Canvas

Software Architecture Document

周植，胡子木，朱任翔，王登泰，李舞桂

Version 1

2019.10.21

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Description of Versions / Changes** | **Responsible Party** | **Date** |
| 1.0 | Add 1.2.3 | Zhu Renxiang | 19.10.17 |

Table of Contents

[1. 简介 1](#_Toc22545255)

[1.1. 目的 1](#_Toc22545256)

[1.2. 范围 1](#_Toc22545257)

[1.3. 定义，首字母缩写词和缩写 1](#_Toc22545258)

[1.4. 参考文献 2](#_Toc22545259)

[1.5. 概观 2](#_Toc22545260)

[2. 架构表现 2](#_Toc22545261)

[3. 架构目标与约束 3](#_Toc22545262)

[4. 用例视图 3](#_Toc22545263)

[4.1. 角色 3](#_Toc22545264)

[4.2. 用例实现 3](#_Toc22545265)

[4.2.1. Login 3](#_Toc22545266)

[4.2.2. Request Analysis (Get Report) 5](#_Toc22545267)

[4.2.3. Retrieve Last Report 8](#_Toc22545268)

[4.2.4. Print Report 9](#_Toc22545269)

[4.2.5. Email Report 10](#_Toc22545270)

[5. 逻辑视图 11](#_Toc22545271)

[5.1. Overview 11](#_Toc22545272)

[5.2. Interface Definitions 13](#_Toc22545273)

[6. 数据视图 22](#_Toc22545274)

[7. 部署视图 24](#_Toc22545275)

Software Architecture Document

# 简介

本文档提供了一个高级概述，并解释了Canvas的体系结构。

该文档定义了体系结构的目标，系统支持的用例，体系结构样式和已选择的组件。 该文档为从概念构思到实现的体系结构和设计决策提供了理论依据。

## 目的

软件体系结构文档（SAD）提供了Canvas系统的全面体系结构概述。 它提供了许多不同的体系结构视图来描述系统的不同方面。

为了尽可能准确地描述软件，本文档使用了基于Philippe Kruchten的“4 + 1”体系结构模型视图[Kruchten]结构。



“ 4 + 1”视图模型允许各种利益相关者找到他们在软件体系结构中需要的东西。

## 范围

该SAD的范围是解释分布式开发监视和挖掘系统的体系结构。

本文档描述了Canvas系统设计的各个方面，这些方面在体系结构上很重要。 这些要素和行为对于指导Canvas系统的构建以及从整体上理解该项目至关重要。 鼓励对 Canvas 系统有技术了解的利益相关者从阅读为该系统开发的项目提案，操作概念和软件要求规范文档开始[PP，ConOps，SRS]。

## 定义，首字母缩写词和缩写

* **Apache** – Web Server
* **ASP.NET** - Microsoft web platform
* **HTTP** – Hypertext Transfer Protocol
* **WWW** – World Wide Web
* **SAD -** Software Architecture Document
* **UML** – Unified Modeling Language
* **User -** This is any user who is registered on the DMM website

## 概观

为了完整记录体系结构的所有方面，软件体系结构文档包含以下小节。

第2节：介绍每种视图的用法

第3节：描述系统的体系结构目标和约束

第4节：描述最重要的用例实现

第5节：描述系统的逻辑视图，包括接口和操作定义。

第6节：描述了重要的持久性元素。

第7节：介绍如何部署系统。

# 架构表现

本文档使用“ 4 + 1”模型[Kruchten]中定义的视图详细说明了体系结构。 用于记录Canvas系统的视图为：

**用例视图**

**受众**: 系统中的所有利益相关者，包括用户终端。

**领域**: 描述代表系统某些重要的核心功能的场景和/或用例集。 描述了系统的参与者和用例，此视图提出了用户的需求，并在设计级别上进行了详细说明，以更详细地描述离散的流程和约束。 该领域词汇表与任何处理模型或表示语法无关。

**相关工件** :用户模型，用例文档。

**逻辑视图**

**受众**: 开发者

**领域**: 功能需求：描述设计的对象模型。 还描述了最重要的用例实现和系统的业务需求。

相关工件: 设计模型。

**数据视图**

**受众**: 数据专家，数据库管理员。

**领域**: 持久性：描述数据模型中在体系结构上重要的持久性元素以及数据如何流经系统。

**相关工件:** 数据模型

**部署视图**

**受众**: 部署经理。

**领域**: 拓扑：描述软件到硬件的映射，并显示系统的分布式方面。 通过在架构中包括已知和预期的部署方案来描述潜在的部署结构，我们允许实施者对网络性能，系统交互等做出某些假设。

**相关工件:** 部署模型。

# 架构目标与约束

有一些关键要求和系统约束对体系结构有重要影响。他们是：

1. 该系统旨在作为将来要构建的更完整的项目预测系统的概念证明。因此，本文档和整个系统的主要利益相关者之一是未来的架构师和设计师，而不是通常的用户。因此，本文档的一个目标是对将来的建筑师和设计师有用。
2. 该系统将使用Microsoft .NET技术编写，但将使用开源RDBMS系统（MySQL）进行数据持久化，并将部署到运行Mono的Linux Web服务器上。这些特殊的部署要求在体系结构的开发中需要额外考虑。
3. 系统必须与多个第三方API，Assembla和Google Predictive通信。定义系统如何​​与这些第三方系统接口是体系结构的主要问题。
4. 软件需求规范的第3.3节概述了应用程序可能会随着时间推移而面临的许多预期更改。系统体系结构的主要目标之一是通过最小化实现这些代码而需要修改的代码量来最小化这些更改的影响。该体系结构试图通过使用模块化和信息隐藏来做到这一点，以将可能会更改的组件与系统的其余部分隔离开来。

# 用例视图

用例视图的目的是为系统的使用及其组件之间的交互提供额外的上下文。 在本文档中，每个组件都被视为用例参与者。第4.1节列出了当前的参与者，并在系统的整体使用环境中对每个参与者进行了简要说明。在第4.2节中，使用UML用例图和序列图概述和说明了最常见的用例，以阐明组件之间的交互。

## 角色

用户：用户是使用系统的载体，一共被分为三类用户——学生、教师以及管理员。用户与操作界面进行交互并使用整个系统。

## 用例实现

### Login

User credentials are authenticated and user is redirected to application home page.

**Figure 4.1** Login Use Case Diagram



**Figure 4.2** Login Sequence Diagram



### Request Analysis (Get Report)

User requests a report for a user-specified project and report is displayed.

**Figure 4.3** Request Analysis Use Case Diagram



**Figure 4.4** Request Analysis Sequence Diagram



### Retrieve Last Report

User requests to view the last generated report.

**Figure 4.5** Retrieve Last Report Use Case Diagram



**Figure 4.6** Retrieve Last Report Sequence Diagram



### Print Report

User requests to print report.

**Figure 4.7** Print Report Use Case Diagram



**Figure 4.8** Print Report Sequence Diagram



### Email Report

User requests a report be sent to a user-specified list of recipients.

**Figure 4.9** Email Report Use Case Diagram



**Figure 4.10** Email Report Sequence Diagram



# 逻辑视图

## 概况

本节主要描述了系统的各个组件以及组件之间相应的接口，可以保证当某一功能修改时，只需要对相应的接口进行一定的修改，从而避免重新设计整个系统。

1. 更改用户验证接口

用户验证接口主要用来在登陆以及需要验证用户身份时实现相应的功能，如果后期再用户登录时需要设计其他复杂的验证信息，则可以通过修改用户验证接口来实现功能的变更。

1. 更改讨论板接口

讨论板接口主要用来在前端实现讨论板的相关功能，在后期如果有对讨论板的功能和表现形式进行调整的需求，则可以通过更改讨论板接口来实现。

1. 更改文件接口

文件接口用于对文件的上传和下载操作，如果后期需要对文件的上传和下载进行其他的限制或者处理，可以通过更改文件接口达到相应的目的。

1. 更改站内信接口

站内信接口用于实现站内信的相关功能，如果需要更改站内信的功能，则只需要对站内信接口进行修改。

1. 更改课程接口

课程接口用于实现与课程有关的功能，在后期需要增加课程功能或修改现有功能时，可以通过修改课程接口来实现。

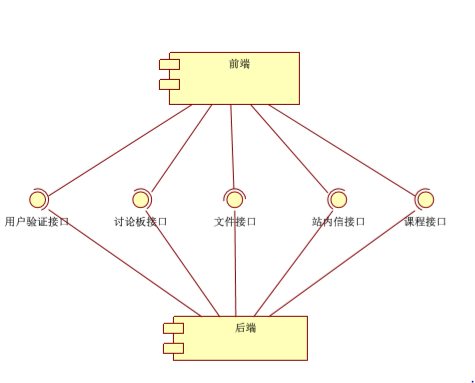


图 5.1 逻辑构件图

**表格5.1** 构件功能

|  |  |
| --- | --- |
| **构件** | **功能** |
| 前端 | * 用于显示用户界面 * 用于接受用户操作并调用相应的接口 |
| 后端 | * 提供用户验证接口 * 提供讨论板接口 * 提供文件接口 * 提供站内信接口 * 提供课程接口 * 组织相应的数据库存储 * 进行逻辑判断及数据计算 |

## 接口定义

前端界面与后台之间使用 Ajax 发送数据进行交互（基本使用 HTTP POST 的方式），所有后端接口均为一个 HTTP 地址，假设 API 的域名地址为 URL，前端将通信需要的数据（下文将作说明）与账户信息（存储在 Cookie 中，下面所有操作均会检查用户是否具有相应权限，如不具有则返回失败）发送给后端，并从后端获取数据。

### 用户接口

**用户登陆**

|  |  |
| --- | --- |
| 调用地址 | URL/user/login |
| 详细描述 | 传入两个字段：uid, pwd，分别代表用户名与加密后的密码。如果登陆成功，将自动设定账户相关信息的 Cookie，否则返回失败原因。 |
| 调用前提 | 无 |
| 接口效果 | 登陆成功或者返回失败原因。 |

**修改账户信息**

|  |  |
| --- | --- |
| 调用地址 | URL/user/modify |
| 详细描述 | 必须传入的字段：pwd，代表加密后的密码。其他字段表示，需要修改的用户信息。如果验证成功则可以修改用户西，否则返回失败原因。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 用户信息修改成功或者返回失败原因。 |

### 文件系统接口

**上传文件**

|  |  |
| --- | --- |
| 调用地址 | URL/files/upload |
| 详细描述 | 传入需要上传的文件 File，一次只能上传一个文件。如果上传成功，则返回文件ID，否则返回失败原因。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回文件ID或者返回失败原因。 |

**获取文件地址**

|  |  |
| --- | --- |
| 调用地址 | URL/files/url |
| 详细描述 | 传入字段：id，表示文件的唯一编号，如果文件存在则返回文件的下载地址，否则返回失败原因。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回下载地址或者返回失败原因。 |

**删除文件**

|  |  |
| --- | --- |
| 调用地址 | URL/files/delete |
| 详细描述 | 传入字段：id，分别表示文件的唯一编号。如果删除成功则返回成功，否则返回失败原因。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回成功或失败原因。 |

### 课程内容接口

**Markdown 接口**

|  |  |
| --- | --- |
| 调用地址 | URL/course/markdown |
| 详细描述 | GET 方法：传入参数 id，表示 Markdown 文本的编号，返回 Markdown 内容。此方法可以拉取一个已有的 Markdown 文本资源。  POST 方法：传入参数 content，表示 Markdown 文本的内容。此方法可以新建 Markdown 文本资源。  PUT 方法：传入参数 id, content，表示 Markdown 文本的编号和内容，此方法可以更新一个 Markdown 文本的内容。  DELETE 方法：传入参数 id，表示 Markdown 文本的编号，此方法删除一个 Markdown 文本资源。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回或失败原因。 |

**问题接口**

|  |  |
| --- | --- |
| 调用地址 | URL/course/problem |
| 详细描述 | GET 方法：传入参数 id，表示问题的编号，返回问题的内容。此方法可以拉取一个已有的问题。  POST 方法：传入参数 content，表示问题的内容。此方法可以新建问题资源。  PUT 方法：传入参数 id, content，表示问题的编号和内容，此方法可以更新一个问题。  DELETE 方法：传入参数 id，表示问题的编号，此方法删除一个问题。  问题可以用于加入到考试或者作业中去。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回内容或失败原因。 |

**问题操作接口**

|  |  |
| --- | --- |
| 调用地址 | URL/course/operations |
| 详细描述 | GET 方法：传入参数 id，表示问题的编号。此方法可以拉取问题的状态与分数。  POST 方法：传入参数 uid, content，表示学生编号与问题修改内容。此方法用于学生作答问题或者老师批改问题。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回内容或失败原因。 |

**课程结构设置接口**

|  |  |
| --- | --- |
| 调用地址 | URL/course/modules |
| 详细描述 | GET 方法：传入参数 id，表示课程编号，返回课程模块内容。此方法可以拉取一个已有的课程模块资源。  PUT 方法：传入参数 id, content，表示课程模块的编号和内容，此方法可以更新一个课程模块内容。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回成功或者失败原因。 |

**学生列表接口**

|  |  |
| --- | --- |
| 调用地址 | URL/course/students |
| 详细描述 | GET 方法：传入参数 id，表示课程编号，返回课程内所有学生的编号。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回内容或者失败原因。 |

### 评论系统接口

**讨论帖接口**

|  |  |
| --- | --- |
| 调用地址 | URL/bbs/post |
| 详细描述 | GET 方法：传入参数 id，表示帖子编号。此方法可以拉取一个已有的讨论帖子。  POST 方法：传入参数 content，表示帖子的主题，此方法可以新建一个帖子。  PUT 方法：传入参数 id, content，表示帖子的编号和内容，此方法可以更新一个帖子。  DELETE 方法：传入参数 id，表示帖子的编号，此方法可以删除一个帖子。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回成功或者失败原因。 |

**讨论帖接口**

|  |  |
| --- | --- |
| 调用地址 | URL/bbs/comment |
| 详细描述 | GET 方法：传入参数 id，表示评论编号，此方法可以拉取一条评论内容。  POST 方法：传入参数 content，表示评论内容，此方法可以新建评论内容。  PUT 方法：传入参数 id, content，表示评论的编号和内容，此方法可以修改评论的编号和内容。  DELETE 方法：传入参数 id，表示评论的内容，此方法可以删除评论。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回成功或者失败原因。 |

### 站内信接口

**列表接口**

|  |  |
| --- | --- |
| 调用地址 | URL/mail/list |
| 详细描述 | GET 方法：此方法可以拉取用户的邮件列表 。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回成功或者失败原因。 |

**邮件接口**

|  |  |
| --- | --- |
| 调用地址 | URL/mail/post |
| 详细描述 | GET 方法：传入参数 id，表示邮件编号。此方法可以拉取一个邮件的内容。  POST 方法：传入参数 content，表示邮件的内容，此方法可以发送邮件。  DELETE 方法：传入参数 id，表示邮件的编号，归档一封邮件。 |
| 调用前提 | 用户登陆 |
| 接口效果 | 返回成功或者失败原因。 |

# 数据视图

user 数据表用于存储用户的信息，通过 Role 字段来分辨用户类型。

mail 数据表用于存储来往邮件，其中 from, to 均为外键，表示收件人与发件人，ref 为外键表示回复的邮件。

file 数据表用于存储上传到服务器中的文件信息，使用两个外键表示此文件挂在在哪一个文本资源上或者属于哪个用户。

problem 数据表表示问题，可以是考试中的一个问题或者是作业中的一个问题，设定内容、状态与问题满分。

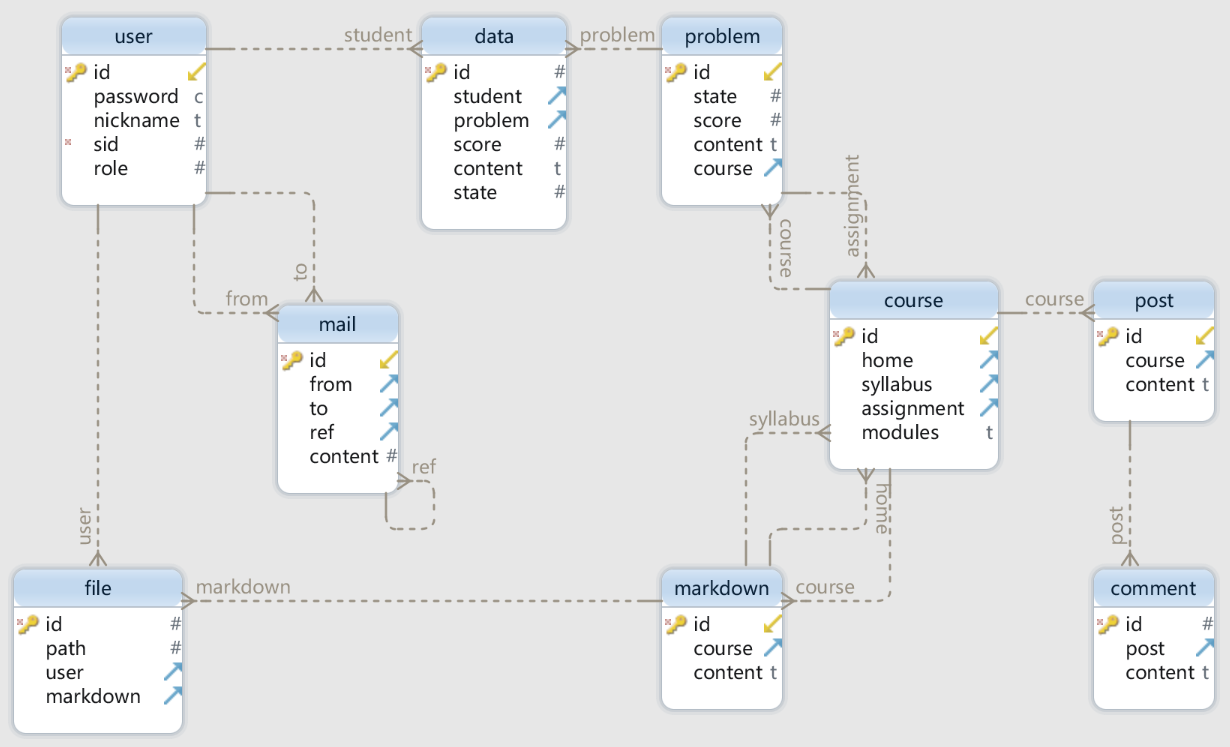
data 数据表存储每一个学生用户回答问题的数据，保存回答内容，教师评分当前状态等数据。

course 数据表存储一门课程的信息，其中大部分内容均为 Markdown 资源文件，markdown 数据表中的一条字段，课程模块为一个 json 文本，表示当前课程模块的划分状况与每一个小模块指向什么哪种。

markdown 数据表，存储一个 Markdown 文本资源。作业连向 problem 数据表中的一个字段，表示当前的作业。

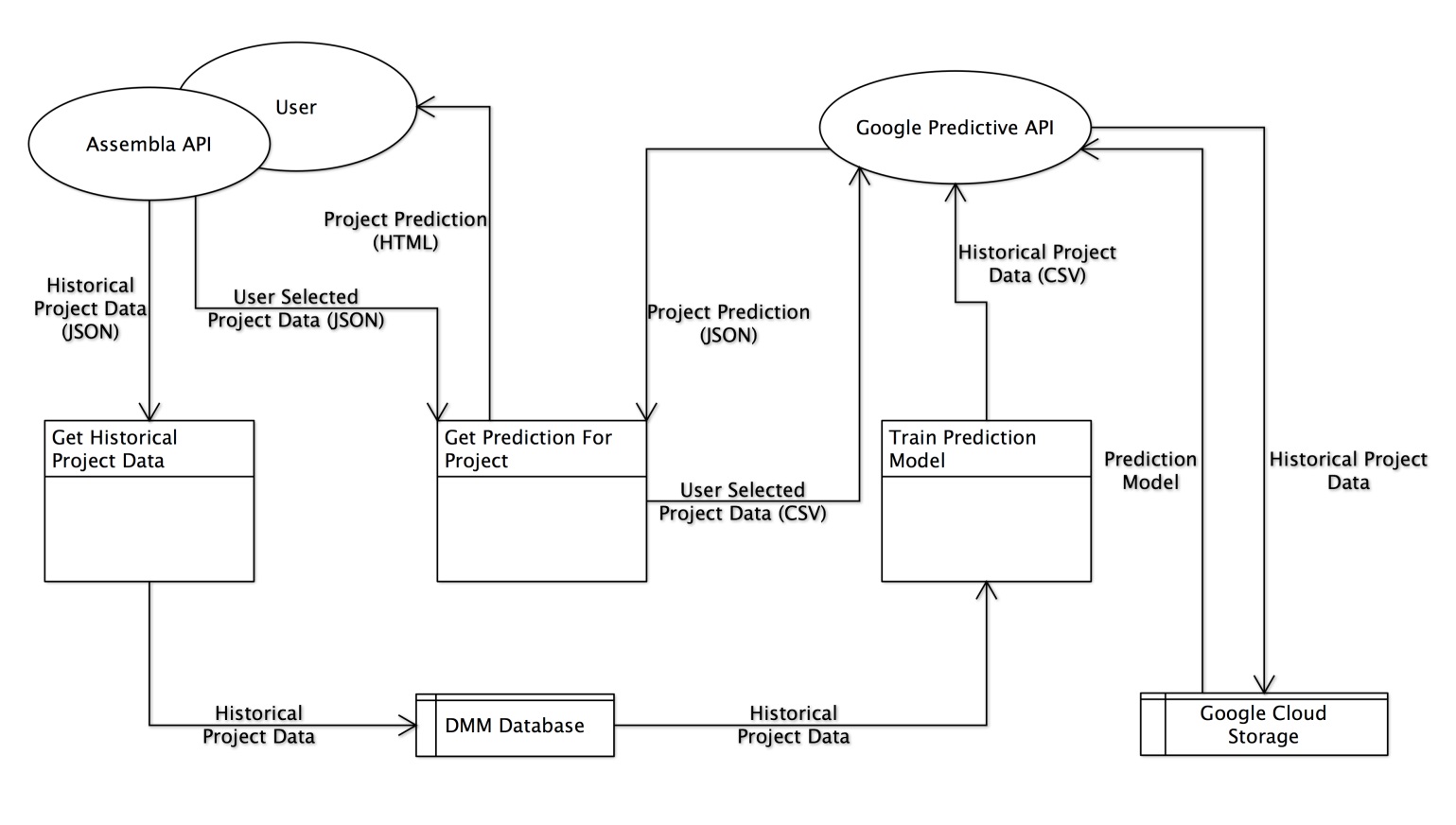
post 数据表，存储一个帖子的信息，外键连向课程。

comment 数据表，存储一条回复的信息，外键连向帖子。



**Figure 8.2** Data Flow Diagram

This diagram illustrates how data will flow between external entities and the DMM application. Ovals represent external entities, tall boxes represent points where data is processed, arrows show the direction of data flow, and short boxes represent persistent data stores.



# 部署视图

Canvas 在线教学系统采用B/S结构模型。该系统运行在 Linux 环境的云服务器中，后台程序使用 PHP 与 MySQL 进行编写，用户可通过移动端和PC端进行访问。MySql服务也有该服务器提供，用于对数据的高效管理。服务器和客服端通过http协议进行数据交互。

部署图如下

