

# MobiTran: 一种面向智能手机的 PC 版 Web 应用转化技术

方译萌 马 鄂 刘讓哲 黄 罡

(北京大学信息科学技术学院软件研究所 北京 100871) (高可信软件技术教育部重点实验室 北京 100871)

**摘 要** 随着移动互联网的快速发展,人们可以通过包括智能手机和平板电脑在内的移动计算设备来访问和使用 Web 应用。然而,目前已有的 Web 应用大多是针对 PC 设计的,因此通过移动计算设备直接访问时,可能出现信息显示不全、界面布局混乱、用户体验变差、需要耗费较多流量等问题。重新开发符合移动设备特性的 Web 应用是一种最直接的解决方案,但是开发成本和代价较大,并且同时维护 PC 版和移动版两个不同的应用版本也会带来更高的成本。提出一种面向开发者的 Web 应用移动版本半自动转化技术并实现了支持工具 MobiTran:首先,对 PC 版本 Web 应用的界面进行自动转化,使之适应移动设备的屏幕尺寸;然后,通过支持开发者手工对界面的样式和布局进行修改,定制折分子页面以控制数据流量,同时加入适应移动设备操控特点的元素;最终,生成一个供移动设备访问的移动版本 Web 应用。对主流网站的测试结果表明:MobiTran 可以将原网页自动地转化为适合移动设备宽度的新网页,转化后的网页清晰无溢出并可以较大幅度地维持新页面与原页面风格一致;在自动转化过程中,可以较少地丢失页面信息并较大幅度地保证原页面的功能;转化后的新页面相比原页面消耗数据流量较小。

**关键词** 移动 Web 应用,适应,转化服务

**中图法分类号** TP317.1 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2014.11.015

## MobiTran: A Technique of Transforming PC Web Application for Smart Phones

FANG Yi-meng MA Yun LIU Xuan-zhe HUANG Gang

(Institute of Software, School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

(Key laboratory of High Confidence Software Technologies (Peking University), Ministry of Education, Beijing 100871, China)

**Abstract** With the development of Internet, a large number of mobile devices begin to get access to Internet. However, Web applications are mostly designed for the PC screen size. The screen size of mobile devices is small relative to the PC. Therefore, when a mobile device gets access to the application for PC, there may be incomplete information display, interface layout confusion, poor user experience, more network traffic and other issues. Developing a mobile version of the Web application from scratch is the most naive choice, but it costs a lot, and maintaining two different versions is a daunting task. So we tried to adapt the view of the Web application instead of making a new one. Therefore, this paper implemented a developer-oriented Web application conversion tool, called MobiTran. The PC version of the Web application can be transformed into mobile version to fit the screen size of mobile devices, other features and data traffic. It also allows developers to edit the style and layout of the user interface, and ultimately a mobile version of this Web application is generated. After testing on mainstream sites, we concluded that MobiTran can adapt the view of Web application while maintaining the style, and it can lose less information and save data traffic.

**Keywords** Web applications, Adaptation, Conversion service

## 1 引言

中国互联网络信息中心 2013 年 1 月发布的《第 31 次中国互联网络发展状况统计报表》显示:截至 2012 年 12 月底,我国网民规模达到 5.6 亿。而在所有上网终端设备(包括台式机、笔记本电脑、手机)当中,手机已经成为了第一大上网终端,使用手机上网的网民数量占整体网民数量的 74.5%。智能

手机的发展使得手机功能越来越强大,第三代移动通信技术的普及也会为手机提供更高速的数据传输服务;其他移动计算设备,比如平板电脑的发展速度也极为迅速,Wi-Fi、3G 和 LTE 的普及将会使得移动计算设备更加广泛和方便地接入互联网。移动互联网正在成为近年来信息技术发展的热点。

移动互联网无疑为人们的生活和工作带来了更多的便利,人们可以通过移动设备来访问和使用各类 Web 应用和服

到稿日期:2013-09-16 返修日期:2013-11-16 本文受国家重点基础研究发展规划 973 项目(2009CB320703),国家高技术研究发展计划 863 项目(2011AA01A202),国家自然科学基金(61121063,61003010,U1201252)资助。

方译萌(1991—),女,硕士生,主要研究方向为服务计算、Web 前端工程,E-mail:fangym13@sei.pku.edu.cn;马 鄂(1989—),男,博士生,主要研究方向为移动计算、服务计算、Web 运行时模型;刘讓哲(1980—),男,博士,副教授,主要研究方向为服务计算、Web 工程和移动计算等;黄 罡(1975—),男,博士,教授,主要研究方向为软件中间件、操作系统和软件工程。

务。但是同时也带来了一些新的问题。

首先,已有 Web 应用多数是针对个人计算机屏幕大小设计的,而相对于个人计算机屏幕,移动计算设备屏幕具有尺寸小、分辨率低的特点。现在主流手机屏幕大小仍为 3.5~5 英寸,主流平板电脑屏幕大小为 7~9 英寸,而个人计算机屏幕大小则为 13 英寸以上。因此,很多移动设备无法完整清晰地显示针对 PC 屏幕尺寸设计的 Web 应用内容,必须放大、拖拽页面才能浏览,用户体验变差。

其次,移动设备的操作方式主要为手指触摸,而非通过鼠标操作,所以点击页面中小按钮会比较困难。同时,部分鼠标事件如 MouseOver 无法通过手指触发,而手指的一些事件如使用一根、两根手指滑动也没有鼠标事件与之对应。

最后,移动设备需要考虑流量问题。移动设备接入网络的方式有 3G 和 Wi-Fi 两种。这两种方式相比于有线网络传输速度都较慢。特别是在 3G 模式下,流量增加不仅仅是降低浏览速度,更意味着增加金钱开销。

针对上述问题,最直接的解决方案是重新开发一个面向移动设备的全新 Web 应用。例如,基于流行的响应式用户界面(Responsive UI)设计方式,Web 应用的拥有者可以针对移动设备的屏幕尺寸、操作方式和通讯方式,重新开发一个与原有应用对应的移动版本应用。尽管这种实现方式的效果最好,但是这种实现方式有两个问题:一方面,重新开发移动版 Web 应用对于个人或中小型企业而言,需要重新编写一套 HTML、CSS、JavaScript<sup>[7]</sup> 代码,开发成本和代价较大;另一方面,PC 版和移动版两个架构不同的应用版本也带来了更高的维护成本。当需要修改应用内容时,必须同时修改 PC 版和移动版两个不同的应用,这无疑会增大 Web 应用的维护成本。

由此可见,尽管重新开发一个移动版本的应用会使用户得到最好的体验,但是开发成本、维护成本都较大;而全自动转化效果不好。针对上述问题,本文提出一个面向开发者的移动版 Web 应用转化工具 MobiTran<sup>1)</sup>。首先,根据用户输入的 URL,自动抓取并转换页面,使 Web 应用初步适应移动设备的显示屏幕同时较大幅度地保证原页面的功能,以满足基本需求,由自动转化部分完成大部分转化工作;然后,Web 应用开发者手动调整页面,可以增加、删除、修改页面中的元素,增加与本地应用的结合,将页面切分为多页以适应流量,手动调整将以所见即所得的方式进行,不会给开发者带来较大负担。

本文第 2 节介绍相关技术发展过程以及学术界和工业界的研究实践;第 3 节介绍方法概述;第 4 节介绍系统实现;第 5 节介绍验证评估;最后总结本文并提出未来工作展望。

## 2 相关工作

从实现方式上来看,PC 版 Web 应用在移动设备上的适应主要由两种技术手段实现,即客户端实现和服务器端实现。最初,人们认为 PC 版 Web 应用在移动设备上用户体验不好,主要是由于移动设备屏幕尺寸小,因此人们认为 Web 应用移动版本转化等于适应移动设备的屏幕尺寸。基于这种目标,此时的转化工具是面向最终用户的。而由于面向最终用户,

因此工具是全自动的,主要在客户端实现。逐渐地,人们发现,由于移动设备存在新的特性,如操作方式不同、本地应用不同和存在数据流量限制,仅仅将原 PC 版 Web 应用适应移动设备的屏幕尺寸是不够的。此时,Web 应用移动版本转化等于适应屏幕、适应操作、与本地应用结合及适应流量。为了支持移动版 Web 应用中的这些新特性,此时的转化工具是面向开发者的,采用自动转化和手工修改相结合的方式。由于需要人工参与,因此转化工具需要在服务器端实现。

围绕 Web 应用的移动版本转化,学术界和工业界均开展了大量的研究实践。例如,Chen 等人通过分析页面结构并将其分为多个逻辑上相关的子页面完成 Web 应用移动版本转化<sup>[5]</sup>。转化工作在浏览器中完成,转化后的页面呈“目录、子页面”两级结构,此工作是面向最终用户的。Kazuki 等人则提供了一个 Web 应用移动版转化服务。该服务支持开发者从页面中抽取页面模型并分割为多个子页面,进而进行导航栏推荐<sup>[6]</sup>,此工作是面向开发者的。

而在工业界中,面向最终用户的 Web 应用移动版本转化工具主要包括 UC 浏览器适应屏幕、Sm@rtAdaptor<sup>[4]</sup>、百度转码。面向开发者的 Web 应用移动版本转化工具主要包括谷歌 GoMo<sup>[11]</sup>、百度 SiteApp<sup>[3]</sup>。

UC 浏览器采用自定义浏览器内核,自带智能适应屏幕功能。当开启功能后,UC 浏览器会将当前网页按照适应屏幕方式显示,页面中元素无论大小均能适应当前屏幕,同时字体足够大,页面内容显示完整并且不会重叠,避免了用户放大缩小、左右拖拽屏幕所带来的麻烦<sup>[1,2]</sup>。

Sm@rtAdaptor<sup>[4]</sup> 通过在网页加载完毕后执行一段 JavaScript 代码完成页面自适应。在 JavaScript 代码中,会对网页的 DOM 结构进行分析,将整个页面分为几个适合移动设备屏幕显示的部分,并生成导航栏,以方便用户在不同部分之间切换。Sm@rtAdaptor 将整个页面视为 DOM 树,将适应后的各个部分视为原 DOM 树的子树,分别视为整体,不支持对个别 DOM 元素的修改。基于这种设计,适应后的各个部分最大限度地维持了原网页中的形态,但是对于原网页中宽度较宽的部分,适应后宽度仍较宽,会出现溢出的情况。Sm@rtAdaptor 通过在网页加载完毕后执行一段 JavaScript 代码完成页面自适应,将整体 DOM 树分割为几棵子树,对局部 DOM 树不做修改。适应工作在客户端实现。

以上两种工具转化过程全自动,不需要开发者参与,然而转化效果不好,并且无法解决流量问题。

百度 SiteApp<sup>[3]</sup>会根据网站结构,自动地对内容进行结构化提取及展示,并支持开发者进行个性化效果定制。通过重新生成一组代码,达到 Web 页面适应屏幕的效果,支持开发者修改页面中的元素,并支持添加拨打电话之类的增值服务至 Web 应用,适应工作在服务器端实现。

GoMo 是谷歌提供的一个自动创建移动版 Web 应用的服务,目前由 DudaMobile 公司提供技术支持。当输入 Web 应用 URL 后,GoMo 会生成一组新的代码,这组代码即为转化后的移动版本 Web 应用。同时,GoMo 包含强大的 Web 应用页面编辑器,可以通过拖拽增加或删除页面元素来修改 HTML 代码和 CSS 代码。它通过重新生成一组代码,达到

<sup>1)</sup> 可在 <http://www.internetworkare.info/mobitrans> 下载 demo

Web 页面适应屏幕的效果,支持开发者修改页面中的元素,并支持添加拨打电话之类的增值服务至 Web 应用,适应工作在服务器端实现。

以上两种工具转化过程为半自动,需要开发者参与,转化效果较好。然而无法保证转化后页面的 JavaScript 代码运行、表单提交。

### 3 方法概述

根据上述相关工作,我们发现,面向最终用户的工具效果相对较差:响应速度较慢,不能适应移动设备的操作方式,不能与本地应用结合,不节省流量;而面向开发者的工具均可以克服这些缺点。然而,目前各种面向开发者的工具均有不完善之处,例如 Kazuki 等人的工具<sup>[6]</sup>不具有自动转化的功能,这会导致开发者负担过重;百度 SiteApp 和谷歌 GoMo 都不具备将页面切分为多页的功能,同时无法保证转化后页面的 JavaScript 代码运行、表单提交。

因此,本文选择了实现面向开发者的移动版 Web 应用转化工具 MobiTran。MobiTran 是一个在线工具服务。首先,根据用户输入的 URL,自动抓取并转换页面,使 Web 应用初步适应移动设备的显示屏幕同时较大限度地保证原页面的功能,以满足基本需求,由自动转化部分完成大部分转化工作;然后,Web 应用开发者手动调整页面,可以增加、删除、修改页面中的元素,增加与本地应用的结合,将页面切分为多页以适应流量,手动调整将以所见即所得的方式进行,不会给开发者带来较大负担。

MobiTran 设计主要包括如下几个方面。

(1)系统需将 PC 版 Web 应用自动转化为移动版本。系统整体目标是半自动地完成 Web 应用移动版转化,为了减轻开发者手工修改的负担,系统需要首先将 PC 版 Web 应用自动转化为移动版本。自动转化部分主要完成页面适应,而页面样式主要由 CSS 控制,部分 DOM(Document Object Model)元素也含有样式,所以需要 CSS 和部分 DOM 元素进行重写。我们尽量维持 DOM 树的原状,较大限度地保证原页面的功能。在系统所在的服务器上,维护了 PC 版 Web 应用对应的移动版应用,当最终用户通过手机访问 PC 版 Web 应用首页时,会重定向到系统所在服务器的移动版 Web 应用首页。点击移动版 Web 应用页面中的链接时,跳转的页面仍为移动版本,这就需要将页面中的链接指向对应的移动版页面。

(2)转化后的移动版 Web 应用应保持原 PC 版 Web 应用的功能。网页一个非常重要的功能就是表单提交,由于我们采取的策略是对网页进行重构,因此在我们的服务器上需要保存一套对应页面,当点击移动版本页面的某一连接时,跳转到的页面仍为移动版本。然而对于表单提交等操作,需要定向到原始服务器,才能保证功能实现。

(3)系统需要为开发者提供所见即所得的切分多页功能。全自动完成这项工作是现实的,而全手工实现时,用户需要重写 HTML、JavaScript、CSS 代码,工作量较大,所以在工具中,我们提供一个用户友好的操作环境,使得开发者可以通过拖拽,完成对页面的切分。

(4)开发者进行编辑后,当原始 PC 版应用出现内容上的

变化时,系统可以将新内容映射到移动版 Web 应用对应位置;当原始 PC 版应用出现结构上的变化时,系统可以提醒开发者重新手工编辑移动版应用。

本文主要介绍前 3 个问题的解决方案,对于动态内容的处理将在未来工作中进一步完善。

根据对系统所要解决的问题进行实际需求分析,我们得到系统流程图如图 1 所示。

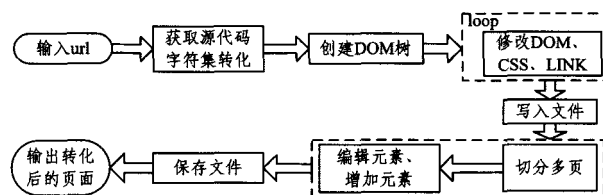


图 1 系统流程图

开发者使用工具时,首先输入 URL,之后工具就会开始自动转化:工具首先获取页面源代码并进行字符集转换,之后创建 DOM 树,对于每个 DOM 节点依次修改 DOM 类型、CSS 属性、链接属性值。在完成自动转换后,写入文件,此时开发者可以进行手工编辑。在手工编辑阶段,开发者可以将页面切分为多页,可以增加元素,编辑元素,最终保存文件,完成 Web 应用的移动版转化,输出转化后的页面。

### 4 系统实现

根据上述分析,我们设计 MobiTran 系统的架构图,如图 2 所示。

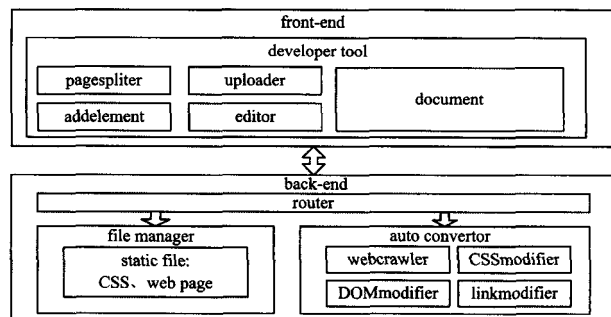


图 2 系统架构图

如系统架构图所示,系统分为前端、后端两个部分。前端是和开发者交互的部分,主要完成手动修改部分。后端是服务器部分,主要完成自动转换、读写文件部分。

前端使用 jQuery UI<sup>[10]</sup>实现。在前端中,document 是开发者进行手工修改时 iFrame 里面页面的 document。document 的 element 上绑定了一系列事件,如 mouseover 绑定了 DOM 元素背景色改变的事件,用于表示选中某个 DOM 元素。pageSplitter 是切分多页模块。该模块会调用 document,用于选中某些页面中的信息。Uploader 是保存模块。该模块会调用 document,获得页面信息。add element 模块是增加元素模块,被 document 模块调用,用于向 document 中添加新元素,如拨打电话按钮。editor 模块是编辑页面模块,与 element 模块发生交互,用于编辑页面中元素。

后端使用 Node.js<sup>[8,9]</sup>实现。在后端中,router 是路由模块,调用 auto convertor 和 file manager 处理前端请求。auto convertor 模块是自动转化模块,负责根据 URL 请求页面并



将页面转化为移动版本。file manager 负责读写文件。

下面将对系统架构图中的主要模块进行介绍。

#### 4.1 路由控制

路由模块主要处理 3 种请求:自动转化请求、手工修改后保存请求、页面访问请求。

开发者使用工具时,首先输入 URL,此时就会向服务器发出自动转化请求。路由模块收到请求后,调用自动转化模块进行处理。

开发者手工修改后,会向服务器发出保存请求。路由模块收到请求后,调用文件管理模块进行文件操作,完成保存功能。

最终用户访问移动版应用时,会向服务器发出页面访问请求。此时服务器先检查是否有已被开发者编辑过的移动版页面,如果存在,则将其返回最终用户;如果不存在,则进行自动转化,并将自动转化后的页面返回给最终用户。

#### 4.2 自动转化

Auto Convertor 模块是 PC 版 Web 网页的自动转换模块,负责根据 URL 请求页面并将页面转化为移动版本。主要算法为 autoConv,调用 crawler, CSS modifier, DOM modifier, link modifier 模块,如代码 1 所示。

##### 代码 1 网页自动转化

autoConv//自动转化模块

get sourcecode//根据 url 请求页面

iconv//对页面进行字符集转化

build DOM tree//创建 DOM 树

travel in head tree//对 head 修改连接

modify link

add js&-css//加载 js 和 css

travel in body tree//对 body 的每个结点,修改 dom,css,连接

modify dom

modify css

modify link

CSS modifier 模块中的 modifyCSS,主要是将不适合的 CSS 属性去掉,并设置适合移动设备显示的 CSS。由于需要对元素统一设定,因此 html 文档中设定的页面样式会出现冗余,具体属性如下: width、height、cssFloat、position、text-align、align、valign。将这些冗余属性从 DOM 属性中删除即可。另外,为了显示清晰,需要设定对所有元素通用的 CSS,这些代码写在外部的 CSS 文件中,通过加载作用于页面。需要设定的属性有: max-width、min-width、width、height、float、left、top、position、margin、padding、word-break、text-indent、white-space。由于 DOM 元素中 CSS 属性优先级最高,因此在外部的 CSS 中需要加 !important 关键字才能使优先级最高。另外,对于 li,我们不希望它横向溢出,所以设置 display 为 block。

DOM modifier 模块的 modifyDOM 思想为将不适合的 DOM 元素替换掉,通过对所有标签进行分析,我们发现,td、tr、table、tbody 意图在页面中建立表格,标签中仅包含样式的含义。为了使页面适应屏幕,我们将上述标签转换为 div。这样会导致一些通过标签选择元素的 JavaScript 代码和 CSS 代码不可用,如 getElementByTagName(),再如 CSS 选择器 td。我们可以通过设立元素 id,避免通过标签类型访问元素,使该问题得到解决。

#### 4.3 手工编辑

Document 是开发者进行自动转换或手工修改时 iFrame

里页面的 document。为了支持手工修改,需要在 Document 的 DOM 元素上绑定一系列事件。为了选中元素,并使其可拖拽,对 DOM 元素的 mouseover 和 mouseleave 事件绑定事件处理程序。为了删除元素,对 DOM 元素的 dblclick 事件绑定事件处理程序。为了修改元素,对 DOM 元素的 click 事件绑定事件处理程序。由于 dblclick 事件触发时会触发两次 click 事件,为了对二者加以区分,设计 click 队列。每次触发 click 事件后都会向队列插入元素,该元素在 500ms 后自动执行编辑元素函数。当触发 dblclick 事件时,清空 click 队列,防止执行自动编辑函数。为了接受元素,对 DOM 元素的 drop、dragover 事件绑定事件处理程序。

add element 模块用于向移动版页面添加元素,具体表现为工具页面的一个支持拖动的元素。

editor 模块的功能是显示选中 DOM 元素的代码,支持用户修改并同步到 document 模块中。

page splitter 模块提供切分为多页的功能,包括工具界面中的部分和操作部分。当切分多页时,首先按下增加页,可以按照需求拖拽元素至新页面。

uploader 模块负责将 iFrame 中的代码取出,并提交给服务器。

#### 4.4 用户界面

根据系统的设计,系统共需两个界面:等待转换界面和手工修改界面。等待转化界面至少需要一个输入框和一个按钮,用于输入待转换的 URL 和用于提交请求。手工修改界面至少需要一个 iFrame 用于显示新生成的网页、一个编辑 HTML 代码的面板、一个切分多页面板、一个增加元素面板以及一个保存按钮。

图 3 为等待转化界面,开发者在输入框中输入 URL,点击按钮,工具就会进行自动转化并跳转到手工编辑界面。图 4 为编辑 HTML 代码面板。开发者鼠标移动到 iFrame 中某元素上并单击鼠标左键后,会打开编辑面板;在编辑面板编辑并确认后,编辑结果会显示在 iFrame 中。图 5 为切分多页面板。开发者点击创建新页按钮后,会打开此面板。在此面板中,开发者可以输入页面标题、拖拽 iFrame 中元素到新页面中。



图 3 等待转化界面

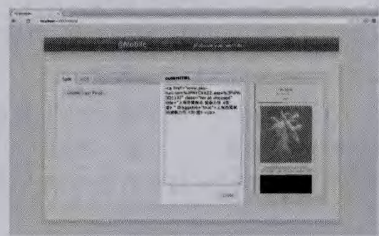


图 4 编辑 HTML 代码面板



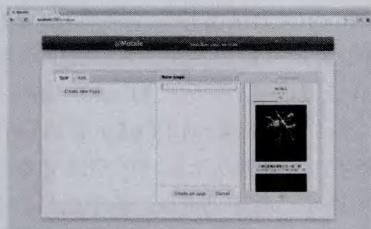


图5 切分多页面版

## 5 验证评估

### 5.1 实验设计

为了验证移动版 Web 应用转化工具,我们对 Alexa<sup>1)</sup> 网站的中国排名前 50 的网站进行了实验。从转化效果和流量数据两方面入手对我们的工具 MobiTran 进行评价。

由于转化结果是个相对主观且比较定性的概念,因此我们设计了如下指标加以衡量。首先页面风格是否跟原来相似,这里我们比较了 CSS 的来源。我们认为,原页面 CSS 所占比例越大代表修改后的页面风格与原页面越相似。之后我们比较了转化后是否出现页面信息缺失。我们认为页面信息存在于有体积的 DOM 结点,所以比较了缺失率和缺失超过 50% 的页面比例。我们使用一段 JavaScript 代码来计算叶结点。之后比较了页面元素是否有重叠,即两个元素重叠在一起使得页面信息混乱;是否有横向溢出,即修改后的页面宽度大于移动设备宽度。基于以上指标,还对 Sm@rtAdaptor 和 GoMo 进行了比较。

数据流量方面,我们使用 MobiTran 对新浪首页进行转换,将其切分为多个子页面,并与原始未转化的页面进行比较,载入首页的数据流量变化。

### 5.2 实验结果

#### 5.2.1 转化效果结果及分析

实验结果如表 1 所列。

表1 转化效果结果

	Sm@rtAdaptor	MobiTran	GoMo
CSS 来源	原网页 CSS	原网页 CSS 516B 新 CSS	新 CSS
原页面有体积 DOM 叶结点缺失率	0	9.2%	45.8%
原页面 DOM 叶结点缺失率超过 50% 的页面比例	0	0	38.1%
页面元素重叠	无	有背景重叠	无
页面横向溢出	115.19%	0	0

从实验结果可以看出,Sm@rtAdaptor 只使用了原网页的 CSS,GoMo 采用了全新的 CSS。而我们的 MobiTran 基本采用了原网页的 CSS。所以对于页面风格而言,Sm@rtAdaptor 与原页面相同,MobiTran 与原页面风格非常相似,GoMo 与原页面风格差别较大。与原页面相似与否无孰优孰劣,与原页面较为相似时,最终用户浏览更为方便;而如采用移动化的设计风格,则会与原页面风格差别较大。

从 DOM 叶结点缺失率可以发现,GoMo 页面信息缺失较多,直接观察转化结果也印证了这一结论。当使用 GoMo 对携程网<sup>2)</sup> 进行转化后,GoMo 将语言栏识别为导航栏,导致

页面中其他内容丢失。

由于 MobiTran 使用了原网页的 CSS,同时对页面布局进行了修改,因此出现了背景重叠的状况。最后,MobiTran 和 GoMo 都没有横向溢出,相比 Sm@rtAdaptor 而言有了进步。

#### 5.2.2 转化效果比较

图 6 和图 7 是以新浪首页为例的转化效果。可以看出,MobiTran 转化后的页面维持了新浪的原始风格,字体、段落格式都和原页面相似,原页面中高亮的段落仍然高亮,而 GoMo 转化后的页面风格则有较大变化,高亮部分被取消。

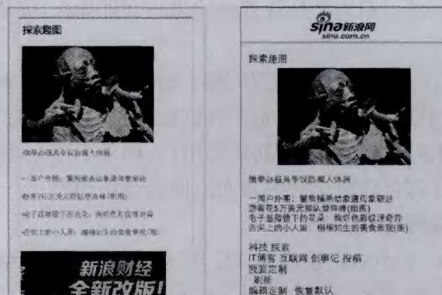


图6 左 MobiTran 右 GoMo

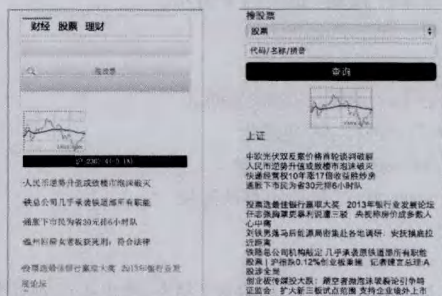


图7 左 MobiTran 右 GoMo

#### 5.2.3 数据流量比较

如表 2 所列,以新浪为例切多页后,将原 710kB 的网页切分为 7 个 100 多 kB 的网页,共计 748kB。以前浏览新浪首页时,需要加载 710kB 的页面和 3.4MB 的文件。而此时浏览新浪首页时,最初仅加载 100kB 的页面和 520kB,减少了 85.1% 的流量。当要浏览原页面全部内容时,仅增加 5.4% 的流量,这是可承受的。

通过对网页的切分,在某些情况下,我们可以大幅地减少用户的流量开销。例如,当用户希望通过新浪首页中新浪微博的连接跳转到新浪微博时,用户不得不加载新浪首页的全部信息,消耗巨大的流量。而转化之后,则可为用户减少 85.1% 的流量,这不仅提高了浏览速度,而且降低了金钱开销。

表2 数据流量比较

单位(kB)	Files	HTML	total
原始页面	3449	710	4159
切分后首页	520	100	620
切分后首页变化百分比			-85.1%
切分后总页面	3449	100+163+167+128+137+128+111=934	4383
切分后总页面变化百分比			5.4%

(下转第 87 页)

<sup>1)</sup> www. alexa. com

<sup>2)</sup> www. ctrip. com

- [7] Cambroner M-E, Valero V, Diaz G, et al. Verification of real-time systems design [J]. Software Testing, Verification & Reliability, 2010, 20(1): 3-37
- [8] Frits Vaandrager. A First Introduction to UPPAAL [R/OL]. [2013-07-01]. <http://www.mbsd.cs.ru.nl/publications/papers/fvaan/uppaaltutorial.pdf>
- [9] Behrmann G, David A, Larson K G, et al. A Tutorial on UPPAAL 4.0 [R/OL]. [2013-07-01]. <http://www.it.uu.se/research/group/darts/papers/texts/new-tutorial.pdf>
- [10] Ohn H, Marburger J H, Kvamme E F, et al. Leadership Under

Challenge: Information Technology R&D in a Competitive World [R/OL]. [2013-07-01]. [http://ostp.gov/pdf/nitrd\\_review.pdf](http://ostp.gov/pdf/nitrd_review.pdf)

- [11] Thacker R A, Jones K R, Myers C J, et al. Automatic Abstraction for Verification of Cyber-Physical Systems [C]// Proceedings of the 1st IEEE International Conference on Cyber-Physical Systems, New York: ACM, 2010: 12-21
- [12] Akshay R, Shangwen C, Schmerl B R, et al. An Architectural Approach to the Design and Analysis of Cyber-Physical Systems [J]. Electronic Communications of the EASST, 2009, 21

(上接第 78 页)

### 5.3 其他实验结果分析

实验结果中,出现页面白底白字的情况。通过分析页面源代码,发现原页面使用了不适合在移动设备上显示的 DOM 元素,而新页面将元素替换成适合在移动设备上显示的 DOM 元素。这导致原来对于该元素的 CSS 选择器失效。这一问题可以通过修改 CSS 文件解决。

实验结果中,也出现了某些相对路径未被替换为绝对路径的情况。通过分析页面源代码,发现这些元素是由 JavaScript 进行加载的,即当页面加载完成后才由 JavaScript 引入,所以工具未对这些相对路径进行转化。这一问题可以通过修改 JavaScript 文件中的相对路径解决。

### 5.4 实验结果分析

根据实验结果, MobiTran 的优势主要体现在以下方面:

(1) 可以将原网页自动转化为适合移动设备宽度的新网页,清晰而无溢出。

(2) 可以较大幅度地维持新页面与原页面风格一致。

(3) 在自动转化过程中可以较少地丢失页面信息。

(4) 转化后的新页面相比原页面消耗数据流量较小。

根据实验结果,仍须完善的部分主要体现在以下方面:

(1) 转化后的页面会出现元素与背景重叠。

(2) 部分类型的 DOM 元素转换效果不好。

**结束语** 本文的主要工作包括:

(1) 介绍应用背景及相关技术工具的选择,比较面向最终用户和面向开发者的 Web 应用移动版本转化工具的特点,分析了本文选择实现面向开发者的 Web 应用移动版本转化工具的原因。

(2) 实现了面向开发者的 Web 应用移动版本转化工具并将其部署为在线应用转化服务, Web 应用开发者只需上传应用的 URL, 该服务即可将其半自动化地转化为移动版本,并发布新的 URL 供移动设备访问。

(3) 使用该服务对一些流行网站进行实例验证,并与现有的工具进行比较。

本文实现了面向开发者的移动版 Web 应用转化工具,未来工作主要集中于完善本工具:

当前工具仅支持最基本的自动转换。开发者进行编辑

后,当原始 PC 版应用出现内容上的变化时,系统应可以将新内容映射到移动版 Web 应用对应位置;当原始 PC 版应用出现结构上的变化时,系统应可以提醒开发者重新手工编辑移动版应用。未来工作在自动转化和手工编辑阶段并不仅限于修改当前页面,而是记录一种映射规则,以此应对页面变化。

当前工具仅支持最基本的自动转换。未来工作可以对某种特定类型的网站给出一种自动转换模板,并支持内容自动匹配。

当前工具仅支持最基本的编辑。对于某一 Web 应用,会有很多内容页结构相同,内容不同。此时编辑某一页面,仅会对当前页面进行修改。未来工作可以给开发者更多选项,支持开发者选择仅修改当前页面,或修改本类页面。

### 参 考 文 献

- [1] UC 浏览器功能介绍[OL]. UC 浏览器官方网站. <http://www.uc.cn/browser/introduce.shtml>
- [2] 杨雪涛. 五大特色解析 UC 浏览器 8.0 安卓版评测[OL]. 手机中国. [2011-9-15]. <http://www.cnmo.com/soft/110022.html>
- [3] 百度 SiteApp[OL]. 百度. <http://siteapp.baidu.com/static/document.html>
- [4] Huang G, Wang D. Adapting user interface of service-oriented rich client to mobile phones[C]// 2011 IEEE 6th International Symposium on Service Oriented System Engineering (SOSE). IEEE, 2011: 140-145
- [5] Chen Y, Xie X, Ma W Y, et al. Adapting web pages for small-screen devices[J]. Internet Computing, IEEE, 2005, 9(1): 50-56
- [6] Nishiura K, Maezawa Y, Ishikawa F, et al. Supporting view transition design of smartphone applications using Web templates [C]// Proceedings of the 12th International Conference on Web Engineering. Springer Berlin Heidelberg, 2012: 323-331
- [7] JavaScript[OL]. WiKiPedia. <http://zh.wikipedia.org/wiki/JavaScript>
- [8] Node.js[OL]. WiKiPedia. <http://zh.wikipedia.org/wiki/Node.js>
- [9] BYvoid. Node.js 开发指南[M]. 北京: 人民邮电出版社, 2012
- [10] jQuery UI. 百度百科. <http://baike.baidu.com/view/2998196.htm>
- [11] GoMo[OL]. <http://gomo.dudamobile.com/>