基于 Elastos 的 WebKit 引擎的研究与移植

谢立丹,陈 榕
(同济大学 基础软件工程中心,上海 200092)

摘 要:在嵌入式系统中,使用 WebKit 作为其显示层内核可以降低程序人员的编程难度和提高程序的运行效率。文中的目的是将 WebKit 移植到 Elastos 嵌入式系统中,使其可以作为 Elastos 操作系统中浏览器的引擎。所采用的方法是利用 CAR 构件技术,完成 WebKit 在 Elastos 上的接口的定义,并将 WebKit 所依赖的图形库与 Elastos 图形库进行衔接。通过对 WebKit 的移植,最终结果是可以在 Elastos 系统上提供一个基于 WebKit 引擎的浏览器。结论是浏览器可以正确地运行在 PC 机上并准确的显示相关网页。

关键词:Elastos;CAR;WebKit;接口
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2011)01-0012-04

Research and Implementation of WebKit for Elastos OS

XIE Li-dan, CHEN Rong
(System Software Engineering Centre of Tongji University, Shanghai 200092, China)

Abstract: It will make programming easy for the embedded programmers and improve efficiency of proceduces by using WebKit as the kernel of layer in embedded systems. The purpose of this article is porting WebKit to Elastos OS, so it could be the browser engine in the Elastos OS. The method which adopting is CAR technology, implement the interface of the WebKit, and connect the graphic library which WebKit depends with the Elastos graphic library. By means of porting WebKit, could implement a browser based on WebKit in Elastos. The conclusion is the browser could run well on the computer, and could display the web page correctly.

Key words: Elastos; CAR; WebKit; interface

0 引 言

随着3G网络的普及,人们上网不再受时间和空间的限制,可以随时随地上网。目前,计算机用户有90%的行为是在网络中靠着浏览器就可以完成,在未来甚至可以提高到95%以上。因此,浏览器的未来将不仅仅是一个简单的应用,而是会成为一个应用软件的平台[1]。

人们拥有一个强大功能的浏览器就可以满足日常生活的需要。同时,基于浏览器的应用大部分是用HTML语言和JavaScript脚本语言编写,具有开发周期短,易于维护的特点[2]。基于这种思想以及浏览器内核技术的成熟,便产生了WebKit内核在Elastos系统上的移植,在Elastos上提供基于WebKit内核的浏览器的方案。

收稿日期:2010-04-23;**修回日期**:2010-07-26
基金项目:国家核高基重大专项(2009ZX01039-002-002)
作者简介:谢立丹(1985-),男,江苏盐城人,硕士研究生,研究方向为嵌入式操作系统、系统软件支撑技术;陈 榕,博士生导师,教授,科泰世纪首席科学家,研究方向为嵌入式系统、构件技术。

1 Elastos 嵌入式操作系统

1.1 Elastos 概述

Elastos 嵌入式网络操作系统是上海科泰世纪科技有限公司开发的拥有自主知识产权的一款新一代适用于网络应用的32位嵌入式操作系统。Elastos操作系统是基于微内核的,具有多进程、多线程、抢占式、线程多优先级任务调度等特性。

同时Elastos操作系统还是一款基于构件化技术的操作系统,Elastos提供关于CAR构件编程模型和CAR构件运行的最优化支持[3]。

1.2 CAR 构件概述

CAR[4]构件是一种二进制构件技术,Elastos是CAR的运行环境。CAR构件是一种类似于微软COM[5]的构件化技术,所不同的是在目标代码中引入了元数据,从而支持元数据计算,进而实现构件的运行组装及状态演化。

CAR构件是一个面向构件的编程模型,它规定了构件间相互调用的标准,包括构件、类、对象、接口等定义与访问构件对象的规定,使得二进制构件可以自描述,能够在运行时动态链接[6]。

一个 CAR 构件由一个. car 文件和一个或多个. cpp 文件组成。CAR 文件主要由构件类、接口、接口方法的定义以及修饰构件类、接口及接口方法的属性和关键字构成。

如 HelloDemo. car:

```
module //构件 HelloDemo
{
    //接口 IHello
    interface IHello {
    Hello([in] Int32 i); //方法
    }
    //接口 IHey
    interface IHey {
    Hey(Int32 I, WString pChar);
    }
    //类 CHello
    class CHello {
    interface IHello;
    interface IHey;
    }
    //applet 类
    Applet THello {
    }
}
```

. cpp 文件是这些接口和类的具体实现,对构件调用者来说是透明的。

2 WebKit 概述

2.1 WebKit 介绍

WebKit[7]是一款浏览器内核,是一个开源项目,是一款优秀的轻量级的浏览器排版引擎,其前身是 KDE 项目,现在主要有苹果公司和 Google 公司的一些组件。WebKit 的 HTML 解析和 JavaScript 代码的解析起源是 KDE 的 KHTML 和 KJS 类库的一个分支。目前基于 WebKit 内核的浏览器有 Safari、Chrome、Konqueror、iCab 等。WebKit 代码具有高效稳定、源码结构清晰易于维护等特点。另外,WebKit 本身兼容性也非常好,很容易移植到不同的平台中。

2.2 WebKit 分析

WebKit 目前支持 HTML4. 0/5. 0、CSS1/2、SVG、RSS2. 0 等。WebKit 主要由布局渲染引擎(WebCore)和 JavaScript 引擎(JavaScriptCore)组成,还包括了平台相关的一些功能,如图形图像、字体、输入法等。WebKit 专注的核心部分主要是分析 HTML, JavaScript 的解析和布局渲染技术。分别在 WebCore/Html, JavaScriptCore 和 WebCore/rendering 里面。

WebKit 的工作的主要流程是,首先用户请求打开一个网页,浏览器将得到的数据发给 HTML 解析器生成相应的 DOM 树,如果有脚本则与脚本解析引擎进行绑定,接着通过 DOM 树生成 Render 树,最终渲染出到屏幕上,整个过程如图 1 所示。

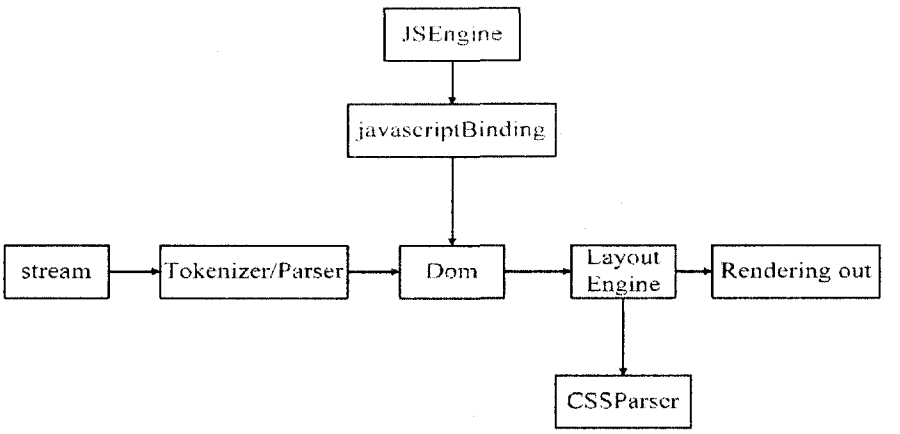

图 1　WebKit 解析渲染图

2.2.1 WebCore 介绍

WebCore 是 WebKit 中的核心组成部分,主要负责 HTML 的解析以及页面内容的排版与渲染,主要包括以下目录:

1. Bindings:主要的工作是将 DOM 树与 JavaScriptCore 方面的代码进行绑定,并生成对应于 JavaScriptCore 的绑定的内容;

2. Bridge:主要包括 NPPlugin 方面的接口访问的相关内容;

3. CSS:主要包括与 CSS 方面相关内容解析;

4. DOM:主要包括不同 DOM 元素的定义与实现、DOM 绑定给 JS 接口定义相关内容;

5. HTML:主要是关于 HTML 相关内容,包括了 HTML 的解析等;

6. Loader:主要是包括加载资源,如 HTML 页面,CSS 等方面资源的加载;

7. Page:主要包括描述一个 Web 页面所涉及到的内容,包括 page, frame, frameview 等;

8. Rendering:主要包括显示方面的内容,包括如何使用样式,组织布局等;

9. SVG:主要包括 SVG(Scalable Vector Graphics)相关方面的内容;

10. XML:主要包括 XML 解析的相关方面内容,包括 XML parser、XSLT 等;

11. Platform:主要包括各种平台以及 WebKit 本身所依赖的外部库方面的内容。

2.2.2 JavaScriptCore 介绍

JavaScriptCore[8]作为 JavaScript 的解释器,主要分为前端和后端框架。前端为解析模块,负责对输入的 JavaScript 代码进行词法和语法的分析、并实现相应的纠错功能,对应的目录主要是 parser 和 yarr 等。后端

为执行模块,执行经过解析后的代码。

3 WebKit 的移植

从 WebKit 设计的体系结构可以看出,WebKit 的核心结构为 WebCore 和 JavaScriptCore 的实现。而这两部分逻辑上是不直接提供接口给外部程序使用的,但 WebKit 作为一个固定的整体,它与外部有一组相对固定的接口,这个接口是在 WebKit 中的 WebKit 目录下定义的,不同的操作系统上有不同的接口的实现。在这个目录中有 win、mac 等目录,分别对应着 WebKit 在 Windows、Mac 等操作系统上接口的定义与实现[9]。这组接口相当于一组协议,这些接口的功能有的是由 WebKit 来实现以供外部来调用,而有的正好相反,是由外部来实现功能供内部来调用的。

因此,在 Elastos 上移植 WebKit 主要做的工作有两个部分,一部分是基于 WebKit 内核的移植,另外一部分是基于 WebKit 外壳的移植。

3.1 WebKit 内核的移植

WebKit 内核的移植中主要有两个问题,WebCore 的渲染引擎与 Elastos 图形系统的结合问题以及 WebKit 内部数据类型与 Elastos 数据类型之间的转换问题。

因为笔者使用 cairo 图形库对二维图形进行渲染,而 Elastos 平台中 ICanvas 对应了一块画布,具体的作图操作都是在这块画布上完成,所以 WebCore 的渲染引擎与 Elastos 图形系统的结合问题主要是如何将 ICanvas 对应的画布与 cairo 中对应画布的 cairo_t 进行转换。笔者采取的做法是先用 ICanvas 申请一块内存构造一块画布,然后加锁,接着在这块内存区域调用 cairo 申请画布的函数,这样这两块画布是基于同一块内存也就是代表了同一块画布。这样也就实现了 Elastos 图形系统与 cairo 图形库的结合。下面是相应的关键代码:

```
ICanvas * pIGfx
    pIGfx->GetClip(&left, &top, &width, &height); //构造一块 ICanvas 画布
    BitmapBuf info;
    info.pitch = 0;
    info.addr = 0;
    IBitmapBuf * bimap = IBitmapBuf::Probe(pIGfx);
    bimap->LockBitmap(&info); //对这块画布加锁
    int stride = (int)info.pitch;
    cairo_surface_t * surface = cairo_elastos_surface_create (
    width, height, stride, (unsigned char *)info.addr);
    //在同样一块内存调用 cairo 申请画布的函数
```

WebKit 内核移植中另外一个重要的问题是关于 WebKit 的数据类型与 Elastos 数据类型的转换。这主要包括三个主要的数据类型,它们分别是:int,bool,String 类型,它们对应于 Elastos 平台的数据类型分别是:Int32,Boolean,WString 类型[10]。

3.2 WebKit 外壳的移植

WebKit 分为内核和外壳部分,在 Elastos 上移植 WebKit 外壳主要做的工作是在 WebKit 目录下定义一个 Elastos 文件夹,在其中提供 Elastos 平台下的 WebKit 的接口的定义以及实现。在 WebKit 中的 WebKit 目录下,每一个不同平台的目录下都有 WebCoreSupport 目录,这个目录下主要定义的是外部程序提供给 WebKit 内部使用的接口。这些接口主要有 WebCore::ChromeClient、WebCore::ContextMenuClient、WebCore::FrameLoaderClient 等。

下面主要论述 WebKit 中的显示模块 WebView 的移植实现。WebView 主要的作用是方便外部程序的嵌入,针对不同的平台 WebView 的定义有不同的实现。但是它们与 WebCore 中的 Page 之间的交互是一样的。在 Elastos 平台上实现 WebView 的移植首先是要写出相应的 CAR 文件,提供相应的接口 IWebView。IWebView 管理着 IWebFrameView 和 IWebFrame 接口。

IWebView 是最重要的接口。为了加载网页,首先是通过 IWebView 的 main frame 发送一个请求,当服务器应答这个请求后,这个 main frame 负责创建一个 IWebFrame,这个 IWebFrame 就是代表了这个网页。IWebFrame 代表了这个网页,但是代表网页视图的是 IWebFrameView,也就是说 IWebFrame 是数据而 IWebFrameView 是表现。IWebFrameView 是附着在 IWebView 上的,所以 IWebView 接口的移植与实现是非常关键的。部分代码如下所示:

```
interface IWebView
    | goBack([out] Boolean * succeeded);
    /*!
    @method goForward
    @abstract Go forward to the next URL in the backforward list.
    @result YES if able to go forward in the backforward list, NO otherwise.
    - (Boolean)goForward;
    */
.....
}
```

这是 webview 众多接口中的一个。在编写好 CAR 文件后,编译,如上文所述,会生成相应的头文件和实现文件[11]。接着就需要将这其中的实现填写完成。最终这个函数的实现的代码如下所示:

```
ECode WebView::goBack(
    /* [out] */ Boolean * pSucceeded)
{
```

```
if (! pSucceeded) return E_INVALID_ARGUMENT;
* pSucceeded = m_page->goBack();
return NOERROR;
}
```

这个函数的主要作用是实现网页向后的功能。其中的 m_page 是 WebCore::Page * 类型。函数调用 WebCore 中的 Page 所提供的功能。其中的 Ecode 是 Elastos 平台的类型,在参数表中的 out 是说明这个参数的属性。其中调用的 goBack() 函数在 WebCore 中的 page 目录下的 page.cpp 文件中的 goBack 函数。

在实现完相应的接口文件后,移植 WebKit 还需对 WebKit 所依赖的库进行移植[12]。比如 WebKit 所依赖的二维图形库 cairo,curl 网络库等。我们已成功将 cairo 和 curl 等 WebKit 所依赖库移植到了 Elastos 平台上,并实现了 cairo 和 Elasto 图形系统的无缝配合。

4　试验结果

将移植结果在主频为 1.66GHz 的 Intel 双核处理器,内存为 1G 的 PC 机中运行,可以正确显示网页。

5　结束语

分析了 Elastos 的相关技术和 WebKit 的核心技术,并针对 Elastos 平台,对 WebKit 做了相应的改动,实现了相应的接口。通过实验表明,可以正确地显示网页。基于 Elastos 平台的 WebKit 嵌入式浏览器可以广泛地应用于多种嵌入式设备中,比如智能手机、PDA 等,该浏览器有着极大的应用前景。

参考文献:

[1] Grosskurth A, Godfrey M W. Architecture and evolution of the modern web browser[M]//David R. Canada: Cheriton School of Computer Science, University of Waterloo,2006.

[2] 周正勇,阳富民,胡贯荣.一种嵌入式浏览器的核心技术及特色[J].计算机工程与设计,2003,24(3):21-23.

[3] Kortide. CAR's Manual[EB/OL]. 2009-07. http://www.kortide.com.cn.

[4] 郑　炜.CAR 构件编程技术中的自描述特性[J].计算机工程与应用,2005(9):95-98.

[5] Box D. Essential COM[EB/OL]. 1997. http://download.cs-dn.net/source/1598215.

[6] 周平东,陈　榕. CAR 构件的四种运行时形态[J].计算机技术与发展,2010,20(4):1-4.

[7] The WebKit Open Source Project[EB/OL]. 2008. http://webkit.org.

[8] JavaScriptCore Framework Reference[EB/OL]. 2008-10. http://developer.apple.com/documentation/Carbon/Reference/WebKit_JavaScriptCore_Ref/index.html#//apple_ref/doc/uid/TP40004754. Apple Inc, October 2008.

[9] 蒋章概,陈　榕.基于 CAR 构件的 WebKit 本地扩展策略[J].计算机应用,2009,29(增刊):195-197.

[10] 杨向科,陈　榕.基于 Elastos 的构件化驱动编程模型的研究[J].计算机技术与发展,2008,18(11):220-222.

[11] 李　辉.一种新型的编程模型——CAR 事件编程模型[J].计算机工程与应用,2005(10):86-90.

[12] 朱剑民,陈　榕,倪光南. 和欣操作系统的浏览器设计模型[J].计算机工程与应用,2003(13):13-15.

(上接第 11 页)

得到了很好的表现,另一方面也能够利用有色 petri 网所具有的丰富分析技术来对并行测试进行分析、验证。

参考文献:

[1] Zhu X P, Xiao M Q. The TPS Development of Parallel Automatic TestSystems[C]//Proc of AUTOTEST 2004 IEEE Systems Readiness Technology Conference. Xi'an, China: [s.n.],2004:248-253.

[2] 陈国良.并行算法的设计与分析[M].北京:高等教育出版社,1995.

[3] Hu Z, Shatz S M. Mapping UML Diagrams to a Petri Net Notation for System Simulation[C]//Proceedings of the International Conference on Software Engineering and Knowledge Engineering. Banffshire: [s.n.],2004:213-219.

[4] 肖明清,朱小平,夏　锐.并行测试技术综述[J].空军工程大学学报(自然科学版),2005,6(3):22-25.

[5] 董　威,王　戟,齐治昌.并发程序的切片模型检验方法[J].计算机学报,2003,26(3):266-274.

[6] Grolleau E, Choquet-Geniet A. Off-Line Computation of real-Time schedules usingPetri nets[J]. Discrete Event Dynamic Systems,2002,12(3):311-333.

[7] 周　维,王明哲. UML 和着色 Petri 网在物资采购建模中的应用[J].计算机系统应用,2002(3):25-28.

[8] 姚绍文,周明天. CPN 原理及其在人工智能中的应用[J].计算机科学,2001,28(1):65-69.

[9] 田保军. UML 类图到 CPN 转化方法的研究[J].系统仿真学报,2007(5):67-70.

[10] 罗雪山. petri 网在 C[4] ISR 系统建模,仿真与分析中的应用[M].长沙:国防科技大学出版社,2007.

[11] 方贤文,赵　艳,殷志祥.基于 Petri 网软件测试分析[J].计算机技术与发展,2007,17(2):96-99.

[12] 袁崇义. Petri 网原理与应用[M].北京:电子工业出版社,2005.

[13] Kazuhiro S. Robust design of flexible manufacturing systems using colored Petri net and genetic algorithm[J]. Journal of Intelligent Manufacturing,2002,13(5):339-351.