基于 XML 的用户界面描述语言分析

A Review and Analysis of XML – Based User Interface Language

吴根祥 （浙江绍兴文理学院计算机系 312000）

摘要：本文对当前正在发展中的基于 XML 的用户界面描述语言进行了研究，并对其中具代表性的 XAML、XUL、XMXL 进行了比较分析，指出了使用界面描述语言对网络应用软件开发的重要性，并就今后的发展趋势及应用前景进行展望。

关键词：用户界面 XML XAML XUL XMXL

1 基于 B/S 架构的应用软件开发模式中存在的问题

在 B/S 架构中，我们一直面临着这样一个问题：如何做到 C/S 那样的表示层精确控制以及良好的用户体验？在基于 HTML 的、以页面为中心的 Web 应用中，把应用程序的表示层建立于 HTML 页面之上，不可避免的页面刷新和频繁的服务器请求既影响系统的性能，也不能满足系统使用者更高的体验要求。当然，利用 DHTML 以及 JavaScript 我们可以实现接近 C/S 时代的表示层控制和展现，但需付出更多的人力物力，这又不符合系统开发者的要求。当单纯的基于 HTML 的 Thin Client/Rich Server 方式不能很好的解决上述问题时，诸如 FLEX（Macromedia 出品的表示层解决方案）等新技术的出现就显得极其自然了。

在这些新技术中，全面提升用户界面及其表现方式是一个重要的内容。与 HTML 只给用户提供了非常有限的界面控制元素相比，新技术提供了灵活多样的界面控制元素及其展现方式，而且这些控制元素可以很好的与数据模型相结合。采用新的用户界面技术，可以从以前的服务器响应影响整个界面，转移到只有收到请求的应用程序部分才会做出相应的变化，这本质上意味着界面被分解成许多独立的模块，并根据用户的操作做出相应的反应。

当 XML 作为一种数据描述、交换的标准格式被广为接受时，选择 XML 作为应用界面描述的语言格式，是各技术组织的一致共识。

2 基于 XML 的用户界面描述语言的优点

基于 XML 的用户界面描述语言，将界面元素（例如对话框，菜单，工具条等等）定义为 XML 标签（Tag），可方便地将界面元素以及相关信息写入 XML 文件中，此 XML 文件即定义了相应的用户界面。显然，基于 XML 的格式文档易于编辑，当然还可由此建立可视化界面设计工具。更为重要的是，此类文档可在程序运行时动态载入或修改，从而赋予了应用程序更为强大的界面处理能力。

对于应用开发者而言，采用基于 XML 的用户界面描述语言，可以获得如下的好处：

（1）分离表示与应用程序逻辑，以实现最大程度地提高开发人员的生产率及应用程序的重复使用率。许多 Web 应用失败的一个重要原因是用户界面元素和客户端应用逻辑的紧密耦合，在团队开发环境中，开发者必须同时具备界面设计及应用逻辑设计的技巧，这往往很难做到。使用基于 XML 的用户界面描述语言分离表示和应用逻辑的问题，使得程序员、界面设计人员可独立的进行各自的工作，相对于开发基于 HTML 的 web 应用，开发人员的工作更易于重组，且对系统的质量和稳定性影响较小。

（2）使得用户界面程序可以跨平台运行。基于 XML 的用户界面描述语言提供了用户界面组件的抽象描述，只要平台有支持该语言运行的环境，可实现“一次编写，到处运行”，从而可降低开发的成本。

（3）可以动态地改变资源文件或结合其他标准（如 CSS），从而改变应用程序的风格和外观。该项特征同样得益于表示层和应用逻辑的分离，开发者仅需维护主要的应用程序代码，通过定制不同的“Skin”，实现应用程序风格和外观的改变。

（4）使方便地结合其他标准从而为实现 MUI（Multimode User Interface）提供一个统一的界面描述方式。

当前关于基于 XML 的用户界面描述语言的研究及应用，展示了一种全新的应用程序组织方式，这将对应用程序的开发模式带来近乎革命性的变革。在已发表的基于 XML 的用户界面描述语言中，受到广泛关注的有：XUL（XML User Interface Language，Mozilla）、XAML（XML Application Markup Language，Microsoft）、MXML（Flex markup language，Macromedia）。

©《计算机系统应用》编辑部 http://www.c-s-a.org.cn

3 主要技术的比较分析

3.1 XUL(XML User Interface Language,Mozilla)

XUL 由 Mozilla 基金会创建,其最初目的是为了快速开发 Mozilla 项目的界面,目前 Mozilla Suit 的界面都是由 XUL 实现的。

XUL 定义了丰富的用户界面元素及布局方式;界面元素的外观(颜色,字体等等)可以通过属性设置以及应用 CSS、图片进行定制;元素的行为则通过脚本 JavaScript 定义,代码可直接放在 Script 元素中,也可通过 Script 元素的 ref 属性导入。

通过 Overlays(覆盖)这一技术,可将描述额外内容的 XUL 文件加载至主 XUL 文件中,可实现 XUL 应用界面的动态扩展和定制,同时也降低了 XUL 应用维护的难度和成本。使用这一技术的另一重要作用是,开发者可扩充已有 XUL 应用的功能而不必修改其源代码,这样开发者可以保持对所做工作的所有权。

通过 XBL(eXtensible Bindings Language),开发者可整合组件(可以来自其它规范或 XML 文档,如 XHTML)、行为、事件、属性和风格来构成新组件,以满足特定的交互目的。这种定制通过 XBL 的 binding 标签来表示,该标签的 id 属性值即为新组件的名称。通过在 CSS 文档中定义引用此组件的类,即可在 XUL 文档中通过设置既定 XUL 组件的 class 属性值为新组件的名称来使用新组件。这种方式避免了标签集的过度扩张,也可保持宿主文档 DOM 树清晰简洁(详见 XBL 文档)。XBL 已递交给 W3C,当前状态为 W3C Note。

基于 XUL 的应用可通过三种方式运行。一是直接在 Mozilla 中运行,不需编译。在 Mozilla 中处理 XUL 文件,首先是解析 XML 文件,构建 DOM(Document Object Model)树,应用相关联的 CSS 来定义标签的外观,建立布局对象,最后将用户界面显示给用户。这个装载过程所需的时间跟所用硬件平台的性能有关,但是 Mozilla 为 XML 文件和脚本提供了一种二进制的方式,从而避免了对 XUL 文件进行不必要的多次解析。。二是通过 XPInstall(Cross Platform Install,Mozilla 的跨平台安装工具),将 XUL 应用组件生成包含安装脚本的安装文件,用户通过网络下载并执行此安装文件,将新 XUL 应用添加到本地 Mozilla 环境中,作为 Mozilla 的一部分运行。三是作为本地程序运行,但需要 XRE (XUL Runtime Environment)或 GRE (Gecko Runtime Environment)的支持。

XUL 不是单纯的用来构筑应用程序界面的,Mozilla 提供的与之相关的技术是一个整体,向用户展现了一种全新的开发网络应用的思想和方式。当然 XUL 应用的运行需要 Mozilla 的支持,或许是一个限制,但也许是仅有的一个限制,因为 Mozilla 提供多个平台的运行版本(Windows、Linux 等等),这一限制似乎微不足道。

3.2 XAML(XML Application markup language,Microsoft)

微软在 Longhorn 平台中开发了新的表现子系统(Presentation subsystem)"Avalon", Avalon 最大的技术亮点便是界面描述语言 XAML。微软的 Longhorn 尚未正式发布,因此 XAML 也许会在此期间发生比较大的改动。

XAML 同样定义了丰富的界面元素及布局方式;元素最初始的外观通过属性来定义,或者使用一个 XAML 特有的 style 元素来定义;元素的行为可以使用任何. NET 兼容的语言来定义,行为与界面元素绑定有两种方式,一种是将行为的定义(代码)附加在 XAML 文件的 CDATA 段落中,另外一种是通过 Code – Behind 方式引用。

XAML 定义的元素都是 Avalon 中定义相应类的抽象描述,基于这种方式,用户可以使用. NET 兼容的语言定义自己的组件,并在 XAML 中进行复用。实际上,任何一个具有公共的、无参数的构造函数以及可设置属性的类均可在 XAML 中使用。

XAML 描述的页(Avalon 中的概念),可以通过两种方式进行预览。一是使用 Longhorn 版的 IE 进行装载预览,但是如果使用. NET 语言为 widget 定义行为,那么不论这个 widget 是自定义的还是 XAML 预定义的,必须要对应用程序进行编译。二是对其进行编译,并像一般 windows 程序一样运行。当然通过设置编译选项可将程序编译为可在 IE 中运行。与 XUL 应用程序的运行方式相比,XAML 稍欠灵活,能否使用 script 脚本使应用可以在 IE 中运行而不需编译,目前还不明确。

从现阶段 XAML 的发展可以发现,微软避开了所有 W3C 的建议,比如 SMIL、SVG 以及 DOM,而为 XAML 实现了一些与上述建议相似但不兼容的技术。迄今为止 XAML 的运行环境限定在 longhorn 上,用户必须升级平台才能使用此技术,而且 longhorn 尚未正式发布,XAML 要成为 WEB 技术的扩展,被用户广泛接受尚需时日。

3.3 MXML(Flex markup language,Macromedia)

Flex 是 Macromedia 发布的表示层服务器,开发者可用此开发新一代的 RIA(Rich Internet Application)。Flex 应用程序框架由 MXML、ActionScript 2.0 及 Flex 类库构成。Flex 类库中包括 Flex 组件、管理器及行为等。

开发人员利用 MXML 定义应用程序用户界面,利用

©《计算机系统应用》编辑部 http://www.c-s-a.org.cn

ActionScript 2.0 定义客户逻辑与程序控制，CSS 被用来定义用户界面的风格。MXML 定义的元素都是 FLEX 中定义相应组件的抽象描述。Flex 组件非常灵活，可对组件外观、组件如何响应用户交互、组件中文本的字体与字号、应用程序中组件的大小以及很多其他特征进行控制。

利用基于 Flex 组件的开发模型，开发人员可在程序中加入预建的组件、创建新组件或是将预建的组件加入复合组件中。开发者使用 MXML 定义新组件（一般为可是组件）的方式与建立 MXML 应用一样，根据需要将预定义组件、ActionScript 代码组合至 MXML 文件中，此 MXML 文件即定义了一个新的组件，组件名与文件名一致。这种定义方式简洁直观，大大提高了代码的复用性。也可使用 ActionScript 来定制组件（一般是非可视的组件）或使用 SWC（FlashMX2004 定制）。这些扩展方式相当灵活方便，在 MXML 应用中可直接使用自定义组件名作为相应的 MXML 标签。

运行 MXML 应用的方式不同于前面介绍的两种语言，MXML 应用部署在 FLEX 服务器上，当客户端首次请求 MXML 应用时，服务器将 MXML 代码转换成 ActionScript 代码，其中每个 MXML 相当于 ActionScript 中的一个 class（类），然后将 ActionScript 代码编译成 FLASH 字节码，即一个 SWF 文件。服务器将此 SWF 文件传送到客户端，SWF 文件将客户端 Flash Player 中被执行。FLEX 服务器提供必要的缓存机制，后续对同一 MXML 应用（未经修改）的请求将跳过编译环节，以减少不必要的编译消耗。

以 SWF 作为客户端承载运行的内容，充分利用了 Flash 技术带给用户的精彩体验，也有效降低了服务器的运行负荷。同时也应重视因为当前 Flex 没有实现共享库，编译得到的 SWF 文件较大，会明显影响装载运行的速度与系统资源的使用。

4 结论与展望

通过上面的描述，可以看到三种语言的实现及其运行环境的差别是明显的。

XUL 的初始目的是为了构建 Mozilla 的用户界面以及面向 web 的应用程序，所以自然地便采用了一系列成熟的 web 技术，比如 CSS 和 JavaScript。由于 Web 开发人员对这些技术都比较熟悉，因此转向 XUL 技术是比较容易和平滑的。XUL 对 W3C 众多标准的支持，一般认为 XUL 代表 XML－GUI 发展的方向。

XMXL 是 Macromedia 的新产品 Flex 中采用的技术，简化了 RIA 的开发，选择编译成 SWF 文件的方式，目的在于充分结合其 FLASH 产品出色的用户支持度。虽然 Flex 已发布，但是其现阶段的功能和性能依然有待进一步改进，而且当前仅可部署在 Java 平台上。

XAML 采用. Net 技术，虽然有众多不确定的因素，但目的是开发全新的应用程序开发模式，充分利用原有的强大技术优势，最大限度的减少 C/S 与 B/S 的差别。问题是对 W3C 标准的支持和尚须等待的 longhorn。

可见，在所推出的新技术中，基于 XML 的用户界面描述语言是下一代网络应用程序开发环境的重要组成部分，也是应用程序开发新模式中的中心部分。围绕这一中心所集成的网络应用开发技术将在今后几年内必将得到更大的发展和应用。同时，当前基于 XML 的用户界面描述语言的发展标准并不一致，兼容性问题将长期存在。

笔者将继续关注新技术的发展并在应用新技术上进行实践。

参考文献

1 David Hyatt, XUL 1. 0 specification [EB/OL], Mozilla. Org, 2001, http://www. mozilla. org/projects/xul/xul. html.

2 Neil Deakin, XUL Tutorial [EB/OL], 2004, http://www. xulplanet. com/tutorials/xultu/.

3 Charles Petzold, Create Real Apps Using New Code and Markup Model[J], MSDN magazine, Microsoft, 2004. 1.

4 http://longhorn. msdn. microsoft. com, Microsoft, 2004.

5 http://livedocs. macromedia. com/flex/, Macromedia, 2004.

6 David Hyatt, XML Binding Language [EB/OL], mozilla. org, W3C Member Submission, January 2001.

©《计算机系统应用》编辑部 http://www.c-s-a.org.cn