基于和欣操作系统的动态文件系统KTFS设计与实现

郭峰, 陈榕

(同济大学 计算机科学与工程系，上海 200092)

摘　要: 和欣嵌入式操作系统是基于 CAR 构件技术、支持构件化应用的操作系统，是国家 863 支持的 TD-SCDMA 的操作系统标准。文中介绍了在“和欣”嵌入式操作系统上，利用 CAR 构件运行环境，构建支持可动态替换、动态扩展的嵌入式文件系统。
关键词: 和欣嵌入式操作系统; 文件系统; CAR 构件技术; 面向侧面

Construction of dynamic file system in elastos

Guo Feng, Chen Rong

(Department of Computer Science and Engineering, Tongji University, 200092, China)

Abstract: Elastos is an embedded operating system based on CAR component technology, and the standard of TD-SCDMA operating system supported by the "863" Program. In this paper, we propose an dynamic file system featured by dynamic replacing and upgrade based on CAR component technology in Elastos.
Key words: Elastos embedded operating system; File system; CAR; AOP

0 引言

回首硬件在过去几年里的发展过程，我们不禁会对某些硬件类型的发展速度感到惊讶。每个人都知道摩尔定律-芯片上的晶体管数量每 18 个月翻一番。但许多人都忽略的一点是，网络带宽和存储技术的发展速度甚至超出了摩尔定律中指出的速度。在企业内部以及通过家庭网络连接到全球网络的单个用户之间，网络带宽都以惊人的速度不断增长。在过去几十年中，服务器和客户端计算机上的存储容量明显增大。1984 年，IBM PC 推出了 10 MB 硬盘。如今，60-80 GB 硬盘已成为便携式计算机的标准配置。在未来的几年里，不难想象便携式计算机将具有 1 TB 甚至更大的存储。伴随着存储以惊人速度增长的同时，用来管理人们每天创建、存储和搜索的数据的文件系统如果不能够跟上时代发展的脚本，必将会大大影响人们的工作效率。

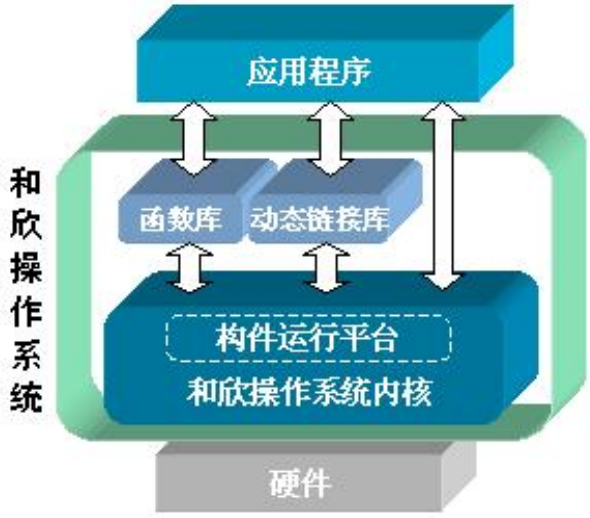
当 Microsoft 大张旗鼓的宣传其代号为“WinFS”的革新性新型文件系统的时候；当桌面操作系统的文件系统面临又一轮技术革命的时候，我们不难发现嵌入式文件系统仍然不能摆脱旧的体系结构的束缚。在以往的嵌入式操作系统架构中，文件系统是内核的必要组成部分，一旦操作系统分发到最终用户，文件系统是不可替

换和升级的，在网络化、构件化的时代，在嵌入式设备日新月异的今天，越来越不能满足多变的时代需求，嵌入式文件系统更需要一次技术革命。

1 关于和欣嵌入式操作系统和 CAR 构件技术

1．1 和欣嵌入式操作系统

和欣嵌入式操作系统是一个基于构件化软件模型的系统，构件化软件设计思想贯穿了整个系统的设计与实现，系统实现本身就是构件模式。除内核中最底层的控制部分外，所有系统功能都是以构件接口的形式提供。另外，操作系统对构件化软件模型提供了必要的运行环境，来源不同的构件可以在该环境上实现互操作。系统提供了构件自动寻址/自动加载机制，用户不必知道调用的构件程序是本地的还是远程的，也就是说，构件运行环境是对用户透明的。构件化系统的实现，使得操作系统本身具有高度的灵活性和扩展性。和欣操作系统的体系结构如下图所示：



1．2 CAR 构件技术

CAR(Component Application Run-Time)是一个国内的自主知识产权的构件系统，是由上海科泰世纪科技有限公司开发的新一代的构件系统。CAR 构件技术定义了一套网络编程时代的构件编程模型和编程规范，它规定了一组构件间相互调用的标准，使得二进制构件能够自描述，能够在运行时动态链接。

CAR 构件技术继承了 COM 的二进制封装思想，面向接口编程。在逐步融合.Net，Java 技术思想之后，形成独有的二进制构件程序模型。其构件模型特征包括以下几个方面：

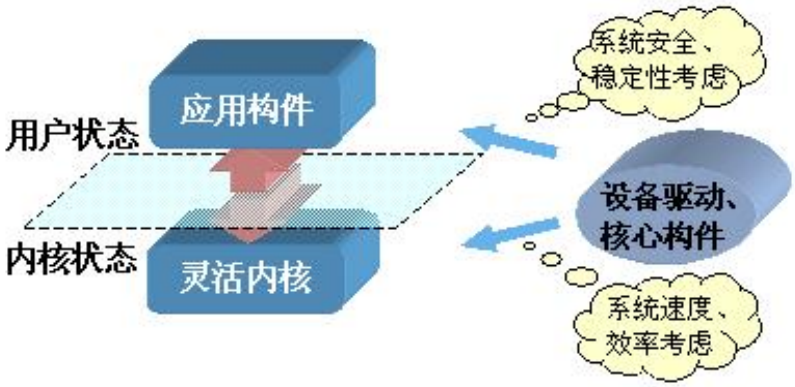
- 面向接口编程
- 构件自描述
- 二进制继承
- 面向侧面编程（AOP）

- 域安全控制
- 构件包组装

其主要目的是从操作系统层面上引入构件的概念，所有的服务由构件来提供，实现软件的目标代码级的重用，为网络编程和 Web Services 提供了强大的支持。

1．3 灵活内核技术

和欣操作系统的实现全面贯穿了 CAR 思想，CAR 构件可以运行于不同地址空间或不同的运行环境。我们可以把操作系统的内核地址区看成是一段特殊的地址空间，用户可以根据运行时的需求，自主选择将操作系统的某些系统服务构件、文件系统、图形系统、设备驱动等构件运行于内核地址空间或用户地址空间。与传统的操作系统的"大内核"、"微内核"体系结构相比，和欣操作系统内核提供的系统服务，完全可以由用户依据系统自身的需求动态决定。因此我们称和欣操作系统内核为"灵活内核"（Agile Kernel）。



1．4 CAR 构件系统对 AOP 的支持

CAR 构件系统对 AOP 的支持是通过 context(构件运行时上下文)来提供的，context 也是一种构件对象，它具有普通构件对象的所有功能。一个构件对象如果进入了某个 context，那么该对象将具有此 context 的特征，一旦对象离开了这个 context，它所具有的 context 特征就会失去（但该对象很有可能又进入了另外一个 context， 拥有新的 context 特征）。比如一只猫（构件对象）把它放到宠物店（context）里去卖，那么这只猫就有了宠物和商品的特征，如果有人买了这只猫，进入了别人家里，这只猫在失去宠物店的 context 特征的同时又重新拥有了新的 context 特征。每个 context 特征可以包括一个或多个侧面，构件通过进入某个 context 就可以组合原有逻辑和 context 特定侧面，从而构成新系统。

CAR 构件库提供如下两个函数分别完成对象环境的进入和离开：

```
STDAPI EzEnterContext(
     /* [in] */ PIOBJECT pIContext,
     /* [in] */ PIOBJECT pIObj)
STDAPI EzLeaveContext(
     /* [in] */ PIOBJECT pIContext,
     /* [in] */ PIOBJECT pIObj)
```

pIContext 为 context 接口指针，pIObj 为进入或离开 context 的构件对象接口指针。

2 和欣嵌入式操作系统中文件系统接口定义

和欣嵌入式操作系统定义了一套文件系统必须实现的接口，该文件系统接口设计目的是为和欣嵌入式操作系统提供必要的文件和目录的操作，接口简洁，为底层实现提供高度的灵活性。在文件系统接口层次，文件系统中的节点对象只有两种类型：一般文件和目录。

和欣嵌入式操作系统要求所有文件系统构件都必须实现下列接口（这里省略了每个接口的详细定义）：

接口名称	功能描述
IBasicFileSystem	文件系统对象接口，是对整个文件系统的抽象
IAbsoluteIO	输入输出接口，包括读、写等操作
IBasicFile	文件对象接口，包括设置文件和目录的相关属性等操作
IBasicDirectory	目录对象接口，包括创建、删除文件和目录等操作
IEzStrEnumerator	字符串枚举器接口

3 和欣嵌入式操作系统中文件系统的新特性

3．1 可动态替换

在“和欣”操作系统的灵活内核技术支持下，文件系统、图形系统等系统服务都可以运行在用户地址空间，这时它们对于操作系统内核来讲都是“外部”服务构件，它们的地位与普通应用程序相比没有任何特殊性。正是因为没有特殊性，用户就可以象卸载普通应用程序一样把文件系统构件“卸载”掉，

然后又可以象安装普通应用程序一样“安装”新的文件系统构件。“和欣”操作系统是完全“面向接口编程”构架的，操作系统对所有服务构件的调用都是通过接口来完成的，因此只要用户“安装”的新文件系统构件实现了操作系统规定的文件系统必须实现的所有接口，操作系统将会“无视”接口实现的任何改变而运行。

“和欣”操作系统提供系统调用EzRegisterService和EzUnRegisterService，它们分别向操作系统注册和反注册命名服务，从而实现服务构件的“安装”和“卸载”。注册命名服务完成后可以通过另一个系统调用EzFindService得到与命名服务相关的服务构件接口指针，进而就可以对接口指针进行操作。

例如，在默认情况下，“和欣”操作系统是通过“filesys”命名服务来得到文件系统构件接口指针，进而获得文件系统构件服务的。为了实现文件系统动态替换，用户可以调用 UnRegisterService 系统调用来反注册默认的 KTFS 文件系统构件，然后以相同的“filesye”命名服务来注册用户指定的文件系统构件，之后操作系统获得的所有文件系统服务都将通过用户指定的文件系统构件来提供（当然进行该操作的用户必须具有指定的权限）。主要代码如下：

```
EzUnRegisterService("filesys");
EzRegisterService("filesys", (PIOBJECT)pINewFileSys);
```

pINewFileSys 为用户指定的文件系统构件接口指针。

3．2 允许多个文件系统共存

由于操作系统和应用程序都可以通过查找命名服务来获得服务构件接口指针，进而获得构件服务，因此用户应用程序可以根据具体应用需求实现具有各自特色的文件系统，并注册为不同的文件系统命名服务，通过查找不同的命名服务来获得不同的文件系统构件服务。每个应用程序都可以获得自己指定的文件系统构件服务，甚至一个应用程序在不同阶段可以获得不同的文件系统构件服务。

例如，在和欣嵌入式操作系统开发的过程中，有三套文件系统构件：KTFS、MEMFS、TFS。其中 KTFS 和 MEMFS 分别提供了对嵌入式设备上 flash 和内存的文件 IO 操作；而 TFS 在 TFTP 网络协议的基础上，实现了对远程文件目录的映射，提供给嵌入式设备对远程主机上文件的 IO 操作。我们把他们注册为不同的命名服务：“filesys”、“memfs”、“tfs”，然后通过系统调用把它们分别映射到不同的盘符，之后通过盘符的切换获得不同文件系统服务，所有的操作与只使用一个文件系统构件没有任何差别。多个文件系统构件的透明使用为我们的开发工作带来了巨大的方便。

3．3 支持远程文件系统

CAR构件运行平台提供自动列集/散集机制（即标准 marshaling/unmarshaling）来支持远程接口调用，通过该机制可以对CAR构件

接口调用的各种标准类型的数据进行不同地址空间数据交互，而用户却如同在同一地址空间里面使用构件，透明地进行接口调用。

同时每一个 CAR 构件的元信息中已经包括该构件的 uunm (Universal Unique Name) 属性，它用来标识该构件在互联网上的地址。例如：uunm(http://……/XXX.dll)，其中 XXX.dll 为该构件的名字。每一个构件定义的时候都必须标明它的 uunm 属性。CAR 构件运行平台解析出构件的 uunm 后会自动的定位出该构件的物理地址，无论它在本地还是在远程。所有对服务构件的接口调用如果需要远程通讯，CAR 构件运行平台会自动完成。

CAR 构件运行平台通过构件自动寻址/自动加载机制，完全向用户屏蔽了底层使用的标准列集/散集以及远程通讯过程，使得操作系统和用户都不需要关心服务构件的物理位置。这样一个位于远程机器上的文件系统构件与位于本地的文件系统构件，对于调用服务的客户（包括操作系统和应用程序）来说，调用方法都是相同的。

支持远程文件系统构件，在下面几种情况下具有明显的优势：

- 由于“和欣”操作系统是面向嵌入式的操作系统，而嵌入式设备往往由于物理设备的限制，对于操作系统的尺寸有着严格的限制，这时我们可以在嵌入式设备上只安装最基本的操作系统模块，而把文件系统构件等服务构件放置在远程机器上。
- 由于嵌入式设备的多变性，它的需求的变化非常快，如果每次文件系统构件的变化都必须在最终用户端进行相应配置的话，无疑会加大用户操作的复杂度。而把文件系统等服务构件放置在远程，会使所有的变化对最终用户来说都是透明。
- 可以使最终用户更合理的、更充分的利用第三方构件厂商提供的文件系统构件。

3．4 支持 AOP

由于 CAR 构件运行平台通过 context 来支持 AOP，而位于 CAR 构件运行平台之上的文件系统构件就可以根据各自需求自定义不同的 context，在适当的上下文中进入适当的 context，从而实现对 AOP 的支持。

例如，用户为了提高整个系统的安全性，需要对所有的文件 IO 操作进行安全检查，当然安全检查并不是文件系统的主要逻辑，如果把这部分代码放在文件系统中，一方面会不可避免的降低构件的模块化程度和复用性，另一方面需要用户重新实现文件系统构件，即使新、旧文件系统构件的主要逻辑都是相同的。这时如果用 AOP，那么操作的复杂度就大大降低了，用户只需要完成很少的代码，甚至直接用第三方提供的 context 就可以实现功能需要，主要实现过程如下：

- 定义 context

```
module
{
  /*定义安全检查接口*/
  interface ISecurity {
    Check();
  }
  /*定义侧面，该侧面具有 ISecurity 特征*/
  aspect ASecurity {
    interface ISecurity;
  }
  /*定义接口 IHello*/
  interface IHello {
    Hello();
  }
  /*定义其他接口*/
  ...

  /*定义 context，该 context 具有 ASecurity 侧面的所有特征*/
  [
    aspect(ASecurity)
  ]
  context CSecurityContext {
    interface IHello;
    ...
  }
}
```

- 进入 context

  EzEnterContext(pIContext, pIBasicFileSystem);

  其中 pIContext 为 CsecurityContext 接口指针；pIBasicFileSystem 为文件系统接口指针。
- 文件系统构件已经具有 ASecurity 侧面所有特征，就像它实现了 ISecurity 接口一样。系统也已经可以通过对文件系统构件查询 ISecurity 接口从而对文件 IO 操作进行安全检查了。
- 离开 context

  EzLeaveContext(pIContext, pIBasicFileSystem);

  其中 pIContext 为 CsecurityContext 接口指针；pIBasicFileSystem 为文件系统接口指针。

4 和欣嵌入式操作系统中文件系统的实现

KTFS文件系统是和欣嵌入式操作系统面向智能手持设备而开发的构件化文件系统，它能够动态、智能识别flash和硬盘等存储设备，具有体积小，速度快，容错性能好的特点。

KTFS 文件系统用四个类实现了操作系统规定的五个接口，它们分别是：

- 文件系统类： CBasicFileSystem，实现了 IBasicFileSystem 接口。
- 一般文件类： CBasicFile，实现了 IBasicFile 接口和 IAbsioluteIO 接口。
- 目录类： CBasicDirectory，实现了 IBasicDirectory 接口和 IBasicFile 接口。
- 字符串枚举器类：CDirStrEnumerator，实现了 IEzStrEnumerator 接口。

CBasicFileSystem 类抽象出了整个文件系统，负责文件系统的初始化等操作；CBasicFile 类实现了对基本文件节点的抽象，负责设置和查询基本文件节点的时间、读写权限等相关属性以及对基本文件的读写操作；CBasicDirectory 类实现了对目录节点的抽象，负责设置和查询目录节点的时间、读写权限等相关属性以及创建、删除文件和目录等操作；CDirStrEnumerator 类实现了对单个目录下所有子节点的枚举操作。

5 总结与展望

和欣嵌入式操作系统在 CAR 构件运行平台的支持下，把文件系统诠释到一个崭新的高度，提供给开发人员和最终用户极大的灵活度以及强大的可操作性。但是灵活度提高对系统的安全性提出了进一步的要求，外部服务构件的身份识别、安全检查将成为进一步的研究目标。

本文受国家 863 计划“软件重大专项” 支持（课题名称：基于中间件技术的因特网嵌入式操作系统及跨操作系统中间件运行平台，课题编号：2001AA113400，所属专题：计算机软件，所属领域：信息技术领域）
郭峰（1981-），男，山西临汾人，硕士研究生，主要研究方向：系统软件支撑技术；陈榕(1952-)，男，北京人，教授，清华大学操作系统与中间件中心副主任，同济大学基础软件工程中心主任，主要研究方向：网络操作系统、构件技术