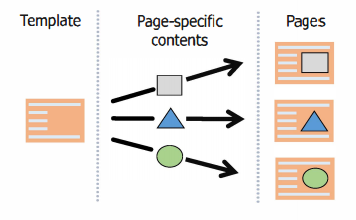
Mvc架构驱动的重构来实现客户端网页构成

摘要：本文呈现了一种将java web应用的页面构成从服务器重定位到浏览器的方式。目的是减少过多重复的操作和传输页面共享的代码或数据。减少量是动过执行重构算法产生的，从mvc架构的视角来看，有效地保证了源和目标应用的一致性，因为当前的问题是要求目标应用要保存源应用上的可观察的行为。研究案例表明，我们的重构工具能有效的支持重构的程序。

关键字：重构，mvc，单页应用，基于模板的网页应用，软件引擎，软件引擎。

1. **介绍**

在一个网页应用中，网页页面通常分享着一些内容，比如页头，页脚和菜单。在一块地方用一个模板去维护这些公共内容分隔页面上具体展示的内容。一个基于模板的web应用(TWA)使用这样一种模板去动态地生成页面，并将页面上具体的内容与模板进行绑定。举个例子，在图1中，这三个页面通过使用一个模板和每个页面具体的内容生成。

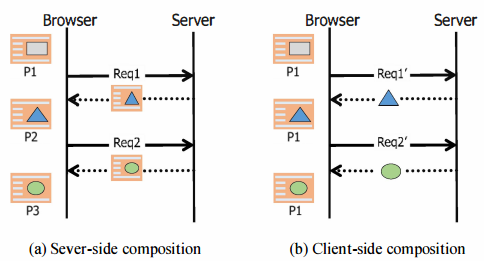


图一.基于模板的web应用

在java web应用中，<jsp:include>标准动作常用于将页面具体内容填充进模板的占位符中

当用户发出一个请求之后，一个网页浏览器下载和展现一个新的页面，TWAs 主要同过多页面应用模型与用户进行交互。如图2a展示的一样，一件最重要的记住的事情就是TWAs是对于每个请求，服务器都会结合对应的模板和页面具体内容并发送。因此对于么每个请求，模板数据被重复进行传输并用整个页面刷新进行渲染。

对于这种重复问题，其中一种解决方案是将页面构成由服务器转交给浏览器。无论用户何时点击一个超链接或是提交一个表单，网页浏览器接收到当前页面和新页面之前的不同处，这时局部地更新当前页面不同之处而不同过整个页面的刷新。如图2b所示。每个请求，请求1和请求2造成一次关于有联系的差异部分的部分页面刷新。页面1是物理上的一个单独页面但不能有不同的统一资源定位地址（URLs）,取决于他的状态（即包含的内容）。这种网页应用称它为单页应用（SPAs）。



图二.web页面组件

MVC架构被普遍用于网页应用开发中。因此，当重构或重建多页面应用成一个单页面应用时，它能很自然地考虑到三个组件和他们之间的交互作为一个基本的研究和改动的单元。然而，这些架构组件还没有在大多数之前的工程中充分地描述关于转化的问题。

这个转化是主要要求在它执行前前提条件能够满足。特别是，关于重复代码和数据的减少量有时候会导致在源和目标应用上的不一致：目标应用上的可观察行为能与他的源应用行为有所不同。然而，大多数之前的关于转换问题的调查确实没有在具体的当转换手段无法应用时的环境进行过。

因此，从MVC模型的组件架构方面来说，非常有必要去确定先前条件和转换的步骤。这篇论文提出一种方式，从MVC架构的层面来说，不通过改变源应用可观察的行为并从一个完整页面的请求能重构成一个部分页面的请求。也推出一种重构算法去描述重构的步骤以及重构的前提条件。我们展示一个通过重构算法生效的重构程序的总体图片。此外，这篇论文还描述了当实现这种重构算法时我们遇到的最重要的问题。此外，我们展示了案例研究的结果，当我们的工具被使用时，重组所需的工作量减少了，web应用程序的性能得到了改善。

1. **问题和背景**
   1. **问题**

本文讨论的重构问题可以被认为是低级的重构，因为用户界面(UI)结构，页面导航模型（即两个页面之间的链接），数据模型和源应用程序的功能都没有改变。目标应用程序具有与源应用程序相同的外观和感觉，但它提高了响应时间的性能和在网络上传输的数据量，并在服务器和浏览器之间提供了异步通信功能。此外，模板和页面特定内容之间的耦合通过重构减少，因此维护（例如，一个额外的新的具体页面内容的请求）变得更加容易。选择低层次结构调整的根本原因是，将一个输入应用程序的多页面架构无缝地重构到单一的页面架构中，而不会让用户感到困惑。

我们假定输入的TWAs是使用采用MVC模式的Java Server Pages (JSP) model 2架构开发的。这个假设是合理的，因为JSP模型2架构在JSP的早期规范中被提及并且在业界得到了认可。这里有许多关于解决将一个多页面应用重构或重建成单页面应用问题的方式。但是，从MVC这个web应用程序开发的流行架构模式的角度来看，很难找到解决这个问题的研究。本文的早期版本明确地考虑过修改模型、视图、控制器及他们之间的交互。但是，这个早期研究有一些限制。重构的先决条件没有描述。算法不能自动进行重构，所以不太容易实现这个方法。在重构过程中不能显示实验结果。

在本文中，我们改进了之前的研究，并提出了一种重构算法，该算法考虑了从MVC模式的角度对输入和输出应用程序的一致性检查。此外，我们还从过程质量的角度提出了案例研究的结果。

* 1. **背景**

本段简要地描述了我们之前的工作，帮助人们更清楚地理解本论文的概念。Java web应用程序主要由JSP、servlet和JavaBean对象组成，它们分别从MVC架构的角度对应到视图、控制器和模型。web模板已被用于为用户提供一致的视图，并减少web应用程序中的代码。模板创建并管理共享组件，比如菜单。当用户请求一个页面时，请求被发送到一个servlet，该servlet执行其业务逻辑。执行的结果这时包括被发送到浏览器上的模板生成的页面。在请求处理过程中，通过servlet和视图来访问JavaBeans对象来操作业务数据。图3a展示了动态的架构处理用户请求。

该架构存在一个问题，即模板在每个用户请求上都会加载并渲染整个页面。因此，我们之前的工作提出了一种将TWA重组为客户端web页面组合来解决问题的方法。主要的想法是在第一次访问应用程序时下载模板，然后，尽可能地绕过模板下载进行后续访问。此外，用户请求从正常的超文本传输协议(HTTP)请求转换为异步JavaScript和XML (AJAX)请求，以便在浏览器中启用部分更新。在JavaScript代码的帮助下，完整的页面完全由浏览器组成，它更新当前文档对象模型(DOM)树，同时用下载的部分替换旧的页面特定部分。如图3b展示的动态架构处理用户请求。因此，输出应用程序是一个SPA。

我们之前的工作还支持向后导航和可书签化，来支持与经典web应用程序相同的方式访问重组的应用程序。，默认情况下，当用户在SPA中单击浏览器的后退按钮时，浏览器不会回到之前的SPA状态，而是回到SPA之前访问的页面(见图4a)。为了解决这个问题，我们之前的工作应用了超文本标记语言5 (HTML5)历史应用程序编程接口，它在用户请求的浏览器历史堆栈中存储当前状态，并在使用back按钮时检索状态(见图4b)。默认的，一个SPA有不同的状态，但是只有一个URL。

1. **将基于模板的应用程序重组为单页应用程序。**
   1. **重构程序**

在本小节中，我们展示了修改TWAs到SPAs的重组过程的总体图(见图5)。