<儿童日历设计>

软件体系结构文档

版本 <2.3>

修订历史记录

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **描述** | **作者** |
| <03/12/2018> | <1.0> | <初版> | <杨劲松> |
| <05/12/2018> | <2.0> | <简介及体系结构部分内容> | <叶苏航> |
| <07/12/2018> | <2.1> | <部分视图的实现> | <凌通> |
| <07/12/2018> | <2.2> | <部署图> | <游增> |
| <10/12/2018> | <2.3> | <最终检查> | <林水泉> |

目录

1. 简介 4

1.1 目的 4

1.2 范围 4

1.3 定义、首字母缩写和缩写 4

1.4 参考资料 4

1.5 概述 4

2. 体系结构表示 4

3. 体系结构目标和约束 4

4. 用例视图 4

5. 逻辑视图 5

5.1 概述 5

5.2 重要体系结构设计包 5

5.3 用例实现 5

6. 流程视图 5

7. 部署视图 5

8. 实施视图 5

8.1 概述 5

8.2 层 5

9. 数据视图（可选） 5

10. 大小和性能 6

11. 质量 6

软件体系结构文档

# 简介

文档介绍了本系统设计时所采用的架构模式：即描述系统组件，组件间的关系，组件对环境的关系，以及指导系统设计和演变的原则。

## 目的

本文档使用许多不同的体系结构视图来描述系统的多个方面，对系统进行了全面的体系结构概述。其目的是捕获和传达对系统作出的重要体系结构决策。

## 范围

由于系统仅针对安卓系统的手机应用软件而设计，因此本文档仅适用于安卓app，并且文档只针对系统的整体逻辑架构做出设计，不涉及详细代码。

## 定义、首字母缩写和缩写

MVC结构：Model-View-Controller结构的缩写，代表着模型-视图-控制器结构。

星星：儿童用户打卡每天刷牙所得到的奖品在软件中的呈现形式。

积分：用户的一个属性。用户打卡刷牙完成任务将获得积分。

等级：用户的一个属性。累积一定的积分可获得等级的提高。

奖品：积分与等级可用于换取奖品。

## 参考资料

1. Karina Sokolova, Marc Lemercier University of Technology of TroyesAndroid, Ludovic Garcia, EUTECH SSII, *Passive MVC: a Novel Architecture Model for Android Application Development,* PATTERNS 2013 : The FIfth International Conferences on Pervasive Patterns and Applications.
2. IEEE1471-2000
3. ISO\_IEC\_IEEE 42010-2011
4. Android Passive MVC [sokolova2013] Android Passive MVC: a Novel Architecture Model for Android Application Development.pdf

## 概述

本文档将从该日历软件的体系结构出发，首先对体系结构的目标和约束进行简要介绍，再结合用例视图、逻辑视图、流程视图、部署视图和实施视图这些必要的视图进行详细介绍，全面地解释、分析本日历软件的软件架构。

# 体系结构表示

本系统采用Android Passive MVC架构，文档使用用例视图、逻辑视图、流程视图、部署视图及实施视图这些必要的视图去解释架构的组成。

# 体系结构目标和约束

[软件目标]

可用性：儿童日历软件具有可用性。本软件是针对儿童的日常使用进行开发的，经测试运行流畅，可用性强。

安全性：本儿童日历软件具有安全性。本软件采用MVC结构，可脱机运行，在遭受恶意攻击时仍能保证系统的正常使用。

隐私性：本儿童日历软件具有隐私性。采用MVC结构，前端界面与后端数据分离，可保证用户数据良好的隐私性。

可移植性：本软件是基于Java语言开发的，可移植性强。

[软件约束]

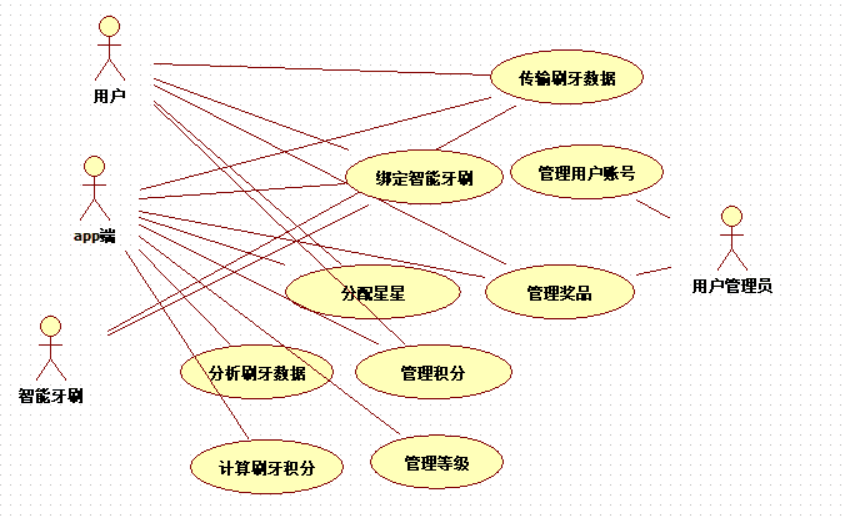
时间约束：本软件的开发大体来说开发所要求时间较短，进度较急。

技术成熟度约束：小组成员技术的成熟度较低，在技术上受到约束。

[人员约束]

人员的技术能力相对较低，花费了较多的时间去学习、调试。

# 用例视图

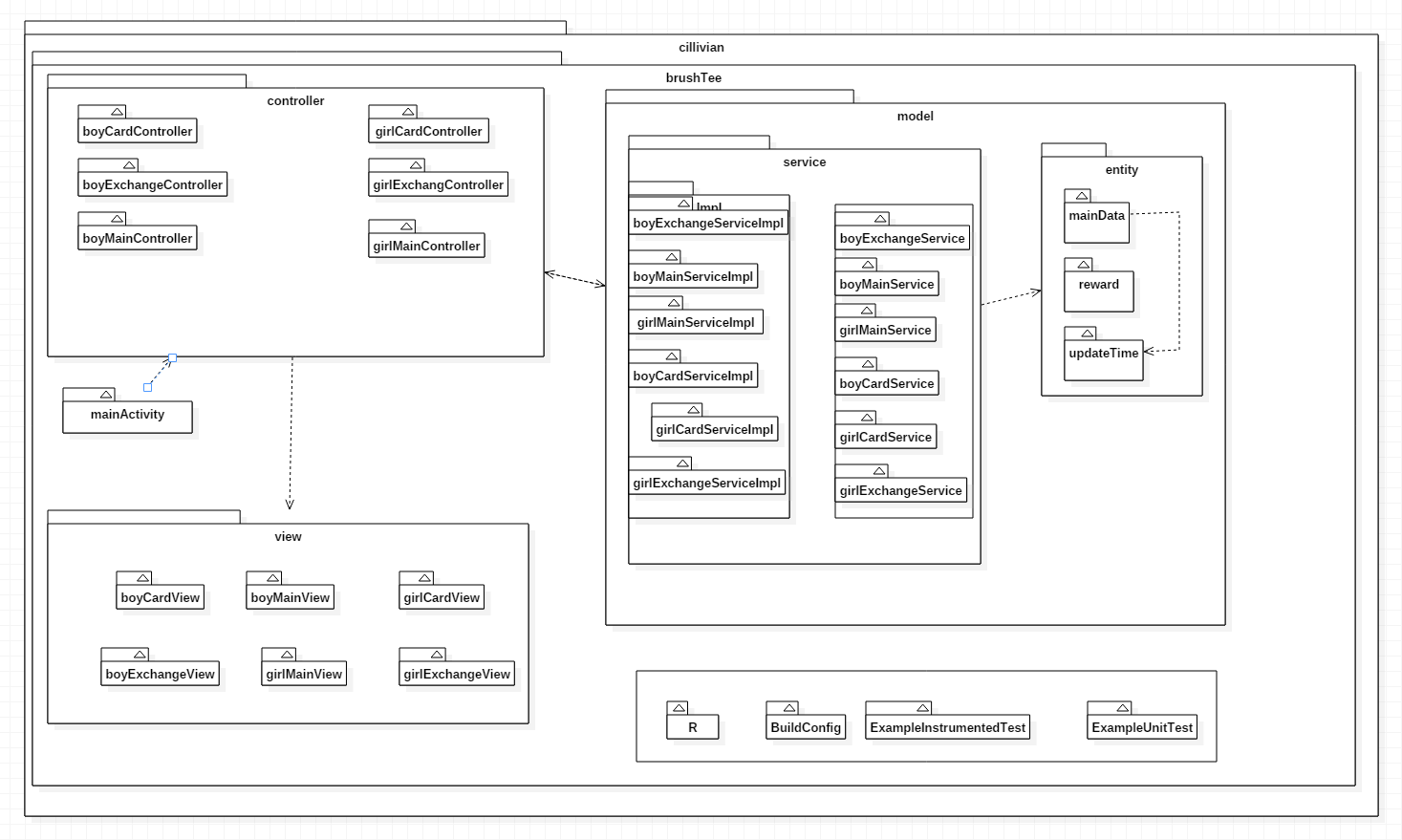


# 逻辑视图

## 概述

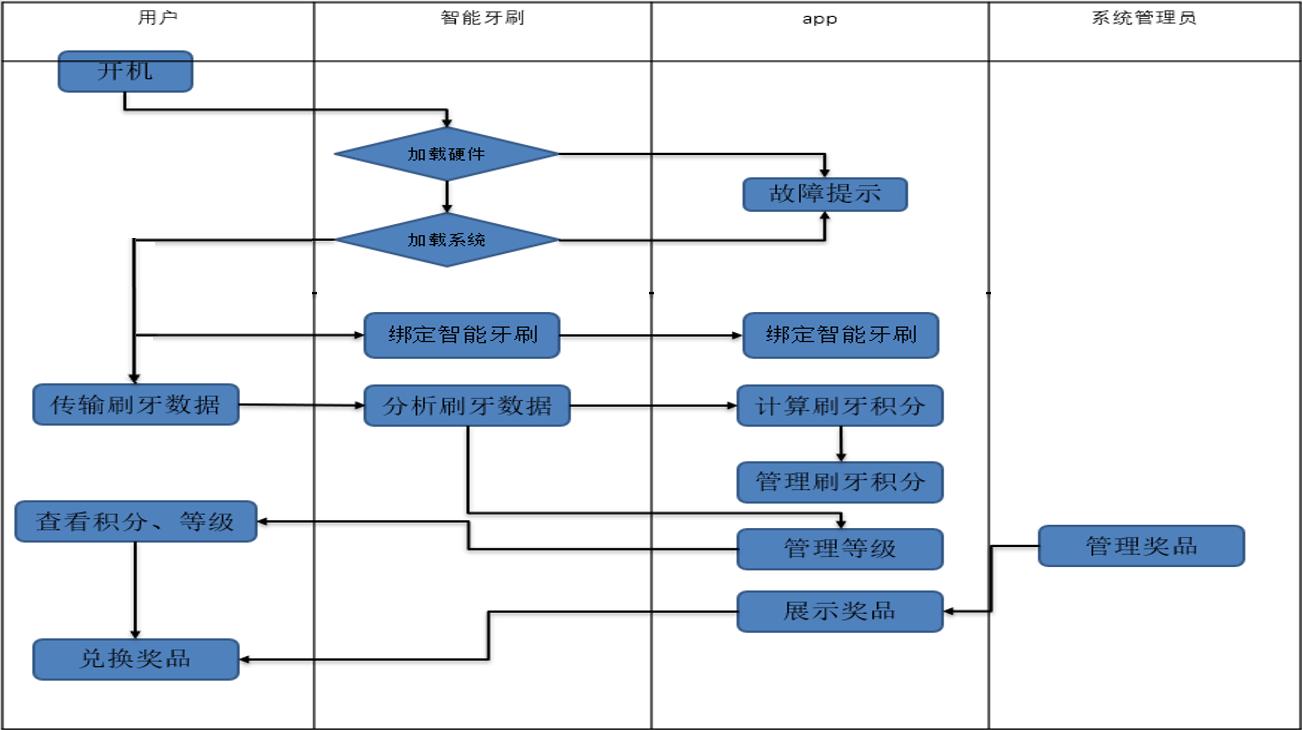
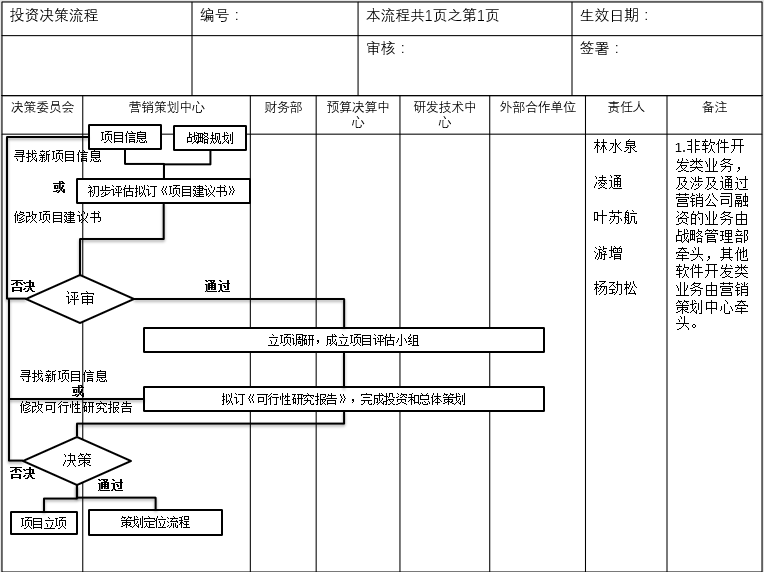
此节描述设计模型在体系结构上很重要的部分，例如将模型分解成众多子系统和包。对于每个重要的包，又描述将包分解成类和类实用程序。

## 重要体系结构设计包

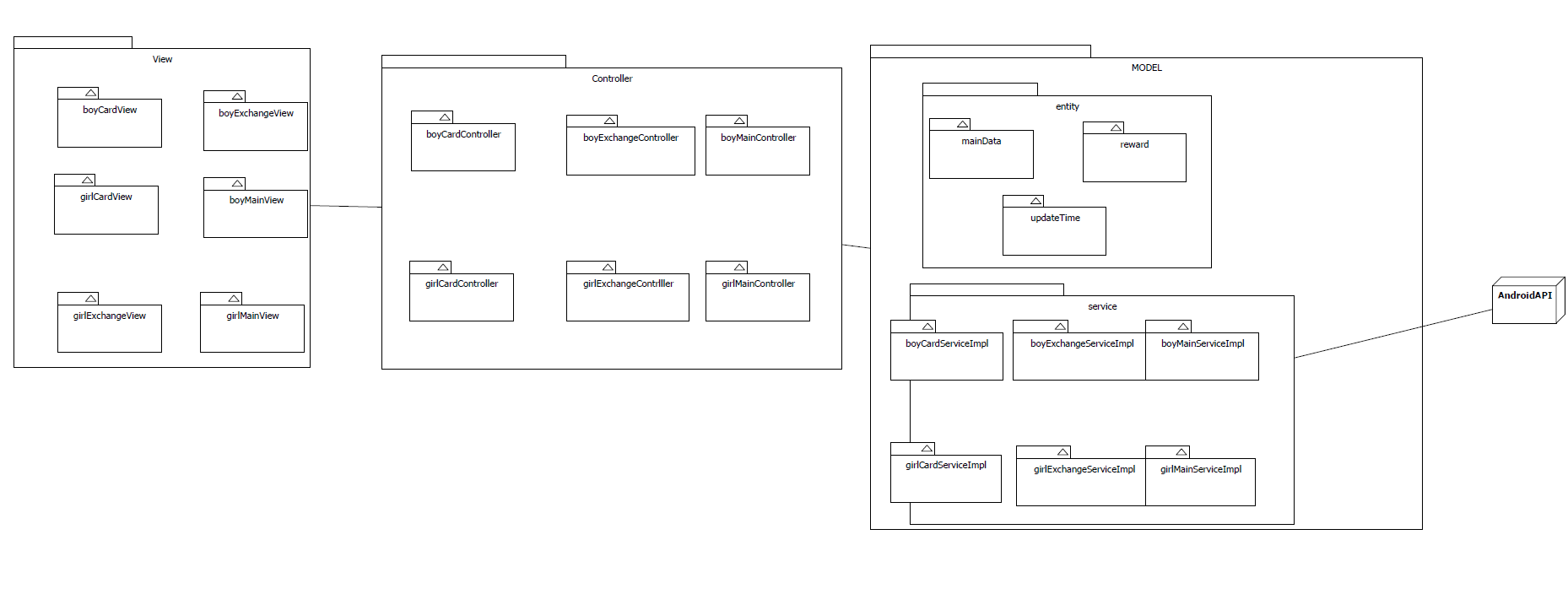


## 用例实现

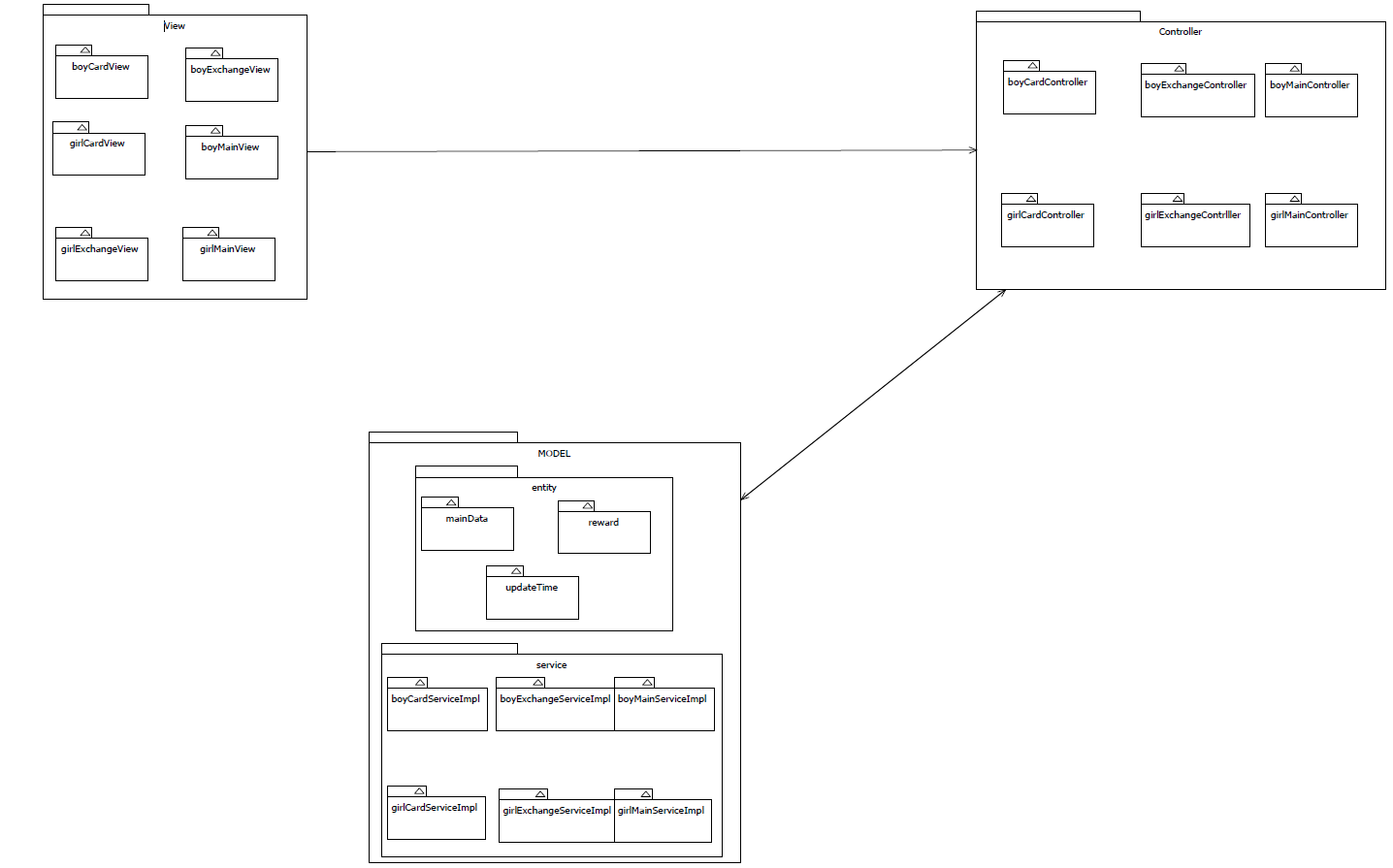
# 流程视图



# 部署视图



# 实施视图

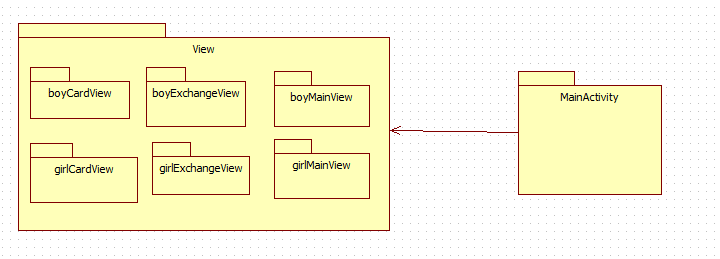


## 概述

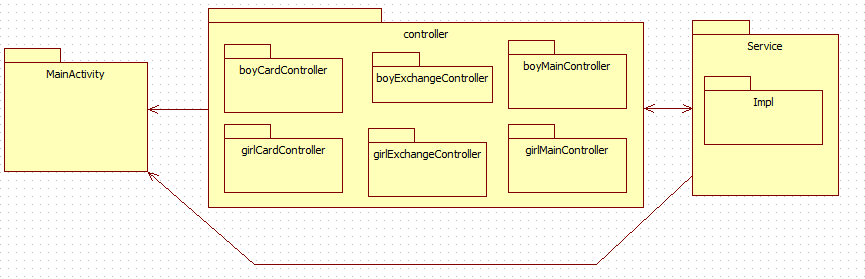
该程序包含了三个主要的层，分别是View层，用于向用户展示界面以及交互，第二个是Controller，用于存取数据，与Model进行交互，第三层是Model层，这一层摆好了数据的存储和访问控制，同时可以调用系统的API

## 层

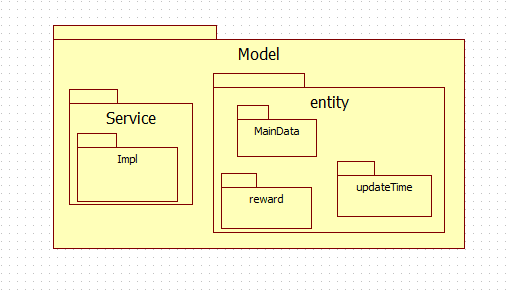
UI层



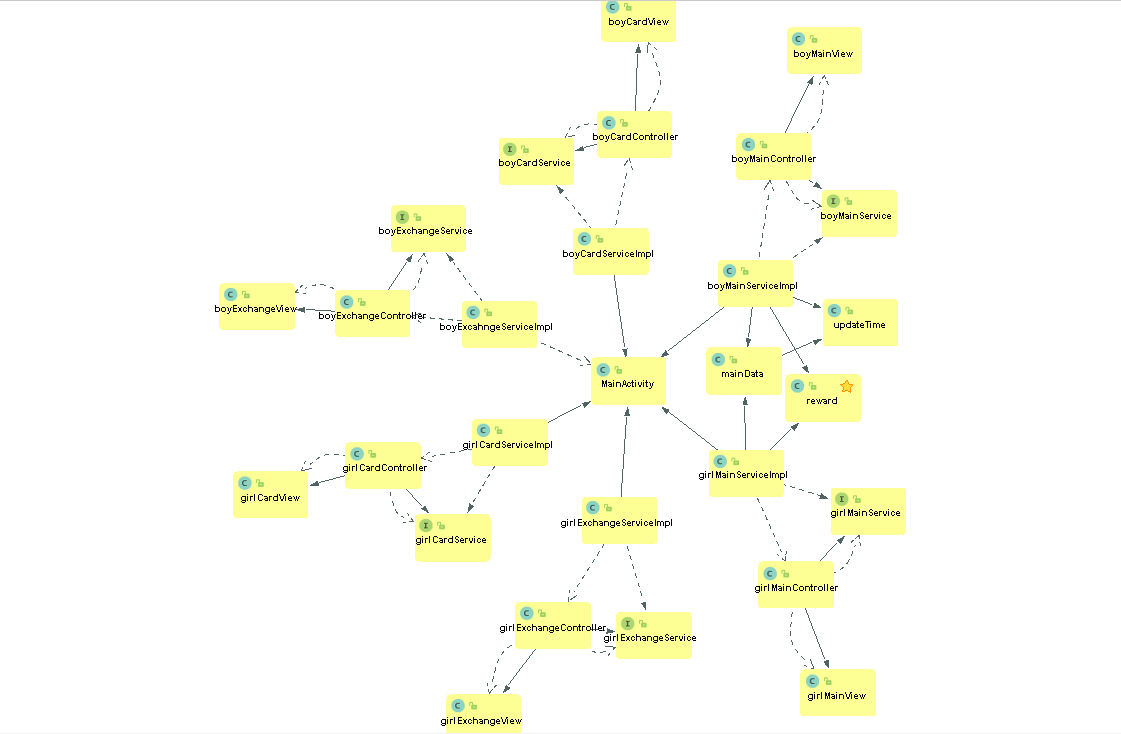
业务层



领域层



# 数据视图（可选）



# 大小和性能

并发访问数：系统每秒可完成100次并发处理

响应时间：系统对于用户交互事件的响应时间最多不超过1.5s

数据库一致性：保证用户的相关数据在数据库中的一致性

# 质量

易用性：

1.界面在简易清晰的基础上卡通化

2.用点击屏幕方式代表打卡记录

3.各个界面跳转只需点击屏幕即可

可扩展性：

1.passive mvc架构使系统内部各个功能模块之间耦合度下降，可以方便新增加的页面或功能，扩展性更强。

可维护性：

1.低耦合系统可独立维护修改某功能组件而不影响其他功能组件

2.对于本系统，用户要求系统维护人员在用户遇到打卡提交和响应时间缺陷时，在一个工作日内及时修复BUG