计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2019,28(4):111−118 [doi: 10.15888/j.cnki.csa.006857]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

单页面技术在试验数据管理系统中的研究与应用①

方　敏, 赵　峰

(江苏徐工工程机械研究院有限公司, 徐州 221004)
通讯作者: 方　敏, E-mail: snmlws@126.com

摘　要: 工程机械行业试验数据具有格式多、数据量大、业务复杂度高等特点, 传统形式的网站构建方式在网页友好度、美观度等方面存在诸多缺陷, 并且在并发处理大量数据情况下容易发生网页加载慢、运行不稳定等情况, 严重影响企业试验业务运行效率, 造成企业成本增加. 单页面应用技术 (以下简称 SPA) 是一种全新的网站构建模式, 是目前网页开发技术的主流, 其运用模块化的编程方式, 为用户提供体验感更佳的网站, 其构建机理很大程度解决传统网站构建方式所带来的问题. 本文主要论述运用 SPA 网站架构构建适用于大型企业的试验业务管理系统, 解决企业试验数据管理面临的问题, 提高试验业务数字化能力, 提升企业试验运营效率.

关键词: 试验数据管理系统; 单页面应用; 模块-视图-控制; 加载速度

引用格式: 方敏,赵峰.单页面技术在试验数据管理系统中的研究与应用.计算机系统应用,2019,28(4):111–118. http://www.c-s-a.org.cn/1003-3254/6857.html

Research and Application of SPA Technology in Test Data Management System

FANG Min, ZHAO Feng

(Jiangsu Xugong Construction Machinery Research Institute Co. Ltd., Xuzhou 221004, China)

Abstract: The characteristics of test data in construction machinery industry are many formats, large data volume, and high business complexity. Traditional website construction has many defects in terms of website friendliness and aesthetics. In addition, slow loading and unstable running of web pages are easy to occur when processing large amounts of data concurrently. The operating efficiency of enterprise test business is seriously affected, resulting in the increase of enterprise cost. Single-page application technology (referred to as SPA) is a brand new website construction mode, which is the mainstream of current web development technology. It applies modular programming method to provide users with better website experience. Its construction mechanism solves the problems largely brought by traditional website construction methods. This paper mainly discusses the use of SPA website architecture to build a test business management system applicable to large enterprises, to solve the problems faced by enterprise test data management, to improve the digitalization ability of test business, and to improve the efficiency of enterprise test operation finally.

Key words: test data management system; single page application; MVC; loading speed

设计数据、试验数据、仿真数据构成企业的研发数据, 研发数据作为企业核心的知识资产, 其科学的管理方式对企业高效应用这些数据至关重要. 近年来, 随着计算机科学、软件工程等技术的发展, 信息化方式成为提升企业研发数据管理的最有效手段. 试验数据作为对设计数据的验证支撑, 其重要度不言而喻[1].

① 收稿时间: 2018-10-16; 修改时间: 2018-11-06, 2018-11-20; 采用时间: 2018-11-22; csa 在线出版时间: 2019-03-28

目前, 国内市场上已有很多对于试验数据管理的解决方案和软件系统. 但多数系统都是采用传统的网站构建方式进行搭建的, 其主要缺点如下[2]:

(1) 每次用户进行请求操作时, 都需要重新加载整个页面, 如果页面很大, 服务器又繁忙, 或者网络连接很慢, 页面就会出现“闪烁”现象, 可能会持续好几秒钟甚至是更长时间.

(2) 数据处理过程、验证、授权和持久存储等大多数应用逻辑都在服务器端, 服务器端压力大, 对大部分的用户输入的响应, 必须等待一个“请求/响应/重绘”的循环周期, 网页响应慢[3].

(3) 针对不同的客户端, 如平板电脑、台式计算机、智能手机等, 往往需要分别进行处理开发, 兼容性较差.

针对传统网页构建模式所带来的缺点, 本文从网页构建架构模式作为切入点, 提出基于单页面应用技术 (SPA) 架构构建管理系统的方法, 通过 MVC 框架、HTML、AngularJS 等网络信息技术的综合应用, 设计开发出类似原生态界面风格的试验数据管理系统. 从底层系统架构层面根本上改善上述描述的问题.

1 相关概念

1.1 试验数据管理系统概念

试验数据管理系统, 行业内称之为 TDM (Test Data Management)[4], 其研究的是试验业务的数字化管理技术, 主要围绕试验数据管理、试验流程管理、试验资源管理, 实现试验业务从试验发起到报告发放的信息化管理, 构建覆盖试验全流程、全数据、全资源的试验业务管理平台.

(1) 全流程管理: 覆盖试验业务全生命周期各个环节, 管理从试验发起、试验准备、试验执行、试验处理全过程中所涉及的各个业务对象. 以流程为导向, 指引试验开展过程, 支撑试验业务顺利高效执行; 以任务管理为方法, 串联试验发起人、审核人及执行人等, 实现试验全过程的实时跟踪; 以企业制度规范为约束, 管理试验相关的执行过程、数据格式、报告模板等, 标准化管理试验全过程.

(2) 全资源管理: 覆盖试验所有相关资源, 如试验人员、试验设备、试验样品、试验知识等. 以资源对象为基点, 建立各资源对象的信息台账看板, 数字化管理试验资源的基本信息、使用信息、维护保养信息、检定信息等. 运用系统结构化管理试验知识, 建立企业试验知识库, 关联相关试验信息, 最终基于系统实现企业试验知识的积累、共享[5].

(3) 全数据管理: 覆盖试验业务所涉及的各类数据, 如试验原始数据、过程数据、结果数据. 以数据集中存储为基础, 建立全数据的结构化管理, 支撑试验报告的生成; 以数据应用为目标, 结合各类分析工具, 深度挖掘数据潜在价值; 以数据多维度展示为方法, 提供 360 度全景式数据查看机制, 指导企业试验运营分析[6].

1.2 SPA(单页面应用) 概念

单页面 Web 应用 (Single-page Web Application), 简称为 SPA[7], 是指用户在加载单个 HTML 页面并在用户与应用程序交互时动态更新该页面的应用程序. SPA 主要体现在浏览器端, 应用到的技术主要是 HTML+CSS+JavaScript, 使用 SPA 模式构建的网站在浏览器端第一次会加载所有程序文件, 而后续所有的刷新都不会重新加载所有文件, 而只加载访问请求的文件, 因此请求响应更快.

图 1 展示了 SPA 网站构建的常规模式, 在此模式中, 将网页视图的创建和管理从服务器端解耦出来, 移到 UI 层 (用户接口层), 这导致的结果即是除非在服务器端需要处理部分渲染, 否则服务器端将不会收到数据呈现方面的请求. SPA 整体设计与传统 Web 应用设计相似, 主要的不同点在于: 没有整页刷新、表现逻辑位于客户端, 服务器端的事务处理可以只涉及数据.

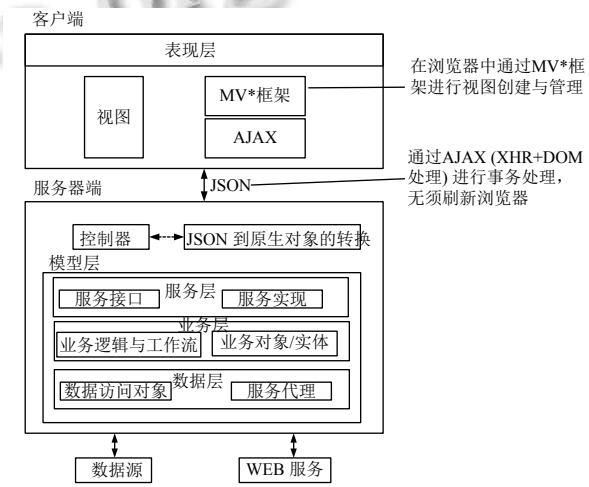

图 1 SPA 网站构建模式

SPA 模式与传统模式相较体现的具体优势包括: 用户体验好、快, 改变内容不需要重新加载整个页面;

服务器端只提供数据, 不用展示逻辑和页面合成, 提高性能; 单页面应用支持跨平台, 可以在电脑、手机和平板电脑等设备上使用.

1.3 MVC 概念

MVC 模式是“Model-View-Controller”的缩写, 中文翻译为“模型-视图-控制器”, 应用于用户交互应用程序中[8]. MVC 把一个应用的输入、处理、输出流程按照 Model、View、Controller 的方式进行分离, 这样一个应用被分成三个层: 模型层、视图层和控制层[9].

MVC 的运行机制是: 事件导致控制器改变模型或视图, 或者同时改变两者. 当控制器改变了模型中的数据或者属性时, 所有依赖的视图都会自动更新; 当控制器改变了视图, 视图会从潜在的模型中获取数据来刷新自己[10]. 图 2 展示的是 MVC 模式的各组件内容和工作原理[11].

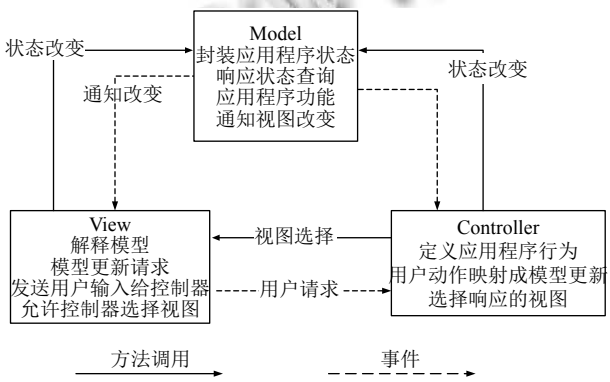

图 2 MVC 模式工作原理

运用 MVC 模式进行软件设计具有以下优势:低耦合性, 高内聚性; 高复用性; 可维护性高; 代码管理更简便.

2 试验数据管理系统业务分析

2.1 企业试验数据管理现状

工程机械企业一般都建立了自己的零部件和整机实验室, 经过多年积累, 试验数据量比较大, 但是在数据存储及安全、试验执行效率、试验管理水平、试验数据分析应用等方面存在短板.

(1) 数据存储及安全: 试验设备未联网, 试验数据存储分散, 个人数据存储于个人电脑, 没有集中管控; 数据传输未管控, 未进行数据备份, 数据安全存在隐患; 数据、报表等查询不便.

(2) 试验执行效率: 流程签审线下执行, 人工“跑签”影响签审效率; 试验业务进度无法监控.

(3) 试验管理水平: 作为管理者, 无法快速调度试验人员、设备等资源; 无法实现试验资源使用信息快速一览, 进行成本分析; 无法实现试验任务台账式管理, 全面了解企业试验运营情况.

(4) 试验数据挖掘分析应用: 试验数据没有集中管控, 数据价值挖掘基础条件不成熟, 无法实现基于大数据的应用分析, 价值挖掘.

2.2 试验数据管理系统功能建设

针对企业面临的问题, 结合当前 TDM 行业经验, 设计出符合企业现状的试验数据管理系统功能架构, 主要功能包括: 试验任务管理、试验流程管理、试验数据管理、试验设备管理、试验样品管理、试验报告管理、试验计划管理等. 针对不同的角色设定不同的访问门户. 图 3 所示为根据业务需求搭建的试验数据管理系统功能架构.

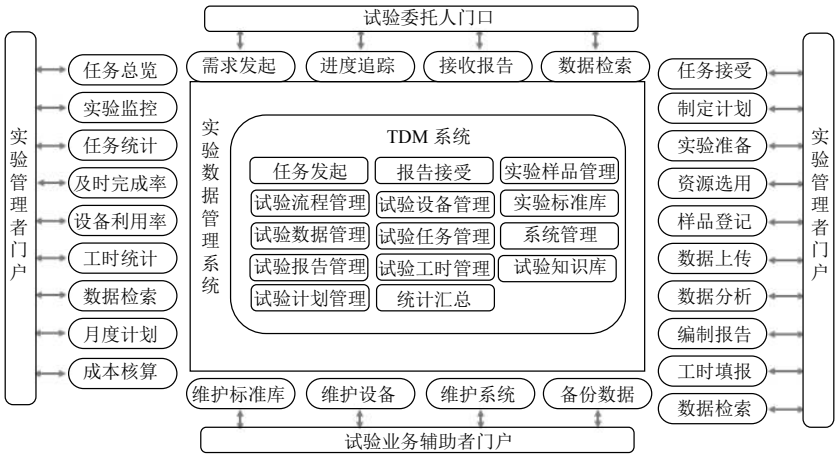

图 3 试验数据管理系统功能架构

3 系统框架设计

系统采用的是 B/S 架构, 即浏览器/服务器架构, 用户可以基于本地浏览器进行服务器访问. 系统设计采用的框架是 SPA, 如之前所描述, 在 SPA 框架中, 服务器端只处理数据模型, 浏览器端处理业务逻辑和视图展示. SPA 主要体现在浏览器端, 所以本文将研究重点放在前端, 服务器端详细设计不在研究范围内, 我们只需要了解本文所描述系统的服务器端与前端通信采用的是 MVC 框架.

3.1 系统功能框架

依据实际的业务需要, 本文所设计的试验业务管理系统整体建设框架如图 4 所示, 可以将其分为以下五层.

(1) 网络环境层: 此类涉及到企业核心数据的系统一般都需要构建统一的网络环境, 并且是独立的局域网, 其要满足两个要求, 一是所有与试验相关的 IT 终端都需要并入此网; 二是要确保此局域网的安全.

(2) 数据库层: 系统采用的企业级的 Oracle 服务器, 并且提供文件服务, 支持文件的存储和管理.

(3) 平台层: 指相关支撑功能, 如权限控制、查询引擎、业务建模、流程引擎等.

(4) 应用层: 系统的主要业务功能层, 主要包括试验数据管理、试验流程管理、试验设备管理、试验报告管理、试验样品管理等.

(5) 界面接口层: 系统提供基于角色的用户访问策略, 并提供响应 API 接口和其他外接接口, 用以与其他系统、工具的集成.

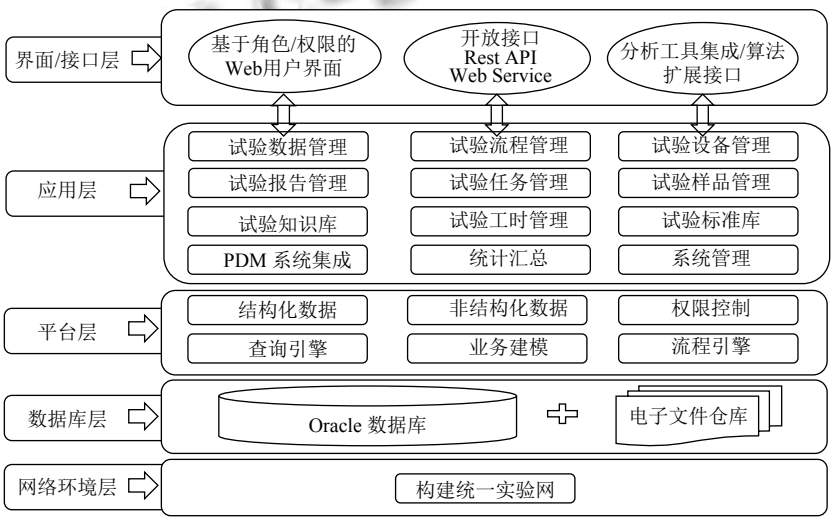

图 4 系统建设框架

3.2 系统整体视图布局

符合通用业务情况的试验数据管理系统的主要功能主要有我的空间、试验任务管理、试验数据管理、工时管理、任务跟踪、设备管理、业务管理、系统管理等. 总体视图布局主要涵盖四个部分, 即页头区、导航区、内容区和页脚区. 页头区用来放置系统的 logo 图标和用户账号管理; 导航区主要作用是放置访问每个视图的链接; 内容区用来展示每个视图的详细内容; 而页脚区则是显示其他相关信息, 如联系方式、站点管理等.

整个页面采用的是 SPA 构建方式, 用户通过账号密码登陆系统后, 初次访问会与服务器通信, 获取相关文件, 数据传输格式主要是 JSON 格式, 初次登陆会刷新整个页面, 加载页面所有内容. 而访问导航区所链接的条目时, 并不会重复刷新整个页面, 只是获取当前导航条目所对应的服务器端数据, 然后刷新当前视图即可. 这种网页构建方式将会大大加快页面的加载速度, 提升用户使用感受.

4 SPA 导航

4.1 Shell 外壳页面

单页面应用程序的“单页面”指的是初始 HTML 页面, 或被称为 Shell(外壳页面). 这个 HTML 页面加载且仅加载一次, 它是充当应用程序其余部分的起始点. 在 SPA 应用中, 这是唯一全页面加载的时机. 应用后续部分的加载将动态并独立于 Shell 页面进行, 无须全页面

加载, 不让用户感受到页面的刷新. Shell 页面在结构上保持最小化, 并通常包含一个空 标签, 该标签容纳应用程序其余内容. Shell 页面文件可看作其他应用视图的母舰. 关键代码:

```
<!DOCTYPE html>
<html>
<head>
<title>TDM Shell</title>…
</head>
</html>
```

上述即为单页面应用 Shell 外壳页面的 HTML 代码, 主要作用: 加载应用程序样式表; 加载第三方脚本文件; 初始 容器标签.

4.2 SPA 导航设计

SPA 的 DOM 元素通常是作为 SPA index 页面中的 Shell 的起点. SPA 模块及 MVC 框架, 包括支持库, 都跟 index 页面一同下载. 当用户导航时, 视图无缝呈现. 整个应用展示过程更加平滑, 极大提升用户体验. 旧页面被清除, 然后下载显示新页面所带来的粗糙体验已不复存在. 一旦初始页面加载了, SPA 的各种动作都不需要重复刷新它.

然而, 在 SPA 应用加载后用户需要有一种方式来访问应用的其他内容. 即通过 SPA 导航来实现.

图 5 展示的就是 SPA 导航的基本过程. 以用户请求“任务跟踪”视图为例, 当用户在前端点击导航栏中的“任务跟踪”时, 会产生一个固定的 URL, 路由器会根据配置路由对该 URL 进行匹配, 如果匹配不上, 将会配对默认路由, 在前端显示默认路由视图; 如果匹配上, 将会根据匹配的路由运行相应的代码、功能函数, 并调用匹配的视图, 在前端进行展示. 关键代码:

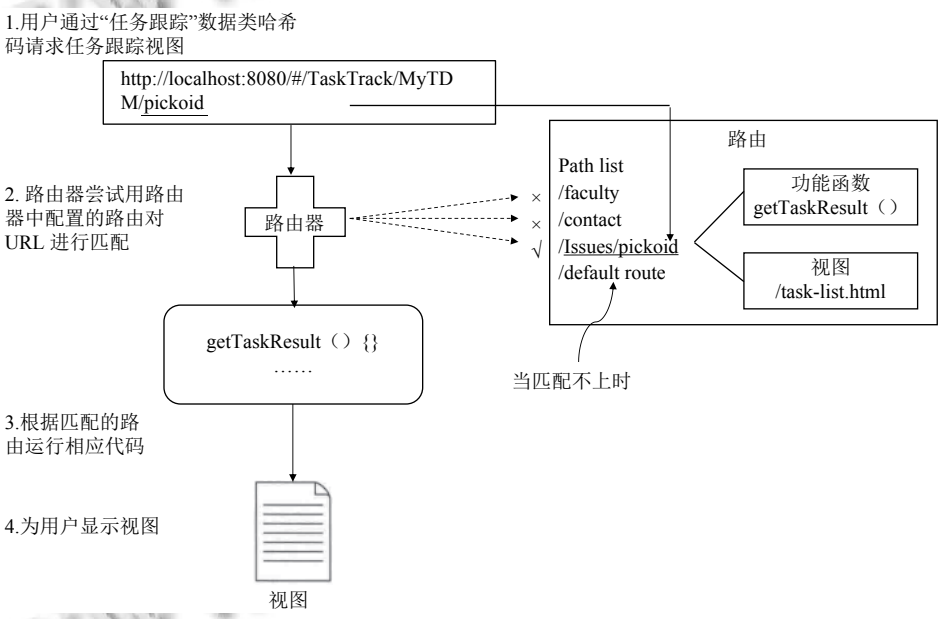

图 5 SPA 导航过程描述

```
angular.module(‘app.layout’, [‘ui.router’]).config(function
( $ stateProvider, $ urlRouterProvider) {
 $ stateProvider
.state(‘Task Management’, {
url: ‘/TaskManagement/MyTDM/217703421’,
templateUrl: ‘TaskManagement.html’})
templateUrl: ‘app/tasktrack/views/task-list.html’
 $ urlRouterProvider.otherwis(‘/myspace/MyTDM/217705104’);}
```

上述为路由配置关键代码, 主要作用: 定义模块, 并导入 ui-router, 对服务进行参数初始化; 定义前端页面导航栏所设计的如“我的空间”、“试验任务管理”、“试验任务跟踪”等的路由, 包括路由名称、URL 地址及视图模板; 定义行为缺省路由, 即登陆系统后的默认路由, 这里设置的默认路由是“我的空间”.

5 应用模块程序设计

本文所述 SPA 构建的试验业务管理系统是通过 MVC 框架的 JavaScript 框架风格来实践的. 1.3 节已经详细描述 MVC 框架的原理及优势, 本文所设计系统采用 Angular JS 进行开发. 因涉及的应用模块较多, 本文尝试用“试验任务跟踪”模块来描述如何使用 MVC 框架来进行模块设计.

5.1 模块 (Module) 定义

每个应用模块都有一个定义文件, 名为 module.js. 该文件主要定义模块的名称, 引用的其它模块和模块的入口及跳转路径等. 下面是 tasktrack 模块的 module.js 的定义. 关键代码:

```
.state('app.tasktrack', {
.state('app.tasktrack.tasklist', {
url: '/tasktrack/tasklist/:schema/:class/:pickoid', }
templateUrl: 'app/tasktrack/views/task-list.html',
controller: 'TaskListCtrl'}
```

上述代码主要作用: 定义模块的主要状态, 包括模块名: app.tasktrack.tasklist. URL 是

```
'/tasktrack/tasklist/:schema/:class/:pickoid';
```

模块所关联的视图模板为:

app/tasktrack/views/task-list.html;

模块所关联的控制器为:

app/tasktrack/controllers/TaskListCtrl.js.

5.2 应用模块视图 (View) 程序实现

针对“试验任务跟踪”应用模块, 用户关注的属性包括名称、进度、状态、负责人、开始时间、结束时间等. 如图 6 所示, 设计的主内容视图基本可以用表格来标示.

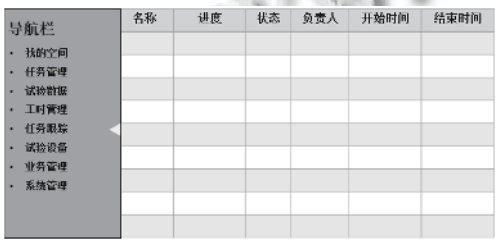

图 6 试验任务跟踪视图

关键代码:

```
<td><a ui-sref='.modalform({schema: dbschema, class: taskclass, oid:
row.obj_id, template: template})'><strong>{{row.Name}}</strong></a>
<div class='progress progress-xs'
data-progressbar-value='{{row.Progress}}'>
<div class='progress-bar'></div></div></td>
<td><span ng-class="row.Progress != 100 ? 'label label-success':
'label label-default'">{{row.Status}}</span></td>
<td><strong>{{row.Owner}}</strong></td>
<td>{{row.StartTime|date: 'shortDate'}}</td>
<td>{{row.EndTime|date: 'shortDate'}}</td>
```

视图代码段主要作用:

配置前端显示界面, 并在内容区绘制相应表格, 通过表达式{{row.Progress}}、{{row.Status}}等绑定显示数据库中的对象属性.

5.3 应用模块逻辑 (Controller) 程序实现

要能够使用路由参数传进来的信息, 每个框架或库都会提供相关的代码访问方式. AngularJS 提供了相应的 $ stateParams 对象. 下述代码展示了控制器对该变量的访问方式.

在“试验任务跟踪”应用模块中, 传进来的路由参数将匹配应用模块的数据库 (dbschema)、数据类 (taskclass)、哈希码 (pickoid), 并通过接口服务 taskTrackService 从服务器获取模块具体数据, 填充到 $ scope 变量, 以让视图显示信息.

关键代码:

```
angular.module('app.tasktrack').controller('TaskListCtrl', function ($scope,
$stateParams, taskTrackService) {
$scope.dbschema = $stateParams.schema;
$scope.callServer= function callServer(tableState) {
taskTrackService.getTaskResult($scope.dbschema,
$scope.displayed = result.data;
tableState.pagination.numberOfPages=
result.numberOfPages;
$scope.searchIssues = function (){
```

此部分代码主要作用: 定义控制器的名称为“TaskListCtrl”和所属的模块为“app.tasktrack”. 其中 taskTrackService 提供数据库操作的接口; 初始化变量; 调用 taskTrackService 的接口从数据库获得任务数据.

每个应用模块基本上都要与后端服务的 API 打交道. 为了更好地复用代码, 将与后端服务交互的代码封

装为 Angular 的 Service. “试验任务跟踪”应用模块将与后端服务交互的代码封装在 taskTrackService 中. 类似于试验业务管理这类系统, 一般都是需要持久化存储数据, 也就是要存在数据库里, 所以大部分数据模型都是映射后台传过来的 JSON 对象. 只有前端需要缓存数据时才会考虑设计模型 (Model), 用于跨模块或 Controller 使用. “试验任务跟踪”应用模块没有将返回的数据进行转换或暂存在内存中, 因而不需要定义数据模型.

图 7 试验任务跟踪最终界面展示

6 实践与结果分析

通过前面方法, 我们设计开发出试验数据管理系统所有应用模块, 其中“试验任务跟踪”模块最终前端界面展示如图 7 所示. 在左侧导航栏直接点击试验任务跟踪 (Task Track) 条目, 在不刷新整个页面的基础上, 右侧显示任务跟踪的详细内容.

6.1 定义测试环境

客户端硬件: 惠普 Z240 Workstation, 内存 16 GB.

客户端软件环境: Windows 7 操作系统, 火狐浏览器 (版本: 52.3.0(32 位)).

网络带宽: 1000 M.

6.2 测试结果

在同样测试环境下, 同一人使用同一个客户端进行登陆测试. 测试首页整页加载时间和单页面应用视图加载时间.

通过测试发现, 整页加载需要加载各类 HTML、JS、CSS、XHR、图像等文件, 本文设计系统共计 45 个请求. 单应用模块加载内容主要是与服务器端沟通的 XHR 对象, 本文设计系统“试验任务跟踪”模块加载请求仅 15 个. 对比结果如图 8 所示.

进行 10 次对比测试, 结果如图 9 所示. 通过测试发现, 单模块视图加载速度明显快于整页面加载速度, 并且在页面视图呈现上更加平滑.

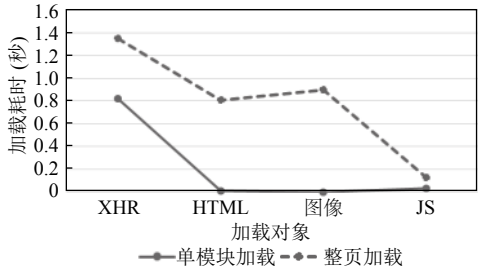

图 8 整页加载与单模块加载内容及时间对比

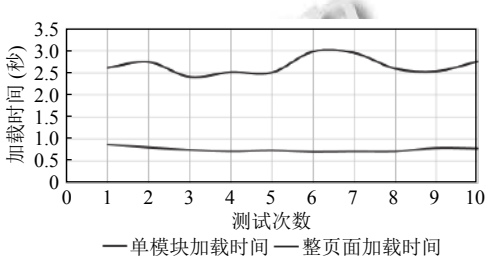

图 9 网页加载速度测试

7 总结与展望

本文针对工程机械行业试验业务的特点, 以及传统网站构建方式的弊端, 提出了基于 SPA 架构构建管理系统的方法, 通过 MVC 框架、HTML、AngularJS 等网络信息技术的综合应用, 设计开发出类似原生态界面风格的试验数据管理系统. 通过实践测试, 验证确实可行, 大大缩短了网页加载时间, 提升了用户体验. 当然, 本文所述系统在实际设计开发过程中是一个庞大的系统工程, 包括实际业务流程、前端、后端、服务器、网络等多方面的内容. 本文仅从网站构建方式的角度来进行描述, 提出构建方法, 论证 SPA 架构对网站构建的突出优势.

目前, 很多知名的网站都已采用 SPA 这种方式来进行网站搭建, 界面美观、浏览流畅等特点已众所周知, 但在工程机械信息化领域, 应用还比较少. 通过本文的论述, SPA 网站构建方式具有重大应用价值, 不仅在试验数据管理系统, 甚至在企业产品数据管理、生产制造管理、财务管理、档案管理等都可以加以应用, 以提升企业现代数字化水平, 提高用户工作体验, 提升企业竞争力.

参考文献

1 李洪奇. 试验数据管理系统的应用. 软件工程师, 2010, (4): 43–44. [doi: 10.3969/j.issn.1008-0775.2010.04.021]

2 丁力, 安海军. 试验数据管理系统的需求与实现. 航空计算技术, 2010, 40(3): 96–98. [doi: 10.3969/j.issn.1671-654X.2010.03.025]
3 李曼丽, 易忠, 肖琦. 航天器磁环境试验数据库的建设. 航天器环境工程, 2009, 26(4): 343–346. [doi: 10.3969/j.issn.1673-1379.2009.04.010]
4 张怡然. TDM 试验数据管理系统的设计与应用[硕士学位论文]. 北京: 北京工业大学, 2012.
5 董冬, 朱成亮, 胡瑛, 等. 试验数据管理平台设计研究. 火箭推进, 2014, 40(4): 67–72. [doi: 10.3969/j.issn.1672-9374.2014.04.012]
6 宋铭利, 王素丽. 试验数据管理系统的设计与实现. 计算机工程与设计, 2011, 32(5): 1680–1683, 1717.
7 Scott EA. SPA 设计与架构: 理解单页面 Web 应用. 卢俊祥, 译. 北京: 电子工业出版社, 2016.
8 曹燕. 基于 Java 的系统开发中 MVC 架构的应用. 气象科技, 2006, 34(S1): 36–39. [doi: 10.3969/j.issn.1671–6345.2006.z1.009]
9 王伟, 贾慧娟. 基于 MVC 模式的分布式作战指挥系统研究. 微计算机信息, 2007, 23(30): 50–52. [doi: 10.3969/j.issn.1008-0570.2007.30.019]
10 张向奎, 黄瑀潇, 郭威. 基于 MVC 模式的冲压模具 CAE 系统构架设计. 北华大学学报 (自然科学版), 2005, 6(4): 378–381. [doi: 10.3969/j.issn.1009-4822.2005.04.026]
11 曾艳阳, 刘淑芬, 高五星. MVC 框架在指控仿真系统中的应用研究. 微计算机信息, 2010, 26(31): 79–80. [doi: 10.3969/j.issn.2095-6835.2010.31.032]