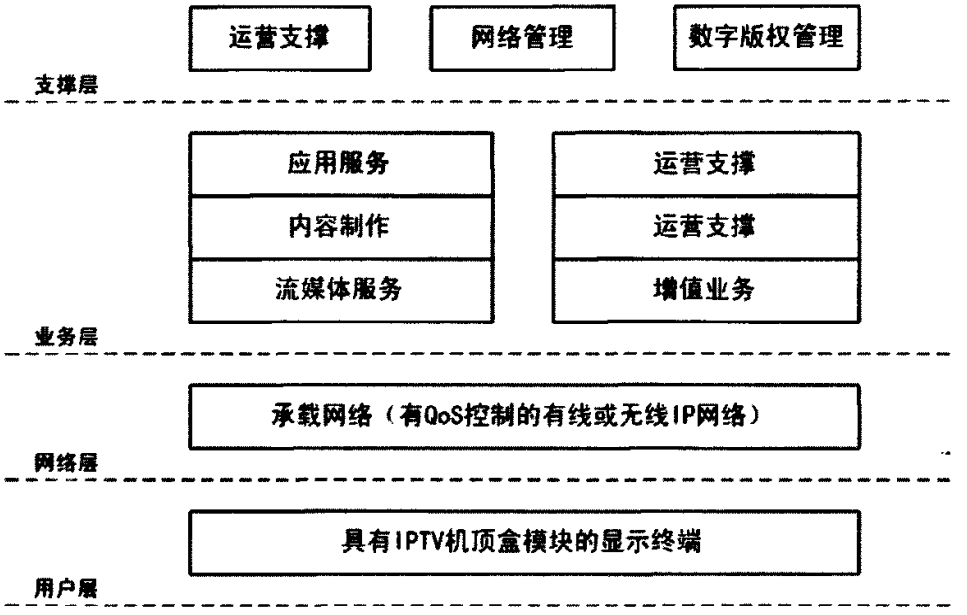

图 1-1 IPTV 总体结构

结合图 1-1 对 IPTV 业务的描述，我们可以从两个方面进行理解：

1) 对于"IP"的理解。首先，与传统 TV 有这最根本区别的就是 IPTV 的传输媒质，IPTV 的传输媒质已经从传统的有线电视网络转变为有 Qos（Quality of Service 关于服务质量相关的文档，在参考文献[4]、[5]中进行了详细的说明）控制的基于 IP 协议的广域有线或无线网络。也就是说，不论是 IPTV 业务中的"支撑层"、"业务层"还是"用户层"都承载在一张 IP 网络上；其次，数据传输的形式不再是传统电视节目的 Cable 传输形式，也不是数字电视的基于 TS 流的节目传输方式，而是通过有 Qos 控制的 TCP/IP 协议和流媒体协议共同控制节目的传输。
2) 对于"TV"的理解。首先，传统的 TV 节目都是由电视台提供的，但是 IPTV 的电视节目内容来源可以是电视台，也可以是网络运营商甚至是 Internet 内容提供商来提供，所有的内容只需要存储在运行于 IP 网络上的内容服务器中；其次，传统电视节目的传播方式是采用广播的形式，即使是有线电视也仅仅是向用户 push 节目，但是 IPTV 却为用户提供了更多的选择，其节目的播放不仅仅可以采用节目提供商 push 的形式，更可以采用用户 pull 的方式，由传统的点对多点的传输方式转变为点对点的传输；最后，传统电视的接收终端必须为模拟电视，目前也允许数字电视的使用，但是 IPTV 则是通过具有类似机顶盒模块的电视、PDA、手机或移动电视作为终端，这样就大大提高了用户对于 IPTV 业务提供方式的选择。

结合上述说明，给出 IPTV 系统的网络结构图如图 1-2 所示，具体内容请查看参考文献[2]：

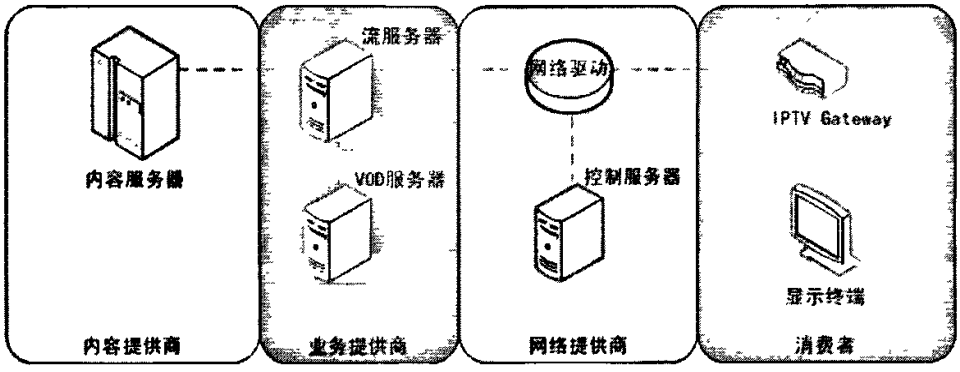

图 1-2 IPTV 网络结构图

总的来说，随着技术的发展，IPTV 技术的出现使得传统的视频终端——电视机不再只是用作看电视节目，而是可以当作其它多种业务(如游戏、股票行情、教学、医疗)的综合型终端使用。电视节目也可以不只是在电视机上收看，也可以在 PC 和手机等设备上收看。

1.2.2 机顶盒的定义

机顶盒（set-top box，STB）起源于 20 世纪 90 年代初，当时主要是欧美国家有线电视台为解决有线电视收视费问题而设计的一个解扰设备。它的主要作用是使用户能够用原有的模拟电视机收看数字电视节目和高清数字电视节目，即提供数 / 模信号转换功能，通常把这类机顶盒称为数字电视机顶盒。

IPTV 如火如荼的发展带动了终端市场的活跃，电信设备制造商纷纷投入机顶盒的研发生产，原有数字电视机顶盒厂商也在尝试制作双向 IPTV 机顶盒。机顶盒已经由原来单一的解扰或数 / 模转换专用机顶盒，发展到支持包括 IP 在内的多种接入方式，具有多种编解码能力和图形浏览器功能，可以支持基于数字电视的视频点播、时移电视、网络浏览、信息服务、远程教学和医疗、互动游戏等业务功能的 IPTV 机顶盒。未来 IPTV 机顶盒将作为一种家用数字平台被广泛用于不断扩大的交互式多媒体数字内容服务领域。

对 IPTV 机顶盒的技术需求，这里主要从两个方面考虑，一是实现业务的需求，二是用户体验的需求。在之前提到过 IPTV 可提供的业务有电视直播、视频点播、时移电视、可视业务、信息业务、网络浏览、互动游戏，以及未来可扩充的业务。由于 IPTV 的主要业务特征体现在交互式多媒体检索和播放形式上，因此需要网络具有双向交互的能力，因为节目信息的获取与传送需要双向信息交互，节目播控也需要交互；播放需要足够的接入带宽和媒体传输协议；多媒体需要音视频编解码能力。因此，为实现 IPTV 业务应用的要求，需要 IPTV 机顶盒具有支持接收网络直播电视节目，进行视频点播、时移点播等基本业务功能，同时还要具有互联网浏览和信息服务、电子节目指南（EPG）和节目检索，以及互动游戏和软件在线升级等扩展功能。作为 IPTV 业务终端的机顶盒设备应该具有开放式系统结构，以适应不断升级和扩展的业务需求；充分考虑网络环境和协议因素，要求其具有不少于两种的上行网络接口，支持多种接入认证协议和 IP 协

议；要求具有音视频媒体编解码功能，以及业务应用和管理功能等。总之，从业务角度需要 IPTV 机顶盒具有开放型业务能力，以满足不断发展的业务需求。

IPTV 机顶盒的应用过程一般为业务内容通过宽带网络接入到机顶盒，利用机顶盒遥控器或遥控键盘向宽带网络索要视频内容，宽带网络将需要的视频信号传送到用户端，经过机顶盒解码后在电视上输出。实现视频点播或网页浏览等。从用户体验的需求考虑，IPTV 业务所带来的改变是将传统互联网 PC 终端观看流媒体视频的书房体验转移到客厅，将前倾坐姿变为后仰坐姿，将键盘操作改为遥控器操作。这些改变要求 IPTV 机顶盒的设计要充分考虑习惯于操作电视机的广大用户群体，应尽量保持操作简单、性价比适中和耐用的特点。

1.3 IPTV 机顶盒软件系统要求

1.3.1 IPTV 机顶盒软件系统功能要求

根据前期进行 IPTV 机顶盒应用的调查，特别是结合了《国家 IPTV 机顶盒技术规范（征求意见稿）》（参考文献[23]）、《中国网通 IPTV 机顶盒技术规范草案 V1.0》（参考文献[18]）以及 ITU-T 的 IPTV FG 各小组（参考文献[13]中列出，组织结构在参考文献[14]中列出）提供的相关文档中关于 IPTV 机顶盒基本功能的定义。根据这些规范的要求目前要符合宽带网络通讯要求的 IPTV 机顶盒，在功能上已经不能仅提供如最早的仅提供单一的解扰或数 / 模转换专用机顶盒一般，而需要以宽带为硬件基础，提供音视频播放、网页浏览、网络游戏等综合多种多媒体业务的 IPTV 机顶盒客户端设备。

也就是说，IPTV 机顶盒除了提供作为最基本功能的接收数字电视广播节目，同时具有所有广播和交互式多媒体应用功能。以下列出 IPTV 电视机顶盒所应该具备的所有功能定义，具体内容请查看参考文献[3]。

1.3.1.1 基本功能

1) 电视功能：
   - 直播电视：用户可以通过 IP 网络实时在线接收各种音视频广播节目，并可以通过节目列表选择不同的音视频频道。
   - 点播电视：在 IPTV 业务平台提供的检索界面（EPG）中选择需要的节目进行点播。
2) 网页浏览、配置及通信功能：
   - HTML 浏览器：能够支持 HTML4.0 的浏览器，用于读取来自 ICP 网站的内容或者进行正常的网页浏览。
   - 自动关机及配置保存：在用户所选的通道无信号时间超过 15 分钟以上的时候，IPTV 机顶盒必须能够自动关机；在关开机后必须能够保留用户的设置。
   - 工厂调试菜单：IPTV 机顶盒必须具有工厂调试菜单，此菜单要比用户可配置菜单更复杂，用于在机顶盒出厂或维护维修时对机顶盒进行配置。
   - 实现符合标准规范的通信接口程序和通信协议栈程序。

3) 管理功能：
   - 日志功能：对本机的版本信息、系统信息的管理和错误信息的诊断；上传日志到管理服务器。
   - 远程配置：用户可通过局域网对机顶盒参数进行配置，机顶盒需提供 WEB 配置界面。
   - 本地配置：用户可通过遥控器对机顶盒参数进行配置。
   - 软件升级：机顶盒可通过网络获取最新软件版本号，判断是否需要升级；也可通过外围存储设备获取软件版本号，判断是否需要升级；通过 HTTP(S)或 FTP（TFTP）协议获取最新版本文件；如果升级失败，机顶盒必须仍能正常工作。

1.3.1.2 扩展功能

1) 电视功能：
   - 时移电视业务：用户可以观看已播放过的广播电视，这包括查看时间表、下载节目、存储节目、自动清空过期节目等功能。
   - 网络广播电视业务：用户可以观看以 IGMP 协议组播的电视节目。
2) 扩展应用程序：
   - HTML 浏览器：进一步支持 JavaScript，CSS1，DOM1 等协议。
   - 局域网文件共享：能够允许用户通过局域网访问其它 IPTV 机顶盒存储的节目，搜索想要的已录制的电视节目。
   - 图片浏览器：支持 JPG，GIF，PNG，BMP 等图片格式.
   - 电子邮件客户端：允许用户通过邮件客户端收发 Email，即要求网络支持 SMTP 以及 POP3 协议。
3) 扩展安全程序：
   - 认证安全：终端在接入 IP 网络时应支持 AAA 认证方式，提供必要的个人信息，包括用户识别码或用户名、密码，经过认证系统认证授权后，才可以使用 IPTV 业务平台提供的各类业务。
   - 数字版权管理（DRM）：机顶盒可以针对业务平台 DRM 的相应版本，提供内容版权的保护。
4) 其它扩展程序：
   - 文本信息业务，信息交互业务，网络游戏业务，可视电话业务等。
   - 网络时间同步，画中画功能，密码设定功能等。

1.3.2 IPTV 机顶盒软件系统结构要求

由于 IPTV 机顶盒作为整个 IPTV 系统的客户端设备，考虑到用户消费能力等因素要求就是体积小、价格便宜。因此在进行 IPTV 机顶盒的开发实际是在进行嵌入式软件的开发。IPTV 机顶盒的运行环境需要一个嵌入式系统，而嵌入式系统仅是对于硬件启动与运行的控制，并没有提供数字机顶盒所需要的控制系统，因此软件系统的设计实际就是完成在嵌入式操作系统之上的，除提供机顶盒所需的基本操作之外，还需要可支持业务扩展的中间件(Midware)层。一个完整的数字机顶盒由硬件平台和软件系统组成，可以分为 4 层，从底层向上分别为：

硬件(Hardware)、实时操作系统(RTOS)、中间件(Midware Layer)、应用层(ApplicationLayer)，如图 1-3 所示。

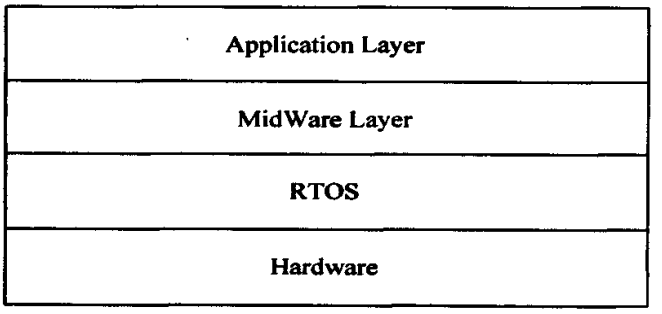

图 1-3 IPTV 机顶盒总体结构

1) 硬件（Hardware）提供机顶盒的硬件平台，为整个 IPTV 机顶盒运行的基础，可以提供音视频输入/输出、网络接口、音视频解码器等功能；
2) 实时操作系统（RTOS）。实时操作系统负责本地资源和网络资源的管理，提供基本的操作功能和设备的访问控制。IPTV 机顶盒的 RTOS 是存储在 FLASH ROM 上的，通过引导启动机顶盒。要求能够在启动时对系统部分硬件进行检测，如 DRAM 等，同时提供各种硬件的驱动；对于 RTOS 中网络资源的要求，请查看参考文献[8]中的说明；
3) 中间件层(Midware Layer)是一种将应用程序与底层的操作系统、硬件细节隔离开来的软件环境。它将应用软件与依赖硬件的底层软件分隔开来，使应用不依赖于特定的硬件环境进行运行，具体要求请查看参考文献[9]中的说明；
4) 应用层(Application Layer)提供包括以下功能，具体要求请查看参考文献[6]、[7]中的说明：
   a) 系统设置。用户可通过遥控器对系统必要的参数进行设置，如网络访问方式、DTV 节目信息表等；
   b) DTV 功能。用户在无点播服务的情况下，可看数字电视节目；
   c) 多媒体数据交互功能。即提供基于 IP 网络的其他数据交互业务。
   d) 系统升级。通过判别系统或台标的名称和版本号，下载升级 FLASH ROM 中的系统或台标数据。
   e) 加解扰技术。采用加扰控制字加密传输的方法，用户端利用 IC 卡解密。其中还包括节目来源、时间、内容分类和节目价格等节目信息。用户要观看节目需经过用户个人分配密钥(PDK)的加密处理，EMMs 中还包括地址、用户授权信息，如用户可以看的节目或时间段，用户付的收视费等。

1.4 完成的工作

针对课题的内容，本人主要完成了以下几个方面的工作：

1）IPTV 业务体系的调查。主要是通过阅读已有的关于 IPTV 的结构定义，

现有系统了解实现IPTV机顶盒在整个IPTV业务体系中所应该扮演的角色和完成的功能；

2）IPTV机顶盒操作系统的选择。通过比较已有的嵌入式操作系统，选择适合IPTV机顶盒软件系统的操作系统，要求系统能够拥有强大的网络支持以及良好的可编程接口，并支持选定的硬件芯片。

3）IPTV机顶盒软件系统的结构设计。根据IPTV机顶盒软件系统的基本功能要求，设计能够完成功能定义要求的，具有稳定性、可扩展性的软件体系结构，并定义软件结构中各部分的基本功能及协作流程。

4）IPTV机顶盒软件结构中部分模块的编写。编写了一些函数基本库，为后续模块开发提供一些基本功能。结合作为参考代码的软件系统，完成结构设计中定义的显示模块、配置模块等，并在开发过程中完成其他模块的基础结构性代码，指导低年级的同学根据软件结构设计要求完成其他模块的代码工作。

5）IPTV机顶盒软件系统的测试。设计了IPTV软件系统的测试方案，并编写了部分测试服务器的代码，在整个测试过程中，在完成部分测试的情况下，指导低年级同学完成测试，并修正相关Bug。

6）IPTV机顶盒软件系统文档编写。在整个项目实施过程中，负责从设计文档到最终测试文档的大部分文档编写。并在整个项目实施过程中，指定项目进度。

第二章 IPTV 机顶盒软件系统的设计

2.1 简述

根据上一章提出的 IPTV 机顶盒所需要具备的基本功能，我们了解了 IPTV 机顶盒除了需要提供作为最基本功能的接收数字电视广播节目，同时具有所有广播和交互式多媒体应用功能。从根据 IPTV 总体功能定义，要完成所有功能需要比较长的时间，因此根据业务功能的重要性在系统的实现过程中，只是先实现了基本要求中的功能，并在要求在软件系统的设计及开发过程中预留进行二次开发的接口，为将来扩展功能的开发提供了基础。以下将通过对 IPTV 机顶盒硬件以及软件的描述对整个实现方案进行说明，相关 IPTV 终端系统要求，请查看参考文献[10]。

2.2 IPTV 机顶盒的硬件环境

2.2.1 IPTV 机顶盒硬件的选型

在硬件方面，综合考虑了目前的市场状况以及各公司产品的性能，决定采用 STMicroelectronics 公司的产品。意法半导体有限公司(STMicroelectronics)是全球独立的半导体公司，它是各种微电子应用系列开发和转让芯片级解决方案的领导者。作为全球最大的半导体公司之一，意法半导体（ST）是全球模拟集成电路、MPEG-2 解码器集成电路和 ASIC（专用集成电路）/ ASSP 的世界级领导厂商。另外，在存储器市场，意法半导体（ST）是 NOR 闪存的第四大供应商。在应用领域，意法半导体（ST）是机顶盒集成电路最大的供应商，智能卡和和硬盘驱动集成电路和 xDSL 芯片的第二大供应商，无线通信业务和汽车集成电路的第三大供应商。公司采用多种制造工艺和专利设计方法来生产和设计产品。意法半导体（ST）也利用其所拥有的广泛知识产权组合，与其它许多主要半导体制造商达成相互许可证协议，从而增强了公司的工艺与设计技术的深度和广度。

在目前国内机顶盒市场，意法半导体（ST）公司的芯片系统占有 70%以上的市场份额，其出产的 ST-710x 系列是目前市场上系统成本最优化,成本最具竞争力的高清 H264/VC1 单片解决方案。ST-710x 系列兼容高清 MPEG2 和 MPEG4,包括 H.264 和 VC1,并支持画中画功能，它还支持数字录像机和 USB 2.0 主控制器接口，因此我们在课题中选择的主板是 ST-7100x 系列中的 ST-7100b 作为硬件基础，硬件具体信息在参考文献[16]中的说明。

2.2.2 IPTV 机顶盒硬件的主要功能

ST7100b 是新一代高清晰机顶盒解码芯片，它在低耗 HD 系统有着很高的性能。ST-7100b 增强了 STi7710 的功能，其中包括一个 266MHz 的 ST40 CPU 用于应用及设备控制；一个用于低比特率 H.264/AVC 应用，并具有 MPEG-2 双向解码能力的视频解码器；一个能够解码 PCM 音频的音频解码器。ST-7100b 基于 Omega（STBus）的结构，能够提供一套完整的后端处理解决方案的芯片系统，主要用于地面、卫星以及电缆传输的，支持 ATSC、DVB、DIRECTTV、DCII、OpenCable 以及 ARIB BS4 规范的高清晰机顶盒。ST-7100b 能够拆分、解密并解码一个有着多音频信道的高清或标清视频流。视频可以输出到两种独立的现实格式：用于电视的全分辨率显示；用于 VCR 或另一个标清电视的下采样显示。视频信号可以采用一个模拟组件接口或一个受版权保护的 DVI/HDMI 接口，连接到一个电视或显示板上。音频信号采用可选的 PCM 混合输出到一个 S/PDIF 接口或 PCM 接口，也可以是一个综合立体上音频 DAC。同时，数字化的模拟节目也可以输入到 ST-7100b 上用于重排和显示，一个双工显示混合器用于针对单独 TV 和 VCR 输出的图形混合及独立视频复合。ST-7100b 还包括了一个能不间断复合和处理来自三条不同节目源的传输流的流复合器。应用支持在前端向卫星或电缆请求 EBP 或数据刘的同时仍然能够进行地面 DVR 时移电视节目的处理。

2.2.3 IPTV 机顶盒硬件的视频处理能力

ST-7100 的数字视频处理器具有很强大的功能，主要特点如下：

1）ITU-R BT.601/656 兼容，720x480, 625/50i 720x576 象征性格式。
2）SQ 像素支持，525/60i 640x480, 625/50i 768x576；
3）针对不提供嵌入码字的视频流（SAV/EAV 协议）的外部同步支持；
4）视频数据以 YCbCr 4:2:2 原始格式被捕捉到系统本地内存（与 2D-和 RDP 显示兼容）；
5）用户定义的捕捉窗口用于选择一个输入视频活动区域的子区域；
6）采用分页循环缓存的通用辅助数据捕捉处理器（SMPTE 291M，ITU-RBT-1364）用于主 CPU 的预处理。

2.2.4 IPTV 机顶盒硬件的音频处理能力

ST-7100 音频子系统能够解码并播放不同标准的多信道压缩音频流。音频解码器是一个 400MHz 的 ST231 CPU，它能够读入从内存中读入编码数据并将解码后的 PCM 数据写入内存。ST231 音频处理器从属于 ST40 主 CPU，能够从 EMI 或 LMI（系统和视频）中执行它的代码。音频流（已编码或已解码）既可以是一个基于数字 PCM 输入接口的外部源也可以是一个内存之上的传输子系统中的内部源。解码后的音频数据可以以模拟或数字的格式输出。ST-7100 音频解码器的功能如下：

1）兼容所有的通用音频标准；
2）来自内部或外部源的 PCM 混合，支持抽样速率转换(32, 44.1, 48 KHz)；
3）在 S/PDIF 输出上的数字音频（IEC 61937）；

4）6 信道至 2 信道的下混合；
5）PCM 音频输入（I2S 格式）；
6）多信道数字 PCM 输出（10 信道）；
7）2 信道描述的信道编码；
8）预处理（信道虚拟）：Dolby 下混合、音量控制、低音复位向。

2.3 IPTV 机顶盒的软件环境

IPTV 机顶盒作为客户端产品，除了需要具有良好的硬件平台外还需要软件系统的配合才能够实现 IPTV 业务功能。目前市场上大多数的机顶盒软件系统均采用分层结构，一般分成三层：应用层、中间解释层和资源层，每一层都包括诸多程序和编程接口，如下图 2-1 所示。

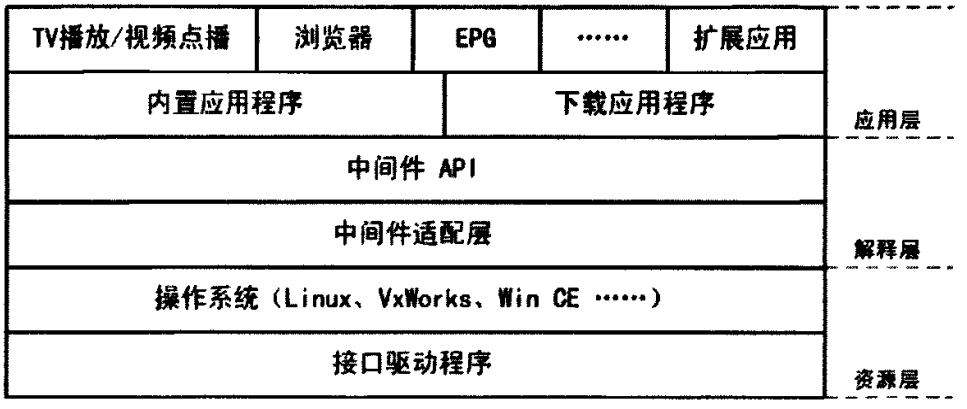

图 2-1　IPTV 软件基本结构

上图划分了机顶盒软件的三层结构，其中资源层软件包括实时操作系统及机顶盒硬件的驱动程序，主要用于完成对硬件设备的操作。解释层的主要功能是将机顶盒应用程序翻译成 CPU 能识别的指令，去调动硬件设备完成相应的操作。应用层可以分成内置应用程序和下载应用程序两部分，实现诸如 TV 播放、视频点播、EPG、DRM（数字版权管理）、游戏下载等业务应用。

在考虑对于操作系统的选择上面，首先要求满足以下一些条件：

- 可装卸性。开放性、可伸缩性的体系结构。
- 强实时性。EOS 实时性一般较强，可用于各种设备控制当中。
- 统一的接口。提供各种设备驱动接口。
- 操作方便、简单、提供友好的图形 GUI，图形界面，追求易学易用。
- 提供强大的网络功能，支持 TCP/IP 协议及其它协议，提供 TCP/UDP/IP/PPP 协议支持及统一的 MAC 访问层接口，为各种移动计算设备预留接口。
- 强稳定性，弱交互性。嵌入式系统一旦开始运行就不需要用户过多的干预，这就要负责系统管理的 EOS 具有较强的稳定性。嵌入式操作系统的用户接口一般不提供操作命令，它通过系统的调用命令向用户程序提供服务。
- 固化代码。在嵌入式系统中，嵌入式操作系统和应用软件被固化在嵌入

式系统计算机的 ROM 中。辅助存储器在嵌入式系统中很少使用，因此，嵌入式操作系统的文件管理功能应该能够很容易地拆卸，而用各种内存文件系统。

- 更好的硬件适应性，也就是良好的移植性。

在比较目前比较流行的嵌入式操作系统 Linux、VxWorks 以及 WindowsCE 后，决定选用 Linux 嵌入式操作系统。Linux 为嵌入操作系统提供了一个极有吸引力的选择，它是个和 Unix 相似、以核心为基础的、完全内存保护、多任务多进程的操作系统。支持广泛的计算机硬件,包括 PPC,ARM,NEC,MOTOROLA 等现有的大部分芯片。程序源码全部公开，任何人可以修改并在 GNU 通用公共许可证(GNU General Public License)下发行。Linux 带有 Unix 用户熟悉的完善的开发工具，几乎所有的 Unix 系统的应用软件都已移植到了 Linux 上。Linux 还提供了强大的网络功能，有多种可选择窗口管理器（X windows）。其强大的语言编译器 gcc、g++等也可以很容易得到。不但成熟完善、而且使用方便。

2.4 IPTV 机顶盒的软件结构设计

2.4.1 软件设计要求

对于高效的软件工程，良好的设计是关键。一个设计得好的软件系统应该是可直接实现、易于维护、易懂和可靠的。设计得不好的系统，尽管可以工作，但很可能维护起来费用昂贵、测试困难和不可靠，因此，设计阶段是软件开发过程中最重要的阶段。

软件设计的这种方法导致了许多动态的和非常昂贵的工程失败。现代软件工程学已经认识到一些完全非正规的表示法，诸如接近于编程语言的流程图，不适用于系统设计的描述和表达。精确的（尽管并不一定是正规的）说明是设计过程的必要部分。软件设计是一个反复的、不能用任何单一表示法来表示的多层次活动。相应地，大量的设计表示法，如数据流图、层次式输入-处理-输出结构图和设计描述语言已经开发出来，这些表示法能比流程图更好地表达软件设计。一个成功的设计应该是：能生成高效的代码，实现尽量紧凑的最小设计，或是一个最易维护的设计。

在给定一个需求定义，软件工程师必须以此导出满足这些需求的程序系统的设计，此导出过程是通过下述步骤来完成的：

1) 必须建立组成程序系统的子系统。
2) 必须把每个子系统分解成分离的成分，并且子系统规范通过定义这些成分的操作来建立。
3) 每个程序可以用相互作用的子成分设计。
4) 每个成分还须进行优化，这通常需要将每个成分规范化成层次式的子成分。
5) 优化过程中的某个阶段，各成分中的算法必须详细说明。

除了程序系统设计中的这些阶段之外，软件工程师也可能需要设计允许系统中各进程之间进行通信的通信机制。软件工程师或许要设计文件结构，并且很可能要设计用于程序的数据结构，还需要设计确认程序的测试事例。

2.4.2 IPTV 机顶盒软件系统设计方案

对于软件开发来说，最重要的部分就是进行系统设计，这是决定一个软件的价值以及生存周期的最关键因素。一个好的软件结构不仅能够应对可能出现的重大错误，增强软件的鲁棒性，同时也需要具有强大的可扩展性以满足消费者不断提出的新要求，增强软件的生命周期，降低二次乃至多次升级开发的难度。

目前最流行的软件设计方式就是采用通过统一的调用接口，将业务需求划分为最基本的业务单元，对于一次业务请求将被划分为多个业务单元的相互组合。这种做法的好处在于降低各业务基本单元之间的关联性，同时增加业务需求的可配置性。一旦当业务流程中有某个业务单元需要更换或者是业务需求发生了变化，需要在某个业务流程中新增一些业务单元以丰富业务内容，那么只需要替换某些需要更新的模块或增加一些必要的业务单元就能够完成软件的升级过程。因此在综合以上的一些因素的考虑，同时考虑到 IPTV 机顶盒对于新增业务的不同要求以及不同厂商或运营商之间对于业务的不同理解，初步设计的软件实现结构如图 2-2 所示：

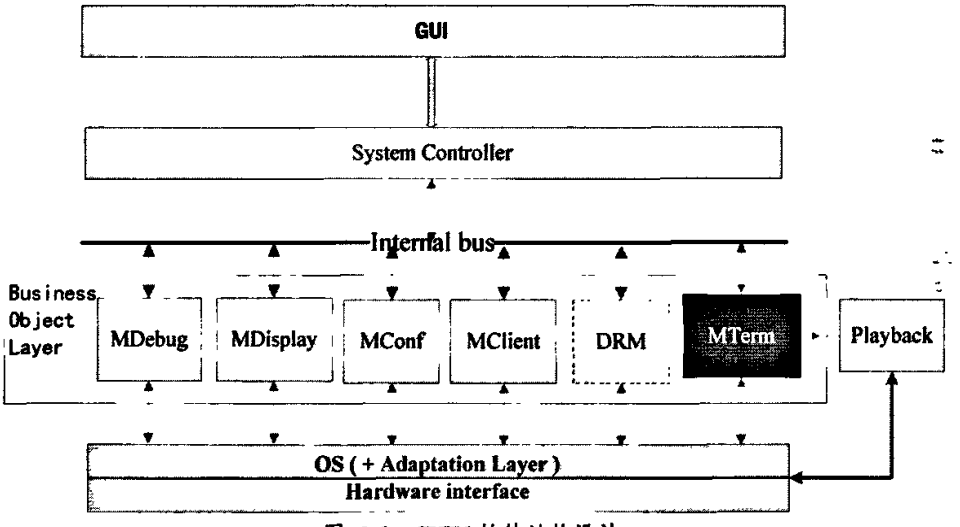

图 2-2 IPTV 软件结构设计

根据图 2-2 设计的系统基本结构可以看出，通过统一的对外用户接口 GUI 层，用户可以进行与软件的交互。在软件内部，通过 SystemController 完成对于各个业务单元的组合与管理，然后通过 InternalBus 作为数据传输的通道向底层的 MDebug、MDisplay、MConf、MClient、MTerm 等模块发送指令或数据。所有的底层业务单元都是基于操作系统之上的，在这中间可以通过一个 AdaptationLayer 进行连接，也允许各模块对 OS 或相关硬件进行直接操作。

2.4.2.1 GUI

GUI 层主要负责将 HTML 文件解析并进行现实，并且标明当前选中业务或发起节目播放业务等等。GUI 层除了能够现实 HTML 文件外，还应该保留有当

前选中的节目或页面跳转对象在软件内建数据结构中的当前位置，并且能够根据用户的操作移动该位置指针，并当用户请求播放的时候采用某中方式获取该指针所指向的节目的 URL 地址，并传递给 SystemController 调用相关模块实现节目播放。

2.4.2.2 SystemController

SystemController是整个软件的系统控制层，用来作为GUI层发起业务的解释器，并根据该次业务所需要的底层业务模块进行模块组合，是系统内部处理实际的起点。1)该层在整个系统中的作用，从GUI层看来，应该是它根据实际的业务需要，定义InternalBus中不同的业务对象，并分配给底层不同的业务实体，进行实际业务的组合与操作，同时当业务内容从业务实体层返回时，为业务内容在GUI层的重新展示做准备；2)从下层的业务实体层（BusinessObjectLayer）看来，SystemController层是来自GUI层业务需求的一个翻译者，完成了从用户事件到业务内容的翻译，是业务的调用者，同时也是一次业务操作结果的接收者，并且将之翻译为GUI层能够接受的形式向上传递（可能是一个HTML结构，也可能就是一个HTML页面）。

2.4.2.3 InternalBus

InternalBus是系统内部的数据总线，其主要功能是连接SystemController以及各功能模块，并提供一种可靠的数据传递以及管理机制。

2.4.2.4 MDisplay 模块

MDisplay 模块应该是管理所有与现实相关的操作，它的主要作用应该包括以下一些：

1） 当用户进行菜单内容请求时，能够根据用户的内容请求生成相关的页面；
2） 当用户的请求与 EPG 菜单相关的时候，能够请求 EPG 服务器获得菜单内容，并形成相关的显示页面。
3） 当用户进行点播业务的时候，能够将节目的 URL 地址返回给 SystemController，并由 SystemController 调用相关模块进行播放。
4） MDiplay 模块的操作应该是与其他模块独立的，它只负责生成需要的页面，而不负责页面的具体标识。

2.4.2.6 MConf 模块

对系统的配置信息进行管理，其中包括对于STB传输平台配置的设定；对STB显示等本地配置的设定等功能。既支持通过IE在局域网对STB进行设置，也支持

来自GUI的用户配置。1)本模块应该在系统启动最先被调用，从文件中读出Config文件中的配置信息后，Conf模块保管所有的配置信息，并将配置信息通过InternalBus传递给SystemController作为系统初始化的依据。2)Conf模块中保存的配置信息应该包含三部分：

1) 网络连接相关配置信息：应该由用户通过IE进行配置。
2) 显示相关配置信息：应该由用户通过系统GUI进行配置。
3) 数据处理配置信息：根据配置文件，完成模块加载，形成特定数据处理链路，能根据需求家在模块适应不同厂家的产品。如果用户在系统运行过程中对网络连接进行重新配置，则有可能需要重新启动MTerm进程，改变连接状态；如果用户改变的是显示相关配置，则需要在MTerm进程中由主控进程进行配置信息的变更。不论是通过IE进行配置还是通过GUI进行配置，都在用户确认更改后都应该直接保存在Config文件中。（注意：有些配置信息在系统运行过程中是不允许更改的，所以需要在MClient中保存当前运行状态，在用户更改时通过SystemController调用MClient模块做必要的检查。）

2.4.2.5 MClient 模块

对系统运行状况的监视，其中主要包括的是

1) 系统版本信息的检查、日志信息的处理、模块升级的管理等等。系统运行过程中任何一次业务的执行，都应该先调用MClient模块进行状态检查。
2) 系统升级检查。在开机时检查是否有最新可用的更新，并在系统运行过程中定期检查是否有可用更新，一旦存在可用更新，即通知SystemController对该模块进行停止，并连接更新服务器对该模块进行更新。
3) 日志管理。每次系统开关机，以及系统运行出现致命错误的时候，写入日志信息为系统维护提供依据。

2.4.2.7 MTerm 模块

Mterm模块主要用于针对特定音视频资源的播放。其结构定义如下：

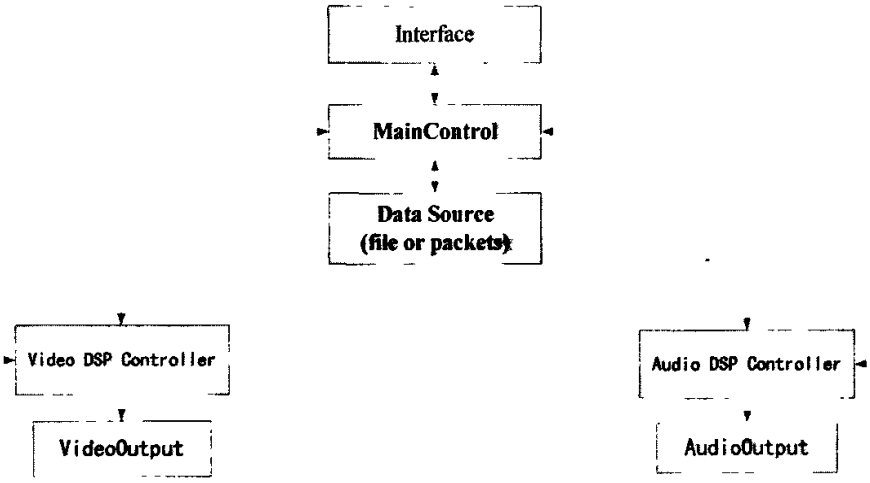

图 2-3 MTerm模块内部结构设计

1) Interface

其对外借口为Mterm Interface。MTerm对外的接口，用于提供与主进程之间的通讯用。它的作用应该是这样的：

- 用户通过遥控器控制STB，需要在电视上显示播放菜单。
- STB接收到信号，通过底层的硬件处理，向主应用进程发送请求。
- 主应用进程在SystemController层捕捉到请求，调用辅助工具显示操作菜单。并在用户有基于菜单的操作的时候组织相应数据，通过MTerm进程提供的对外接口，发送用户操作信息。
- 对于主进程SystemController发送的操作信息，由Interface捕捉到在MainController模块中根据请求作出对应处理。

2) MainControl

模块内部主控程序，其主要负责以下功能：

- 用户事件管理(Event Management)，确定文件打开等用户操作。并根据DataSource状况，确定A/V的状态。同时，将操作结果返回GUI（非视频/音频播放部分）。
- 控制STB的物理时间(Phy Clock Management)，为A/V输出提供绝对时间作为标准。
- 错误处理(Error Management)：当系统模块出现错误的时候，需要调用MainControl提供的统一的错误处理函数，将错误代码传递给MainControl，由它统一根据错误代码调用各线程提供的根据不同错误级别定义的错误处理API。
- 内部消息管理：对于来自Mterm系统内部的操作请求，比如说内部的系统错误的处理，MainControl需要向下提供错误处理的接口，以便在系统出现错误的时候能够正确设置Mterm的状态。

3) DataSource

主要用于获取作为播出数据源的音视频，主要功能描述如下：

- Signaling：网络传输中的信令交互以及传输过程中的状态监测。
- DataFeeding：需要考虑如何处理时延及抖动以及丢包情况下的处理。
- ClockMangement：需要根据服务器时间或者网络时间（RTCP-based）来进行ClockSync。
- Demultiplex：无论对于文件还是网络输入，都需要对读入的内容进行解复用，并将音视频分别存入A/V编码控制器的Buffer中。
- A/V Sync：对于存入A/V编码控制器Buffer中的数据需要提供包中的TimeStamp来做音视频同步。
- Decryption：可以根据DRM模块提供的数字证书进行拥护认证和获取密钥。

4) Video Controller

主要用于控制视频解码芯片，功能要求如下：

- 控制视频解码DSP输入Buffer的管理。涉及从Buffer中读取数据送至DSP的速率。
- 根据时间戳和MainControl提供的绝对时间确定视频输出的时间。
- 有可能会涉及视频输出的管理。

5) Audio Controller

主要用于控制音频解码芯片，功能要求如下：

- 控制音频解码DSP输入Buffer的管理。涉及从Buffer中读取数据送至DSP的速率。
- 根据时间戳和MainControl提供的绝对时间确定音频输出的时间。
- 有可能会涉及音频输出的管理。

2.4.2.7 MDebug 模块

MDebug此模块的功能是，STB在进行调试时所需要显示的一些信息，主要是关于MTerm模块接收到的音视频数据包的各种性能测试数据，并对其进行整理后，显示在STB系统软件的主界面上。

以上就是针对IPTV机顶盒的软件结构的总体设计，各个部分之间相互的相互协作以及软件的可扩展性得到了进一步的实现。接下来需要考虑的就是如何结合现有的一些软件对这个结构进行实现。因为针对不同的IPTV ICP他们的网站内容都是不同的，因此可以有两种实现方式

1）针对不同的业务提供商，采用HTML页面提取算法将页面的内容全部提取出来，然后根据不同的情况形成IPTV机顶盒自己的EGP形式。这样的做法能够令用户得到更好的交互界面体验，也有利于节目的展现，但是缺

点在于HTML页面提取算法比较困难，并且针对目前的网站运行情况以及项目实施安排，这种做法的可行性比较低。

2） 直接采用IE浏览的方式，基于嵌入式浏览器在上面显示IPTV ICP的EGP。这种做法所有的用户交互界面均基于HTML的形式，优点在于能够适应目前大部分ICP供应商所提供的服务，并且类似于传统的电视界面。

综合以上两种选择，特别是在考虑了程序开发周期以及多业务的适用性上面，决定采用方案（2）进行实施。

第三章 IPTV 机顶盒软件系统的实现

3.1 IPTV 机顶盒软件结构设计的实现方案

根据上一节关于 IPTV 机顶盒软件结构设计的说明，软件的实现需要完整的考虑可实现性及可扩展性。软件的具体实现可以在已有的一部分代码的基础上进行实现，在考察了多个已有开源项目后，决定选用 Dillo 进行开发（参考文献[27]）。整个 IPTV 机顶盒软件系统的 GUI 页面设计，请参考文献[17]。

3.1.1 Dillo 的介绍

目前采用的是在嵌入式Linux系统下的一个比较程序的开源浏览器——Dillo。Dillo是一款小巧的网页浏览器（源代码约420 KB，二进制程序约350KB。），遵循GPL协议。采C语言进行编写，在其中使用了 GTK toolkit，该浏览器特别适合运行于老计算机，以及嵌入系统。 目前最新版本的Dillo是0.8.4，还不支持 CSS 、JavaScript 等，对框架的支持也很有限。 Dillo作为一个基于Linux Gtk+实现的嵌入式浏览器，采用了封装Gtk Widget的方法来实现HTML页面渲染，这些最基本单元被称为Dw Widget，Dillo提供了一种基本GTK+ Framework的框架结构，它与GTK+ Framework非常类似。之所以采用Dw Widget而不是GtkWidget的原因是GTK太大了，有很多部分是用不到的；HTML与Desktop GUI渲染的要求是不同的，具体区别见下表：

表 3-1 Dw Widget与Gtk Widget的比较

Widget Type	窗口模式	窗口大小	XService
Dw Widget	无窗口	在Vieport中展现，无需限制大小	模拟大部分Xservice
GTK Widget	有窗口	有大小限制	直接使用Xservice

说明：由于HTML页面的大小未知，因此无法确定实际的窗口大小，所以说是无窗口，无大小的；XService是由X-Window提供的，由于X-Window组件非常大，因此无法全部放入嵌入式Linux中。

由上面的比较可以看出，DwWidget既具有一部分GtkWidget的功能，又有针对HTML应用的扩展，所以Gtk、GtkWidget、DwWidget、Dw Widget之间的关系

应该如下图所示：

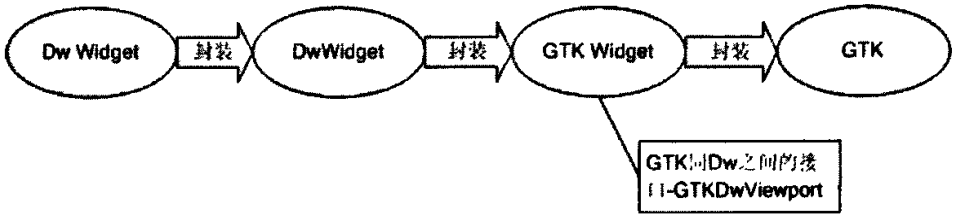

图 3-1 Dw Widget与Gtk Widget等的关系

3.1.1.1 Dillo 基本单元

除了Dw Widget以外，Dillo还有其它的一些基本单元，以下对所有基本单元进行介绍：

1）Dillo Widget —— 主要用于数据分析结果存储以及显示用，是 Dillo 现实的最基本单元， Dillo Widget 是基于 GTK+ 进行编写的，由于 Gtk Toolkit 本身的 Wdget 是针对 X-Window 的桌面现实而定制的，但是 HTML 页面的显示并不需要如此丰富的现实属性，因此 Dillo 对采用了 GtkWidget 中最基本的一部分，并根据需要进行封装，同时也自行编写了一部分的 Widget，这些 Widget 被统称为 Dw Widget。DwWidget 作为页面显示的最基本元素，它提供了一种用于保存基本的数据结构以及进行 Rendering 的机制；

2）Dillo Cache —— 主要用于网络/本地文件的访问缓存。当Dillo接收到了来自网络或者本地的文件（HTML或其它文件）的时候，在读取文件后需要将文件的内容放入Cache中。Cache采用信号(Signal)驱动的I/O引擎，用来对文件进行描述。Cache在整个Dillo程序中所处的位置是介于网络层(Network Layer)和渲染层(Rendering Layer)之间的虚拟层，因此它作为数据的缓存对于网络层（Network Layer）来说，是数据存储的终点；但是对于渲染层（Rendering Layer）来说，它则是页面渲染的数据源。

3）HTML Parser —— 主要是一个流解析器(Stream Parser)。由于 Dillo 是一个浏览器，因此它需要提供将 HTML 页面解析并形成程序数据结构的功能，而 HTML Parser 则提供了这样的一种转换机制。HTML Parser 用于连接 Cache(文件读取/网络连接)和 Widget(数据结构存储/画面显示)，它从 Cache 中读取已经被缓存的网络或本地的 HTML 文件，然后将该文件进行数据结构转换，然后将这些数据放入不同的 Widget 中，交由渲染层（Rendering Layer）进行页面的描画。

4）Image Processing —— 主要负责图像的处理。在HTML页面中对于图像应该作单独的处理，因为图像并不像其它的HTML标签一样属于HTML页面本身，而是有自己单独的数据源，需要浏览器能够<img>标签中的图片来源属性中读取图片的源地址URL，然后对图片进行单独的接收，并针对不同格式的图片进行解码，因此对于图像需要采用单独的处理单元。本处理单元的主要作用就是用于图片的接收、解码、缓存以及显示，它支持的图片格式包括gif/jpg/png格式的图片，

同时也要求本处理单元能够支持网络图片以及本地图片的操作；

5）DPI Framework —— Dillo内外部的程序，用于保证dpi1（DPI 版本1）的正常工作。由于Dillo的业务实现是由基本的业务单元组合来完成的，这些基本的业务单元被称为DPI程序，Dillo具有一套完整管理机制来完成DPI的管理。Dillo通过内部代码向被称为dpid的守护进程发送命令，并由该守护进程根据dillo的需要调用不同的基本业务元素——dpi。对于整个的一套业务调度机制需要进行准确的描述，因此DPI Framework就是整个一套调度机制的统称，其中包含了Dillo内部的一部分dpi调用代码、dpid程序以及基本的dpi程序。

3.1.1.2 Dillo 基本概念

Dillo作为一个浏览器来说，其基本的HTML解析、显示采用的都是类似于Gtk的一套实现方案，只需要懂得Gtk编程就可以轻松完成，但是Dillo中最核心的部分应该是整个的一套业务调度机制，这对于理解Dillo程序来说是至关重要的，因此在这里我们需要对Dillo的模块调用机制进行一个简单的描述。在此之前我们先提出几个基本的概念。

- dpi（Dillo Plugin Interface）：dillo插件系统的统称，是Dillo所有的基本业务单元，以插件（Plugin）的形式进行编写；
- dpi1（Dillo Plugin Interface 1）：dillo插件系统版本1；
- dpi framework：Dillo内外部的程序，用于保证dpi1的正常运行，是整个一套调度机制的统称，其中包含了Dillo内部的一部分dpi调用代码、dpid程序以及基本的dpi程序；
- dpip（Dillo Plugin Interface Protocol）：Dillo插件协议，是Dillo实例与被调用的dpi程序之间的通信协议。Dillo实例与dpi程序之间通过socket传递XML/HTML来完成数据交互。
- dpid（Dillo Plugin Interface Daemon）：Dillo守护进程，主要用于监听Dillo实例对插件的调用请求。Dpid实例在Dillo启动的时候就一同启动，始终监听这来自Dillo实例的请求，然后根据Dillo请求的dpi名称寻找相应的Dpi程序，并与之建立Socket连接，并将该连接转交给Dillo实例。
- server plugin：dpi插件的一种类型，它是在Socket上用于接收连接的插件，它的主要特点是在dpid的同一个时间只能够运行一个server插件进程。
- filter plugin：dpi插件的一种类型，它通过stdio进行读写的程序或脚本，在dpid上可以运行多个filter plugin进程，因此需要保证当filter plugin在写stdout的操作是安全的。
- Service Request Socket(srs)：在dpid上运行的用于监听来自dillo实例的Service请求的dpid Socket，它是dpid与dillo实例通信的基本信道，在dillo实例启动时即开始运行；

- Unix Domain Socket（UDS）：Unix系统的socket，是系统内部进程间进行通信的基本方式。它的操作方式与普通的socket操作没有任何区别，我们可以通过打开多个socket或者是打开stdio所对应的socket来完成dillo实例与dpi插件之间的通信。

对于针对Dillo程序的二次开发来说，最重要部分就是DPI Framework，它是Dillo能够更加丰富自身功能的基本保证。在Dpi Framework中除了dillo内部外，最重要的外部代码就是dpid和dpi插件。以下将针对这两部分做出详细的说明

3.1.1.3 Dpid

Dpid守护进程主要是用来管理dpi连接的程序，它的设计目的用来服务于使用Unix Domain Socket（UDS）的Dillo实例，它能够运行dpi程序，并为Dillo实例和dpi之间的Socket通讯的。在dillo实例启动的时候，dpid守护进程也随之启动，并建立一个针对所有Dillo实例的Service Request Socket(srs)用于监听来自Unix Domain Socket（UDS）的Dillo实例的服务请求，并返回管理该服务的dpi Socket名称，然后启动该dpi插件，将之转交给发出请求的Dillo实例，完成两者之间的通信，它同样也监视着非活动插件的socket，并在有连接请求的时候对它进行激活。

对于dpid来说，它的存在是唯一的，也就是说不论启动多少个dillo实例或者是dpi插件，始终只允许一个dpid守护进程的存在。但是一个dpid守护进程可以服务于多个Dillo实例，也可以是一个Dillo实例下的多个窗口，即允许同一个Dillo实例的2个或两个以上窗口能够同时访问dpi插件；对于dpi插件来说，dpid的存在实际上是对服务的多个实现，即一个dpi程序就是对一个服务的实现，那么多个dpi的启动也就允许了针对不同dillo实例的多个业务实现。同时，dpid的存在也简化了业务更新的过程，也就是说我们至需要通过更新dpi插件就能够实现服务的更新，这大大降低了二次开发的难度。

以下给出了Dpid守护进程的启动过程：

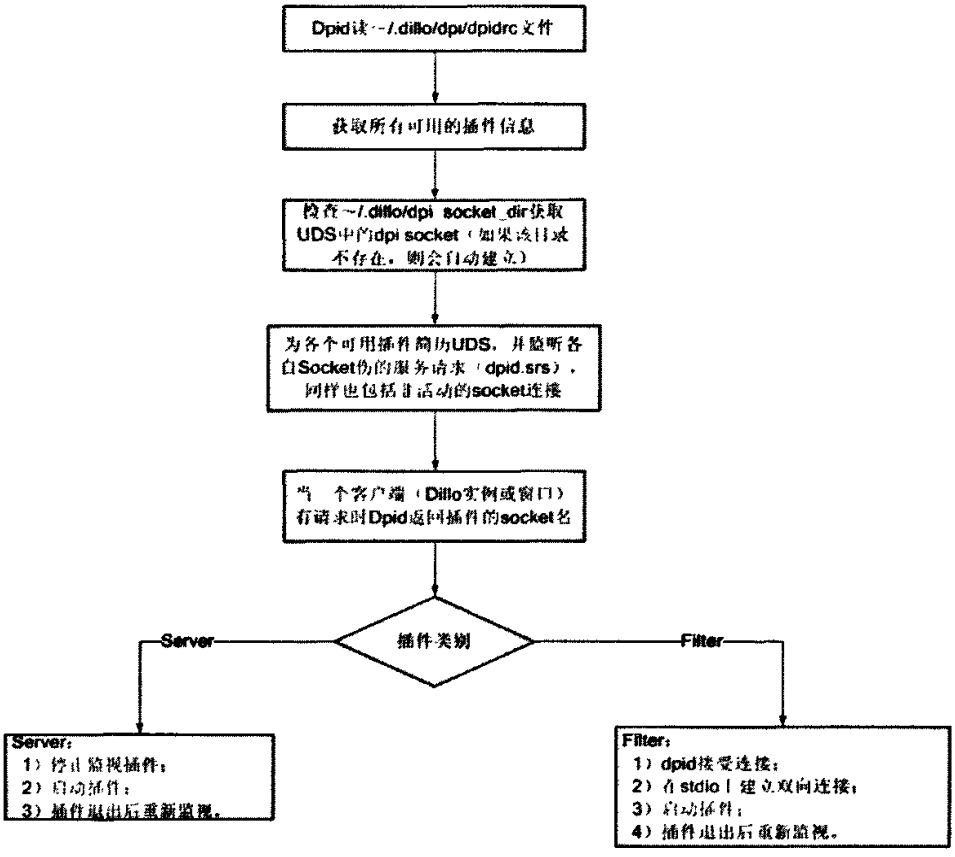

图 3-2 dpid启动过程

现在需要了解一下当用户在一次业务操作中,通过dpid守护进程调用某一dpi插件的过程，只有这样才能够对整个dpi插件的开发有一个总体的了解，以下将以Server类型的dpi插件的调用方式进行说明。

1）首先，当一个dillo实例被启动的时候，那么dpid就会为这个dillo实例开启一个相应的Service Request Socket(srs)来监听来自该dillo实例的请求。与此同时，dpid从一开始运行就通过Dpi Plugin Socket对所有的dpi插件Socket进行监听（如图3-3所示）。

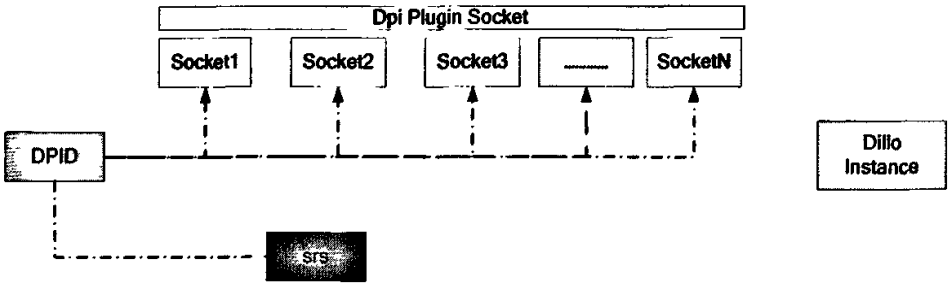

图 3-3 Dillo调用dpi插件过程图（I）

2）当 dpid 守护进程通过 Service Request Socket(srs)监听的某一个 dillo 实例发送了一个服务请求（Service Request），即发送了某个已经存在的 dpi 插件的名称。Dpid 守护进程会通过 Service Request Socket 进行接收，获取该 dpi 插件在 dpid 中注册的 socket 名，并返回给 dillo 实例。（如图 3-4 所示）

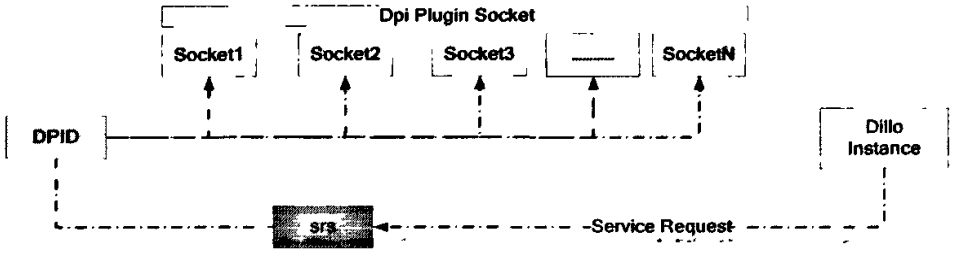

图 3-4 Dillo调用dpi插件过程图（II）

3）Dillo实例根据dpid提供的dpi插件打开的socket名，连接到该socket上。（如图 3-5所示）

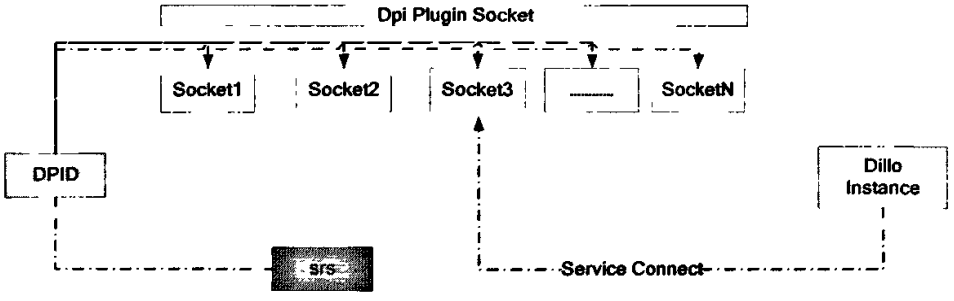

图 3-5 Dillo调用dpi插件过程图（III）

4）当被请求的dpi插件所对应的socket被激活，即获得了来自Dillo实例的Service Data后，dpid开始为该socket调用其所对应的dpi program（即dpi插件编译后的二进制文件）。（如图 3-6所示）

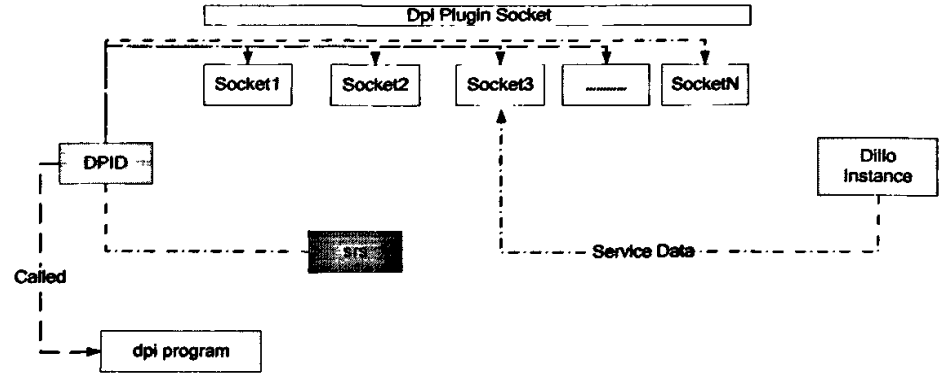

图 3-6 Dillo调用dpi插件过程图（IV）

5）dpid守护进程负责对dpi插件进行启动，启动后的dpi插件是一个独立的进程，如果该dpi插件已经在运行，那么dpid会根据dpi插件类型的不同分别处理，即如果dpi插件是Server类型的插件，则会再复制一个进程；如果dpi插件是Filter类型的，则不进行进程复制或程序启动。之后，dpid将启动后的dpi插件（实际是dpi program）交出发出请求的Dillo实例进行控制。（如图 3-7所示）

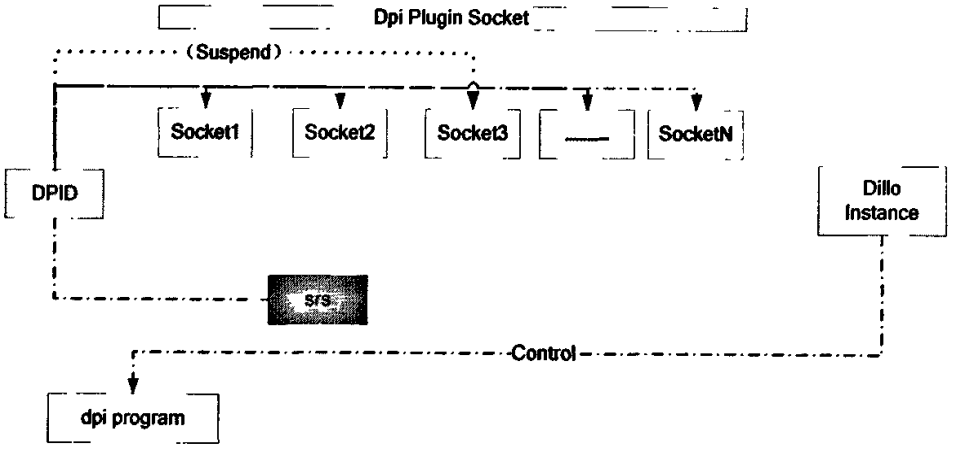

图 3-7 Dillo调用dpi插件过程图（V）

6）Dillo实例在与dpi插件通信的过程中，始终保持了两者之间的Socket连接，即一旦dpi和dpid之间有一个socket连接，所有的连通讯过程便在任务完成前均保持。两者之间的连接只有在dpi program退出的时候才会断开，而dpid守护进程会重新恢复监听socket。（如图 3-8所示）

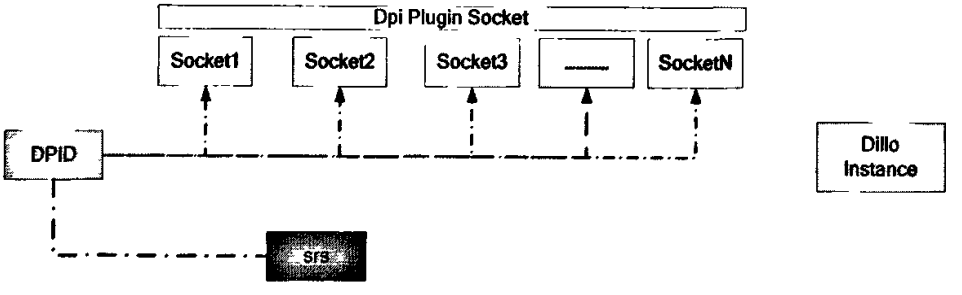

图 3-8 Dillo调用dpi插件过程图（VI）

在Dillo实例调用dpi插件的过程中，之所以需要dpid守护进程作为连接的管理部分而不是直接采用Dillo实例自行管理，是处于以下的考虑：首先，如果多个Dillo实例同时运行的时候，对于不同的插件和不同的Dillo实例之间的管理会变得十分复杂，而当有一个主实例运行的时候，切换到其它实例的插件管理操作方

式也会变得很难；其次，当一个dpi插件被多个Dillo实例调用的时候，数据之间的同步也会产生问题；再次，如果dpid的功能由一个Dillo实例作为主管理程序进行管理，那么一旦当这个主管理Dillo实例关闭，那么其它Dillo实例便会丢失调用DPI的PID；最后，如果一个Dillo程序中集成了插件管理系统，那么不仅增加了内核的大小，同时也增加了插件开发的难度。

3.1.1.4 DPI

Dpi插件作为Dillo程序中的最基本业务单元，在整个Dillo运行过程中起着至关重要的作用，它的存在大大的丰富了Dillo可以实现的功能，同时也为基于Dillo的二次开发提供了一套完整的机制。

设计dpi插件的主要原因在于,由于Dillo的主要目标是完成一个小体积、速度快的客户端浏览器，因此需要将辅助功能性质的代码在"核心"中提取出来。从软件设计的角度来说，在设计中应该尽量减小核心部分代码并简化新功能的开发，以使功能开发者在不必了解Dillo整个结构的情况下就可以进行新功能的开发；从技术的角度来讲，Dillo插件并不是属于Dillo的一部分而是通过一些信道与Dillo程序进行通信的协作进程，因此对于软件的升级来说能够更加轻易的完成。

Dpi插件与dillo之间的通信，目前的实现是利用Unix Domain Socket（UDS）来实现的，其原理如图 3-9所示：

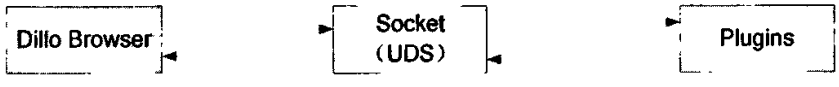

图 3-9 Dillo与dpi插件的关系

采用这个设计的主要目的就是为了能够让浏览器和插件之间建立一个双向通道，并且在这个通道中形成以某种协议（dpi protocol）进行HTML标记的结构。例如以下是一个从插件传往浏览器的消息：<dpi cmd='send_status_message' msg='Hello browser!'>。这个dpip（dpi procotol）的目的就是在Dillo实例中显示一个"Hello browser!"的字符串。这种模式的采用，不再需要在Dillo实例和dpi插件中通过其它复杂的方法进行通信，而仅仅只需要程序的开发者在了解已有dpip的情况下，直接进行dpi插件的开发即可。当然，如果情况需要的话，程序开发者也可以自行定义或者是扩展dpip的内容以达到程序的要求。

一个dpi的插件是有Dillo实例通过dpid的帮助被调用的，以下给出了一个dpi插件被调用的过程：

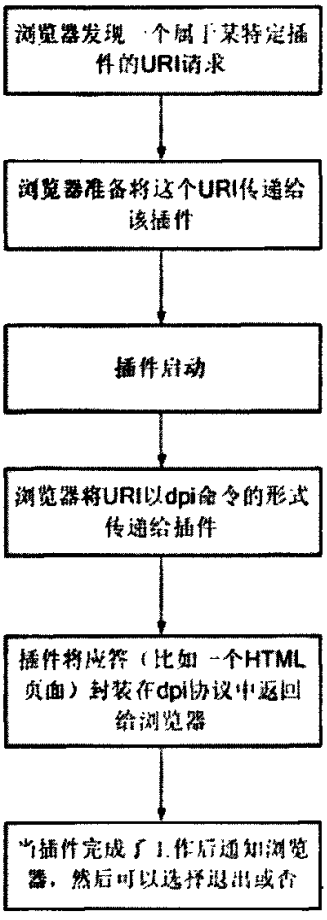

图 3-10 dpi插件被调用过程

从上图中我们肯定清楚的看到dpi插件作为dillo实例业务流程中的一个基本业务单元被调用，在进行了一定的处理后再将结果返回给 Dillo 实例。这个处理结果可以是某些数据，也可以是一个特定的 HTML 页面，甚至是某些特定的操作。Dpi 插件的存在大大降低了 Dillo 程序本身的大小，同时也加强了业务实现的扩展性和程序升级的可能性。目前 Dillo 程序本身已经开发了一部分的 dpi 插件，他们分别是：

1）Downloads —— 这个 dpi 插件主要负责的是文件的下载。在这个 dpi 插件被调用的时候，当前 Dillo 实例的的 Cache 会为将被下载的文件分配内存，并将这个内存移交给一个 Downloads 插件进行控制。在 Downloads 插件中，会根据不同的情况选择采用的代理方式（根据下载采用的协议是 FTP 或 HTTP 而决定是采用 wget 还是 curl），并且将反馈（HTML）返回给 Dillo 实例内部的下载接口。这种做法的好处是，当 Dillo 实例非正常退出的时候下载并不终止。

2）FTP —— 这个 dpi 插件主要负责的是 FTP 下载。它支持浏览 FTP 目录，并将 FTP 目录内容形成 HTML 文件返回给 Dillo 实例显示。FTP 插件下载是采用的方式由 Downloads 插件决定。在下载过程中的所有处理信息，包括用户登录等信息，都能够在状态栏中显示。

3）Bookmarks —— 这个 dpi 插件实现了书签的功能。它维护着自己的数据

库（一个文件），用于保存用户已经进行标记的页面。当用户需要对已经标记的页面进行浏览的时候，该插件能够通过与浏览器进行相互通讯（包括URI，标题，添加书签等等）为Dillo建立一个含有书签内容的HTML页面。同时它支持类似于"清空"的操作，使得书签页不在浏览器中显示。

针对这些已有的dpi插件，dillo提供了一些简单的dpip命令。基本的dpip命令包括：

1）start_send_page：这个命令的作用是将HTML块封装为固定的长度，然后让HTML页面被发送后能够与Dillo实例进行通讯。同时它还允许在命令中追加一些其它的信息，比如说，Bookmark插件在发送完主页面后能够发送一个状态消息以追加一些针对已有操作的额外的信息（例如，"删除三个 bookmark"之类的消息……"）。这个命令是最后被发送的，之所以不在最初就发送这个命令是为了能够尽快的访问一个工作Frame以完成所需要的任务。目前，dpi Framework使用start_send_page来判断所接收到的数据，直到EOF为止，都是HTML页面的一部分。

2）send_data：这个命令是最常用的，用于数据交换的命令。可以用于发送/设置实例的参数。

3）chat：这个命令是用来测试dillo和dpi之间的双向数据交换；

3.1.2 基于Dillo的结构设计的实现

我们在了解了这么多关于Dillo实现的方法后，发现Dillo整个一套实现机制与我们的设计思想非常贴近，因此在结合了Dillo与已经设计好的程序结构，提出了以下的总体结构：

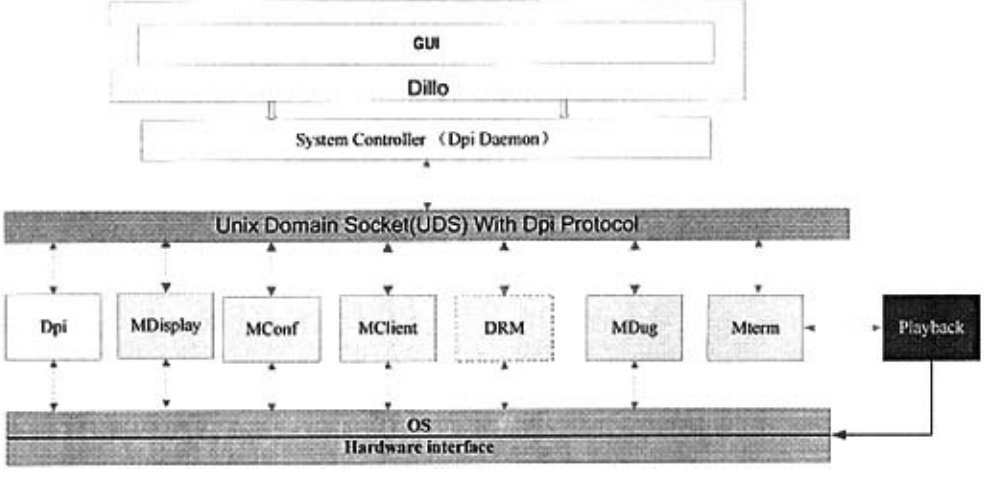

图 3-11 基于DilloIPTV机顶盒软件结构设计

1）GUI（由Dillo实现）是用于HTML页面描画。GUI部分采用继承自GtkWidget并进行了一部分的重写的Dw Widget进行页面元素描述，以适应HTML显示的要求。它针对不同的页面组件，有不同的Dw Widget进行对应，并具有有完整的页

面描画、调整机制，同时对于来自用户的操作事件也能够及时进行相应。

2）SystemController(Dpi Daemon)保存了IPTV软件内部业务传递用数据结构体。它是用于管理dpi插件（业务模块）连接的程序，设计用来服务于使用Unix Domain Socket（UDS）的Dillo实例的，它能够运行dpi程序，并为Dillo实例和dpi之间的Socket通讯的。其实现方式描述如下：dpid监听来自UDS的Dillo实例的服务请求，并返回管理该服务的dpi Socket名称；它同样监视着非活动插件的socket，并在有连接请求的时候对它进行激活。

3）Unix Domain Socket(UDS) With Dpi Protocol用于实现系统内部的IternalBus，它采用了Socket用于管理传递的数据，并采用XML的格式对数据进行保存。这个设计的主要目的就是为了能够让浏览器和插件之间建立一个双向通道，并且在这个通道中形成以某种协议（dpi protocol）进行HTML Tag重标记的结构。例如以下是一个从插件传往浏览器的消息：<dpi cmd='send_status_message' msg='Hello browser!'>

4）Dpi Plugin(dpi 插件)用于实现基本的业务模块。这个设计的主要目的就是结合了Dillo本身的插件系统，将IPTV机顶盒的业务分割为多个小模块，通过Dillo作为中转将所有的数据进行传递。这种做法增强了程序的可移植性和可扩展性，一旦当程序需要进行部分更改的时候，只需要对相应dpi进行替换即可，同时也减小了Dillo内核的大小。

3.2 XMLReader 静态库的实现

3.2.1 XML 介绍

XML(Extensible Markup Language)是一种可扩展的无置标语言。(置标——为了处理的目的，在数据中加入附加信息，这种附加信息称为置标。用置标方法描述的形式语言即为置标语言）在现实生活中，我们经常在语句下加下划线，对重要信息加星号，标记为不同颜色，这些都是置标的应用。XML 来源于 SGML（标准通用置标语言），但 XML 是它的简化，但保留了其精华，W3C 已于 1998 年 2 月批准了 XML 的 1.0 版本，目前 XML 已经成为一个国际标准。

XML 作为一种元标记语言，所谓"元标记"就是开发者可以根据自己的需要定义自己的标记，比如开发者可以定义如下标记 ，任何满足 xml 命名规则的名称都可以标记，这就为不同的应用程序打开了的大门。同时，xml 是一种语义/结构化语言。它描述了文档的结构和语义。xml 的文档是有明确语义并且是结构化的。 XML 是一种通用的数据格式从低级的角度看，xml 是一种简单的数据格式，是纯 100%的 ASCII 文本，而 ASCII 的抗破坏能力是很强的。

目前，XML 在应用程序中可以有多种用途，比如说，XML 可利用于数据交换 主要是因为 XML 表示的信息独立于平台的，这里的平台即可以理解为不同的应用程序也可以理解为不同的操作系统。由于它描述的是一种规范，因此利用它使得不同程序或者是不同类型的文件之间交换信息称为可能。XML 文档有

DTD 和 XML 文本组成，所谓 DTD（Document Type Definition）,简单的说就是一组标记符的语法规则.，表明 XML 文本是怎么样组织的，比如 DTD 可以表示一个必须有一个子标记，可以有或者没有子标记 等等。当然一个简单的 XML 文本可以没有 DTD。

3.2.2 XMLReader 的实现

在本系统中，XML 被用来保存配置信息以及传递 dillo 实例与 dpi 插件间的数据，因此我们有必要对已经编写的 XML 文件进行分析并提取其中的数据，而 XMLReader 就承担了这样的一个功能。

我们先来简单分析一下 XMLReader 实现的原理。首先，无论是用来保存配置信息的 XML 文件还是用来传递数据的 XML 格式的字符串，每一个标记的名称以及属性都是由程序编写人员自行定义的，也就是说 XMLReader 要能够知道 XML 中所允许的标记名称以及属性名称；其次，XMLReader 需要能够根据 XML 结构提取每个标记中的内容，并形成一定的数据结构。因此，我们通过两个枚举类型分别列举了 XML 中所出现的所有标签名称以及属性。

XMLReader 的工作流程如图 3-12 所示：

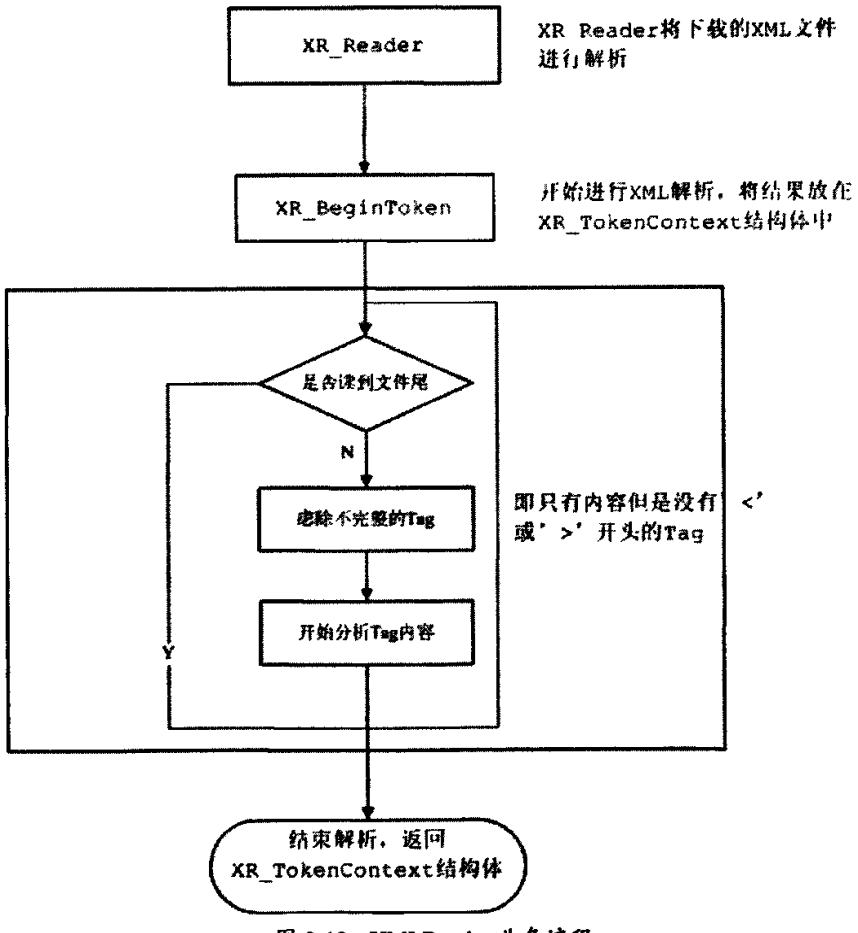

图 3-12 XMLReader 业务流程

XMLReader 根据程序提供的 XML 文件路径或含有 XML 内容的字符串进行解析，在整个流程中，XMLReader 循环读取 XML 内容，并根据"<>"来判断每个标签的开始，根据"</>"判断每个标签的结尾，每当读取一个到一个标签后，就针对该标签中的属性进行比较。

当 XMLReader 在读取到一个标签的开始后，需要针对该标签的内容进行分析，具体的分析过程如下：

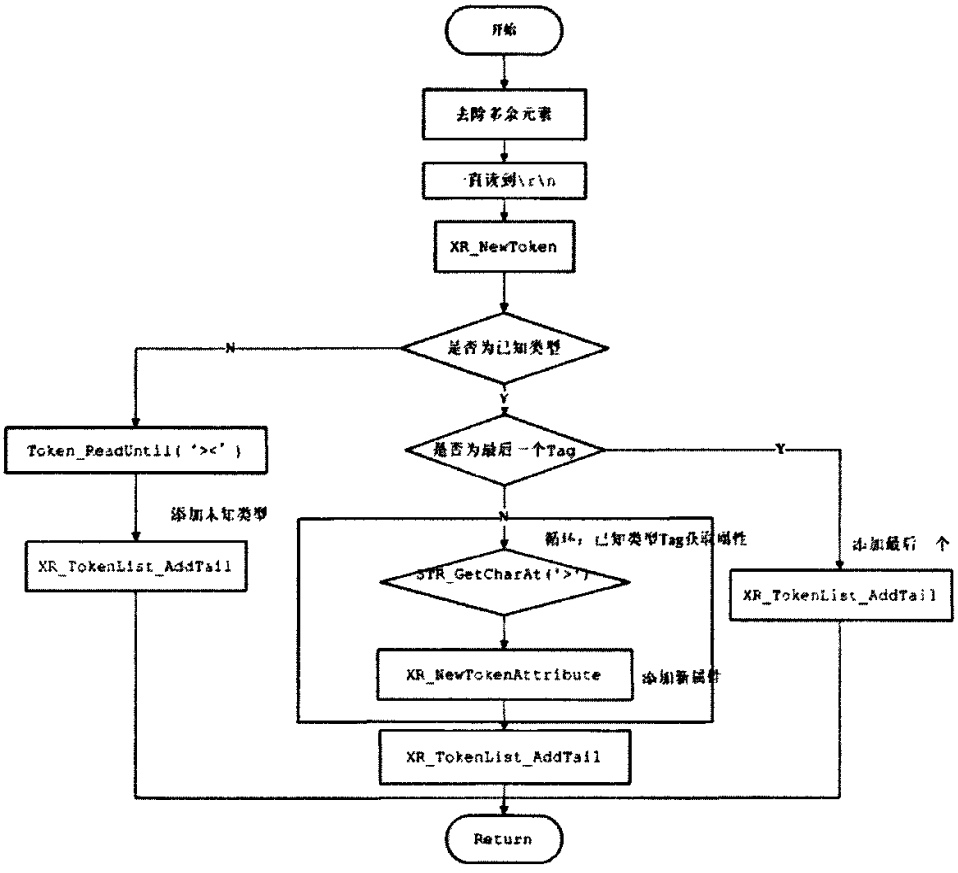

图 3-13 XMLReader 解析单个标签的流程

首先，XMLReader 会在去除了 XML 标签中不必要的一些元素，诸如 tab 等，然后会新建一个 Token 结构体，以提供一个容器以存储将要读到的 XML 标签内容。

其次，XMLReader 会将读到的标签中紧跟"<"字符之后，第一个空格之前的字符串提取，并判断该字符串内容是否为已知的标签类型，如果该类型并非为已知标签类型，则会一直读到"><"，并将该标签内容解析后直接添加到标签树结构体的尾部。（因为本标签结尾和下一个标签开始必然是以"><"进行分割的）

在判定读取到的标签是已知标签后，XMLReader 会判断该标签是否为最后一个标签，如果该标签为最后一个标签，那么直接将该标签进行解析并放入标签树的尾部，结束 XML 解析。

如果该标签不是最后一个 XML 标签那么将会根据标签中的内容进行分析，

在分析过程中判断该标签是否已经结束，在标签属性解析的过程中，XMLReader会将新解析到的标签属性添加到Token结构体中，形成属于该标签的属性树。

最后，当这个已知标签全部解析结束后，新建的Token结构体中将存有该段标签的所有内容，包括标签名，标签中的属性树以及该标签是否为结尾标签，即包含"</>"的标签。并将新解析到的Token结构体放入标签树中，进而开始解析下一个标签。

XMLReader作为一个静态库，为所有的模块提供了一个将已有数据转化为程序能够理解的结构体。它是整个程序能够正常运行的基础，由于标准的XML的标签内容以及属性是有DTD来提供的，所以如果需要完成一个能够解析所有用户自定义的XML文件，XMLReader需要能够解析DTD，并且将该DTD在解析过程中与XML文件相结合，保证了程序的可扩展性。但是在这里XMLReader仅实现了对于有限的用户自定义的XML标签名称以及属性的解析，因此如果需要更加丰富本系统的可扩展性和可移植性，需要对XMLReader进行进一步的改进，允许XMLReader能够根据DTD的内容进行XML的解析。

3.3 Dillo实现用户遥控事件

3.3.1 Dillo的启动

在dillo启动时，除原有的一部分启动相关代码，还需要增加关于各业务模块的启动代码，其中包括MDisplay、MConf、MClient以及MTerm模块的启动。

启动初期，首先需要读入整个IPTV机顶盒软件系统结构配置文件，需要读取菜单配置文件，以获得菜单各级目录之间的关系以及各自的模版文件信息。每个节目项由一个统一的全局结构体进行保存。在该结构体中，除了保存各级菜单信息外还保存了dillo的当前运行状态。菜单配置文件的具体内容请参考3.5节中关于menu_relation.xml的说明。

在获得了页面菜单结构后，需要针对各个模块进行启动，首先被启动的是MDisplay模块用于显示欢迎页面，通过设置dilloDoc，用于在Dillo中保存dpi插件返回的信息，并想MDisplay模块发送dpi命令，以模块能够理解的内容启动模块。

3.3.2 Dillo实现用户遥控事件

初始化完成后，Dillo会调用Gtk的gtk_main函数，进入由Gtk提供的系统循环。由于gtk是基于事件驱动的，所以当用户有按键事件发生时，dillo通过在gtk_main函数中注册的键盘事件处理函数进行捕获，并作处理。

此处针对不同的按键，如上、下、确认等按键事件，调用按键事件处理函数进行相应的事件内容处理，针对每个按键，Dillo会根据全局的页面菜单结构体中对当前所在页面的描述，分别进行不同的处理。在确认了当前页面以及按键内容后，会形成相应的dpi命令并且修改在全局结构体中软件的当前各项运行状态。最后可以通过a_TvMenu_Process（ int key ）函数将该命令发送给相对应的dpi插件，而各插件能够根据相应的命令，完成自己的功能。

1）MDisplay部分：当其接受到dpi命令后，根据该命令完成显示页面的html

文件，然后送回给 dillo 做显示。

2）MTerm 部分：接受到 dpi 命令，做出相应的操作。如开启新的窗口，播放视频文件。

3.4 MDisplay 模块的实现

MDisplay模块作为Dillo的插件而存在，其插件类型为Filter类型。Filter类型的插件一个是交由Dpid控制的进程，通过UDS与STDIO的映射完成进程间的通讯。MDisplay模块在整个Dillo中的作用是在用户进行遥控器操作时进行页面跳转管理页面现实的一个模块，用户通过遥控器的上下左右以及播放、Menu、Back等按键进行页面跳转控制，而MDisplay模块则结合用户的遥控事件以及当前所在页面和即将跳转的页面判断如何进行页面显示。页面显示内容则由MDisplay模块结合页面配置XML以及EPG菜单内容进行组合显示。

MDisplay模块由3个部分组成：（1）页面关系配置文件，用于表明页面跳转关系，及其它一些页面内容、系统相关的配置文件；（2）HTML模版，提供三种类型的模版作为基础，其详细内容根据（1）中所说的配置文件决定详细内容路径决定；（3）显示HTML文件，该类型的HTML文件包括三个类型的内容：Selected，Unselected，nbsp，分别表示选定的项目、为选定的项目以及空项目用于显示不同类型的节目内容。

3.4.1 MDisplay 模块的配置文件

MDisplay模块的显示是以一些的配置文件为基础的，这些配置文件总共分为三个部分：

- 用于表明系统页面关系的配置文件——menu_relation.xml；
- 用于表明各页面内容的配置文件——menu.xml等一系列文件；
- 用于表示系统现实相关的配置文件——display_param.xml。

以下将对各个文件进行详细说明

1）menu_relation.xml文件

menu_relation.xml文件主要用于为Dillo提供页面跳转关系，使得Dillo能够通过读取该文件确定用户页面之间的跳转关系，并建立相应数据结构，确定向Mdisplay模块发送什么样的命令以决定页面显示。这样的做法，使得应用人员能够自由组合用户界面之间关系，从而大大提高了页面组合的灵活性。

menu_relation.xml文件最多支持4级深度的文件跳转，即从起始页面开始最多可以访问到同一棵树枝的第四级深度，各个页面之间的关系，通过XML文件中的标签关系来实现。以下以一个最简单的xml，对其内容结构进行说明。

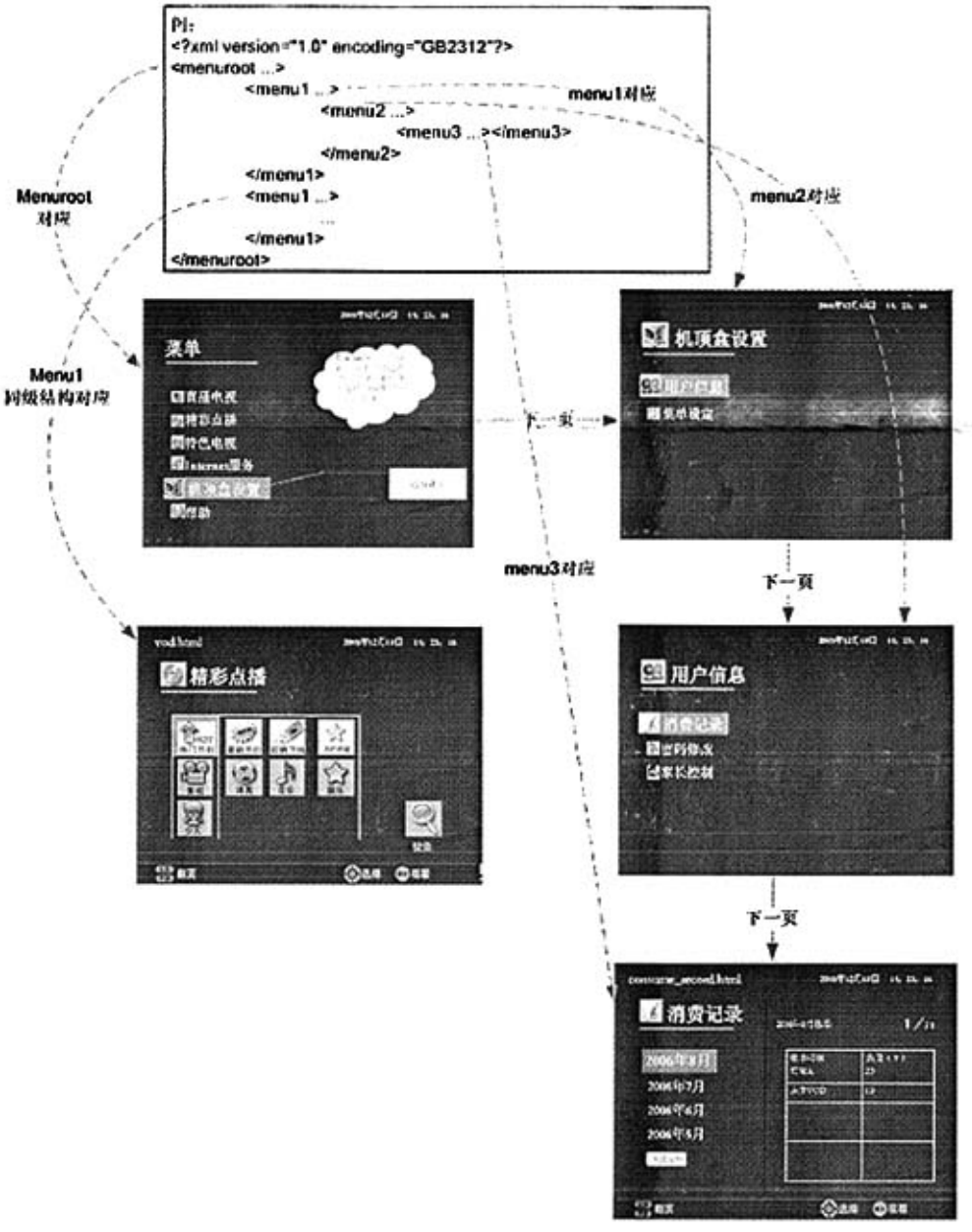

图 3-14 menu_relation.xml 文件说明

其中每一个标签中又包含了描述该页面基本信息的一些属性。

例：

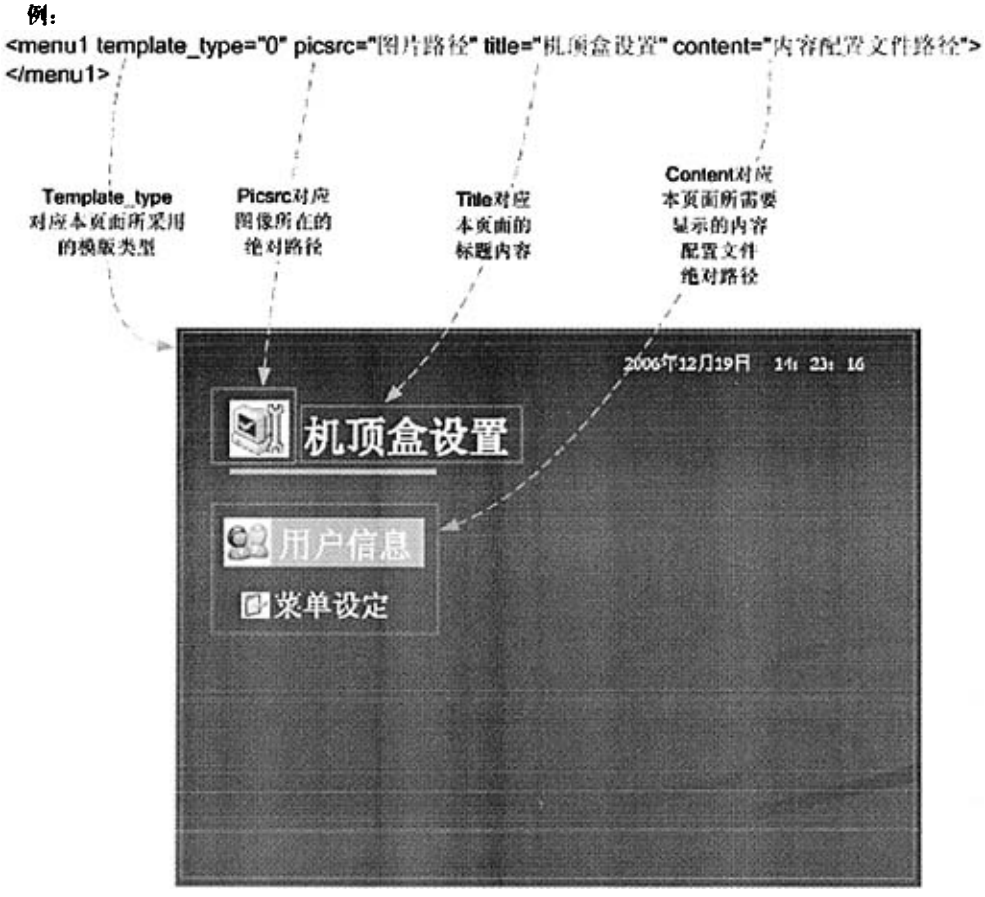

图 3-15 menu_relation.xml 单个标签说明

2）menu.xml文件

对于如menu.xml类型的配置文件说明了当前页面的内容，该文件的地址是保存在menu_relation.xml文件中，并由由dpi命令传输过来的。文件中每一个标签均代表了页面中的一个项目，具体内容如图3-5所示：

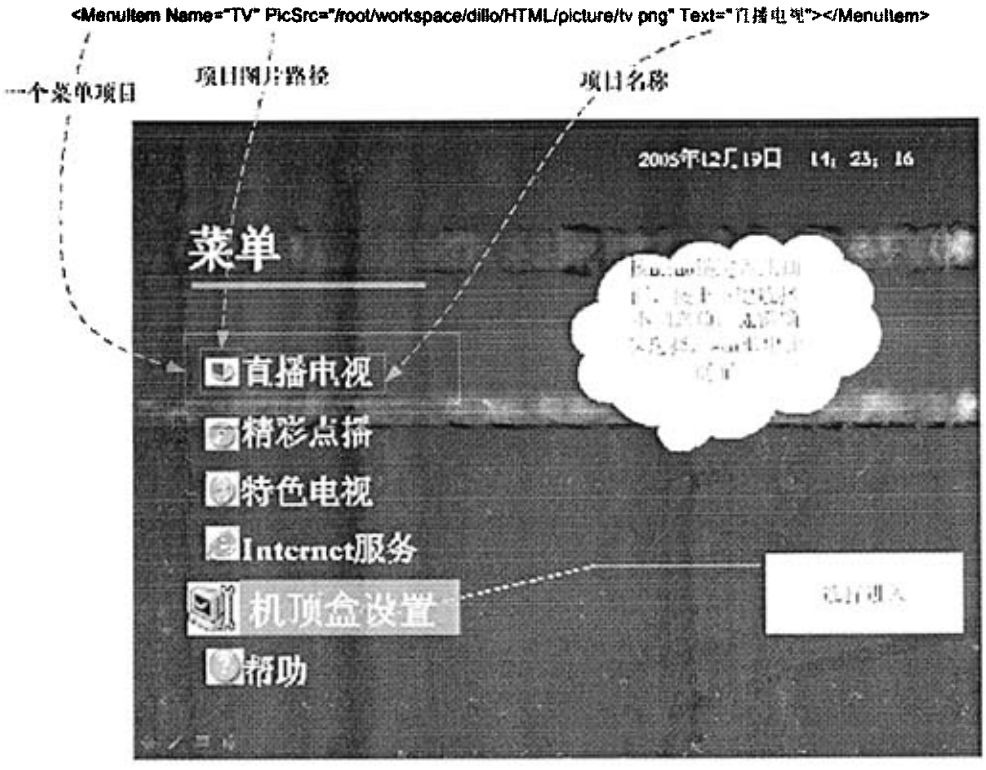

图 3-16　menu.xml 文件说明

每个页面的内容配置文件与Dillo获取的页面关系配置文件相结合就能够获取相关页面的所有内容，作为每个页面显示的基础。

3）display_param.xml文件

这个配置文件中包含了一些与系统的现实相关的设置信息，具体内容如下：

<DisplayMode Code="1"></DisplayMode>

用于描述现实模式是选择宽屏(16:9)还是标准屏幕(4:3)，相应的配置页面如下：

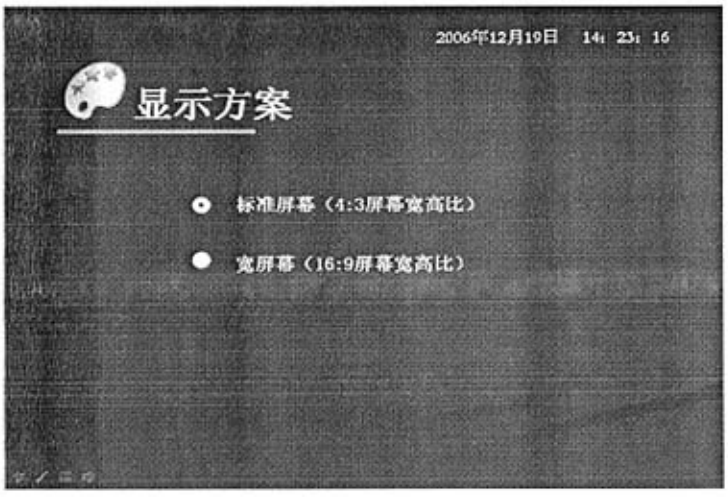

图 3-17 display_param.xml 文件显示——显示方案

<SystemLanguage Code="0"></SystemLanguage>用于描述系统语言，0表示中文，1表示英文。

图 3-18 display_param.xml 文件显示——系统语言

<SysStyle Code="0" Color="#000099"></SysStyle>

用于表示系统风格，code属性表明选择的风格代号，color表明了页面的背景颜色

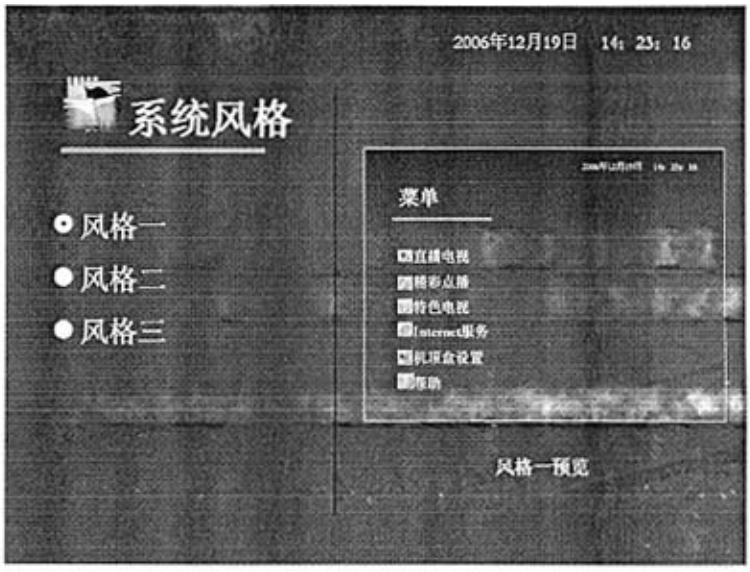

图 3-19 display_param.xml 文件显示——系统风格

3.4.2 MDisplay 模块显示处理流程

Mdisplay模块通过Dillo程序将三部分结合在一起，动态的显示所需要的HTML页面。整个显示过程如图3-20所示：

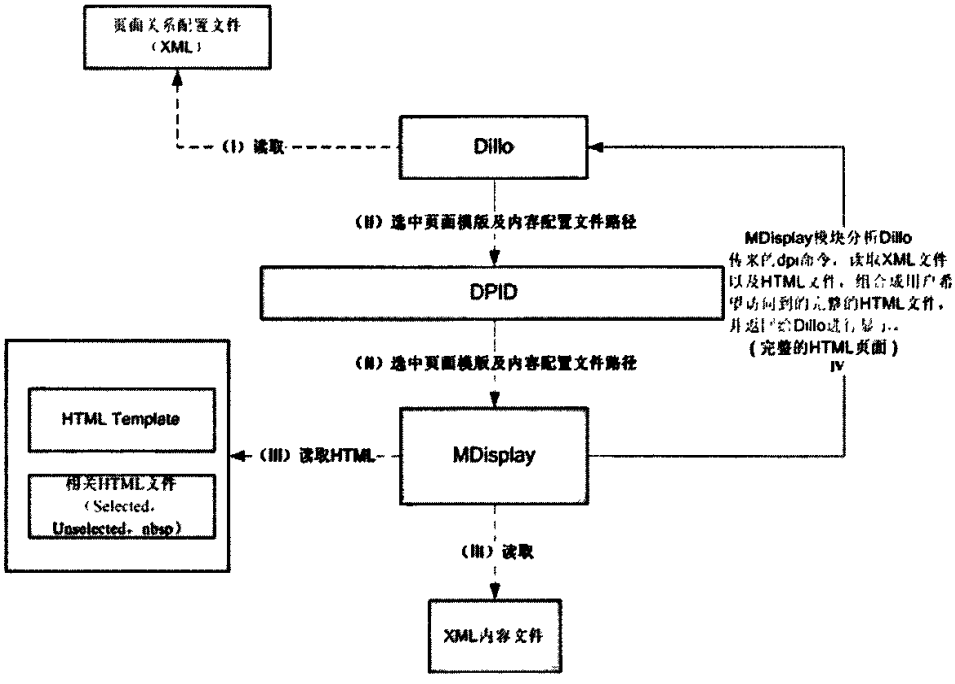

图 3-20 Dillo 页面显示流程

首先，Dillo在启动的时候从指定位置读取一个采用XML编写的页面配置文件，该文件中包含了整个系统的页面结构。Dillo在读取了该文件后能够建立一个关于整个系统页面关系的结构体，所有的用户页面跳转操作结果都是基于这个数据结构来实现的。

当用户发出一个页面跳转的请求后，Dillo会接收到该操作，并根据当前页面所在位置判断在页面关系数据结构中用户操作所指向的页面内容，然后根据该记录组成一个XML格式的指令字符串，并发送个Dpid请求调用MDisplay模块。

Dpid将MDisplay模块启动后，MDisplay模块会接收到Dillo发送的dpip命令行，在解析了该字符串并提取出相应内容后，提取命令中关于页面显示需要的相关命令组合成用户请求的HTML页面，并将这个结果返回给Dillo进行显示。

MDisplay模块在接收到Dillo命令后的处理流程如下图所示：

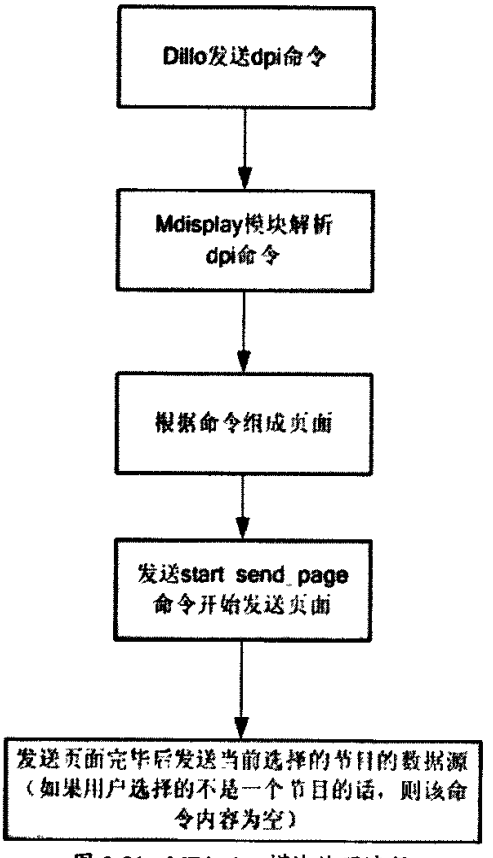

图 3-21 MDisplay 模块处理流程

Mdisplay模块具体页面的显示是模块根据Dillo发送的命令来进行页面显示后的结果。Dillo根据从menu_relation.xml中获得的页面关系结合用户操作，请求Mdisplay模块拼接完整的HTML页面进行显示。一个典型的dpi命令如下所示：

<dpi template_type='%d' pic='%s' title='%s' seq='%d' cmd='mdisplay_key_down'
content='%s'>"

其中，dpi是这个命令的标签名，所有的dpi命令都需要以此开始；Template_type属性说明了当前请求的页面所需要使用的模版类型，所有的模版一共分为6种（分别为（1）不带页数的Menu类型；（2）带页数的Menu类型；（3）节目Detail类型；（4）系统显示设置页面；（5）系统语言设置页面；（6）系统风格设置页面）；Pic属性说明了当前页面的标题图片所在的绝对路径；Title属性表明当前页面的标题内容；Seq属性说明用户当前选择的项目序列号；Cmd：用户遥控器事件（上下左右、menu、pagedown、pageup等），用于判定是否需要进行翻页等操作；Content属性说明了页面内容配置文件所在绝对路径，如果当前页面需要显示的是来自节目服务器的EPG菜单，则本项内容为空，而向EPG服务器请求菜单内容后进行显示。

当用户选中某一个页面并进行遥控器操作后，Mdisplay模块根据dpi命令进行解析，Mdisplay模块不需要知道当前的页面到底是什么内容，只需要根据命令进行合理的解析即可。根据Content的路径读取相应的配置内容，然后结合template_type配合相应的HTML内容进行显示。具体显示方式如图3-22所示：

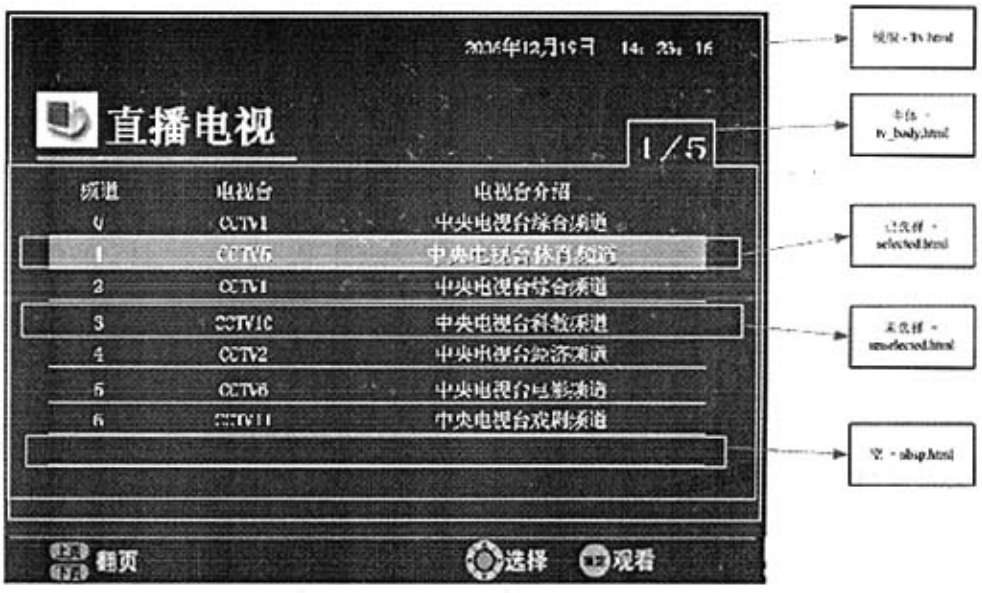

图 3-22　MDisplay 单个页面显示描述

由以上例子可以看出，一个用户访问界面最多由6个部分组成：

1）Template.html：组成显示页面的最基本框架；

2）content.xml或EPG菜单（XML）：提供页面显示所需要的所有内容；

3）body.html：基于template.html的页面主体内容通常为一个table；

4）selected.html:表明用户当前选择的节目内容；

5）unselected.html：表明未杯选中的节目内容；

6）nbsp.html：表明空的节目内容。

其中3中所填充的内容是固定的，但是4 — 6的组合则是根据解析menu.xml

等服务配置文件或EPG菜单而生成的，根据特定的规则以及dpi命令中用户命令进行页面显示的不同组合，并将EPG菜单或menu.xml等服务配置文件的实际意义在页面上进行显示。

当Mdisplay模块解析了dpi命令后，并根据命令内容组成了所需要显示的HTML页面，便将该页面内容发往Dillo，由dillo进行显示。

3.5 MConf 模块的实现

Mconf模块的主要作用是读取、保存一些与系统相关的配置内容。在系统启动时Mconf模块会被启动，并读取指定位置的配置文件，并根据配置内容对系统进行一定的设置。当Dillo或其它模块需要一些由Mconf模块保存的配置信息时，可以通过Dillo程序向Mconf模块发出请求，Mconf模块在解析dpi命令后，根据不同情况将请求的信息返回给Dillo，由Dillo接收或直接转发给请求模块。同时，由于IPTV机顶盒还运行着一个Http服务器，为用户提供远程配置机顶盒的服务。当用户针对机顶盒进行配置后，如果需要配置生效，则需要将更改的配置信息写入XML配置文件中，同时通知Dillo实例，由Dillo实例转发给Mconf模块进行配置信息的重新读取。因此，在本节中也将对Http服务器进行一定的描述。

3.5.1 MConf 模块的配置文件

由于IPTV机顶盒功能的需要，配置文件部分被分为网络部分以及服务部分，其中网络部分主要集中了一些关于网络连接和网络设置相关的配置内容，而服务部分主要集中了与IPTV服务相关的配置内容。

网络相关配置文件中包含了：1）用于检测配置文件版本的<Network>标签；2）用于设置本机静态IP地址的<StaticIP>标签；3）用于设置Mterm模块播放节目时采用的网络通信协议信息的<MTermPro>标签；4）用于描述升级服务器相关信息的<UpdateInfo>标签；5）用于描述DRM服务器的<DRM>标签。以下给出了网络配置XML文件的具体内容，其中包含了对各个标签以及标签属性的说明。

```
<!-- 网络配置文件版本号 -->
<Network version="1.0"></Network>
<!-- 静态IP配置信息 ip=静态IP地址；gateway=网关；dns1,dns2=DNS服务器；是否启用静态IP：enable=0(不使用)；enable=1(使用) -->
<StaticIP ip="192.168.1.44" gateway="192.168.1.1" dns1="202.56.34.1" dns2="202.56.34.1" enable="0"></StaticIP>
<!-- MTerm播放时采用的网络通信协议 2=RTP 3=IGMP -->
<MTermPro protocol="2"></MTermPro>
```

<!-- 升级服务器协议地址 0=HTTP, 1=FTP；server=升级服务器地址；port=端口号 -->

<UpdateInfo protocol="1" server="192.168.1.1" port="3128"></UpdateInfo>

<!-- DRM server=DRM服务器地址 -->

<DRM server="192.168.1.1"></DRM>

服务相关配置文件中包含了：1）含有配置服务器相关信息<ConfServer>标签；2）服务配置文件版本号相关的<Service>标签；3）描述鉴权服务的服务器信息的<AuthServer>标签；4）用于描述业务提供商信息的<ProviderList>标签组；5）用于描述用户登录信息的<AccessInfo>标签。以下是对网络配置XML文件的具体内容，在改写该XML文件时，需要注意的是根据业务要求，首先需要根据配置文件的版本进行配置文件版本检查，<ConfServer>标签一定要放在第一个标签位置，并且在读取网络配置文件之前需要先读取服务配置文件，以获取配置服务器相关信息以便能够获取到配置服务器的信息进行配置文件版本号的检查。

<!-- 配置服务器 address=配置服务器地址 port=配置服务器端口 -->

<ConfServer server="192.168.1.1" port="8080" protocol="0"></ConfServer>

<!-- 服务配置文件版本号 version=版本号 -->

<Service version="1.0"></Service>

<!-- 鉴权服务器相关信息 server=鉴权服务器地址；port=鉴权服务器端口号；protocol=鉴权时所采用的协议-->

<AuthServer server="192.168.1.1" port="8080" protocol="0">

<!-- 服务提供商列表 name=供应商名 code=供应商编号 enable=是否加载该供应商配置 server=EPG访问路径 PORT=端口-->

<ProviderList>

<Provider name="联通" code="0" enable="1" server="www.domain.com" port="8080" protocol="0"></Provider>

<Provider name="电信" code="1" enable="0" server="www.domain.com" port="8080" protocol="0"></Provider>

<Provider name="华为" code="2" enable="0" server="www.domain.com" port="8080" protocol="0"></Provider>

</ProviderList>

<!-- 用户业务接入信息 name=用户名；pwd=密码-->

<AccessInfo name="test" pwd="test"></AccessInfo>

3.5.2 MConf 模块的命令处理流程

MConf模块在整个IPTV机顶盒软件系统的运行过程当中应该是首先被调用的，因为在系统启动时需要对所有的配置信息进行检查，如IPTV机顶盒的鉴权、配置文件版本检查、用户登录操作等。下图描述的就是在系统启动是MConf模块被调用的过程。

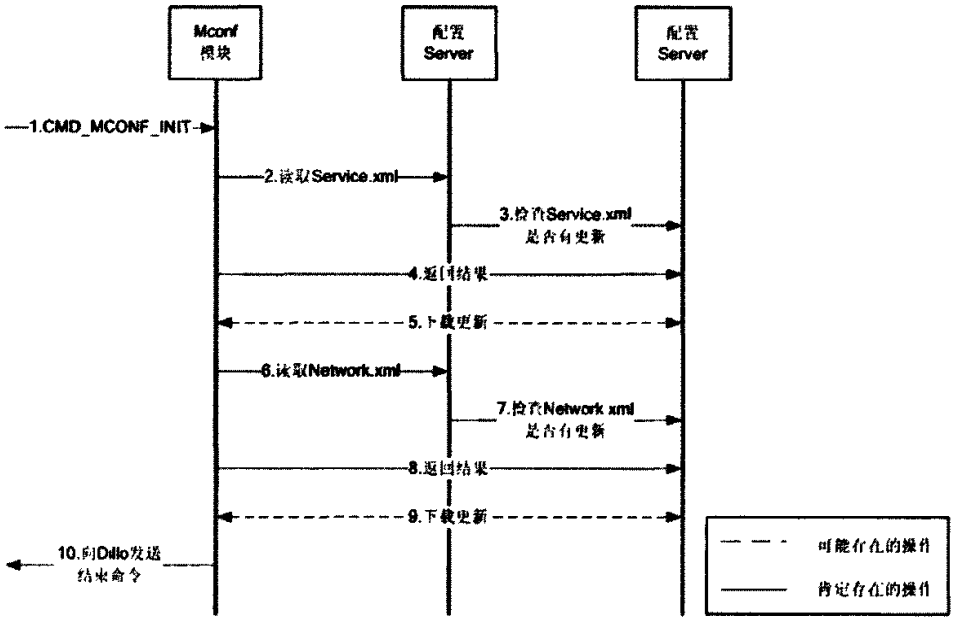

图 3-23 MConf 模块处理时序图

在上图中我们可以看到，当系统启动的时候，Dillo实例会发送cmd_mconf_init命令给MConf模块，当MConf模块接收到该命令后，首先开始解析Service.xml，然后根据服务相关XML中提供的配置服务器信息以及服务配置文件的版本信息进行Service.xml配置文件版本呢的检查，如果检查到该文件有更新，则首先进行配置文件的更新。其次，再次读入Network.xml配置文件，检查是否有更新，如果同样也存在新的网络相关配置文件，则进行网络配置文件的更新。

由于MConf模块保存的是所有的配置信息，当其它的dpi插件需要相关信息的时候，需要通知Dillo向MConf模块获取相关配置内容，因此Dillo与MConf模块之间的交互的dpi命令就显得十分重要，以下给出了在MConf与Dillo之间传递的dpi命令的交互过程以及内容。

1）Mclient模块请求升级服务器地址

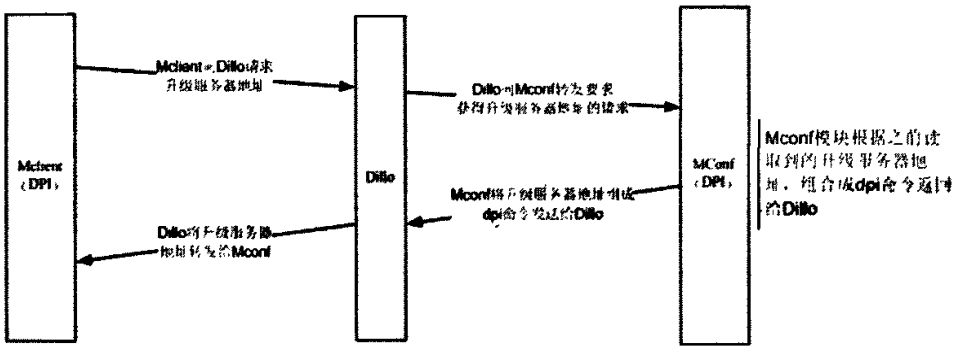

图 3-24　MConf模块请求升级服务器地址操作流程

MConf响应获取升级服务器地址的回复：<dpi cmd='mconf_update_server' server='%s' port='%ld' protocol='%d'>

说明：

- cmd = mconf_update_server 用于表明本次要求获取升级服务器内容；
- server = 升级服务器地址；
- port = 升级服务器端口号；
- protocol = 升级采用的协议

2） Mclient模块请求鉴权服务器地址

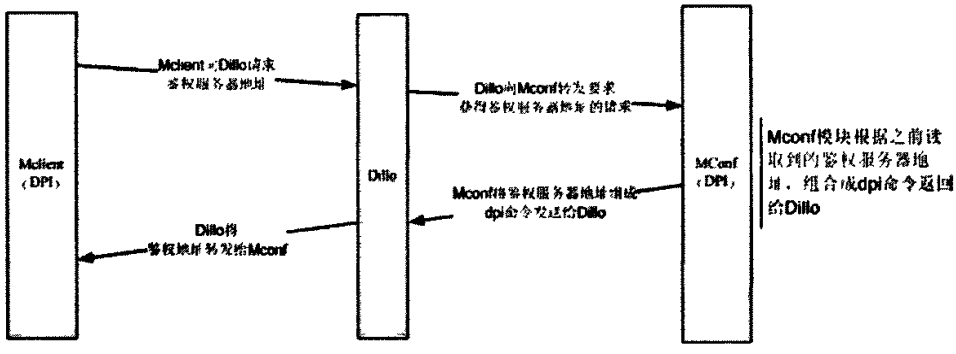

图 3-25　MConf模块请求鉴权服务器地址操作流程

MConf相应获取鉴权服务器地址的回复：<dpi cmd='mconf_auth_server' server='%s' port='%ld' protocol='%d'>

说明：

- cmd = mconf_auth_server 用于表明本次要求获取鉴权服务器内容；
- server = 鉴权服务器地址；
- port = 鉴权服务器端口号；
- protocol = 鉴权服务采用的协议

3） Mterm模块请求网络访问方式

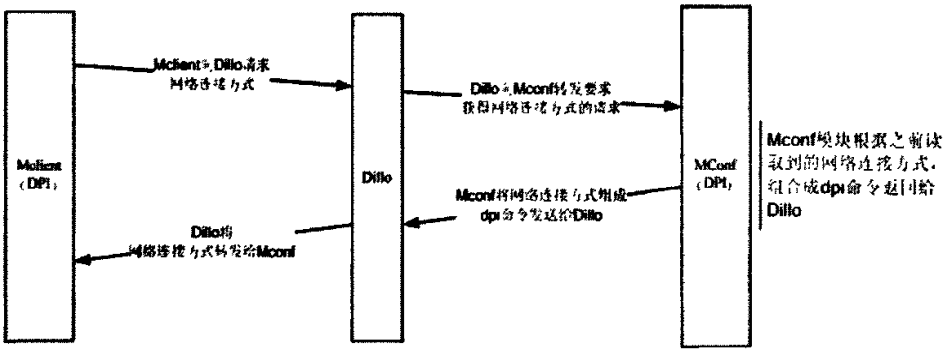

图 3-26 MConf模块请求 MTerm 网络访问方式操作流程

MConf相应获取Mterm播放采用协议的回复： <dpi cmd='mconf_mterm_protocol' protocol='%d'>

说明：

- cmd = mconf_mterm_protocol 用于表明本次要求获取域服务器内容；
- protocol = 播放采用的协议；

4） 从页面保存登录用户名和密码

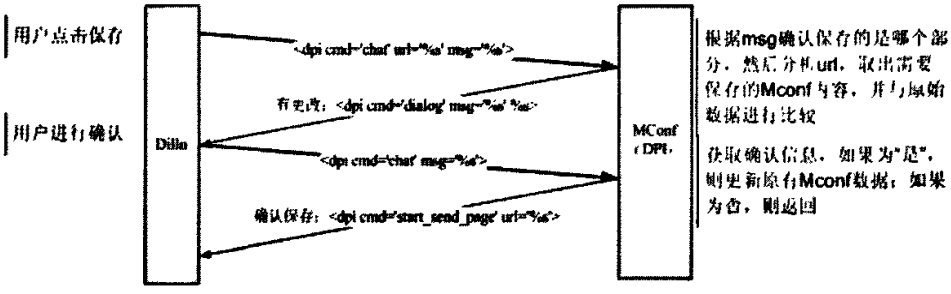

图 3-27 MConf模块保存用户信息操作流程

MConf询问对话框：<dpi cmd='dialog' msg='%s' %s>

说明：

- cmd = dialog 用于表明本次操作为显示对话框；
- msg='%s' %s 第一个%S用于表明对话框显示的内容，第二个%s用于表明按钮的内容；

5） 关机时的全部保存

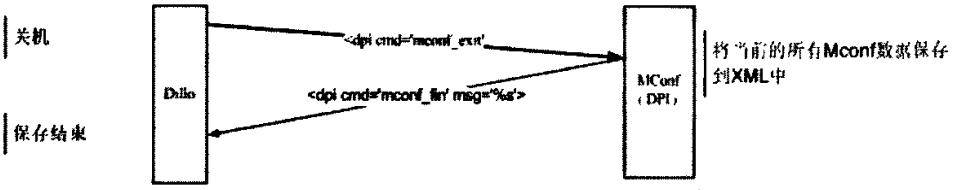

图 3-28 MConf 模块保存配置信息操作流程

Mconf保存结束命令：<dpi cmd='mconf_fin'>

说明：

- cmd = mconf_fin Mconf操作结束；

3.5.3 Http Server

随着嵌入式系统和互联网的发展。通过 Web 方式，对嵌入式系统本身进行监控和设置，成为一些应用于网络服务的嵌入式体统的必备功能。人们通过网络可以做到异地管理和配置嵌入式系统。同时，对一些没有显示界面的系统，利用 Web 页面的方式可以提更好的人机互动，和所见即所得的管理功能。

虽然本 STB 系统，使用电视机作为系统的显示设备，并提供了本地进行系统设置功能，而系统也可以从服务器上下载配置文件。但它仍然有必要，提供通过 Web 方式对本 STB 系统进行设置的功能。因为对于一个 STB 在日常收看 IPTV 的情况，没有必要接入和支持键盘等可以输入更多字符的设备。因此，提供在网络上通过一台计算机远程访问 web 页面的方式，可以更方便的对系统进行设置。HTTP Server 就是在机顶盒软件启动的情况下，用于用户远程登录到机顶盒，通过在远端 Web 页面的方式，对机顶盒的参数进行设置。

HTTP Server 服务器软件选用 Boa 作为的服务器。Boa 是一个单任务的 HTTP 服务器(对每一个连接并不复制一个新的进程或 web 服务器副本)。BOA 支持能够实现动态 WEB 技术的 CGI 技术，源代码开放、性能高。同时服务器程序本身所占空间很小。可以满足本程序的要求。它可以提供的 HTTP 服务，发布 Web 页面和基本的 Cgi 的功能，足够小巧，速度快。提供 Cgi 功能，可以动态的对用户提交的表单数据进行处理和支持。可以满足通过 web 页面，进行 STB 动态设置的功能。

Cgi 程序通过 Unix Domain Socket 与 MConf 模块进行通信。UDS 由 MConf 建立。作为与 Cgi 程序通信的信道。MConf 通过 select()函数的 IO 多路复用的功能，同时检测与 Cgi 通信的 UDS 和与 Dillo 端通信的 UDS。对 Cgi 程序进行服务。通信协议遵循 dpip 的原则，与现有协议体系兼容。

客户端提交的表单数据，可以有 Get 和 Post 两种方法提交到 Web 服务器中。由于 Get 方法提交的数据，将显示在 URL 上。因此，选用 Post 方法更加保密一些。CGI 程序从环境变量“CONTENT-LENGTH”中获得字符串的长度，此环境变量有 web 服务器建立，并设置。CGI 获得文字长度后，就可以从标准输入中获得表单提交的字符串。此字符串的格式为：name1=value&name2=value&...其中，name 为各输入控件字段名，value 为字段值。原数据中的空格字符被替换成为'+'字符。特殊字符被替换为%xx，xx 为此字符的 ASCII 码。

Cgi 获得上面的字符串后，进行解析。生成发往 MConf 的 dpi 的设置命令。
<dpi cmd='mconf_setservice' name='…' pwd=' '>
并使用 UDS 发往 MConf，等待 MConf 的处理结果。如果 MConf 处理成功，则向 cgi 发送<dpi cmd='MConf_setservice_success' msg='…'>dpip 字符串,然后 Cgi 再通过标准输出返回处理结果页面。

3.6 MClient 模块的实现

MClient 模块主要进行与系统相关的工作。协助 Dillo 及 dpid 对系统和各模块进行管理。它的主要任务有，用户认证、系统状态管理、系统软件升级、记录系统日志。

3.6.1 MClient 的用户认证功能

作用：与用户认证服务器联系，进行用户认证的工作，以取得服务供应商的内容，和相关操作的权限。其具体操作过程如下：

1）当系统要求 MClient 向系统要求进行认证后。即 Dillo 向 MClient 发送 "<dpi cmd='MClient_start_Identify'>" 。

2）MClient 就向系统要求获得 MConf 所保存的用户信息。即向 Dillo 发送，要求从 MConf 获得升级用户信息。"<dpi cmd='MClient_requireAccessInfo'>"

3）MCient 收到由 MConf 传回的用户信息"<dpi cmd='MClient_usrIdentify' usr='%s' pwd='%s' >"。则 MClient 启动一个线程(不影响其它 MClient 的任务)，连接认证服务器，并向认证服务器发送信息认证 XML 字符串。

<?xml version=\"1.0\" encoding="gb2312"?>
<ACCESSInfo Name="%s" Pwd="%s"></ACCESSInfo>

2） 服务器返回结果字符串，即返回鉴权成功与否的消息。如：
<ACCESSREPLY msg='0'></ACCESSEPLY>然后线程向 Dillo 报告认证结果。
<dpi cmd='mclient_access' msg='%s'>(msg='userOK' or 'userfail' or 'noserver' or 'nouser') 等。如果认证成功,返回 0;msg ='0'；认证失败,返回 1; msg ='1'。

Mclient 本身，需要的用户认证服务器信息，存储在由 MConf 管理的文件中。所以，要通过 Dillo 向 Mconf 询问用户认证服务器的地址和端口号。这时 MClient 模块会发送"<dpi cmd='MClient_requireAccessServer' >"命令向 Dillo，由 Dillo 返回结果<dpi cmd='MClient_setAccessServer' server='%s' port='%s'protocol='%s'>。

3.6.2 MClient 的系统状态管理功能

MClient 保存系统各模块的运行状态。为系统管理提供依据。

1）Dillo 向 MClient 要求模块状态。
"<dpi cmd='mclient_getModulestate' modulename='%s'>"
2）MClient 向 Dillo 返回状态
"<dpi cmd='mclient_tModulestate' modulename='%s' modulestate='%s' >"
3）Dillo 向 MClient 设置模块状态。
"<dpi cmd='MClient_setModulestate' modulename='%s' modulestate='%s' >"

3.6.3 MClient 的系统软件升级

本功能要求可以自动定时的向升级服务器查询升级信息、并下载相关文件、替换原文件。使系统软件能够得到及时的更新。与用户认证一样，要求系统升级由一个单独的线程负责。此线程在 MClient 启动时与之一同启动。而在 MClient 退出时才退出.

Mclient 主线程和 Update 线程之间的通信，除了线程同步机制和利用传入线程的数据结构。此结构包括 Mclient 主线程控制 Update 线程的信息和升级所需要的模块信息结构和升级信息结构。

使用 pthread_cond_t 结构及其相关函数的线程同步机制，可以由 MClient 主过程控制线程启动或达到 timeout 就可以重启动线程进行系统升级的各种任务。

MClient 的 Update 线程达到 timeout，就向升级服务器发送升级信息。MClient 读取保存模块信息的 XML 文档，包含系统所含模块，及其版本等信息。组成的 XML 格式的消息字符串。线程连接服务器，并将模块信息，发送到升级服务器。当升级服务器返回结果后。若有下载，服务器发送下载文件，及其在 STB 系统中的替换位置。

当 Update 线程分析结果后，发现有可用升级。向 Dillo 发送升级消息。如果用户选择升级，Dillo 则会向 MClient 发送消息。MClient 会重启动线程，下载所需的升级文件。下载完成后，线程向 Dillo 发送下载文件完毕的消息。Dillo 获得该命令后，要求 MClient 进行替换文件的工作。MClient 模块替换相关文件并向 Dillo 返回替换成功的消息。然后更新模块信息的 XML 文档。MClient 还提供了手动进行从服务器检查和手动停止当前升级的功能。

3.6.4 MClient 的系统系统日志记录功能

本功能的主要作用是将系统发生的事件和系统发生的错误，记录下来。以供浏览和咨询。

日志从系统运行时间状态来说分为三个阶段，启动，运行，退出。而从日志事件分类上说，分为四类。系统信息 Infomation，系统升级信息 Update，错误 Error 和警告 Warning。当某一模块要求记录日志时，向 MClient 发送一个包含要记录日志事件的 ID 代码。ID 的格式定义如下：1）模块 ID（2 位数字）；2）类型名(1 位字母- I 表示 Infomation，U 表示 Update，E 表示 Error，W 表示 Warning)；3）级别（1 位数字）；4）消息号（3 位数字）。一个具体的消息 ID 应该是"01I0001"这样的形式。

当 MClient 获得事件 ID 后，通过从记录信息编码的 XML 文档中获得日志时间 ID 相对应的描述字符串。并将发生日志事件 ID，时间描述，与此信息事件发生的时间一起，记录到日志文档中。

一个具体的 Log 信息如下所示：

<ERROR code="01E0001" Content=" MConf 模块启动错误" Time="2007 年 3 月 13 日 18:33:23"></ERROR>

<INFO code="01I0013" Content="MTerm 模块启动成功" Time="2007 年 3 月 13 日 18:34:00"></INFO>

3.7 MTerm 模块的实现

3.7.1 MTerm 模块的主要功能

MTerm的主要功能是，根据上层（dillo）所提供的URL地址发起流媒体业务，与流媒体服务器建立连接，并通过必要的协商后，进行媒体数据的接受、解码以及回放。目前MTerm所支持的业务类型包括

（1） 直播电视业务，用户可实时在线接收各种电视节目，并可以选择不同的电视频道；

（2） 点播电视业务，在IPTV业务平台提供的检索界面中选择需要的节目进行点播。

该模块所支持的音视频压缩码流主要包括H.264标准码流，AAC标准码流，MP3标志码流。

3.7.2 MTerm 模块的内部结构

MTerm主要由4个子模块构成，分别为媒体控制和描述子模块、媒体接收子模块、媒体解码子模块、输出显示子模块。各模块主要功能如下：

1) 媒体控制和描述子模块

在与流媒体服务器建立连接之后，媒体数据传输之前，该子模块通过RTSP和SDP协议与流媒体服务器进行的协商，同时该模块还负责对媒体数据的播出控制，包括暂停、随机播放和停止等。

2) 媒体接收子模块

该子模块主要负责接收直播或点播的媒体数据。建立RTP/RTCP接收线程，接收来自流媒体服务器的媒体数据和控制数据等。

3) 媒体解码子模块

该子模块主要功能是调用STAPI解码AdaptionLayer中的API函数，利用专用解码芯片对接受到的媒体数据进行解码。

4) 媒体展现子模块

该模块主要功能是调用STAPI展现AdaptionLayer中的API函数，将解码后的媒体数据展现在终端设备上。

3.7.3 MTerm 模块的外部接口

MTerm 与系统其它模块的接口主要有如下两个：

1) MTerm与dillo的接口

MTerm 与 dillo 的通信的机制采用的是 UDS(Unix Domain Socket)，目的是将其设计成 dillo 的一个 dpi（dillo plugin），以方便 dillo 对 MTerm

的管理。

2) MTerm与STAPI解码和展现AdaptionLayer的接口

MTerm对压缩的音视频数据的解码是由硬件实现的，为了方便MTerm代码的管理和调试，所有的媒体数据解码以及解码后媒体数据的展现分别由STAPI解码AdaptionLayer和STAPI展现AdaptionLayer来完成，由AdaptionLayer负责与底层硬件通信。其中，MTerm与STAPI解码AdaptionLayer的接口函数主要有适配层初始化、适配层关闭、各解码器的初始化、各解码器的重置、各解码器的解码以及各解码器的关闭；MTerm与STAPI展现AdaptionLayer的接口函数主要有适配层初始化、适配层关闭、视频的展现以及音频的展现。

3.8 MDebug 模块的实现

此模块的功能是，统计 MTerm 接收的音视频数据包的各种性能测试数据，并对其进行整理后，显示在 STB 系统软件的主界面上。

Mterm 发送信息到 Dillo，Dillo 转发到 Mdebug，Mdebug 组成 HTML 页面，交由 Dillo 进行显示。由于其功能的实现与测试点的选择有关，在此仅对其基本功能做除了简要的描述。

第四章 IPTV 机顶盒软件系统的测试

本测试方案的主要目的是进行与服务器相关的IPTV机顶盒操作的测试。测试过程中需要编写针对不同测试目的的服务器测试程序，关于测试程序的编写，可以是从文件中读取，并返回固定结果，也可以与数据库相关联，从数据库中取出结果并返回。相关业务如果具有一定的关联性，可以考虑在测试程序中一并进行测试。

4.1 主要测试目标

Client端测试的主要内容主要包括以下几个方面：

1）Mdisplay模块对于HTML页面Template与EPG或本地文件(XML)结合的正常显示；

2）Mconf模块的配置文件读取、升级、更新以及保存；

3）Mclient模块的STB状态管理、模块升级等功能；

4）Dillo对于HTML页面关系配置文件的正确解析与跳转；

5）Dillo对于本页面操作的正常描画；

6）Dillo对于各模块应答的正确解释及操作；

7）Mdebug模块对于调试信息的正常显示；

8）Mterm模块对于节目源的连接、通信以及解包、播放等功能的正常使用；

9）Mterm模块被调用过程中Dillo对于用户操作事件的捕捉，并在Mterm模块对于用户操作事件的正常响应；

10）各模块协同工作的测试；

总的系统测试结构图如下：

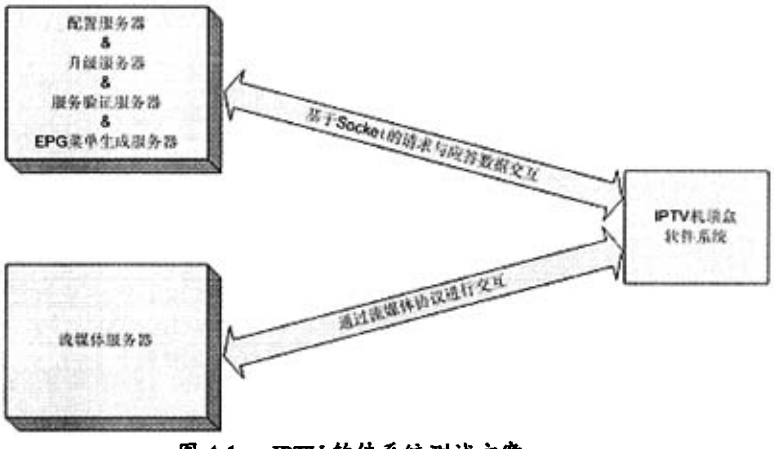

图 4-1 IPTV 软件系统测试方案

由于在整个系统中，涉及到各个部分的测试工作，在本章中，除了针对 IPTV 机顶盒软件系统服务器端测试功能的描述，仅会针对涉及到 MDisplay 模块以及 MConf 模块相关的测试内容，其余部分的测试将会在项目具体的测试报告中涉及。

4.2 IPTV 机顶盒服务器端功能描述

在本测试方案中，需要对软件系统提供的以下网络功能进行测试

1）配置文件版本验证（在配置文件版本低于配置服务器上的版本时，需要能够进行配置文件下载）；

2）模块升级验证（在模块版本低于升级服务器上的版本时，需要能够进行模块升级）；

3）用户登录验证（当程序启动时，能够根据配置文件中的用户名和密码向服务验证服务器发送用户登录认证请求）；

4）EPG菜单生成服务器（能够根据用户登录的用户名进行节目内容交互）；因此Server端需要提供以上基本功能。下图是IPTV机顶盒与Server端交互的框图。

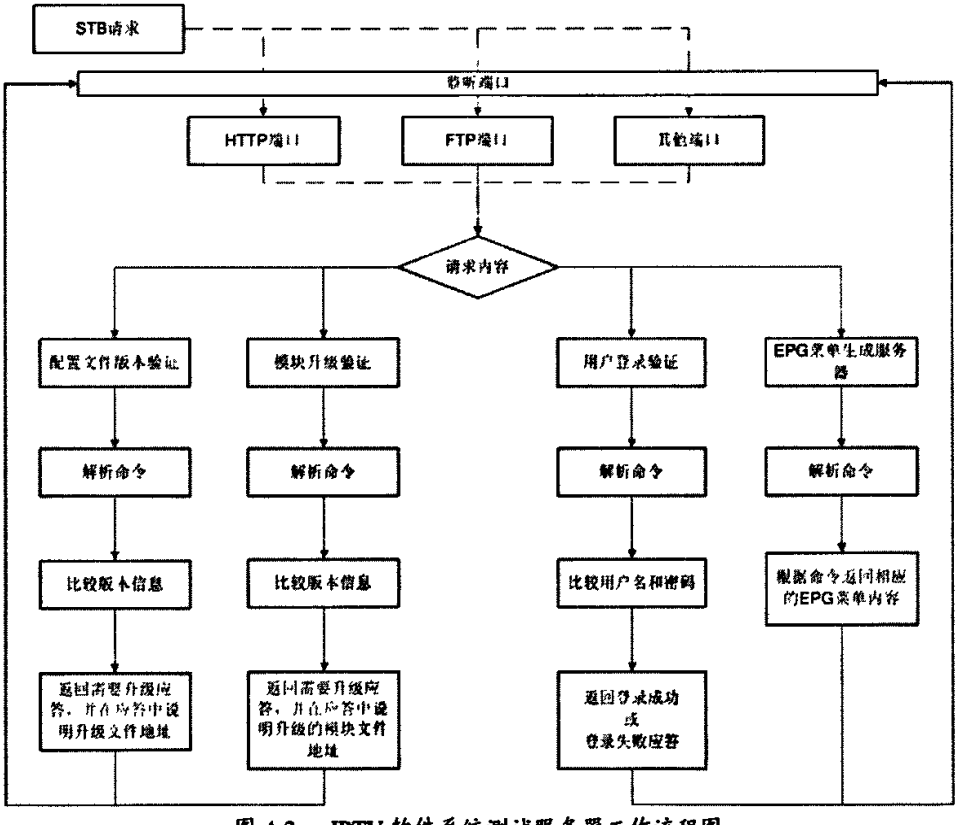

图 4-2　IPTV 软件系统测试服务器工作流程图

4.3 IPTV 机顶盒软件系统模块测试

4.3.1 目标模块：Mconf 模块

测试目标：配置文件版本验证（服务配置文件及网络配置文件的验证与升级）

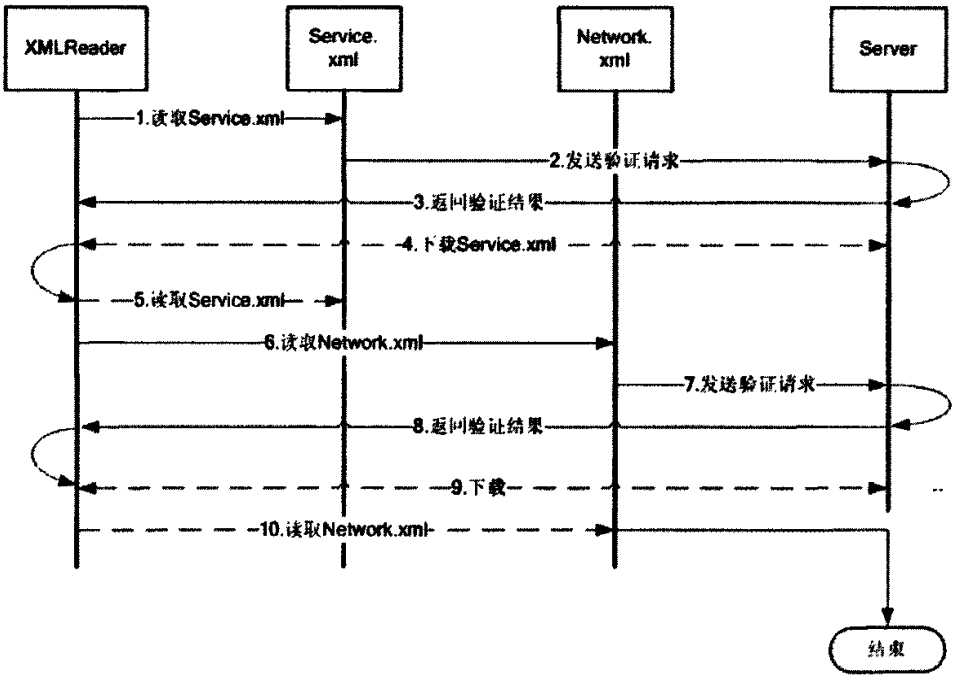

图 4-3　IPTV 软件系统 MConf 模块工作时序图

本测试的目的是根据配置文件提供的配置服务器信息（包括配置服务器地址、端口号以及验证时采用的协议类型），在Dillo启动的时候调用Mconf模块进行读取，并验证配置信息版本。当配置信息版本有更新时，进行新版本的下载，并替换老的版本信息。

在测试过程中，需要Mconf模块以及配置服务器进行通信以确认当前配置文件的版本是否为最新，整个测试需要留有Mconf模块以及Server端的Tace Log，记录下整个操作流程。

要求测试服务器端能够实现以下功能：

1）接收Mconf发送的验证请求，并根据服务器保留的配置文件版本号返回结果；

2）为Mconf模块提供从指定路径下载配置文件的方法；

具体的测试结果，请参考附录1中关于本测试结果的说明。

4.3.2 目标模块：Dillo、Mconf 模块、Mclient 模块

测试目标：模块升级验证（在模块版本低于升级服务器上的版本时，需要能够进行模块升级）；

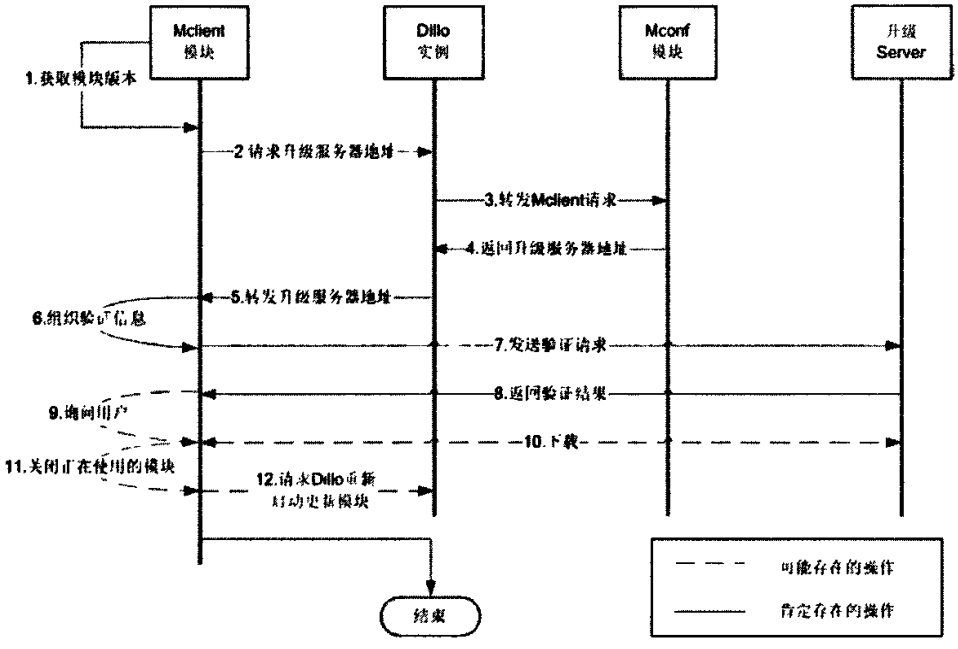

图 4-4　IPTV 软件系统软件升级时序图

本测试的目的是测试Dillo、Mconf模块以及Mclient模块之间相互协作进行模块升级。Dillo实例作为中间的命令转发者，为Mclient模块获取转发升级服务器地址请求给Mconf模块，然后将Mconf模块的应答转发给Mclient模块。Mclient模块在获取了升级服务器地址后进行保存，并组织升级验证信息进行升级验证。当检测到有需要升级的模块时，询问用户是否需要升级，并根据用户应答进行升级或相关操作。

在测试的过程中，需要检查Dillo实例是否能够正确转发Mclient模块的请求以及Mconf模块的应答；Mconf模块是否能够正确解析Dillo转发的请求并正确返回升级服务器地址；Mclient模块在接收到Dillo转发的升级服务器地址后，组织升级信息，发送向升级服务器；在用户确认升级请求后，正确下载文件；Mclient模块在下载完需要升级的模块后，能够正确停止相关模块，并进行升级；在相关模块升级结束后能正确重新启动。

要求测试服务器端能够实现以下功能：

1）接收Mclient发送的验证请求，并根据服务器保留的模块版本号返回结果；

2）为Mclient模块提供从指定路径下载升级模块的方法；

具体的测试结果，请参考附录2中关于本测试结果的说明。

4.3.3 目标模块：Dillo、Mconf模块、Mclient模块

测试目标：用户登录验证（当程序启动时，能够根据配置文件中的用户名和密码向服务验证服务器发送用户登录认证请求）；

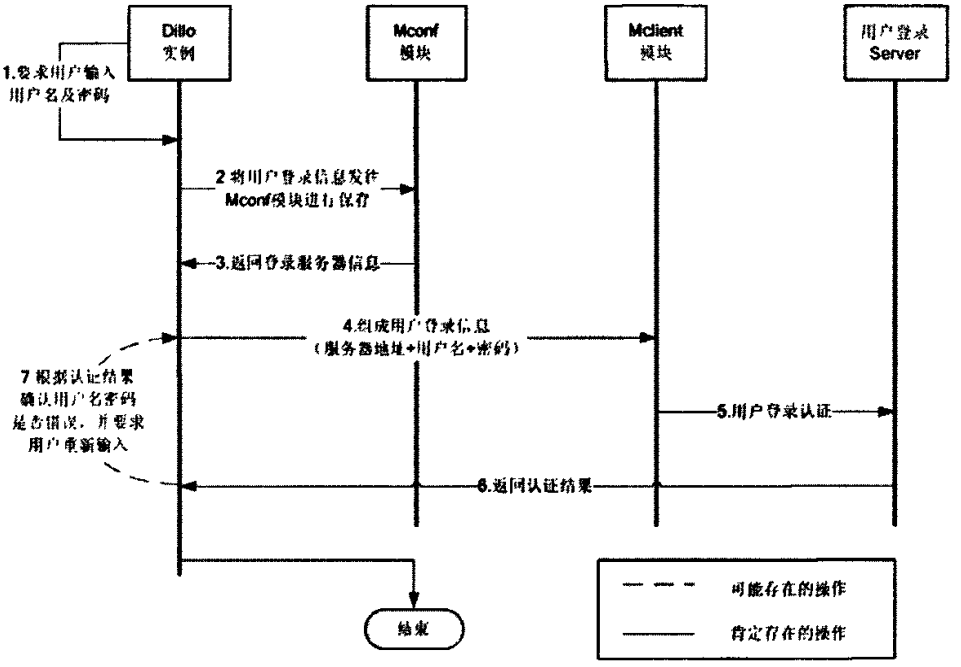

图 4-5　IPTV 软件系统用户登录工作时序图

本测试的目的是测试Dillo、Mclient模块以及Mconf模块相互协作进行用户登录认证。首先，用户通过Dillo实例解析HTML页面的GET/POST方法获取用户输入的用户名以及密码；Dillo实例将用户输入的用户名和密码组成dpi命令发送至Mconf模块；Mconfi模块在接收到dpi命令后获取用户名和密码，并进行保存，再重新结合用户登录服务器相关信息组成dpi命令发送给Dillo实例；Dillo实例在接收到Mconf发来的用户登录信息后，转发给Mclient模块；Mclient模块在接收到了用户登录信息后，进行解析，取出登录服务器信息、用户名和密码后，访问用户登录认证服务器进行认证；用户认证服务器根据用户名和密码进行认证，并返回认证结果；如果用户登录成功，则显示Memu页面，如果认证不成功，则使用对话框提示用户，并要求用户重新输入；

在测试过程中，需要检查以下内容：1）登录用户名以及密码是否正确进行了保存；2）Mclient是否收到用户登录服务器以及用户名和密码；3）Mclient是否能进行用户认证，并根据返回结果进行相关判断；3）Dillo是否能正确转发Mclient以及Mconf模块之间的命令；

要求测试服务器端能够实现以下功能：

1）接收用户登录请求，并根据服务器保存的用户名密码返回结果；

具体的测试结果，请参考附录3中关于本测试结果的说明。

4.3.4 目标模块：Dillo、MDisplay 模块

测试目标：EPG菜单生成服务器（能够根据用户登录的用户名进行节目内容交互）；

时序图（以TV菜单为例）

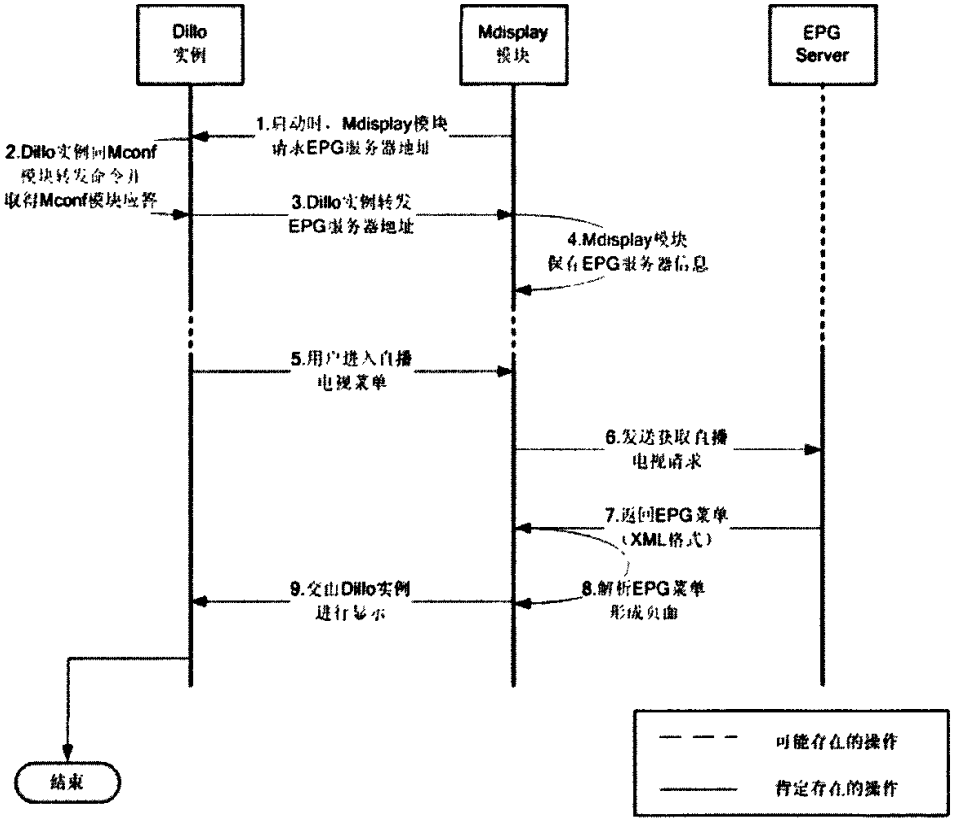

图 4-6　IPTV 软件系统 EPG 菜单获取工作时序图

本测试的目的是测试当用户访问到需要获取EPG菜单的页面时，Mdisplay模块是否能够正确获取EPG菜单，并且将EPG内容解析后形成显示页面。Mdisplay模块主要处理的是页面跳转以及显示的部分，并且当用户选择了特定的节目时，能够返回该节目对应的节目源。

在测试过程中，需要检查以下内容：1）Dillo实例是否能够正确发送页面显示请求，标识需要访问的页面内容；2）Mdisplay模块是否能够正确解析Dillo实例发送的页面显示请求，并判断该页面是否需要访问EPG服务器；3）Mdisplay模块是否能够正确连接EPG服务器；4）EPG服务器在返回EPG菜单后，Mdisplay模块是否能够正确解析，并形成显示页面。

要求测试服务器端能够实现以下功能：

1）是否能正确接收EPG菜单请求，并根据不同请求页面返回不同EPG内容；

2）是否能根据不同用户名返回不同内容（可选）；

具体的测试结果，请参考附录4中关于本测试结果的说明。

4.3.5 目标模块：Dillo、MDisplay 模块、Mterm 模块

测试目标：用户在本页进行节目选择，在选中某一条记录后，进行节目播放

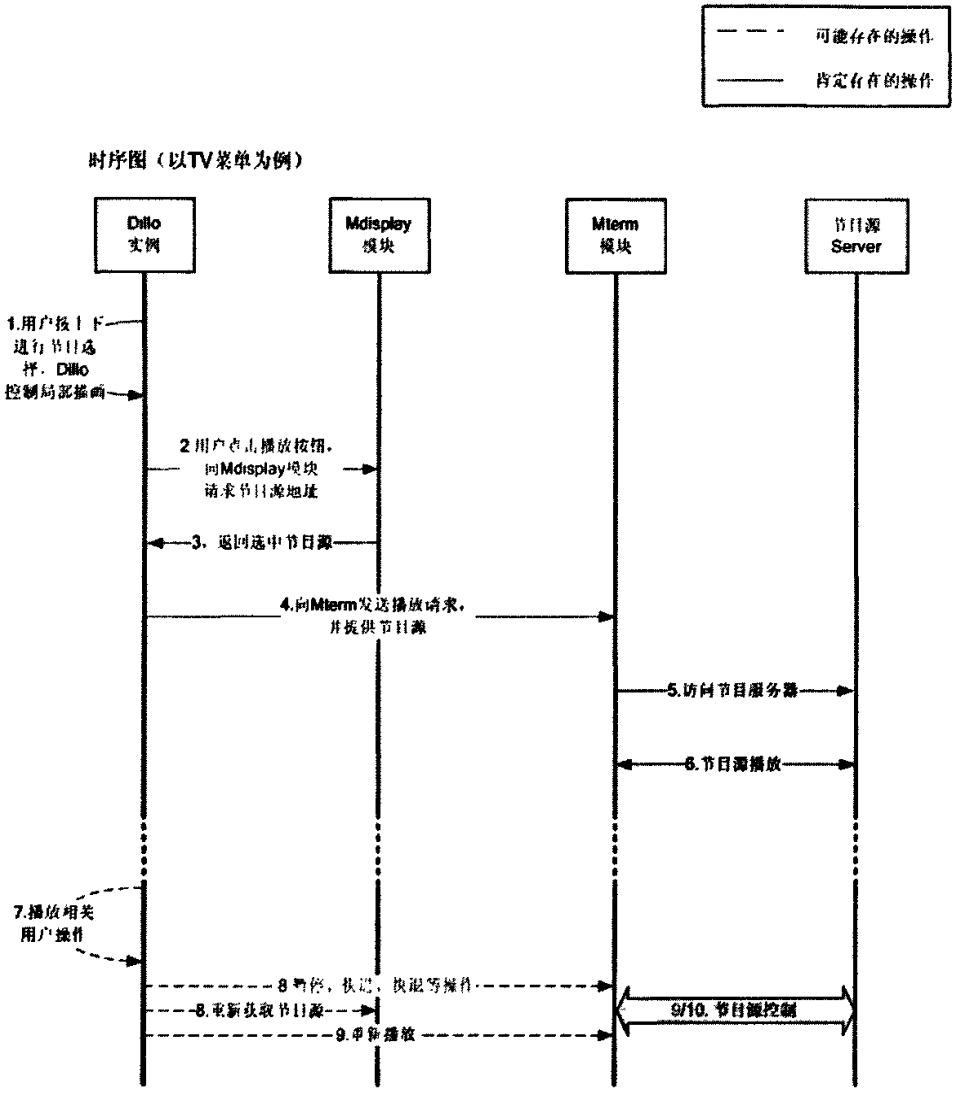

图 4-7　IPTV 软件系统节目播放工作时序图

本测试的目的是测试当用户在节目选择页面进行节目选择时，是否存在画面闪烁以及当选中某条节目时，Mdisplay模块能够返回节目源地址，Dillo实例是否能启动Mterm模块进行节目的播放。当用户在同一个页面进行上下选择的时候，

不应该出现页面闪烁；当用户选中某条节目，并按下“播放”按钮的时候，Dillo实例向Mdisplay模块询问选中的节目源地址；当Dillo获取节目源地址后，调用Mterm模块进行节目播放；Mterm模块连接节目源服务器进行节目的实时及播放，并在播放过程中能够接受来自Dillo实例的控制。

需要测试的内容包括：1）Dillo实例在同一个页面上进行节目浏览时不应该产生闪烁；2）当选中节目时，Mdisplay模块是否能够返回正确的节目源地址；3）Dillo实例是否能在收到节目源地址后调用Mterm模块并将地址传给Mterm模块；4）Mterm模块是否能正确连接节目源并进行播放；5）节目播放过程中，Dillo是否还能够控制Mterm；

要求测试服务器端能够实现以下功能：

1) 是否能够根据Mterm的请求，如播放、暂停、快进、后退等做出相应反应。

具体的测试结果，请参考附录5中关于本测试结果的说明。

第五章 总结与展望

5.1 总结

近年来，已经有越来越多的人开始关注 IPTV 的应用，并投身到 IPTV 的开发中。本课题的研究也是在这样的情况下展开的。在整个课题的设计与实施过程中，本人对 IPTV 业务体系有了比较深刻的理解，并针对作为用户终端接入部分的 IPTV 机顶盒进行了研究。由于 IPTV 机顶盒软件系统 IPTV 业务体系来说有着至关重要的作用，并且在各类 IPTV 业务体系定义文档中均对 IPTV 机顶盒的功能进行了最详细的描述，由于相关规范中规定的 IPTV 机顶盒所需要完成的所有功能，因此在整个课题实施过程中，仅在软件设计过程中为今后的业务扩展提供了接口，但是在 IPTV 机顶盒软件系统的实施过程中完成了其基本功能。

本文在第一章主要针对课题的背景、什么是 IPTV 以及 IPTV 机顶盒软件系统的基本要求进行了描述。从第二章开始，针对第一章中提出的 IPTV 软件系统的功能要求结合现代软件设计要求，在选定了硬件以及软件操作系统的前提下，对整个软件系统进行了设计。在整个设计过程中，充分考虑了 IPTV 机顶盒软件系统的功能要求以及软件开发中针对程序健壮性、扩展性的要求，并在整个设计过程中保留有完整的程序设计文档和模块功能定义文档，为开发者对于软件结构的理解，扩展功能的开发提供了比较详细的资料。

第三章针对提出的 IPTV 机顶盒软件系统的系统结构设计，选定以嵌入式浏览器 Dillo 为基础，在说明了 Dillo 原有的体系结构后，结合已有的系统结构，对 IPTV 机顶盒软件系统进行了实现。在整个描述过程中，主要侧重于对 Dillo 已有软件结构和与设计结合的描述，以及各个模块在实施过程中如何针对已有的设计进行具体实现。其中，对本人完成的 MDisplay 模块以及 MConf 模块进行了比较详细的描述。

第四章中描述了如何对 IPTV 机顶盒软件系统进行测试，以及一部分测试的实施情况。一个优秀的软件系统应该是经过测试的，由于篇幅原因，在本章中只针对一些基本的情况进行了描述。

从本文的描述中，应该可以比较清晰的了解 IPTV 机顶盒软件系统的设计、实施以及最后的测试过程。

5.2 展望

本文针对 IPTV 机顶盒提出了符合其功能定义的软件系统的设计，在设计的过程中能够充分考虑到系统扩展和二次开发的要求。在软件系统实施过程中，根据 IPTV 机顶盒的功能定义完成了其中基本功能，并针对已经完成的功能进行了系统测试。但是由于时间的原因没有办法进一步对整个软件系统的扩展功能进行实施，扩展功能的实现可以通过完善各业务基础模块的功能或增加新的业务模块来完成。

仍然需要完成的扩展功能包括：

1）用户认证功能。本功能要求 IPTV 机顶盒软件系统能够通过用户输入的用户名和密码在整个 IPTV 机顶盒运行过程中，当用户请求观看节目时，提供相应的服务内容。

2）DRM 认证功能。本功能能够通过用户 CA 认证，得到加密的节目流，并通过对该节目流的解密观看相关节目，DRM 相关要求请参考文献[15]、[22]中的定义。

3）时移电视。本功能支持用户观看那些他们喜欢，但是却没有办法观看的已播出的节目。功能要求 IPTV 机顶盒能够向管理服务器发送请求，并要求保存用户感兴趣的节目。当用户请求该节目内容时，能够为用户提供相关内容的播放服务。

4）画中画。本功能允许用户在观看一个节目的同时，能够观看在小画面中的其他节目源的节目，为用户提供更多的选择。

5）其他辅助功能。本功能要求 IPTV 机顶盒能够提供除观看节目、Internet 浏览以外的其他数据业务功能，如邮件客户端、股票信息等相关内容。

总之，如果要使得整个 IPTV 机顶盒能够真正符合市场需求，还需要进一步丰富本系统的安全性以及其他扩展功能，这需要更多的人通过更多的努力来完成。相信随着课题研究的进一步深入，能够真正使得课题设计并实现的 IPTV 机顶盒得到商业应用。

参考文献

[1] ITU-T FG IPTV, FG IPTV-ID-0025 , "Overall definition and description of IPTV in the business role model", 1st FG IPTV meeting:Geneva, 10-14 July 2006

[2] ITU-T FG IPTV, J.290, "Next generation set-top-box core architecture", SERIES J: CABLE NETWORKS AND TRANSMISSIONOF TELEVISION, SOUND PROGRAMME AND OTHER MULTIMEDIA SIGNALS, 2006-11

[3] ITU-T FG IPTV, J.292, "Next generation set-top-box media independent architecture", SERIES J: CABLE NETWORKS AND TRANSMISSIONOF TELEVISION, SOUND PROGRAMME AND OTHER MULTIMEDIA SIGNALS, 2006-11

[4] ITU-T FG IPTV FG, IPTV-OD-0002 , "Outline for draft text of WG2 deliverable on QoE requirements for IPTV", WG2 "QoS and Performance Aspects", 1st FG IPTV meeting:Geneva, 10-14 July 2006

[5] ITU-T FG IPTV FG, FG IPTV-OD-0003, "Outline for draft text of WG2 deliverable on Traffic Management for IPTV", WG2 "QoS and Performance Aspects", 1st FG IPTV meeting:Geneva, 10-14 July 2006

[6] ITU-T FG IPTV FG, FG IPTV-OD-0004, "Outline for draft text of WG2 deliverable on application layer reliability solutions for IPTV", WG2 "QoS and Performance Aspects", 1st FG IPTV meeting:Geneva, 10-14 July 2006

[7] ITU-T FG IPTV FG, FG IPTV-OD-0005, "Outline for draft text of WG2 deliverable on performance monitoring for IPTV", WG2 "QoS and Performance Aspects", 1st FG IPTV meeting:Geneva, 10-14 July 2006

[8] ITU-T FG IPTV FG, FG IPTV-OD-0010, "Working Document on "Requirements of IPTV Network control aspects"", WG4 Leader, 1st FG IPTV meeting:Geneva, 10-14 July 2006

[9] ITU-T FG IPTV FG, FG IPTV-OD-0018, "IPTV Middleware, Applications, and Content Platforms", WG6 (Middleware, Applications, and Content Platforms) Leaders, 1st FG IPTV meeting:Geneva, 10-14 July 2006

[10] ITU-T FG IPTV FG, FG IPTV-OD-0019, "WG 5 Working Document for IPTV End System", Working Group 5, 1st FG IPTV meeting:Geneva, 10-14 July 2006

[11] ITU-T FG IPTV FG, FG IPTV-OD-0026, "Working Document on Service Scenario for IPTV", WG1 leaders, 1st FG IPTV meeting:Geneva, 10-14 July 2006

[12] ITU-T FG IPTV FG, FG IPTV-OD-0028, "Gap Analysis Document", WG1 – Architecture Chair, 1st FG IPTV meeting:Geneva, 10-14 July 2006

[13] ITU-T FG IPTV FG, FG IPTV-OD-0029, "List of the standardisation organisations relevant to IPTV", WG1 Chairs, 1st FG IPTV meeting:Geneva, 10-14 July 2006

[14] ITU-T FG IPTV FG, FG IPTV-OD-0030, "WG 1 Living List (Architecture)", WG1 chair, 1st FG IPTV meeting:Geneva, 10-14 July 2006

[15] ITU-T, J.197, "High level requirements for a Digital Rights Management (DRM) Bridge from a cable access network to a Home Network", SERIES J: CABLE NETWORKS AND TRANSMISSIONOF TELEVISION, SOUND PROGRAMME AND OTHER MULTIMEDIA SIGNALS, 2005-11

[16] STMicroelectronics, "STx7100 (STb7100, STi7100, STm7100) Low-cost HDTV set-top box decoder for H.264/AVC and MPEG-2", STMicroelectronics, 2005

[17] 微软（中国）研究开发中心 MSTV 事业部，《VOD 视频平台项目建议方案 V1.0》，北京, 2006 年 5 月

[18] 中国网络通信集团公司，《IPTV 机顶盒技术规范草案 V1.0》，北京，2006 年

[19] 中国网络通信集团公司，《中国网通 IPTV 体系架构技术规范(Version1.0)》，北京，2006 年

[20] 中国网络通信集团公司，《中国网通 IPTV 业务 EPG 技术规范(Version1.0)》，北京，2006 年

[21] 中国网络通信集团公司，《中国网通 IPTV 业务规范(Version1 0)》，北京，2006 年

[22] 中国网络通信集团公司，《中国网通 IPTV 业务平台 DRM 技术规范（建议稿)》，北京，2006 年

[23] 中国通信行业标准，《IPTV 机顶盒技术要求（征求意见稿)》，北京，2006 年

[24] 中国通信行业标准，《IPTV 内容运营平台与业务运营平台接口要求（征求意见稿）》，北京，2006 年

[25] 中国通信行业标准，《IPTV 业务系统总体技术要求（征求意见稿）》，北京，2006 年

[26] 中国通信行业标准，《机顶盒与 IPTV 业务平台接口规范 V1.0（征求意见稿）》，北京，2006 年

[27] 开源项目，Dillo 翻译文档，http://www.dillo.org

附 录

附录 1 Mconf 模块的测试结果分析

测试目标：配置文件版本验证（服务配置文件及网络配置文件的验证与升级）

根据测试要求中给出的测试方案，针对 MConf 模块的功能进行了测试，在 Client 端——即 Dillo 实例被启动后，针对 MConf 模块的初始化操作进行测试，测试的结果如下图所示：

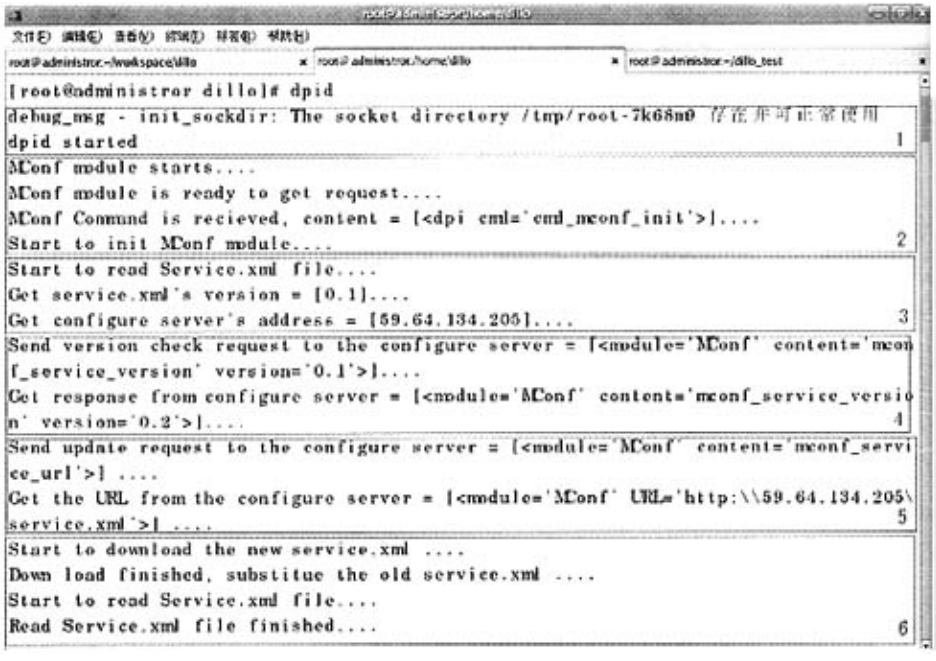

图 附录 1-1 MConf 模块测试结果（客户端）-1

图 附录 1-2 MConf 模块测试结果（客户端）-2

1）方框[1]中，作为 InternalBus 的 dpid 被成功启动；
2）方框[2]中，Dillo 向 MConf 模块发送"cmd_mconf_init"的命令，针对 MConf 模块进行初始化启动；
3）方框[3]中，接收到初始化命令的 MConf 模块会首先读取 service.xml 中的版本信息以及版本配置服务器地址；
4）方框[4]中，MConf 模块在读取到 service.xml 的版本信息后会向版本配置服务器发送版本请求命令，经版本配置服务器返回 service.xml 的版本内容；
5）方框[5]中，MConf 模块向版本配置服务器发送请求，要求获取新版本 service.xml 的下载地址，并获得来自版本配置服务器的应答；
6）方框[6]中，MConf 模块根据版本配置服务器的提供的下载地址进行新的 service.xml 文件的下载，并在下载结束后重新读入 service.xml 文件；
7）方框[7]中，MConf 模块读取 network.xml 文件的版本号，并向版本配置服务器请求验证是否有新版本；
8）在确认没有新版本后直接读取 network.xml 文件中的内容。并在成功读取后，发送"mconf_finish"命令给 dillo，结束 MConf 模块的初始化工作。

在测试 Server 端，接收到 MConf 模块的命令，如下图所示：

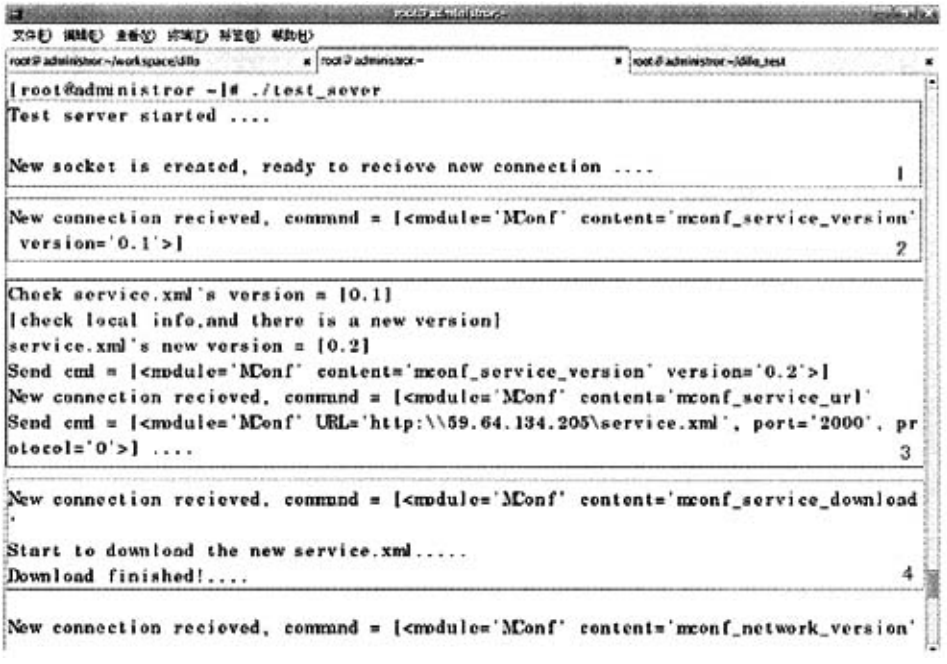

图 附录 1-3 MConf 模块测试结果（服务器端）-1

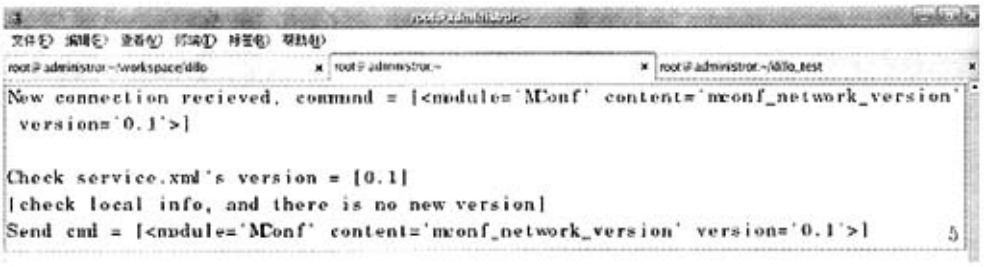

图 附录 1-4 MConf 模块测试结果（服务器端）-2

1）方框[1]中，测试服务器被启动，并打开一个 Socket 用于接收新的连接；

2）方框[2]中，测试服务器接收到了一个来自 MConf 模块的命令，要求获取 service.xml 的版本号；

3）方框[3]中，测试服务器在获取了客户端 Service.xml 文件的版本号，并检查自身的 Service.xml 文件的版本号后，向 MConf 模块发送最新的版本号；当 MConf 模块接收到新版本号，并要求获取新 Service.xml 文件的 URL 地址，由测试服务器组成命令，发往 MConf 模块；

4）方框[4]中，Mconf 模块向测试服务器发送下载请求，并进行最新 Service.xml 文件的下载；

5）方框[5]中，测试 Server 接收到 MConf 模块关于 network.xml 的版本信息，检查 MConf 模块中的 network.xml 是否有新版本，在没有新版本的情况下，直接将版本信息返回给 Mconf 模块。

附录 2 Dillo、Mconf 模块、Mclient 模块测试结果分析

测试目标：模块升级验证（在模块版本低于升级服务器上的版本时，需要能够进行模块升级）；

根据测试要求中给出的测试方案，针对 MConf 模块、MClient 模块的功能进行了测试，测试的结果如下图所示：

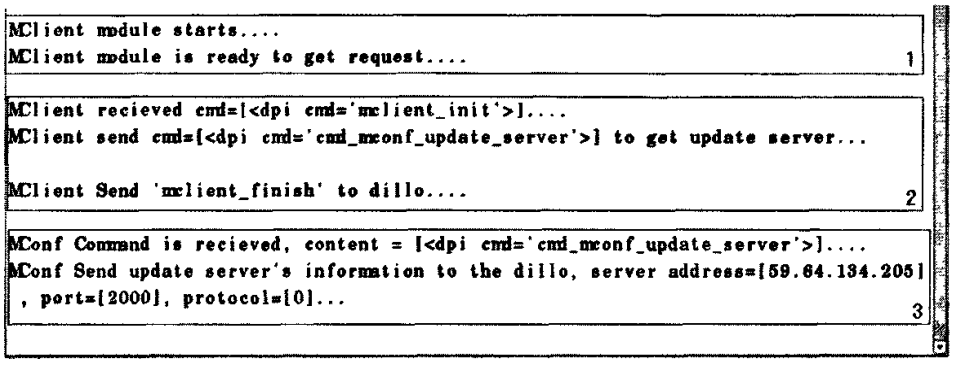

图 附录 2-1 MConf 模块 Mclient 模块测试结果（客户端）-1

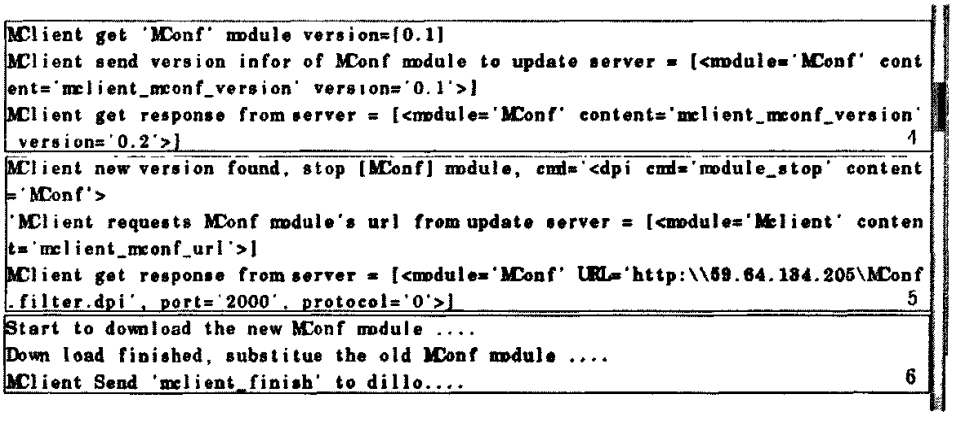

图 附录 2-2 MConf 模块 Mclient 模块测试结果（客户端）-2

1）方框[1]中，MClient 模块被启动；
2）方框[2]中，Mclient 模块向 Dillo 发送获得升级服务器信息的请求，Dillo 会将该消息进行转发；
3）方框[3]中，Mconf 模块接收到来自 Dillo 转发的 MClient 模块要求获得升级服务器信息的请求，形成升级服务器信息命令，并返回给 Dillo。MClient 模块会接收到该内容；
4）方框[4]中，MClient 模块得到 MConf 模块的版本号，并将该信息发送到模块升级服务器，并获得模块升级服务器的应答；
5）方框[5]中，MClient 模块在收到服务器应答后，发现有新版本的 MConf 模块，在用户确认需要进行下载后，则向 Dillo 发出请求，要求停止 MConf

模块的运行，同时要求进行 MConf 模块的下载；

6）方框[6]中，MClient 模块下载了新版本的 MConf 模块，并进行替换。在替换结束后，MClient 模块向 Dillo 发送结束命令，要求 Dillo 重新启动 MConf 模块；

在测试 Server 端，接收到模块的命令，如下图所示：

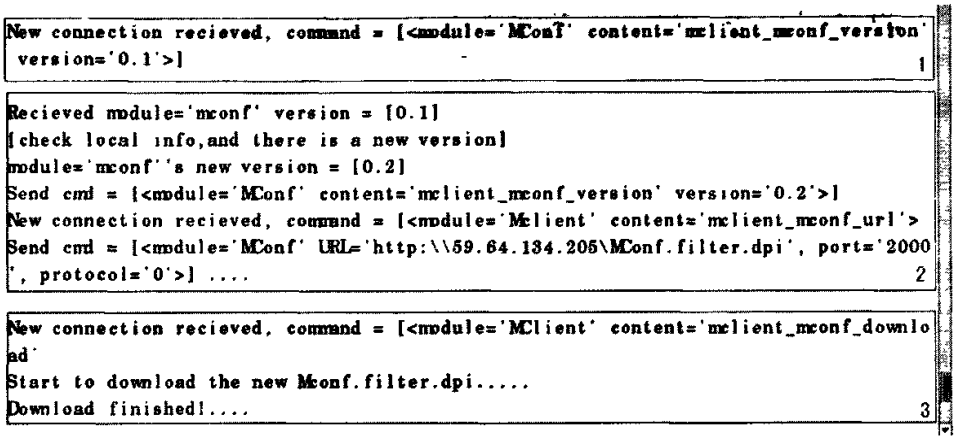

图 附录 2-3 MConf 模块 Mclient 模块测试结果（服务器端）-3

1）方框[1]中，测试服务器接收到来自 MClient 模块的关于 MConf 模块的版本信息；

2）方框[2]中，测试服务器分析 MConf 模块的版本，并寻找发现 Mconf 模块的新版本，并组成版本信息命令返回给 MClient 模块；Mclient 模块在接收到该命令后，通过询问用户，要求进行 MConf 模块的下载；

3）方框[3]中，测试服务器接收到来自 MClient 模块的关于 MConf 模块的下载请求，MClient 模块根据测试服务器给出的下载地址进行 MConf 模块的下载。

附录 3 Dillo、Mconf 模块、Mclient 模块测试结果分析

测试目标：用户登录验证（当程序启动时，能够根据配置文件中的用户名和密码向服务验证服务器发送用户登录认证请求）；

根据测试要求中给出的测试方案，针对 MConf 模块、MClient 模块的功能进行了测试，测试的结果如下图所示：

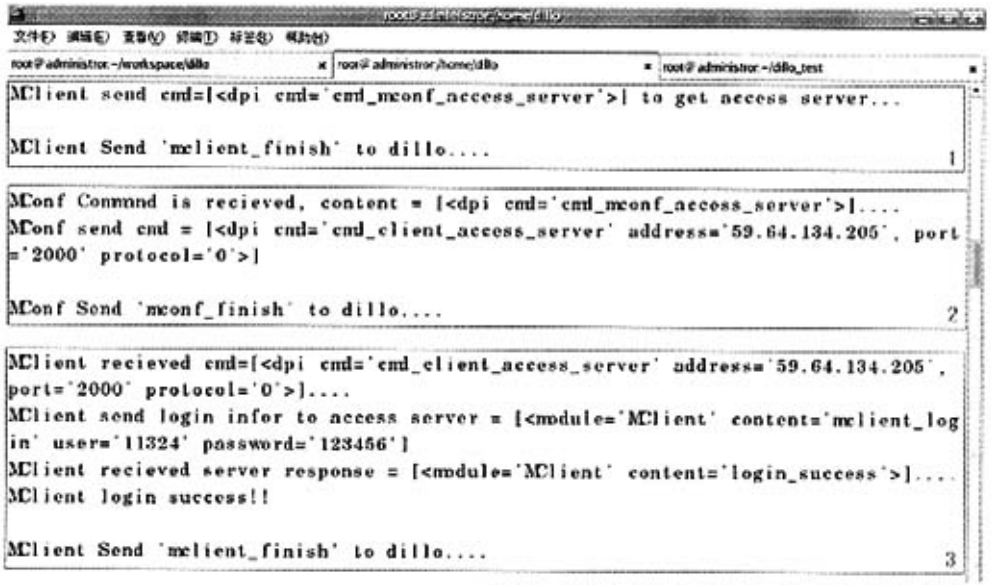

图 附录 3-1 MConf 模块 Mclient 模块测试结果（客户端）-1

1）方框[1]中，MClient 模块向 Dillo 发送请求，要求获得关于用户登录服务器的信息，并由 Dillo 进行转发；

2）方框[2]中，MConf 模块接收到来自 Dillo 转发的 MClient 模块关于用户登录服务器信息的请求，将用户登录服务器的信息返回给 Dillo；

3）方框[3]中，MClient 模块向用户登录服务器发送用户登录验证请求，并获得登录服务器应答，确认登录成功；

在测试 Server 端，接收到模块的命令，如下图所示：

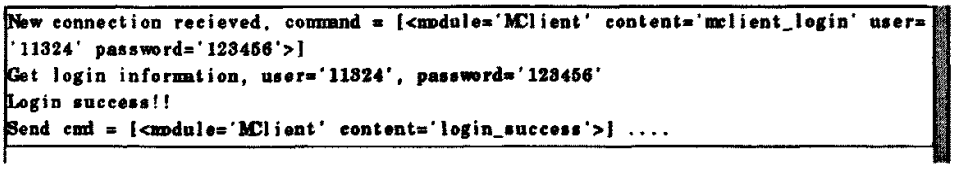

图 附录 3-2 MConf 模块 Mclient 模块测试结果（服务器端）-1

1）方框中，用户登录测试服务器接收到来自 MClient 模块的用户登录请求，提取其中的用户登录信息，在验证后向 MClient 模块返回登录成功应答；

附录 4 Dillo、MDisplay 模块测试结果分析

测试目标：EPG菜单生成服务器（能够根据用户登录的用户名进行节目内容交互）；

根据测试要求中给出的测试方案，针对 MDisplay 模块的功能进行了测试，首先是针对直播电视页面进行测试，测试的结果如下图所示：

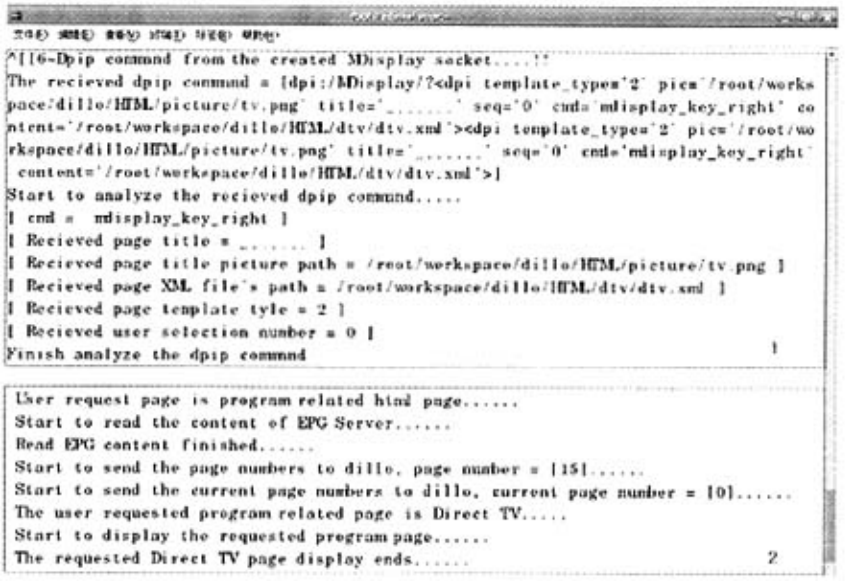

图 附录 4-1 MDisplay 模块测试结果（客户端）-1

1）方框[1]中，用户操作进入直播电视画面，Dillo 向 MDisplay 模块发送页面相关信息；；

2）方框[2]中，MDisplay 模块通过与 EPG 服务器的交互，获得 EPG 菜单内容，并在页面上进行现实，如下图：

图 附录 4-2 MDisplay 模块显示结果（客户端）-1

针对点播电视页面以及热门节目页面进行测试，测试结果如下：

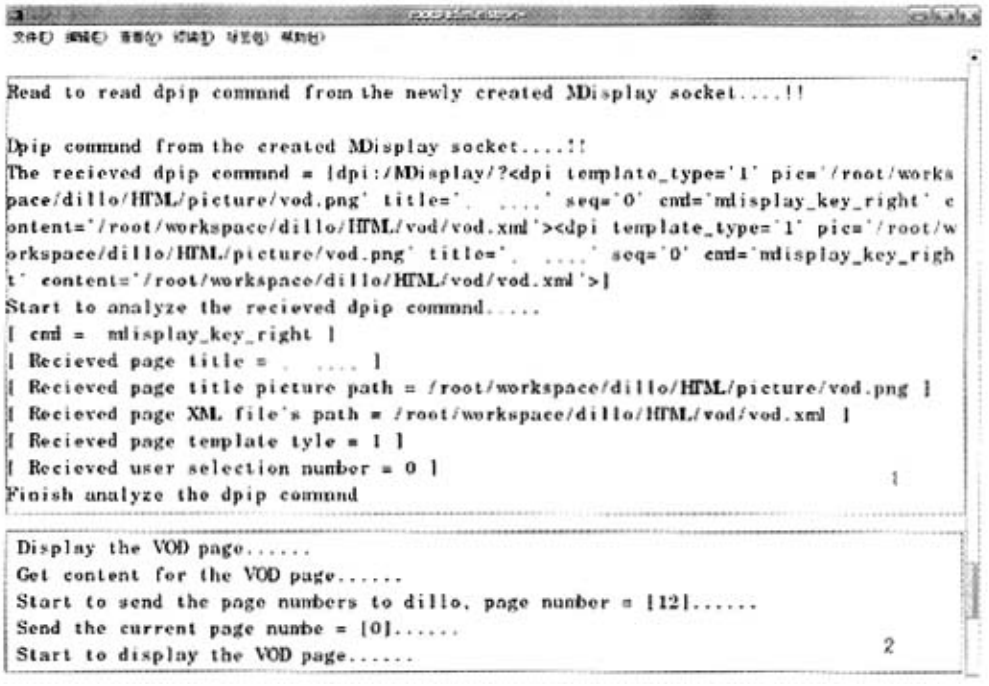

图 附录 4-3 MDisplay 模块测试结果（客户端）-2

1）方框[1]中，用户操作进入 VOD 画面，Dillo 向 MDisplay 模块发送页面相关信息；；

2）方框[2]中，针对 VOD 页面进行操作，现实结果如下：

图 附录 4-4 MDisplay 模块显示结果（客户端）-2

针对热门节目页面进行测试，测试结果如下：

```
Read to read dpip command from the newly created MDisplay socket....!!

Dpip command from the created MDisplay socket....!!
The recieved dpip command = [dpi:/MDisplay/?<dpi template_type='5' pic='/root/works
pace/dillo/HTML/picture/hot.png' title='...Ż.Ł' seq='0' cml='mdisplay_key_right' co
ntent='/root/workspace/dillo/HTML/hot/hot.xml'><dpi template_type='5' pic='/root/wo
rkspace/dillo/HTML/picture/hot.png' title='...Ż.Ł' seq='0' cml='mdisplay_key_right'
 content='/root/workspace/dillo/HTML/hot/hot.xml'>]
Start to analyze the recieved dpip command.....
[ cml =  mdisplay_key_right ]
[ Recieved page title = ...Ż.Ł ]
[ Recieved page title picture path = /root/workspace/dillo/HTML/picture/hot.png ]
[ Recieved page XML file's path = /root/workspace/dillo/HTML/hot/hot.xml ]
[ Recieved page template tyle = 5 ]
[ Recieved user selection number = 0 ]
Finish analyze the dpip command                                          1

User request page is program related html page with a detail page......
Start to get the content of the program related html page with a detail page from
EPG Server......
Read EPG content finished......
Start to send the page numbers to dillo, page number = [15].....         2
```

图 附录 4-5 MDisplay 模块测试结果（客户端）-3

1）方框[1]中，用户操作进入热门节目页面，Dillo 向 MDisplay 模块发送页面相关信息；

2）方框[2]中，MDisplay 模块根据 Dillo 发送的信息，从 EPG 服务器获取页面内容，并形成热门节目页面，如下图所示：

附录 4-6 MDisplay 模块显示结果（客户端）-3

EPG 测试服务器端的结果如下图：

```
New connection recieved, command = [<module='MDisplay' content='mdisplay_tv_epg'
Read EPG content for tv from tv.xml!!
Send EPG content to MDisplay....                                              1

New connection recieved, command = [<module='MDisplay' content='mdisplay_hot_epg'
Read EPG content for hot from hot.xml!!
Send EPG content to MDisplay....                                              2
```

图 附录 4-7 MDisplay 模块测试结果（服务器端）-1

1）方框[1]中，EPG 服务器接收到关于直播电视的 EPG 菜单内容请求，并从相关文件中读取 EPG 菜单内容返回给 MDisplay 模块；
2）方框[2]中，EPG 服务器接收到关于热门电视的 EPG 菜单内容请求，并从相关文件中读取 EPG 菜单内容返回给 MDisplay 模块；

附录 5 Dillo、MDisplay 模块、Mterm 模块测试结果分析

测试目标：用户在本页进行节目选择，在选中某一条记录后，进行节目播放

```
Got a cmd!
MTerm validated the url inputed
Before parseRTSPURL
connect() successfully
Before sendRequest
Sended DESCRIBE
Before getResponse1
After getResponse1
Got OPTIONS Response
Opened URL "rtsp://59.64.134.205/MTV/Alizee-Gourmandises-x264.AAC.384x288.mp4", ret
urning a SDP description:                                                        1
```

图 附录 5-1 MDisplay 模块测试结果（客户端）-1

1）方框[1]中，Mterm 模块接收到来自于 Dillo 转发的 MDisplay 发送的当前节目地址，并开始进行节目的连接；

2）节目连接成功后，开始进行节目播放，如下图所示：

图 附录 5-2 MDisplay 模块 Mterm 模块显示结果（客户端）-1

致 谢

首先，我要感谢为我提供良好学习和生活环境的北京邮电大学以及曾经教育我、关心我的所有老师。

在我硕士研究生学习与工作期间，得到了王雷老师的悉心指导和帮助。老师渊博的知识、开阔的思想、严谨的治学态度和勤恳的敬业精神，对我在研究生求学期间所起的作用，以及今后在社会上，工作中的帮助是难以言语的。在此谨向老师表示由衷的感谢。

感谢实验室诸位师弟的关照，感谢你们的坦诚交流和帮助，使我在实验室的课题研究能够得到很快的进展。

还要感谢和我同届的兄弟姐妹们给予我学习上和生活上无私的帮助，帮我度过了有意义的三年研究生生活。

最后感谢我的父母，他们给予我的关怀和爱护使我能顺利的完成学业。

IPTV机顶盒软件系统的设计与实施

作者：沈昕
学位授予单位：北京邮电大学