

**毕业论文（设计）文献综述**

**[](http://bysj.zjgsu.edu.cn/)**

**题目：前端整合框架在OEM电子商务系统中的应用**

**学 院：电子商务与工程管理学院**

**专 业：电子商务**

**班 级：商务16A**

**学 　 号：1612200132**

**学生姓名：王勇**

**指导教师：吴云标**

**二○ 一七 年 12月**

**《前端整合框架在OEM电子商务系统中的应用》**

**文献综述**

**摘要：**在现代Web系统的开发的过程中，MVC模式中的View层交互功能越来越强大，业务逻辑也越来越复杂。传统的前端开发模式已经满足不了复杂的电子商务系统的业务需求的扩展与维护。为了使得前端应用的开发更加有效率，近些年出现了许多前端框架和自动化构建工具。本文在使用传统的方法实现电子商务系统的提前下，使用前端整合框架React.js+ Redux + Reat-Router对该系统前端页面进行模块化重构，由此比较两种开发模式的主要区别。

**关键词:**Web前端,，React.js，Redux, 单页应用，JSX，单项数据流，电子商务系统前端，前端模块化开发

**“The Application of Front - end Integration Framework in OEM E - commerce System”**

**Abstract:** In the development of modern Web system, interaction in view layer becomes more and more powerful and business logic is also more and more complex. The traditional front-end development model has been unable to meet the complex e-commerce system’s business needs expansion and maintenance. In order to make the development of front-end applications more efficient, in recent years there have been many front-end framework and automated building tools. In this paper, the traditional method is used to build the e-commerce system in advance, using the front-end integration framework React.js to modulate the front-end applications of the system, thus comparing the main differences between the two development models.

**Keywords:** Web front end; React.js; Redux; single-page-application; Single data stream; front-end reconstruction; front-end modular development

1. **引言**

在最近几年的前端领域中，优秀的框架层出不穷，使用web前端框架搭建的单页应用在不向服务器发起HTTP请求的情况下就可以实现路由的转发，极大的提升了用户的体验。其中FaceBook开源React.js引起了业界极大的关注。由于FaceBook前端业务需求越来越复杂，MVC的前端框架已经满足不了其复杂的业务场景的需要，FaceBook推出了React + Flux 来解决前端复杂的交互场景。React.js采用了分而治之的方法，把一个大应用拆分几个小组件，每个组件都会完成自己需要完成的特定的功能。React.js还采用了JSX将JavaScript代码、CSS代码和HTML代码混合在一个组件中，实现了真正的组件封装。Flux实现了更加严格的单向数据流，页面上的UI只能通过派发Action改变状态从而导致组件重新渲染，阻止了View层直接和Model直接交互。正是因为React技术栈实现了前端架构的高内聚，低耦合，国内很多大型互联网公司包括阿里巴巴利用React进行了前端的重构。

1. **传统web前端开发**
   1. **开发流程**

传统web应用的开发，将不同的种类的技术分门别类。放到不同的文件夹中，先编写HTML页面，再编写CSS代码，最后编写JavaScript 页面的交互。这样的分工并没有做到逻辑上的分治。只是单纯把不同类型的技术分开了而已

* 1. **存在的问题**

在传统的web前端运用的开发中,开发人员常常会使用jQuery在DOM节点上绑定事件，一旦用户与界面的交互变得复杂之后，事件和元素的关系会变得纠缠不堪。导致难以维护，难以扩展。而且在DOM节点绑定大量的事件很容易造成内存的泄露。

为了加强代码的复用性，在开发的过程中常常会封装大量的组件，但是组件的样式又部分在CSS文件夹中，导致了组件复用的复杂程度。而且在使用jQuery编写组件的过程中会多次操作DOM节点，严重影响性能。

View层直接和处理数据的业务逻辑交互，导致处理相同逻辑业务代码在重构代码的时候才能抽离出来，降低了代码的复用，严重影响开发的效率。

为了解决这一个现象，出现了一些前端的MVC框架比如Backbone. Js。

* 1. **前端MVC**

前端MVC结构中，并不一定有Controller, 客户端根据用户的行为修改客户端Model,客户端更新和该Model相关的View, 客户端更新和该Model送sync请求到服务器，服务器返回修改是否成功。如果不成功，返回上一个view的显示结果。

1. **MVC前端框架存在的问题**
2. **单项数据流管理数据状态**
3. **React.js特点和优势**
4. **MVC与Flux**

在MVC框架中我们把应用分成了三个部分：Model来管理数据，View层来负责渲染用户的界面，Controller来负责接收用户的响应，并根据用户的行为来改变Model，再把产生的数据结果交给View层，刷新用户界面。

但是在前端MVC框架实现中，并不一定有Controller, 客户端根据用户的行为修改客户端Model,客户端更新和该Model相关的View, 客户端更新和该Model送sync请求到服务器，服务器返回修改是否成功。如果不成功，返回上一个view的显示结果。所以前端的MVC框架并不是严格意义上遵循MVC的，常常为了开发的便捷，View层和Model层会直接进行通信。这就是导致了，View层和Model的逻辑会交错复杂。

**Fackbook采用了更加严格的单向数据流来解决传统的MVC框架带来的问题，**