lab7 详细设计文档

1. 修订记录

姓名	修订版本	修订时间	修改内容
倪奕晨	v1.0.0	2022.07.01	创建并撰写引言、产品描述
邓楚宸	v1.2.0	2022.07.07	完成结构视角
何青云	v1.2.1	2022.07.08	补充动态模型
邓楚宸	v1.3.0	2022.07.09	完成依赖视角服务端
倪奕晨	v1.3.1	2022.07.09	补充依赖视角浏览器端
邓楚宸	v1.3.2	2022.07.10	修正结构视角部分内容

2. 引言

2.1 编制目的

本报告详细完成对ERP系统的详细设计，达到指导后续软件构造的目的，同时实现和测试人员及用户的沟通。

本报告面向开发人员，测试人员及最终用户而编写，是了解系统的导航和基础。

2.2 参考资料

1）骆斌,丁二玉,刘钦.软件开发的技术基础[M].机械工业出版社, 2012.

2）ERP系统体系结构文档

3. 产品描述

一民营企业专业从事灯具开关行业，是某著名开关品牌的南京地区总代理，主要在南京负责品牌的推广及项目的落地销售、分销的批发等工作，服务对象包括项目业主、施工单位、分销商、设计院、终端用户等。现公司规模扩大，企业业务量、办公场所、员工数都发生增长，希望可以帮助员工适应新的环境，提高工作效率和用户满意度。

该ERP系统就是为满足该公司新阶段业务发展要求而开发的，主要包括库存管理、销售管理、财务管理、人事管理和企业经营管理。

4. 结构视角

4.1 业务逻辑层的分解

开发包图见软件体系结构设计文档5.1

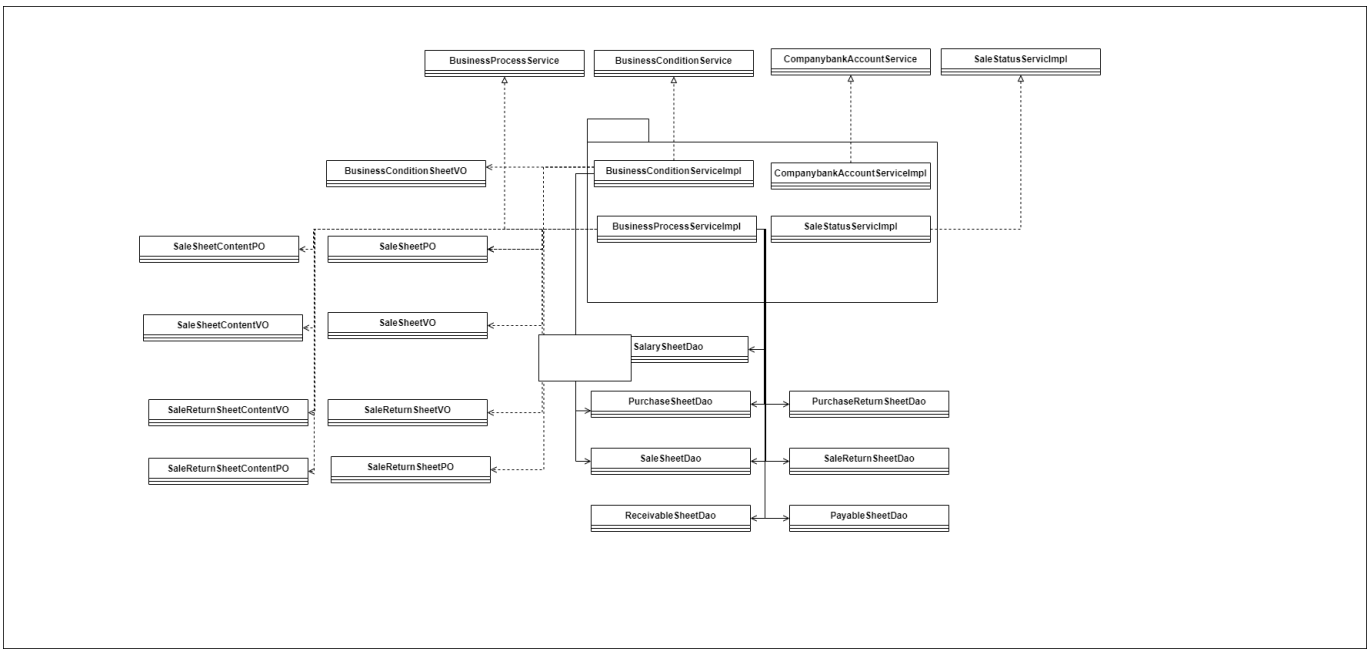
4.1.1 customerBusinessImpl模块（略）

4.1.2 overallManagementBusinessImpl模块

4.1.2.1 模块概述

根据分层的设计风格，系统后端分为Controller层（负责前后端的交互）、service层（接口）、serviceImpl层（负责具体业务逻辑的实现）、dao层（接口）、mapper层（负责与数据库交互），另有对应VO、PO等数据载体。（下同）

4.1.2.2 整体结构



4.1.2.3 内部类接口规范

账户管理：
service.overallManagementBusiness.CompanyBankAccountService
serviceImpl.overallManagementBusinessImpl.CompanyBankAccountServiceImpl

供接口：

```
    /**
     * 新建公司银行账户
     * @param companyBankAccountVO 公司银行账户
     */
    void createCompanyBankAccount(CompanyBankAccountVO companyBankAccountVO);
    /**
     * 删除公司银行账户
     * @param id 公司银行账户id
     */
    void deleteCompanyBankAccount(String id);
    /**
     * 修改账户属性(只能修改账户名)
     * @param companyBankAccountVO 公司银行账户
     */
    void changeCompanyBankAccountName(CompanyBankAccountVO companyBankAccountVO);
    /**
     * 根据账户名模糊查找
     * @param companyBankAccountName 公司账户名
     * @return 模糊查找得到的账户
     */
    List<CompanyBankAccountVO> getCompanyBankAccountByName(String
companyBankAccountName);

    List<CompanyBankAccountVO> getAllCompanyBankAccount();
```

需接口：
来自dao.overallManagementDao.CompanyBankAccountDao的：

```
    /**
     * 存入新增加的公司账户
     * @param toSave
     * @return 影响的行数
     */
    int saveAccount(CompanyBankAccountPO toSave);
    /**
     * 删除公司账户
     * @param companyBankAccountName
     * @return 影响的行数
     */
    int deleteAccount(String companyBankAccountName);
    /**
     * 修改公司账户属性（名称）
     * @param id, companyBankAccountName
```

```
     */
    int changeAccountName(String id, String companyBankAccountName);

    /**
     * 根据账户名模糊查找
     * @param companyBankAccountName 公司账户名
     * @return 模糊查找得到的账户
     */
    List<CompanyBankAccountPO> findAllByName(String companyBankAccountName);

    List<CompanyBankAccountPO> findAll();
```

销售明细表
service.overallManagementBusiness.SaleStatusService
service.overallManagementBusiness.SaleStatusServiceImpl

供接口：

```
     /**
     * 根据起始时间，结束时间，商品名，客户，业务员查询
     * @param
     * @return 销售明细列表
     */
    List<SaleStatusSheetVO> getSaleStatusSheet(String startTime, String endTime,
String name, Integer supplier, String salesman);
```

需接口

来自dao.warehouseDao.ProductDao的

```
    ProductPO findById(String id);

    ProductPO findByName(String name);
```

来自dao.saleDao.SaleSheetDao的

```
    /**
     * 用与选取符合条件的(销售明细列表
     * @param startTime
     * @param endTime
     * @param pid
     * @param supplier
     * @param salesman
     * @return
     */
    List<SaleStatusSheetPO> findSheet(Date startTime, Date endTime, String pid,
Integer supplier, String salesman);
```

经营历程表
service.overallManagementBusiness.BusinessProcessService
service.overallManagementBusiness.BusinessProcessServiceImpl

供接口：

```
    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 销售单
     */
    List<SaleSheetVO> findSaleSheet(String startTime, String endTime, Integer
supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 销售退货单
     */
    List<SaleReturnsSheetVO> findSaleReturnsSheet(String startTime, String endTime,
Integer supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 进货单
     */
    List<PurchaseSheetVO> findPurchaseSheet(String startTime, String endTime, Integer
supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 进货退货单
     */
    List<PurchaseReturnsSheetVO> findPurchaseReturnsSheet(String startTime, String
endTime, Integer supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 收款单
     */
    List<ReceivableSheetVO> findReceivableSheet(String startTime, String endTime,
Integer supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 付款单
```

```
     */
    List<PayableSheetVO> findPayableSheet(String startTime, String endTime, Integer
supplier, String salesman);

    /**
     * 根据起始时间，结束时间，客户，业务员查询
     * @param
     * @return 工资单
     */
    List<SalarySheetVO> findSalarySheet(String startTime, String endTime, Integer
supplier, String salesman);
```

需接口：

来自dao.saleDao.SaleSheetDao的

```
    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<SaleSheetPO> findSaleSheet(Date startTime, Date endTime, Integer supplier,
String salesman);

    /**
     * 查找指定销售单下具体的商品内容
     * @param sheetId
     */
    List<SaleSheetContentPO> findContentBySheetId(String sheetId);
```

来自dao.saleDao.SaleReturnsSheetDao的

```
    /**
     * 通过saleReturnsSheetId找到对应的content条目
     * @param saleReturnsSheetId
     * @return
     */
    List<SaleReturnsSheetContentPO> findContentBySaleReturnsSheetId(String
saleReturnsSheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
```

```
     * @return
     */
    List<SaleReturnsSheetPO> findSaleReturnsSheet(Date startTime, Date endTime,
Integer supplier, String salesman);
```

来自dao.purchaseDao.PurchaseSheetDao的

```
    List<PurchaseSheetContentPO> findContentByPurchaseSheetId(String purchaseSheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<PurchaseSheetPO> findPurchaseSheet(Date startTime, Date endTime, Integer
supplier, String salesman);
```

来自dao.purchaseDao.PurchaseSheetReturnsDao的

```
    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<PurchaseReturnsSheetPO> findPurchaseReturnsSheet(Date startTime, Date
endTime, Integer supplier, String salesman);

    /**
     * 通过purchaseReturnsSheetId找到对应的content条目
     * @param purchaseReturnsSheetId
     * @return
     */
    List<PurchaseReturnsSheetContentPO> findContentByPurchaseReturnsSheetId(String
purchaseReturnsSheetId);
```

来自dao.purchaseDao.PayableSheetDao的

```
    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
```

```
     * @param supplier
     * @param salesman
     * @return
     */
    List<PayableSheetPO> findPayableSheet(Date startTime, Date endTime, Integer
supplier, String salesman);

    List<PayableSheetContentPO> findContentBySheetId(String sheetId);


    List<ReceivableSheetContentPO> findContentBySheetId(String sheetId);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */
    List<ReceivableSheetPO> findReceivableSheet(Date startTime, Date endTime, Integer
supplier, String salesman);

    List<ReceivableSheetContentPO> findContentBySheetId(String sheetId);
```

来自dao.staffDao.SalarySheetDao的

```
    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param staffId
     * @param salesman
     * @return
     */
    List<SalarySheetPO> findSalarySheet(Date startTime, Date endTime, Integer staffId,
String salesman);
```

经营情况表
service.overallManagementBusiness.BusinessConditionService
service.overallManagementBusiness.BusinessConditionServiceImpl

供接口：

```
    /**
     * 根据起始时间，结束时间
     * @param
     * @return 经营情况
     */
    BusinessConditionSheetVO getBusinessConditionSheet(String startTime, String
endTime);
```

需接口：

来自dao.saleDao.SaleSheetDao的

```
    /**
     * 根据时间返回销售单
     * @param startTime,endTime
     * @return 销售列表
     */
    List<SaleSheetPO> findAllByTime(Date startTime, Date endTime);
```

来自dao.staffDao.SalarySheetDao的

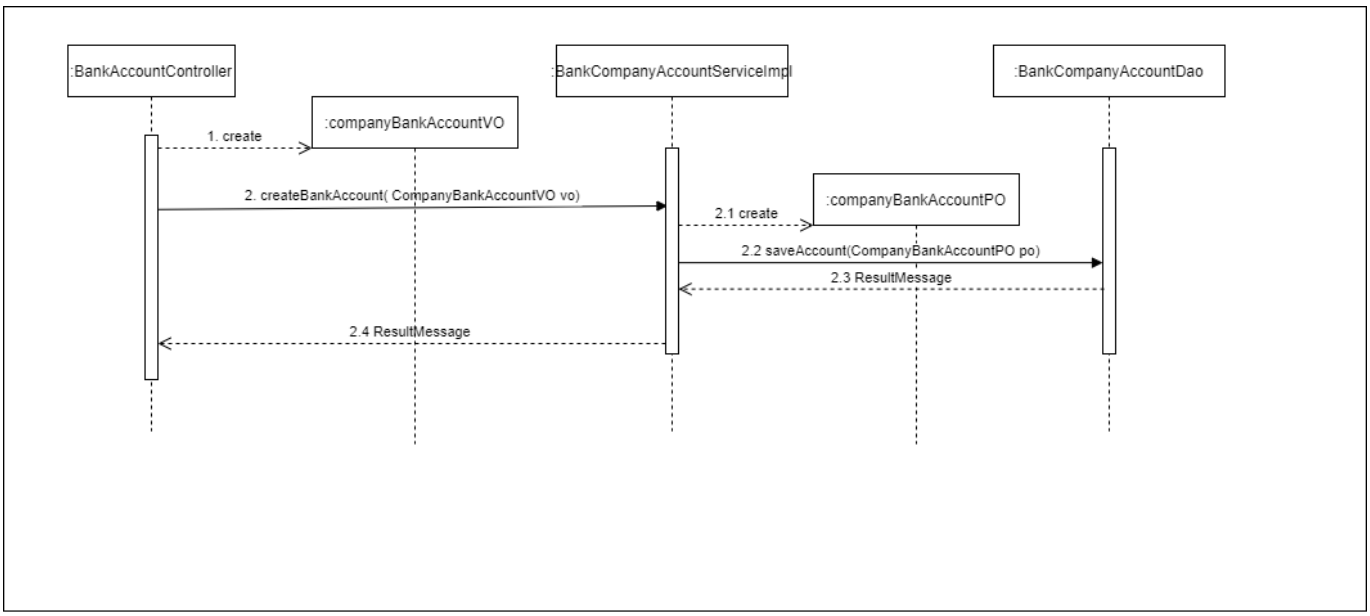
```
    /**
     * 根据时间返回工资单
     * @param startTime,endTime
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByTime(Date startTime, Date endTime);
```

来自dao.purchaseDao.PurchaseSheetDao的

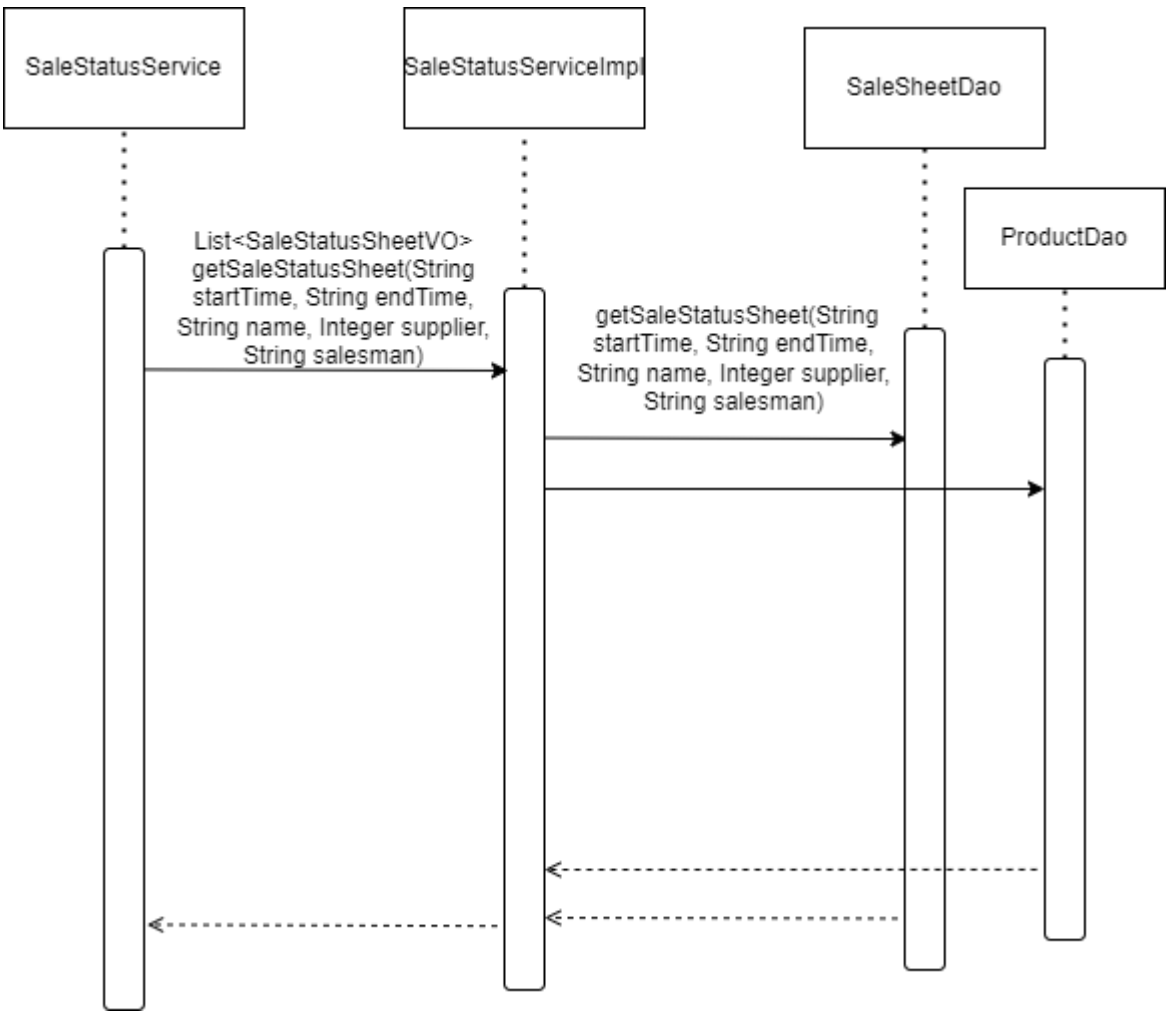
```
    /**
     * 根据时间返回进货单
     * @param startTime,endTime
     * @return 进货列表
     */
    List<PurchaseSheetPO> findAllByTime(Date startTime, Date endTime);
```

4.1.2.4 动态模型

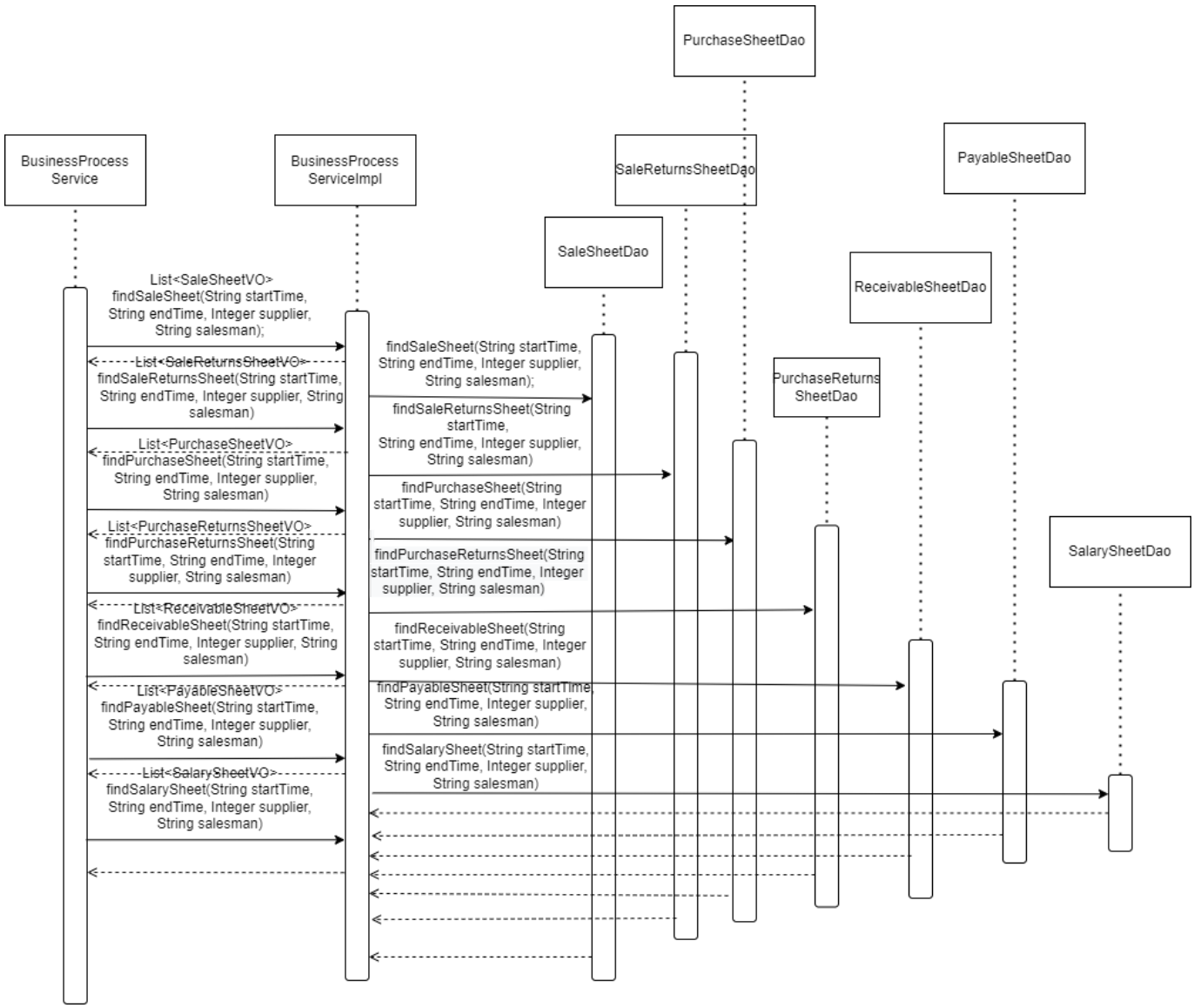
账户管理：通过controller调用serviceImpl函数，service调用Dao层接口实现，以增加一账户为例。



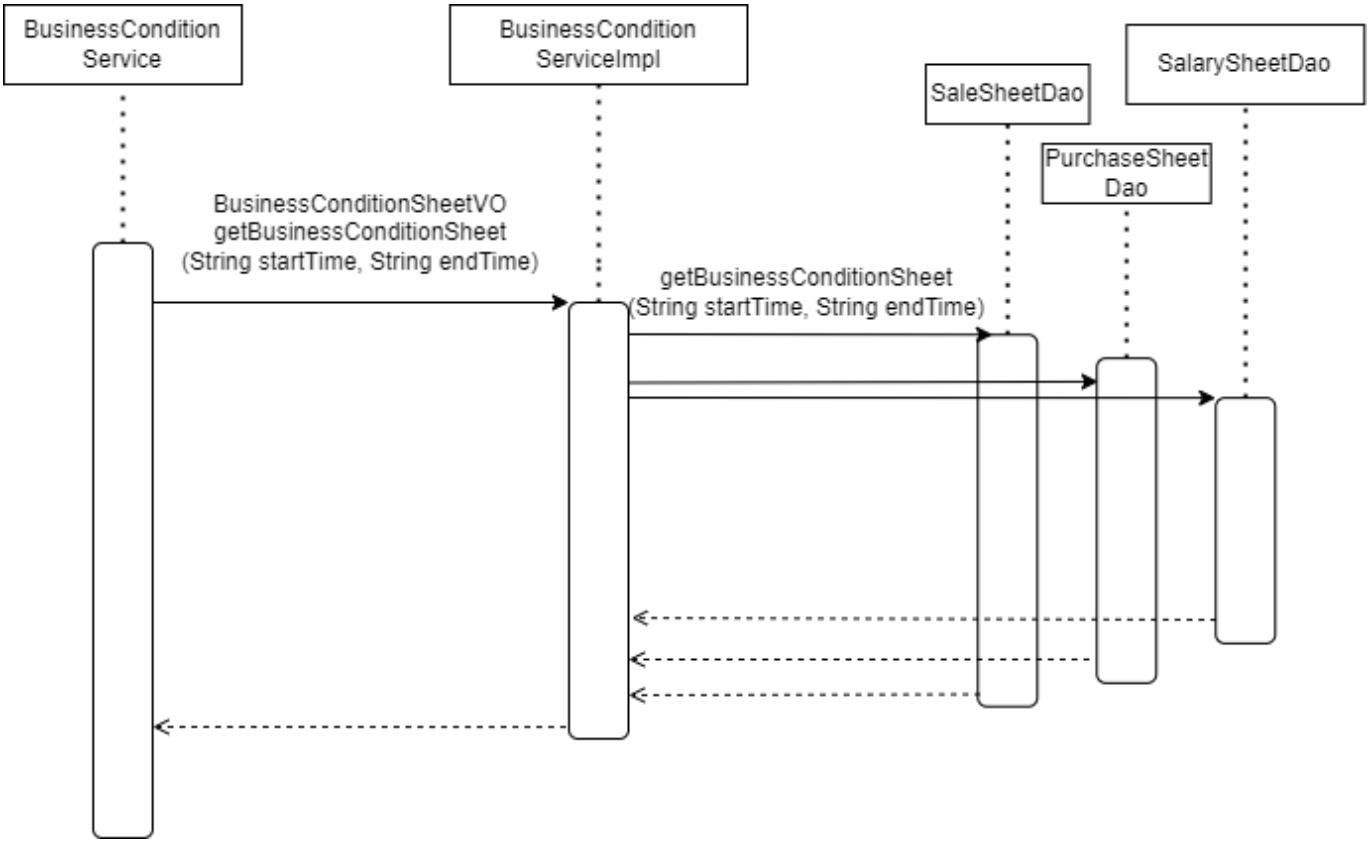
查看销售明细：service通过调用Dao层实现serviceImpl中的各方法



查看经营历程：service通过调用Dao层实现serviceImpl中的各方法



查看经营情况：service通过调用Dao层实现serviceImpl中的各方法

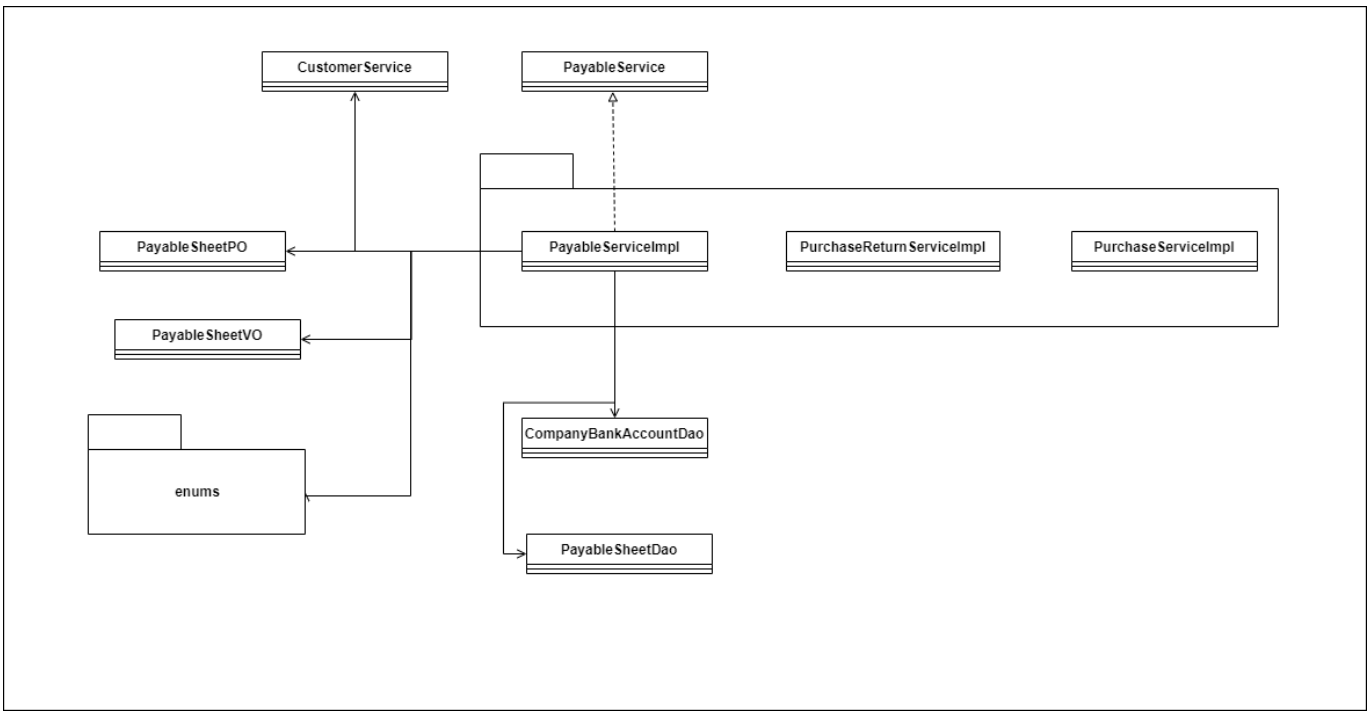


4.1.3 purchaseBusinessImpl模块

4.1.3.1 模块概述

略

4.1.3.2 整体结构



4.1.3.3 内部类接口规范

制订、审批、查看付款单

service.purchaseManagementBusiness.PayableService
serviceImpl.purchaseManagementBusinessImpl.PayableImpl

供接口：

```
    /**
     * 制定付款单
     * @param payableSheetVO 付款单
     */
    void makePayableSheet(UserVO userVO, PayableSheetVO payableSheetVO);

    /**
     * 根据付款单id进行审批(state == "审批完成"/"审批失败")
     * 在controller层进行权限控制
     * @param payableSheetId 付款单id
     * @param state 付款单修改后的状态
     */
    void approval(String payableSheetId, PayableSheetState state);

    List<PayableSheetVO> getAllPayableSheet();
```

需接口：
来自dao.purchaseDao.PayableSheetDao的

```
     /**
     * 存入一条付款单记录
     * @param toSave 一条付款单记录
     * @return 影响的行数
     */
    int save(PayableSheetPO toSave);
    /**
     * 把付款单上条目清单的存入数据库
     * @param payableSheetContentPO 付款单上的条目清单
     */
    void saveBatch(List<PayableSheetContentPO> payableSheetContentPO);
    /**
     * 获取最近一条付款单
     * @return 最近一条付款单
     */
    PayableSheetPO getLatest();


    int updateState(String payableSheetId, PayableSheetState state);


    PayableSheetPO findOneById(String payableSheetId);

    /**
     * 用与选取符合条件的(经营历程列表
```

```
     * @param startTime
     * @param endTime
     * @param supplier
     * @param salesman
     * @return
     */

    List<PayableSheetPO> findPayableSheet(Date startTime, Date endTime, Integer
supplier, String salesman);

    List<PayableSheetPO> findAll();

    List<PayableSheetContentPO> findContentBySheetId(String sheetId);
```

来自dao.overallManagementDao.CompanyBankAccountDao的：

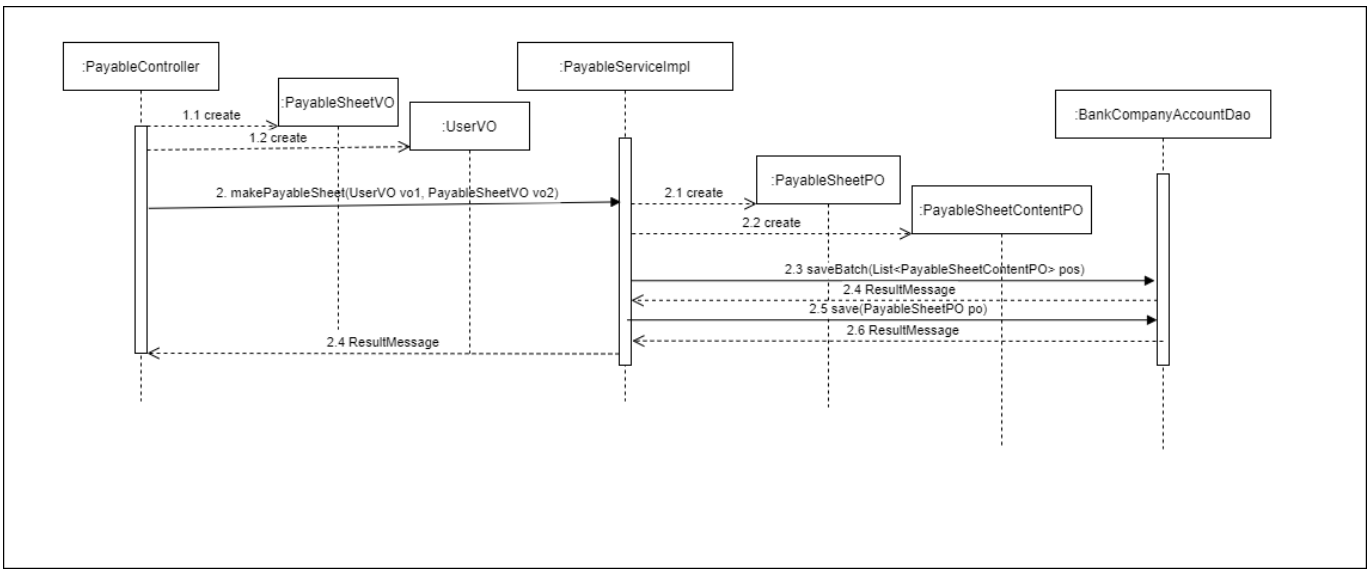
```
void reduceAccountMoney(String bankAccount, BigDecimal totalAmount);
```

来自service.customerManagementBusiness.CustomerServiceImpl的：

```
CustomerPO findCustomerById(String customerId);

void updateCustomer(CustomerPO po);
```

4.1.3.4 动态模型

制订、审批、查看付款单。

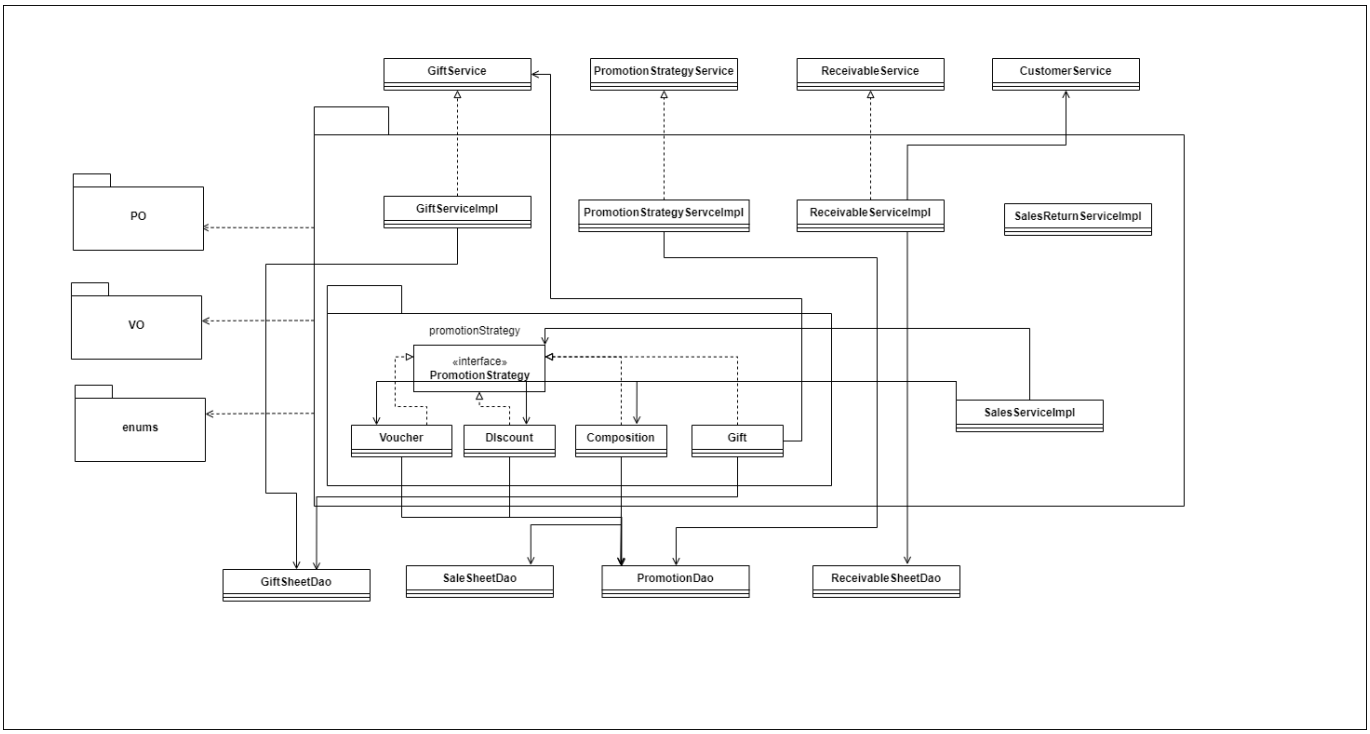


4.1.4 saleBusinessImpl模块

4.1.4.1 模块概述

略

4.1.4.2 整体结构



4.1.4.3 内部类接口规范

制订促销策略：

serviceImpl.saleManagementBusinessImpl.PromotionStrategyServiceImpl

供接口：

```
    /**
     * 制订代金券策略
     * @param po
     */
    void addVoucher(PromotionPO po);

    /**
     * 制订赠送策略
     * @param po
     */
    void addGift(PromotionPO po);

    /**
     * 制订折扣策略
     * @param po
     */
    void addDiscount(PromotionPO po);
```

需接口：

来自dao.saleDao.SalarySheetDao

供接口：

```
    /**
     * 获取最近一条工资单
     */
    SalarySheetPO getLatestSheet();

    /**
     * 存入一条工资单记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveSheet(SalarySheetPO toSave);

    /**
     * 查找所有工资单（根据状态查找时若为null需要用到这里
     */
    List<SalarySheetPO> findAllSheet();

    /**
     * 根据state返回工资单
     * @param state 工资单状态
```

```
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByState(SalarySheetState state);

    /**
     * 查找指定id的工资单
     * @param id id
     */
    SalarySheetPO findSheetById(String id);

    /**
     * 根据时间返回工资单
     * @param startTime,endTime
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByTime(Date startTime, Date endTime);

    /**
     * 更新指定工资单的状态
     * @param sheetId 编号
     * @param state 状态
     */
    int updateSheetState(String sheetId, SalarySheetState state);

    /**
     * 制订薪酬规则
     */

    int makeSalaryRule(String sheetId, SalaryRule rule);
```

制订、审批、查看库存赠送单

serviceImpl.saleManagementBusinessImpl.GiftServiceImpl

供接口:

```
    /**
     * 根据单据状态获取库存赠送单
     * @param state
     * @return
     */
    List<GiftSheetPO> getGiftSheetByState(GiftSheetState state);

    /**
     * 审批单据
     * @param giftSheetId
     * @param state
     */
    void approval(String giftSheetId, GiftSheetState state);


    /**
     * 根据销售单Id搜索库存赠送单信息
     * @param giftSheetId Id
     * @return 库存赠送单全部信息
     */
    GiftSheetPO getGiftSheetById(String giftSheetId);
```

需接口：
来自dao.saleDao.GiftSheetDao的：

```
    /**
     *  根据id寻找
     * @param giftSheetId
     * @return
     */
    GiftSheetPO findOneById(String giftSheetId);

    /**
     * 审批
     * @param id
     */
    void updateState(String id, GiftSheetState state);


    /**
     * 根据状态查询所有库存赠送单
     * @return
```

```
     */

    List<GiftSheetPO> findAllByState(GiftSheetState state);
```

制订、审批、查看收款单

serviceImpl.saleManagementBusinessImpl.ReceivableImpl

供接口：

```
     /**
     * 制定收款单
     * @param receivableSheetVO 收款单
     */
    void makeReceivableSheet(UserVO userVO, ReceivableSheetVO receivableSheetVO);

    /**
     * 根据收款单id进行审批(state == "审批完成"/"审批失败")
     * 在controller层进行权限控制
     * @param receivableSheetId 收款单id
     * @param state 收款单修改后的状态
     */
    void approval(String receivableSheetId, ReceivableSheetState state);

    List<ReceivableSheetVO> getAllReceivableSheet();
```

需接口：

来自dao.saleDao.ReceivableSheetDao的

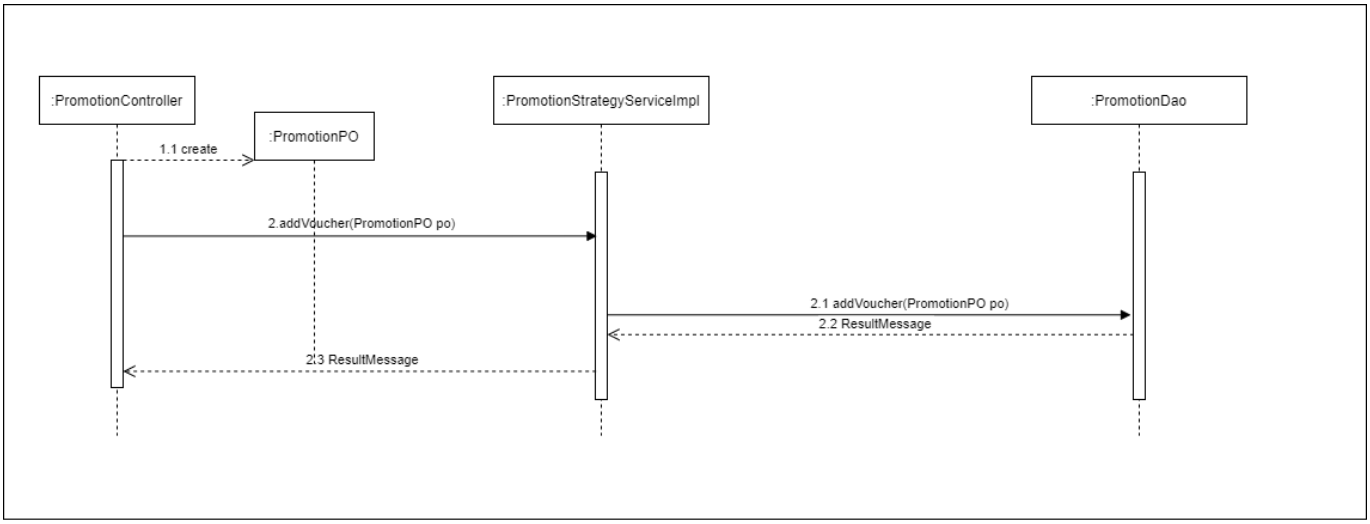
```
    /**
     * 存入一条收款单记录
     * @param toSave 一条收款单记录
     * @return 影响的行数
     */
    int save(ReceivableSheetPO toSave);
    /**
     * 把收款单上转账列表的存入数据库
     * @param receivableSheetContentPO 收款单上的转账列表
     */
    void saveBatch(List<ReceivableSheetContentPO> receivableSheetContentPO);
    /**
     * 获取最近一条收款单
     * @return 最近一条收款单
     */
    ReceivableSheetPO getLatest();


    int updateState(String receivableSheetId, ReceivableSheetState state);


    ReceivableSheetPO findOneById(String receivableSheetId);

    List<ReceivableSheetPO> findAll();

    List<ReceivableSheetContentPO> findContentBySheetId(String sheetId);
```

使用销售策略
serviceImpl.salesBusinessImpl.PromotionStrategy
需接口:
来自dao.saleDao.PromotionDao的

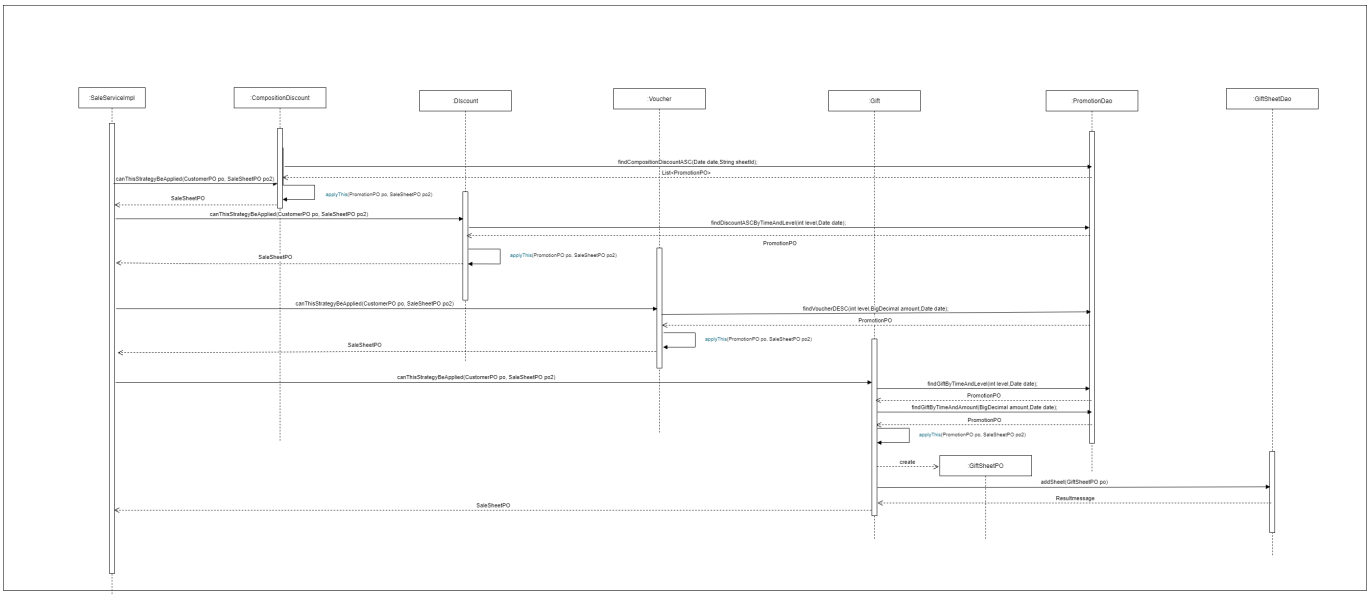
```
 /**
- 查找符合条件的最大的代金券
- @param level 客户等级
- @param date 使用时间
- @param amount 总金额（raw）
- @return
  */
  PromotionPO findVoucherDESCByTimeAndLevel(int level,BigDecimal amount,Date date);


/**
 * 查找符合条件的商品组合降价策略
 * @param date 使用时间
 * @param sheetId sale sheet id
 * @return
 */
List<PromotionPO> findCompositionDiscountASC(Date date,String sheetId);

/**
 * 查找符合条件的最大折扣
 * @param level 客户等级
 * @param date 使用时间
 * @return
 */
PromotionPO findDiscountASCByTimeAndLevel(int level,Date date);

/**
 * 寻找符合条件的赠品策略,并选取对等级要求最严的一个
 * @param level 客户等级
 * @param date 时间
 * @return
 */
PromotionPO findGiftByTimeAndLevel(int level,Date date);

/**
 * 寻找符合条件的赠品策略，并选取对总金额要求最严格的一个
 * @param amount 总金额
 * @param date 时间
 * @return
 */
PromotionPO findGiftByTimeAndAmount(BigDecimal amount,Date date);
```

4.1.4.4 动态模型

制订销售策略，以制订代金券销售策略为例。



使用销售策略。



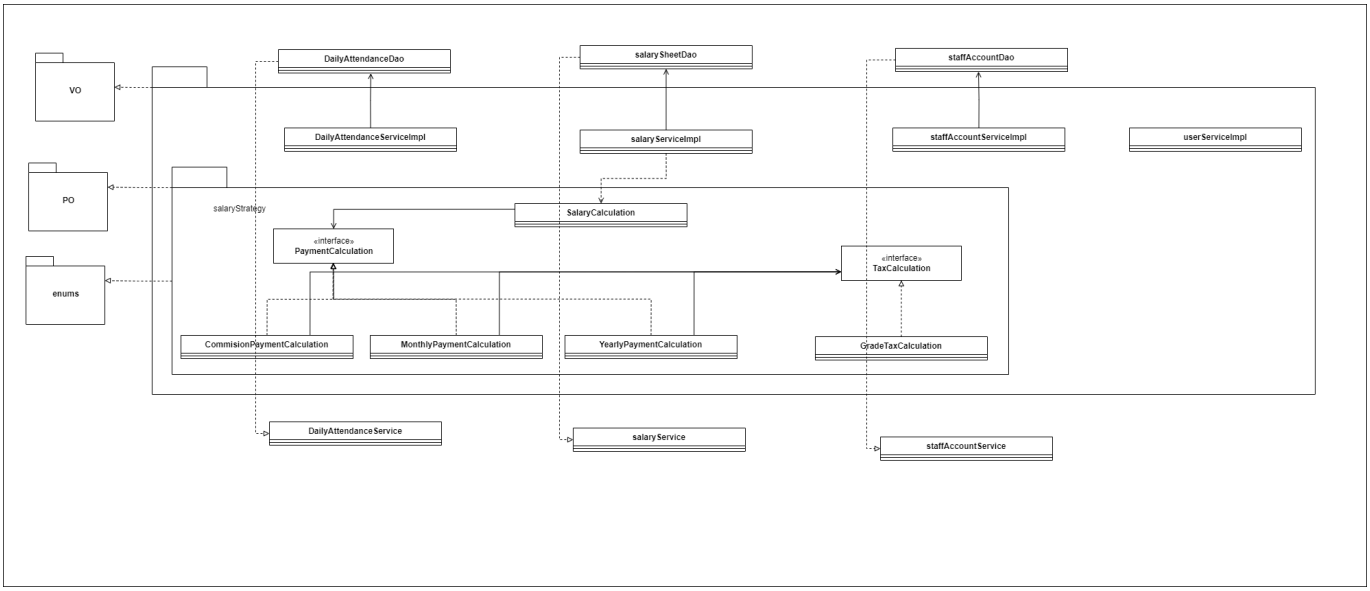
制订审批查看库存赠送单据、制订审批查看收款单与其余单据类似。

4.1.5 staffBusinessImpl模块

4.1.5.1 模块概述

略

4.1.5.2 整体结构



4.1.5.3 内部类接口规范

员工打卡：
serviceImpl.staffbusinessImpl.DailyAttendanceServiceImpl

供接口：

```
    /**
     *  查询某人是否某日已打卡
     * @param staffId 员工id
     * @param date
     */
    boolean doesThisDayHeAttend(String staffId, String date);

    /**
     * 查询所有员工某年某月打卡记录
     * @param date
     */
    List<AttendanceVO> findALLStaffAttendance(String date);

    /**
     *  查询某位员工某年某月打卡记录
     * @param date
     * @param staffId 员工id
     */
    List<AttendanceVO> findOneStaffAttendance(String date,String staffId);

    /**
     *  打卡
     * @param staffId 员工id
     */
    void attendanceRegister(String staffId,String name,String role);

    String findIdByName(String name);
```

需接口：

来自dao.staffDao.DailyAttendanceDao的：

```
doesThisDayHeAttend(String staffId,Date beginDate,Date endDate);
findAll(Date beginDate,Date endDate)
findOne(Date beginDate,Date endDate,String staffId)
findIdByName(String staffName)
AttendanceRegister(AttendancePo po);
```

员工管理：增加改变查询员工信息
service.staffBusiness.StaffAccountService
serviceImpl.staffBusinessImpl.StaffAccountServiceImpl

供接口：

```
    /**
     * 登记入职员工信息，系统创建账户
     * @param staffInformationVO
     */
    void creatStaffAccount(StaffInformationVO staffInformationVO);
    /**
     * 修改入职员工信息
     * @param staffInformationVO
     */
    void changeStaffInfo(StaffInformationVO staffInformationVO);
    /**
     * 根据staffId找员工
     * @param staffId
     */
    StaffInformationPO findByStaffId(Integer staffId);

    List<StaffInformationVO> showAllStaffAccount();
```

需接口：

来自dao.staffDao.StaffAccountDao的：

```
     /**
     * 存入一条员工信息记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveAccount(StaffInformationPO toSave);

    /**
     * 根据id寻找员工
     * @param staffId
     * @return
     */
    StaffInformationPO findById(Integer staffId);

    /**
     * 更新员工信息
     * @param toChange
     * @return
     */
    int changeInfo(StaffInformationPO toChange);

    List<StaffInformationPO> findAll();
```

制订薪酬规则，制订、审批、查看工资单

serviceImpl.staffManagementBusinessImpl.SalaryServiceImpl

供接口：

```
    /**
     * 制定工资单
     * @param salarySheetVO 工资单
     */
    void makeSalarySheet(UserVO userVO, SalarySheetVO salarySheetVO);

    /**
     * 根据状态获取工资单(state == null 则获取所有工资单)
     * @param state 工资单状态
     * @return 工资单
     */
    List<SalarySheetVO> getSalarySheetByState(SalarySheetState state);

    /**
     * 根据工资单id进行审批(state == "审批完成"/"审批失败")
     * 在controller层进行权限控制
     * @param salarySheetId 工资单id
     * @param state 工资单修改后的状态
     */
    void approval(String salarySheetId, SalarySheetState state);

    /**
     * 根据工资单Id搜索进货单信息
     * @param salarySheetId 工资单Id
     * @return 工资单全部信息
     */
    SalarySheetVO getSalarySheetById(String salarySheetId);

    /**
     * 改变id为staffId员工的薪酬规则为rule
     * @param staffId 员工Id
     * @param rule 改变后的规则，间SalaryRule枚举类
     */
    void makeSalaryRule(String staffId, SalaryRule rule);
```

需接口：

来自dao.staffDao.SalarySheetDao的

```
    /**
     * 获取最近一条工资单
     */
    SalarySheetPO getLatestSheet();

    /**
     * 存入一条工资单记录
```

```
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveSheet(SalarySheetPO toSave);

    /**
     * 查找所有工资单（根据状态查找时若为null需要用到这里
     */
    List<SalarySheetPO> findAllSheet();

    /**
     * 根据state返回工资单
     * @param state 工资单状态
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByState(SalarySheetState state);

    /**
     * 查找指定id的工资单
     * @param id id
     */
    SalarySheetPO findSheetById(String id);

    /**
     * 根据时间返回工资单
     * @param startTime,endTime
     * @return 工资单列表
     */
    List<SalarySheetPO> findAllByTime(Date startTime, Date endTime);

    /**
     * 更新指定工资单的状态
     * @param sheetId 编号
     * @param state 状态
     */
    int updateSheetState(String sheetId, SalarySheetState state);

    /**
     * 制订薪酬规则
     */

    int makeSalaryRule(String sheetId, SalaryRule rule);

    /**
     * 用与选取符合条件的(经营历程列表
     * @param startTime
     * @param endTime
     * @param staffId
     * @param salesman
     * @return
     */
```

```
    List<SalarySheetPO> findSalarySheet(Date startTime, Date endTime, Integer staffId,
String salesman);
```

员工管理：增加改变查询员工信息
service.staffBusiness.StaffAccountService
serviceImpl.staffBusinessImpl.StaffAccountServiceImpl

供接口：

```
    /**
     * 登记入职员工信息，系统创建账户
     * @param staffInformationVO
     */
    void creatStaffAccount(StaffInformationVO staffInformationVO);
    /**
     * 修改入职员工信息
     * @param staffInformationVO
     */
    void changeStaffInfo(StaffInformationVO staffInformationVO);
    /**
     * 根据staffId找员工
     * @param staffId
     */
    StaffInformationPO findByStaffId(Integer staffId);

    List<StaffInformationVO> showAllStaffAccount();
```

需接口

来自dao.staffDao.StaffAccountDao的

```
    /**
     * 存入一条员工信息记录
     * @param toSave 一条工资单记录
     * @return 影响的行数
     */
    int saveAccount(StaffInformationPO toSave);

    /**
     * 根据id寻找员工
     * @param staffId
     * @return
     */
    StaffInformationPO findById(Integer staffId);

    /**
     * 更新员工信息
     * @param toChange
     * @return
     */
    int changeInfo(StaffInformationPO toChange);

    List<StaffInfoPO> findAll();
```

制订年终奖
serviceImpl.staffBusinessImpl.AnnualBonusServiceImpl

供接口：

```
    /**
     * 设置年终奖
     * @param staffId
     * @param annualAmount
     */
    void setAnnualAmount(String staffId, BigDecimal annualAmount);

    /**
     * 根据员工id查找年终奖
     * @param staffId
     * @return
     */
    BigDecimal findAnnualAmountById(String staffId);
```

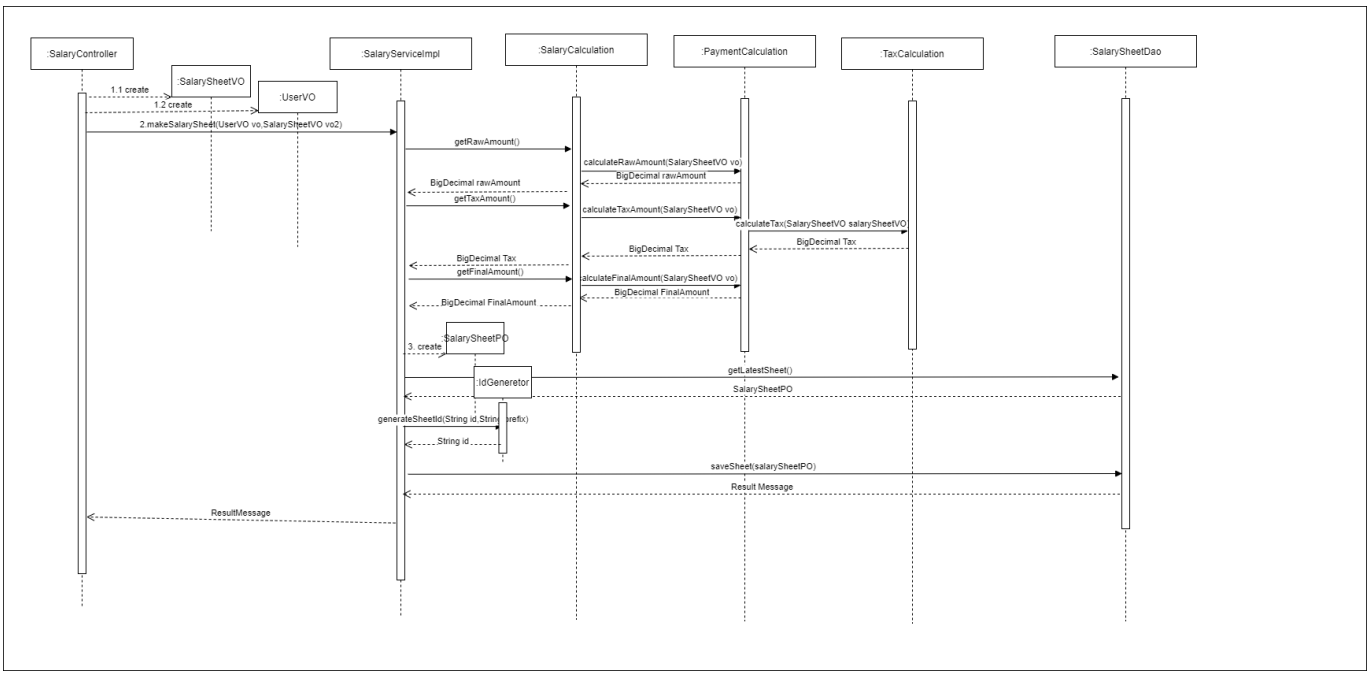
需接口：
来自dao.staffDao.AnnualBonusDao的
save(AnnualBonusPO po);
getAmount(Sting staffId)

4.1.5.4 动态模型

员工进行打卡；查询打卡信息（略）

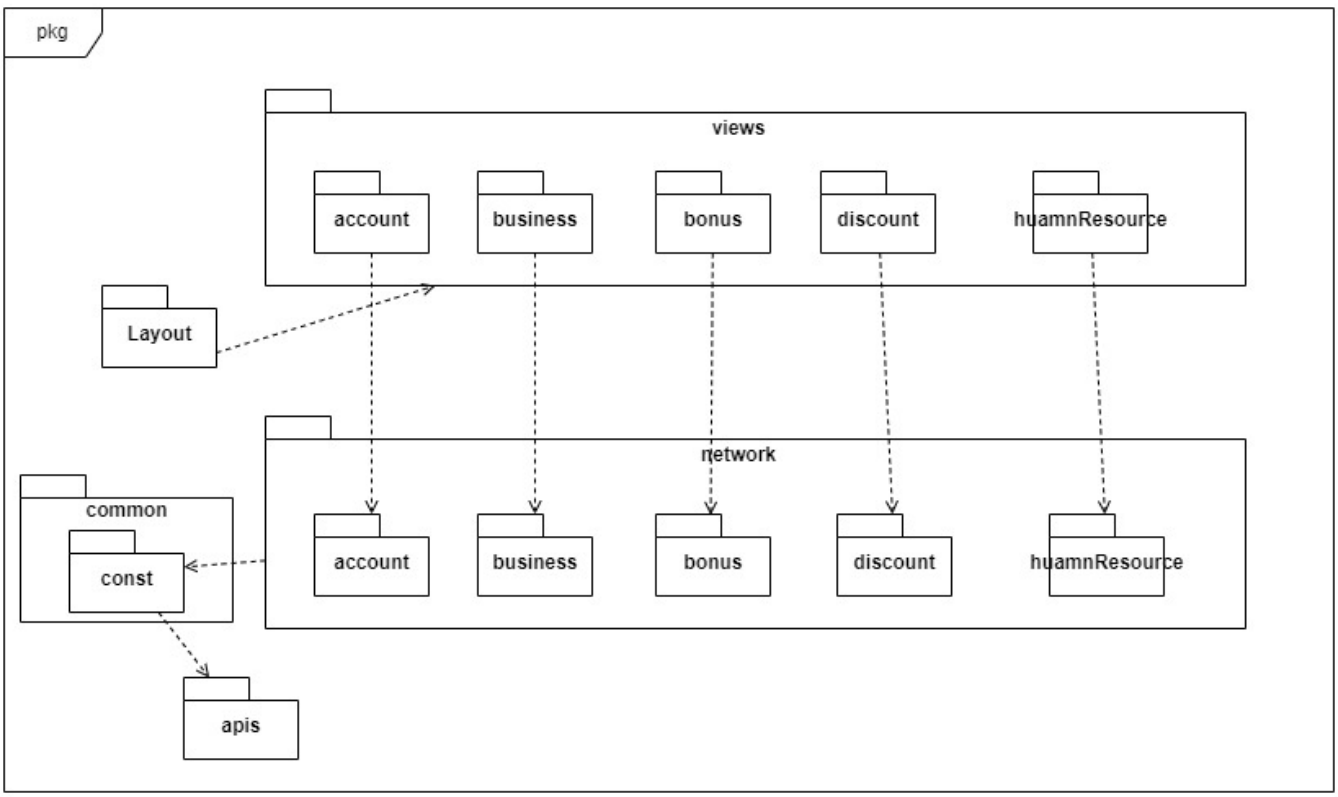
增加改变员工信息（略）

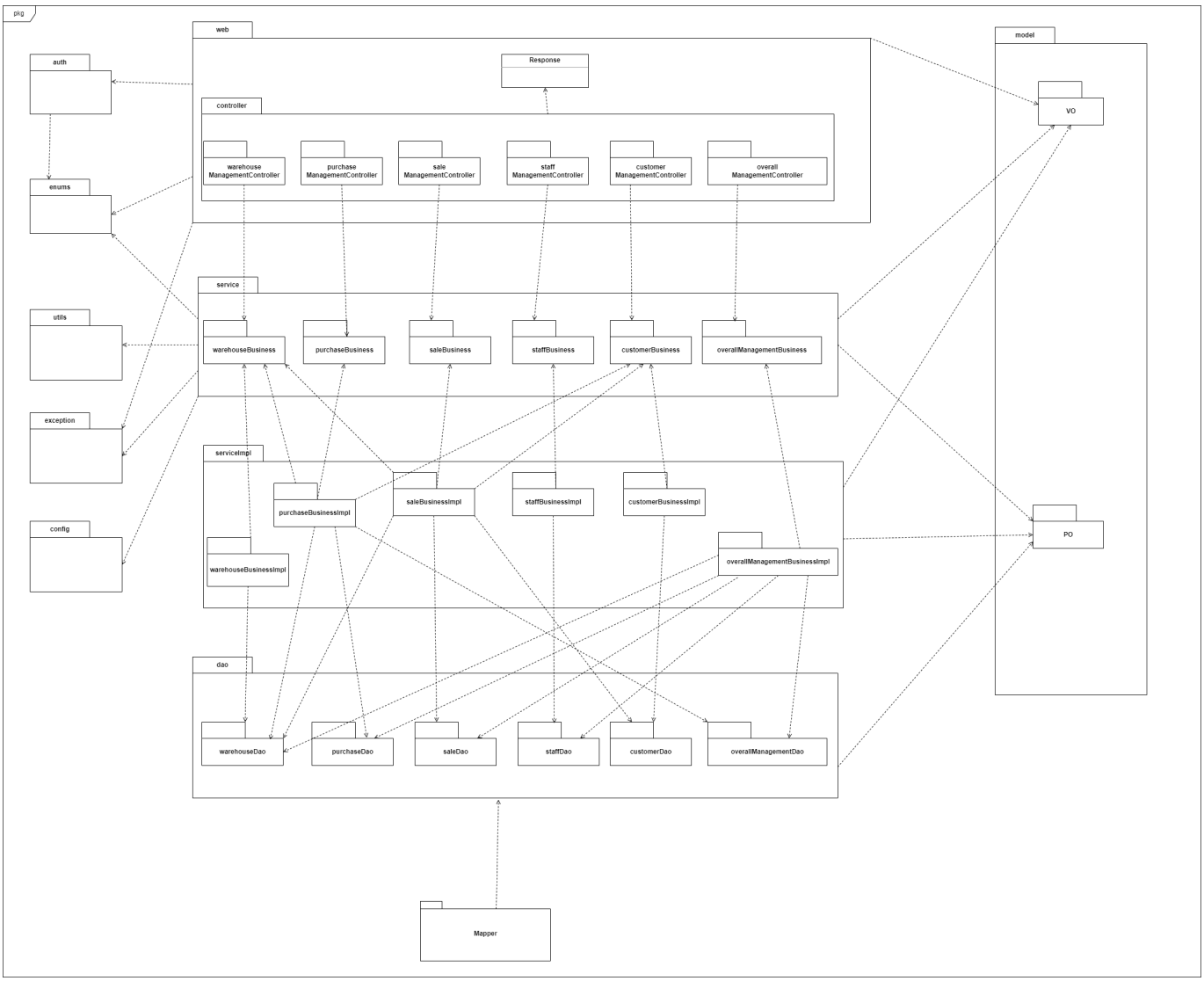
使用工资策略，顺序图如下



4.1.6 warehouseBusinessImpl模块(略)

5. 依赖视角




pkg
web
auth
Response
model
VO
controller
warehouse ManagementController
purchase ManagementController
sale ManagementController
staff ManagementController
customer ManagementController
overall ManagementController
enums
service
utils
warehouseBusiness
purchaseBusiness
saleBusiness
staffBusiness
customerBusiness
overallManagementBusiness
exception
serviceImpl
purchaseBusinessImpl
saleBusinessImpl
staffBusinessImpl
customerBusinessImpl
config
warehouseBusinessImpl
overallManagementBusinessImpl
PO
dao
warehouseDao
purchaseDao
saleDao
staffDao
customerDao
overallManagementDao
Mapper