胡炫光
港澳

/thoughtworks

1 业务分析和建模

/thoughtworks

业务分析和建模

业务分析

/thoughtworks

业务背景

本页概括描述业务需求

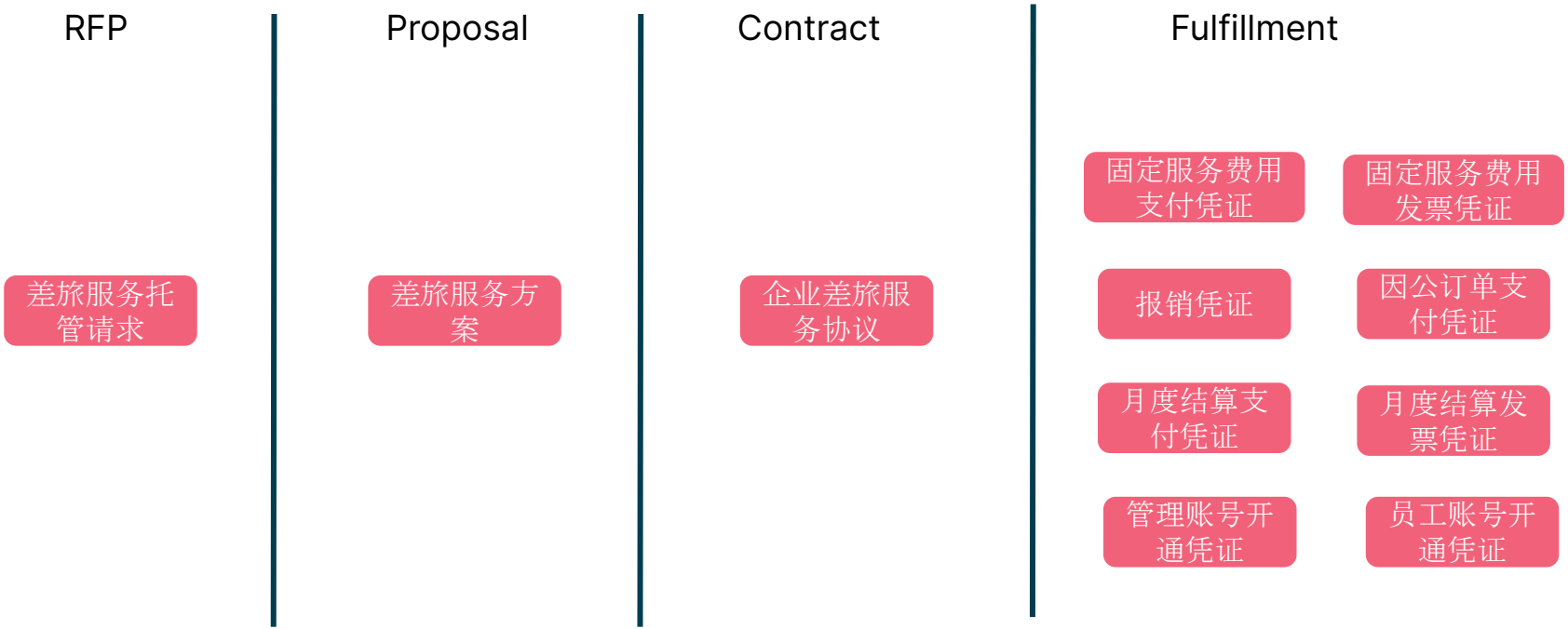
“任你行”差旅服务公司希望我司为其设计与开发一套数字化平台，该平台借助“任你行”差旅服务公司拥有的海量机票/酒店/专车服务提供商，为中小企业客戶提供差旅预订与管理服务，并通过收取一定的服务费实现盈利。

现有主要业务场景：

- 企业客户的管理人员可以在平台中完成企业配置、差旅管理、财务结算等。
- 企业客户的出行人员可以使用“任你行”进行因公和因私的机票服务的预订，如因公，则按企业预配置的流程，由部门主管完成审批，“任你行”代企业支付完成预订过程。如因私，则按个人订票流程，无须审批，由出行人员自行支付完成过程。
- 企业客户需按年为周期向“任你行”支付固定服务费用，并按月为周期向“任你行”结算上一个自然月的因公订单。

企业差旅服务协议 - 合约分析

寻找凭证与合同，分析合约上下文



企业差旅服务协议 - 合约文本

明确业务的权责关系:权利方的权利、责任方的责任以及履约时限等

甲方:企业

乙方:任你行平台

权责说明:

- 固定服务费用支付凭证:甲方需在协议签订后5个工作日内通过银联对公转账向乙方按签订的服务周期支付固定服务费用,否则乙方有权终止协议。
- 固定服务费用发票凭证:甲方在支付固定服务费用后,可向乙方申请开具固定服务费用发票,乙方在收到申请后,需在24小时内完成发票开具,否则甲方有权追究法律责任。
- 报销凭证:乙方需在每月3日前向甲方提供上一个自然月所产生的所有行程单作为报销凭证,否则甲方有权追究法律责任。
- 因公订单支付凭证:甲方员工选择因公支付订单后,乙方需在15分钟内代甲方支付订单费用,否则订单失效且甲方有权追究法律责任。
- 月度结算支付凭证:甲方需在每月10日前根据乙方提供的上一个自然月所有因公订单支付凭证进行结算,并通过银联对公转帐向乙方支付费用,否则乙方有权终止协议且追究甲方法律责任。
- 月度结算发票凭证:甲方在支付月度结算费用后,可向乙方申请开具月度结算费用发票,乙方在收到申请后,需在24小时内完成发票开具,否则甲方有权追究法律责任。
- 管理账号开通凭证:在固定服务费用支付后,甲方可向乙方提供企业员工信息,乙方在收到甲方提供的员工信息的5个工作日内,完成企业信息录入并开通管理账号,否则甲方有权终止协议。
- 员工账号开通凭证:甲方管理员完成企业初始化配置后,乙方需在5个工作日内通过短信通知甲方员工账号已开通,并在员工登录成功后提供商旅卡号,否则甲方有权追究责任。

机票订单买卖合同 - 合约分析

寻找凭证与合同，分析合约上下文

机票订单买卖合同 - 合约文本

明确业务的权责关系:权利方的权利、责任方的责任以及履约时限等

甲方:企业出行人

乙方:任你行平台

权责说明:

- 因公机票订单支付凭证:乙方需在合同签订后15分钟内代甲方所在公司支付订单费用，否则合同终止且甲方所在公司有权追究责任。
- 因私机票订单支付凭证:甲方需在合同签订后15分钟内支付订单费用，否则合同终止。
- 机票行程单凭证:机票订单支付完成后，乙方需在5分钟内向机票承运公司完成预定并生成机票行程单，且需告知甲方机票行程单信息，否则甲方有权追究赔偿。

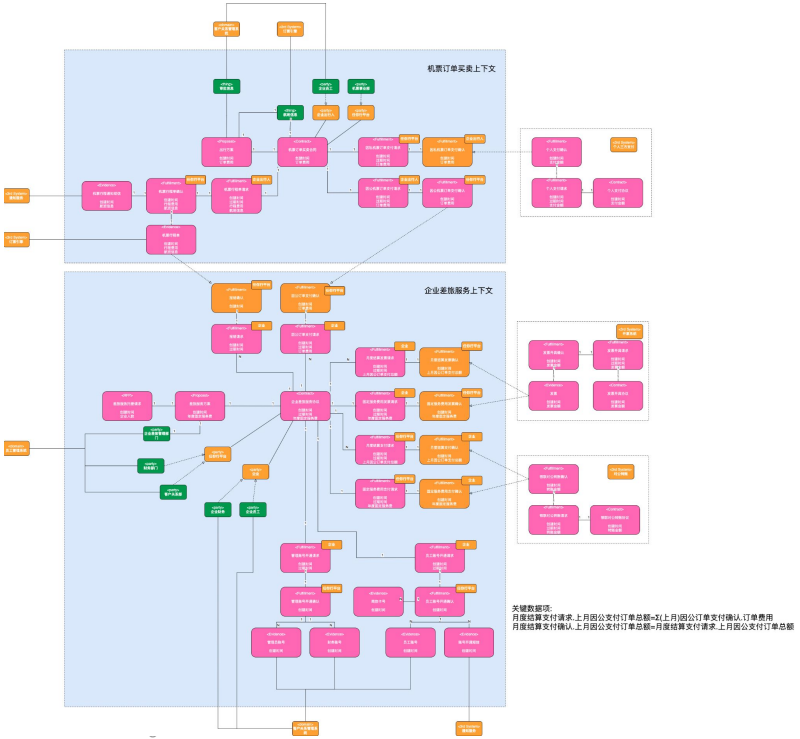
业务分析和建模

业务建模

/thoughtworks

业务建模图

按照图例和规范绘制建模图、描述合同、履约凭证、参与者、角色、领域逻辑/三方系统/凭证角色化以及它们之间的数量关系，使用划线标出上下文边界，并为每个上下文命名。凭证名称使用中文。



大图见:[业务建模图]

2 架构设计

/thoughtworks

架构设计

服务与业务能力

/thoughtworks

业务能力表 - 企业差旅服务上下文

业务建模将模型实现为 RESTful API

角色	Method	URI	业务能力	业务能力服务
平台	POST	/travel-delegation-requests	任你行差旅管理部门替企业客户提交差旅服务托管请求	企业差旅服务
平台	POST	/travel-delegation-requests/{rid}/proposal	任你行差旅管理部门提交差旅管理方案	企业差旅服务
平台	POST	/travel-contracts	任你行差旅管理部门与客户签订企业差旅服务协议	企业差旅服务
平台	POST	/travel-contracts/{cid}/fixd-fee	任你行财务部门请求支付固定服务费用	企业差旅服务
3rd System/企业	POST	/travel-contracts/{cid}/fixd-fee/confirmation	企业财务确认固定服务费用支付结果	企业差旅服务
企业	POST	/travel-contracts/{cid}/cancellation	企业财务终止企业差旅服务协议	企业差旅服务
企业	POST	/travel-contracts/{cid}/manager-account	企业提交管理账号开通请求	企业差旅服务
平台	POST	/travel-contracts/{cid}/manager-account/confirmation	任你行客户关系部设置管理账号	企业差旅服务

业务能力表 - 企业差旅服务上下文

业务建模将模型实现为 RESTful API

角色	Method	URI	业务能力	业务能力服务
企业	POST	/travel-contracts/{cid}/account	企业管理员提交员工账号开通请求	企业差旅服务
平台	POST	/travel-contracts/{cid}/account/confirmation	任你行客户关系部设置员工账号	企业差旅服务
企业	POST	/travel-contracts/{cid}/reimbursements/{mid}	企业财务提交月度报销请求	企业差旅服务
平台	POST	/travel-contracts/{cid}/reimbursements/{mid}/confirmation	平台确认月度报销记录	企业差旅服务
企业	GET	/travel-contracts/{cid}/reimbursements/{mid}	企业财务查看月度报销记录	企业差旅服务
企业	GET	/travel-contracts/{cid}/business-orders/{mid}	企业财务查看月度因公订单记录	企业差旅服务
平台	POST	/travel-contracts/{cid}/monthly-settlements{mid}	任你行财务部提交月度结算支付请求	企业差旅服务
企业/平台	GET	/travel-contracts/{cid}/monthly-settlements{mid}	企业财务和平台财务部门查看月度结算支付详情	企业差旅服务

业务能力表 - 企业差旅服务上下文

业务建模将模型实现为 RESTful API

角色	Method	URI	业务能力	业务能力服务
企业	POST	/travel-contracts/{cid}/monthly-settlements/{mid}/confirmation	企业财务确认月度结算支付结果	企业差旅服务
企业	POST	/travel-contracts/{cid}/fixed-fee-invoice	企业财务请求开具固定服务费发票	企业差旅服务
3rd System/平台	POST	/travel-contracts/{cid}/fixed-fee-invoice/confirmation	平台确认开具固定服务费发票	企业差旅服务
企业	POST	/travel-contracts/{cid}/monthly-settlement-invoices/{mid}	企业财务请求开具月度结算发票	企业差旅服务
3rd System/平台	POST	/travel-contracts/{cid}/monthly-settlement-invoices/{mid}/confirmation	平台确认开具月度结算发票	企业差旅服务
企业	GET	/travel-contracts/{cid}/monthly-settlements	企业财务查看所有月度结算支付详情	企业差旅服务
企业	GET	/travel-contracts/{cid}/monthly-settlements-invoices	企业财务查看所有月度结算发票详情	企业差旅服务
企业	GET	/travel-contracts/{cid}/reimbursements	企业财务查看所有报销	企业差旅服务

业务能力表 - 企业差旅服务上下文

业务建模将模型实现为 RESTful API

角色	Method	URI	业务能力	业务能力服务
企业	GET	/travel-contracts/{cid}/business-orders	企业财务查看所有因公订单记录	企业差旅服务
企业/平台	GET	/travel-contracts/{cid}/fixed-fee-invoice	企业/平台查看固定服务费发票开具详情	企业差旅服务

业务能力表 - 机票订单买卖上下文

业务建模将模型实现为 RESTful API

角色	Method	URI	业务能力	业务能力服务
企业出行人	GET	/air-ticket-request/proposals	企业出行人查看出行方案	机票订单服务
企业出行人	POST	/air-ticket-order-contracts	企业出行人完成预定	机票订单服务
企业出行人	GET	/air-ticket-order-contracts/{cid}	企业出行人查看订单	机票订单服务
企业出行人	GET	/air-ticket-order-contracts/{cid}/personal-payment	企业出行人查看因私机票订单支付请求	机票订单服务
3rd System/企业出行人	POST	/air-ticket-order-contracts/{cid}/personal-payment/confirmation	企业出行人确认因私机票订单支付结果	机票订单服务
企业出行人	POST	/air-ticket-order-contracts/{cid}/business-payment	企业出行人提交因公机票订单支付请求	机票订单服务
3rd System/平台	POST	/air-ticket-order-contracts/{cid}/business-payment/confirmation	平台确认因公机票订单支付结果	机票订单服务
企业出行人	GET	/air-ticket-order-contracts/{cid}/business-payment	企业出行人查看因公机票订单支付请求详情	机票订单服务

业务能力表 - 机票订单买卖上下文

业务建模将模型实现为 RESTful API

角色	Method	URI	业务能力	业务能力服务
企业出行人	GET	/air-ticket-order-contracts/{cid}/itinerary	企业出行人查看机票行程单	机票订单服务
3rd System/平台	POST	/air-ticket-order-contracts/{cid}/itinerary/confirmation	平台确认机票行程单结果	机票订单服务
企业出行人	POST	/air-ticket-order-contracts/{cid}/cancellation	企业出行人取消订单	机票订单服务
企业出行人	GET	/air-ticket-order-contracts	企业出行人查看订单列表	机票订单服务

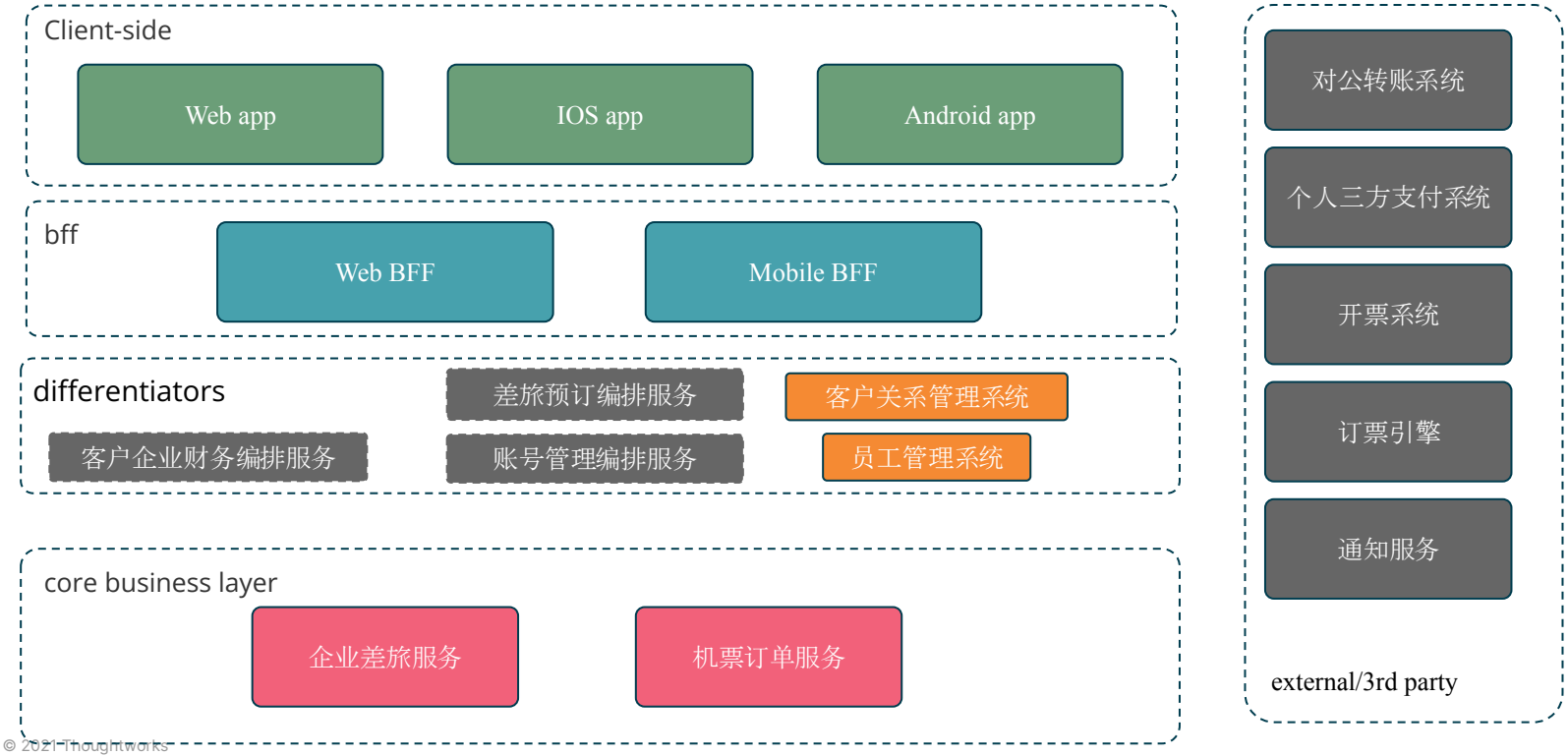
架构设计

进程间架构设计

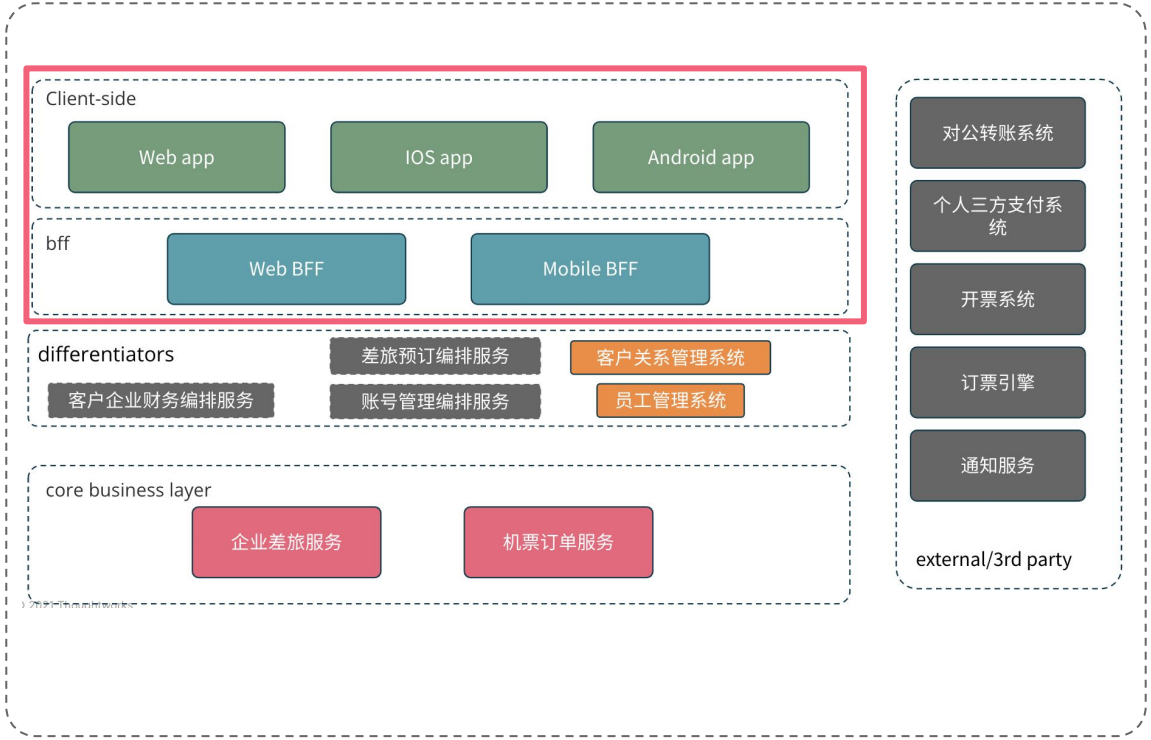
/thoughtworks

进程间架构设计 - 架构图图例参考

借鉴Pace Layered Application Strategy中Pace-Layer概念指导进程间架构设计。下图中，虚线框代表一组同类进程，使用不同颜色图例进行标记。**虚线框和实现组件可自行扩展，不需要拘泥于图中给出的结构；其中，数据库可省略，编排服务可选，如选择，需要在下一页给出支撑理由。**



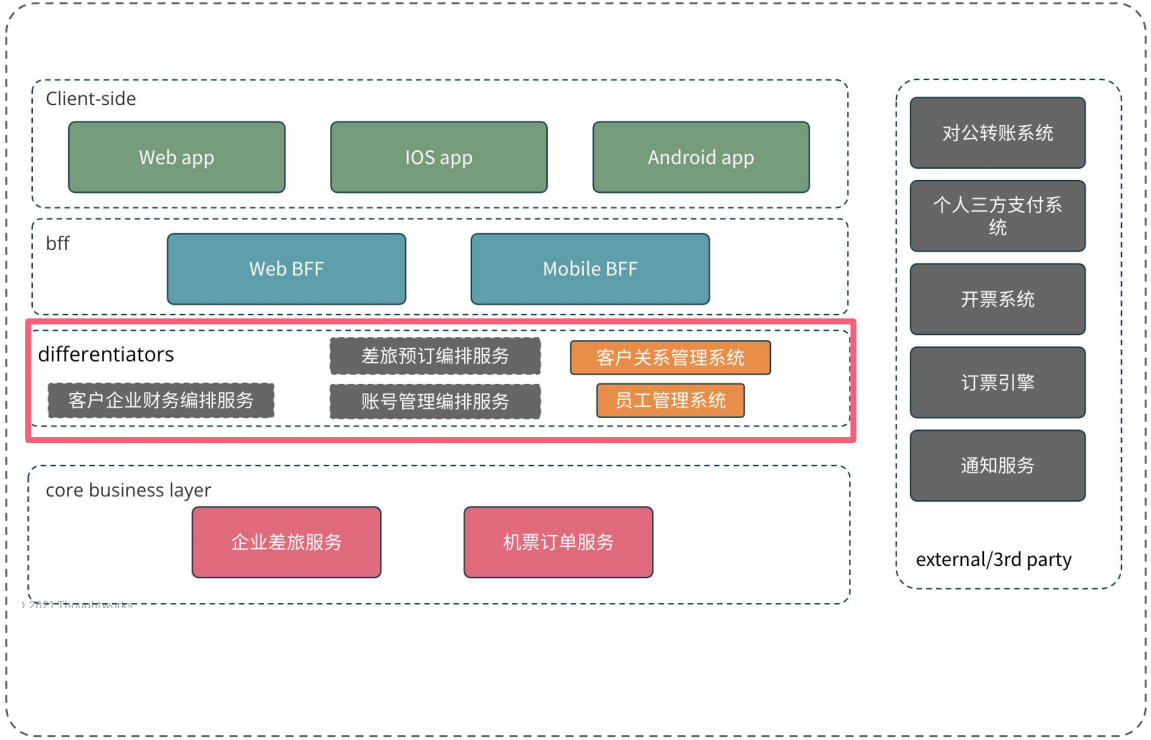
进程间架构设计 - 分层策略和组件职责说明



Client-side层，负责信息展示
进程Web app：提供浏览器Web端信息展示
进程IOS app：提供IOS端信息展示
进程Andorid：提供Android端信息展示

bff层，负责客户端与服务端的差异化数据转换
进程Web BFF：为Web端提供数据转换
进程Mobile BFF：IOS和Android端提供数据转换

进程间架构设计 - 分层策略和组件职责说明



differentiators层，负责领域逻辑以及业务编排

客户企业财务编排服务：依赖企业差旅服务与机票订单服务，提供对账、开票、确认、结算、企业支付以及报表导出等业务能力

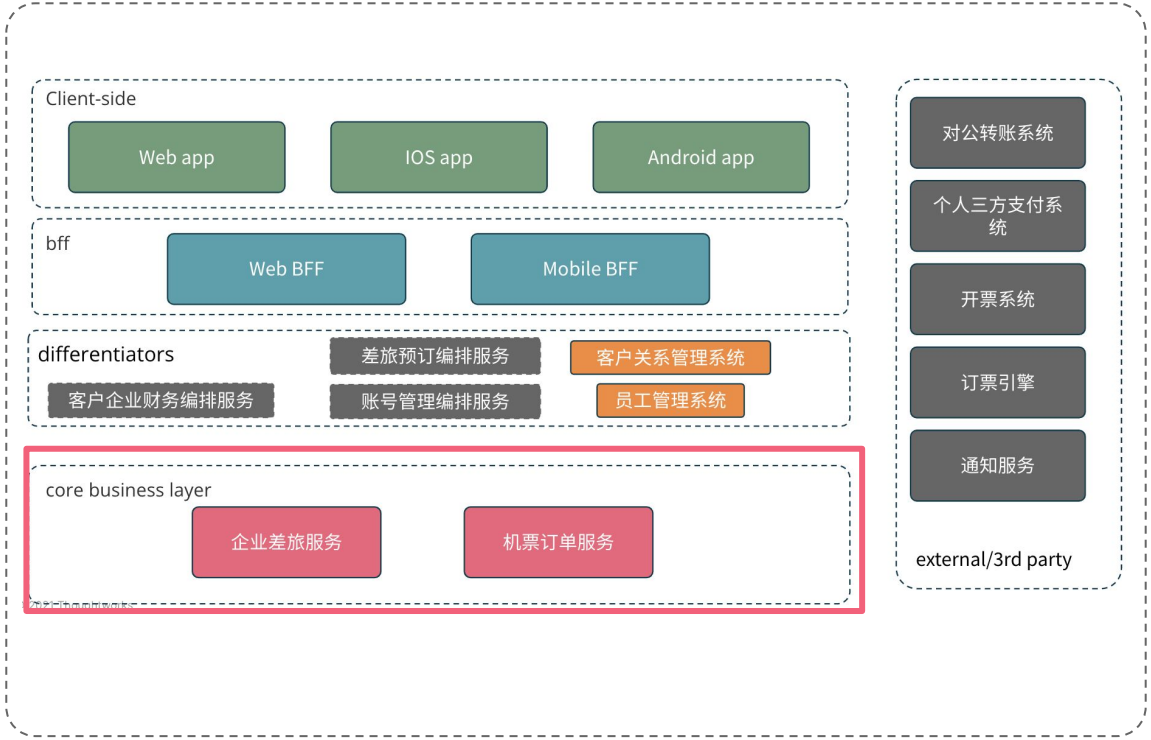
差旅预订编排服务：依赖客户关系管理系统与机票订单服务，避免机票订单服务反向依赖客户关系管理系统。提供因公出行方案审批以及设置用车规范等业务能力

账号管理编排服务：依赖客户关系管理系统与企业差旅服务，避免企业差旅服务反向依赖客户关系管理系统。提供客户员工账号设置与注册等业务能力

客户关系管理系统：提供客户企业员工信息

员工管理系统：提供内部员工信息

进程间架构设计 - 分层策略和组件职责说明

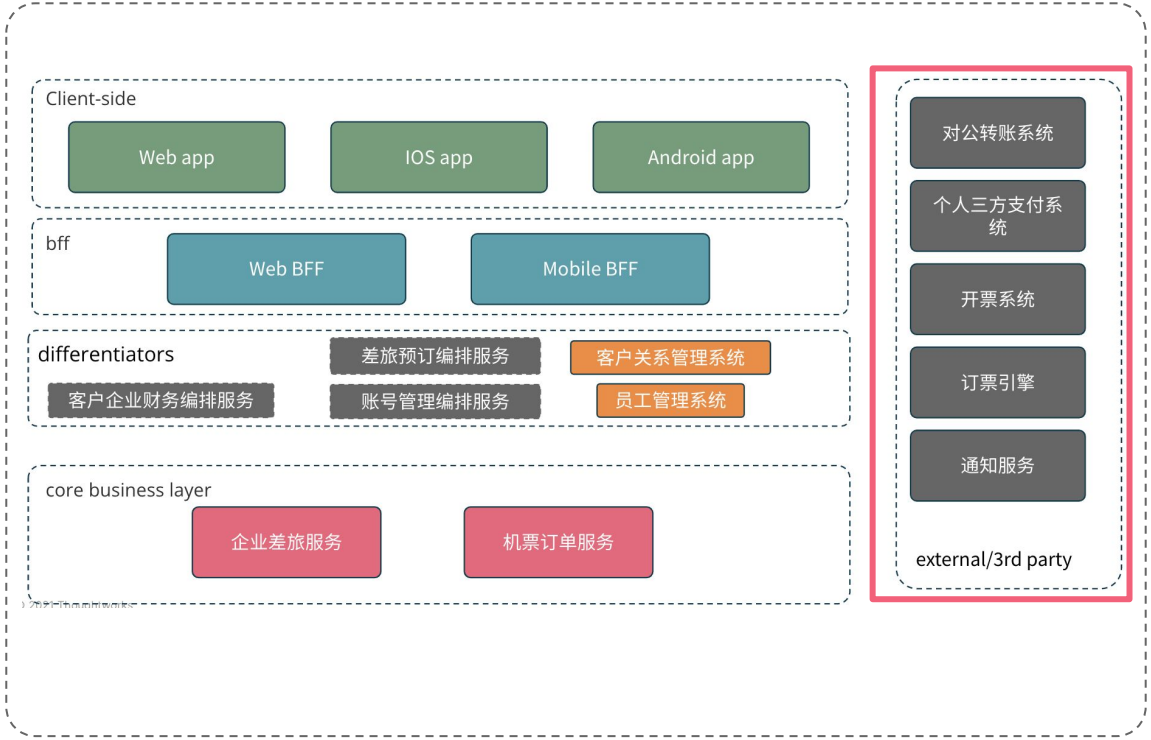


core business层，负责核心业务逻辑

企业差旅服务：提供企业差旅服务上下文的业务能力

机票订单服务：提供机票订单买卖上下文的业务能力

进程间架构设计 - 分层策略和组件职责说明



external/3rd party，负责实现业务能力所依赖的第三方系统

对公转账系统：对外对接银联对公转账系统，提供企业支付的业务能力，包括固定服务费用支付与月度结算支付

个人三方支付系统：对外对接微信支付和支付宝支付，提供因私出行时个人支付的业务能力

开票系统：提供开具发票的业务能力，包括固定服务费用发票与月度结算发票

订票引擎：提供搜索、预订、退改机票、生成机票行程单等业务能力

通知服务：听过发送通知短信的业务能力，包括账号开通短信、审批短信以及机票行程单短信

架构设计

进程内架构设计

/thoughtworks

进程内架构设计 - 组件选择

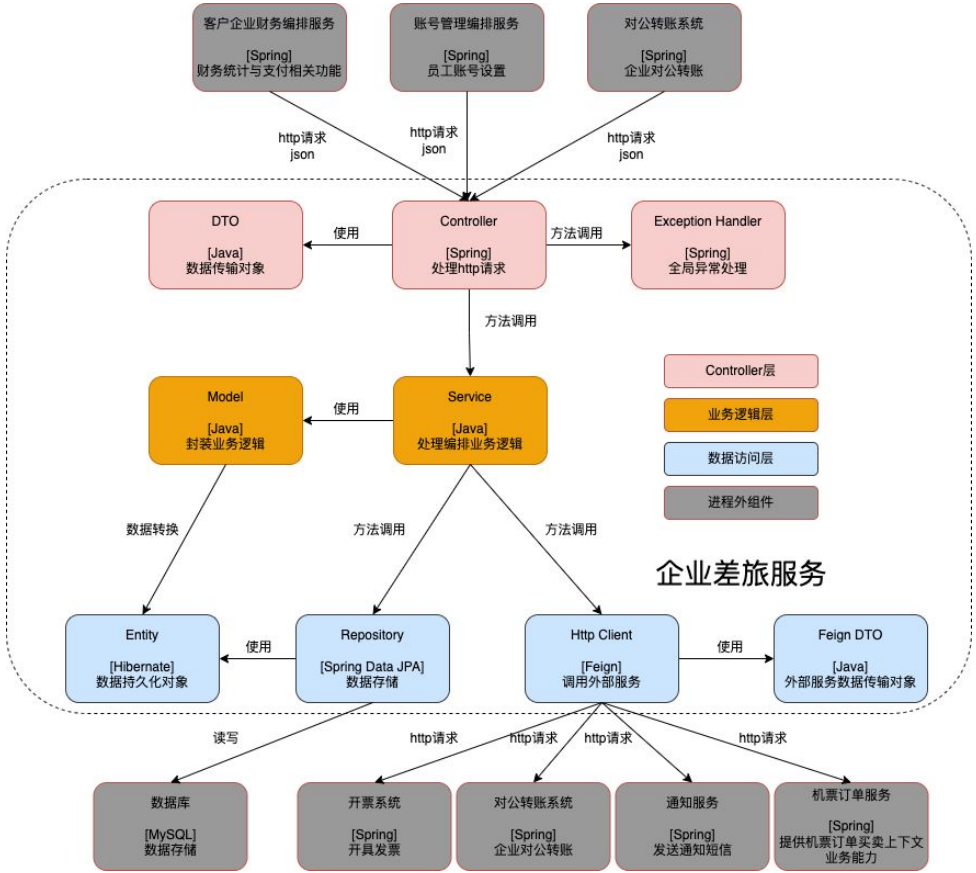
该进程为企业差旅服务

其职责是提供企业差旅服务上下文的业务能力

它以 RESTful API 的形式对外提供业务能力

采用的技术选型为SpringBoot、Java、MySQL、Spring Data JPA、Hibernate、FeignClient

进程内架构 - 架构图



Controller层负责处理外部服务调用的http请求
组件 Controller：定义业务API，处理转发请求
组件 DTO：对应于请求中的request和response，将json反序列化Java对象
组件 Exception Handler：将异常转化为error response

业务逻辑层负责处理业务逻辑
组件Service：负责编排被Model封装好的业务逻辑顺序，完成业务流程
组件 Model：负责定义业务对象，封装业务逻辑

数据访问层负责数据访问操作
组件Entity：是数据库持久化的对象
组件 Repository：负责操作数据库
组件Http Client：负责调用外部服务
组件Feign DTO：调用外部服务的数据传输对象，将Java对象序列化为json

应对进程间分区 - 保证业务准确性

举例说明分区场景和采用CP的应对方案

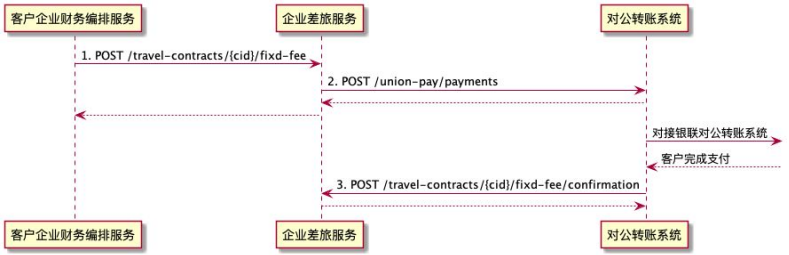
注意:【业务能力】必须来自架构设计的业务能力表

正常情况下，企业差旅服务调用对公转账系统完成请求支付固定服务费用，如右上图步骤2。

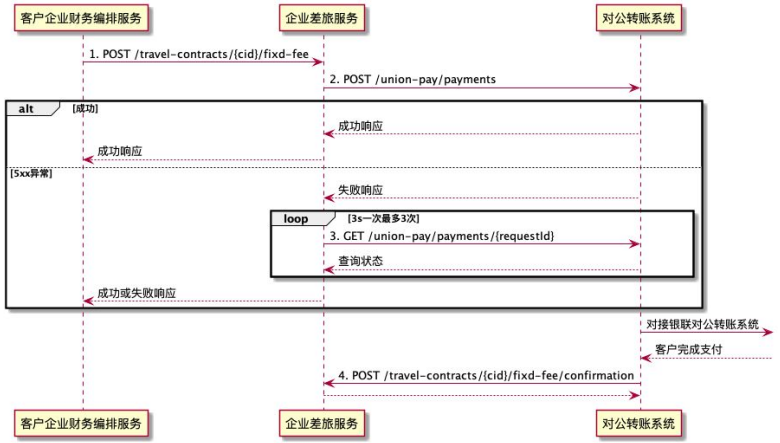
当以下分区异常发生时：
1. 请求因网络原因没送达、对公转账系统处理请求前宕机等，导致请求未被对公转账系统处理；
2. 响应因网络原因没返回、对公转账系统处理完请求后宕机等，导致响应异常；该进程调用对公转账系统会导致请求支付固定服务费用失败。

在此失败场景下，业务方基于准确获取固定服务费用支付状态，避免重复支付的业务目标，需确保固定服务费用支付状态在进程间的准确性，且后续请求开具固定服务费发票业务不可用。

为实现如上业务准确性的诉求，在设计上(架构，交互方式等)采用了以下应对方案，如右下图：
当分区异常发生时，调用对公转账系统提供的/union-pay/payments/{requestId}查询请求状态的接口。当查询到：
1. 请求未创建，对应于请求未被对公转账系统处理，则抛出异常给客户企业财务编排服务，终止业务流程，等待客户企业财务编排服务再次调用重试；
2. 请求已创建，对应于请求被对公转账系统处理，则业务流程成功；
3. 5xx异常，每隔3s重试3次查询，如果3次都是5xx异常，则抛出异常给客户企业财务编排服务，终止业务流程，引入人工处理；
另外，由于涉及到对/union-pay/payments接口的重试，因此对公转账系统需要对该接口实现幂等。同样企业差旅服务的/travel-contracts/{cid}/fixd-fee也涉及到重试，所以也需要实现幂等。



原交互关系



应对方案

应对进程间分区 - 保证业务可用性

举例说明分区场景和采用AP的应对方案

注意:【业务能力】必须来自架构设计的业务能力表

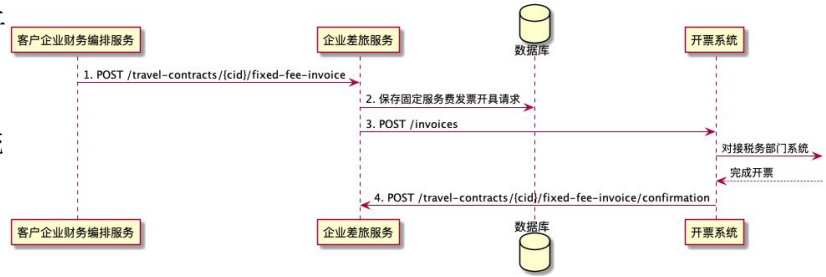
正常情况下，企业差旅服务调用开票系统完成请求开具固定服务费发票，如右上图步骤3

当以下分区异常发生时：
1. 请求因网络原因没送达、开票系统处理请求前宕机等，导致请求未被开票系统处理；
2. 响应因网络原因没返回、开票系统处理完请求后宕机等，导致响应异常；
该进程调用开票系统会导致请求开具固定服务费发票失败。

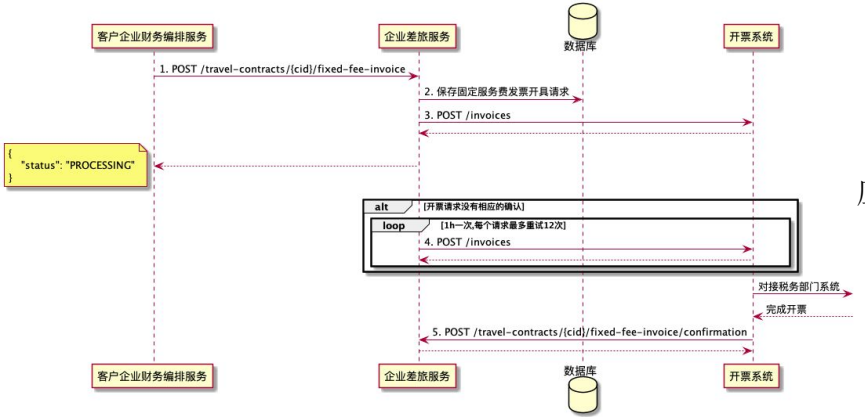
在此失败场景下，业务方基于平台在收到申请后24小时内完成固定服务费发票开具，可容忍进程间发票开具状态的暂时不一致，且查看固定服务费发票开具详情时向客户企业财务编排服务返回发票开具中的中间状态。

为实现如上的业务可用性诉求，在设计上(架构，交互方式等)采用了以下应对方案，如右下图：
企业差旅服务调用开票系统不论成功与否，均向客户企业财务编排服务返回开票状态处理中。之后由一个异步定时任务，将所有没有相应响应的开票请求调用开票系统的/invoices接口进行重试，重试间隔为1h一次，每个请求最多重试12次。当重试超过上限，需要引入人工处理。在重试过程中，如果遇到：
1. 开票系统之前未处理过该请求，则可以接收请求并进行处理；
2. 开票系统之前处理过请该求，则可以忽略该次请求；
同时在重试过程中遇到查看固定服务费发票开具详情的业务时，向其返回PROCESSINIG的状态。
另外，由于涉及到对/invoices接口的重试，因此对开票系统需要对该接口实现幂等。

原交互关系



应对方案



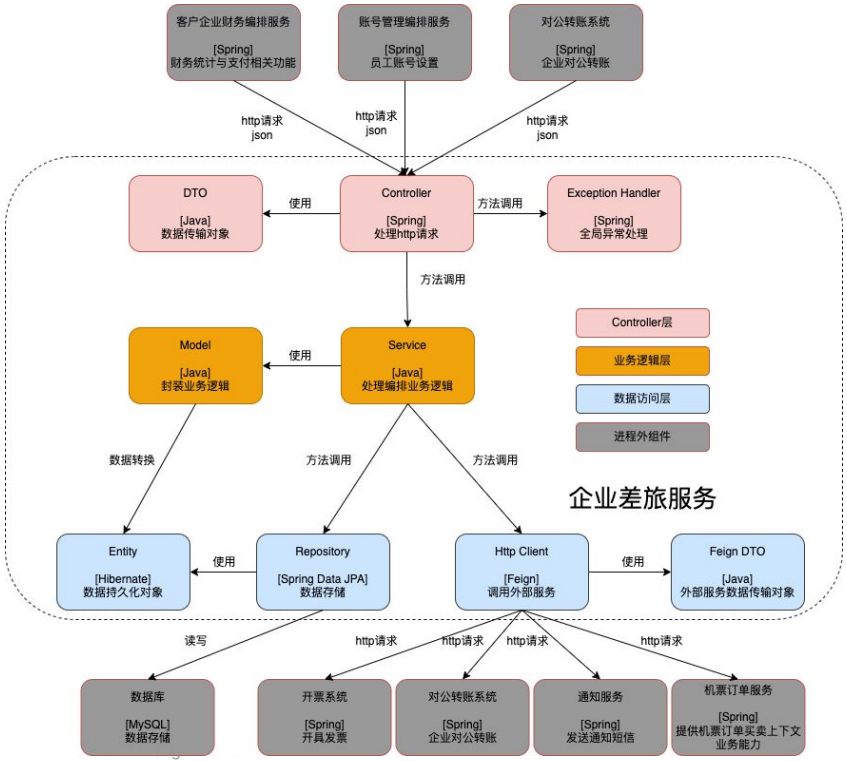
架构设计

3 进程内架构的测试策略

/thoughtworks

测试策略 - 全景

进程内架构图

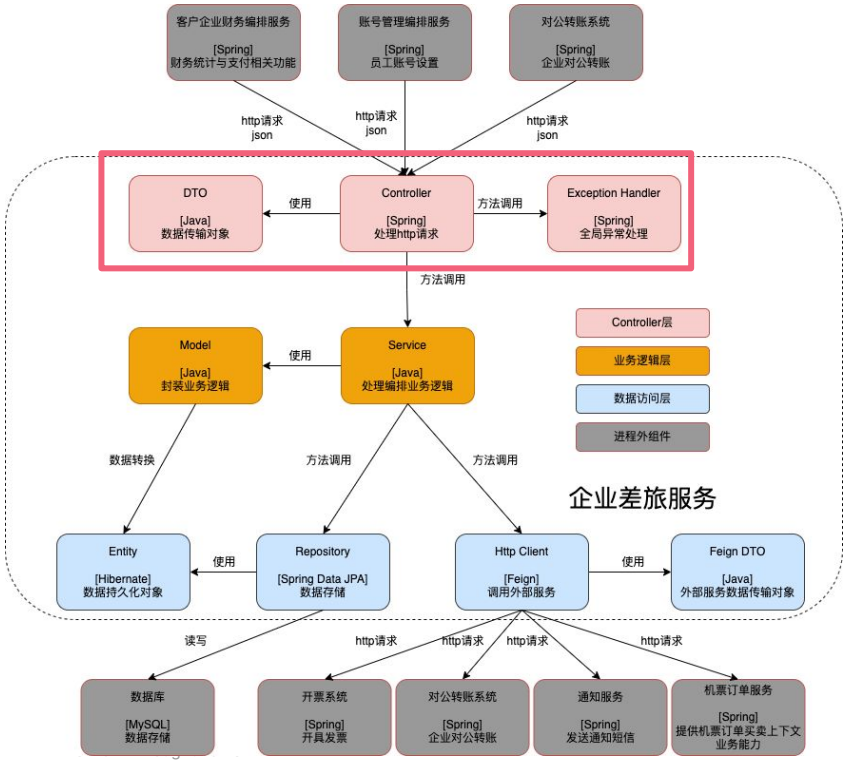


针对当前架构，采用以下测试策略覆盖进程中各个组件并保证进程边界处的测试有效性，来帮助发现问题和定位问题：

序号	被测组件 （一个或多个组件的组件名称）	测试类型 组件测试/单元测试/功能测试等	测试象限 （Q1 - Q4）
1	Controller+DTO+Exception Handler	单元测试	Q1
2	Service+Model	单元测试	Q1
3	Repository+Entity	组件测试	Q1
4	Http Client+Feign DTO	单元测试	Q1

测试策略 - 1 - Controller+DTO+Exception Handler

进程内架构图



被测组件:

Controller+DTO+Exception Handler

采用单元测试，属于第1象限的测试，

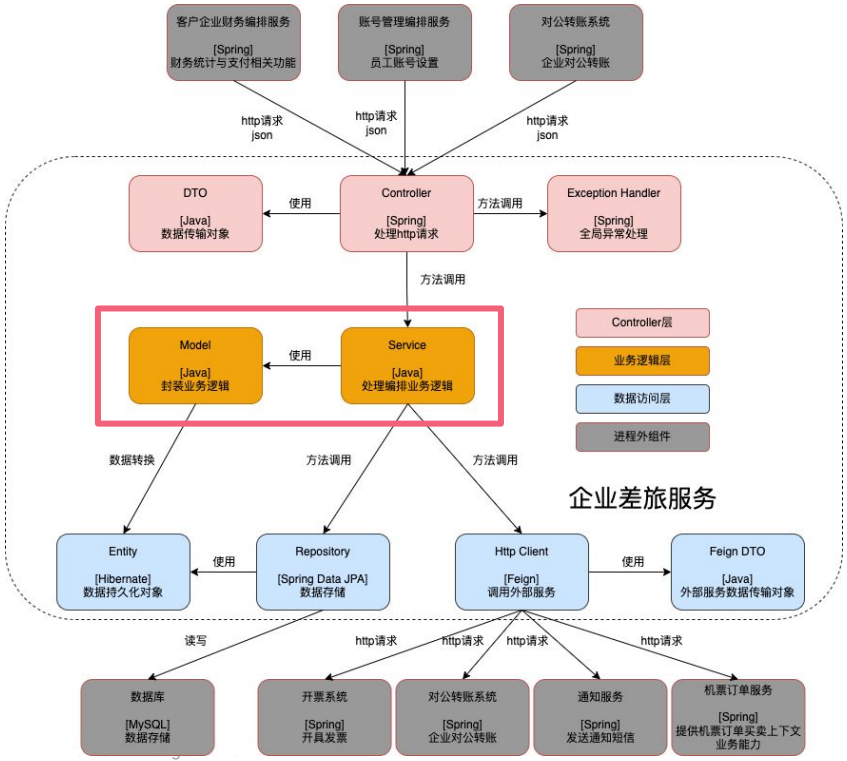
测试这些组件是因为Controller需要将Http请求以及对应的响应进行反序列化以及序列化，同时也需要根据不同的异常返回不同的error response。

测试时，会依赖:

- 进程内组件Service，采用Stub来替代该组件，因为Controller不关注业务逻辑，只关注Service的返回结果。

测试策略 - 2 - Service+Model

进程内架构图



被测组件:

Service+Model

采用单元测试，属于第1象限的测试，

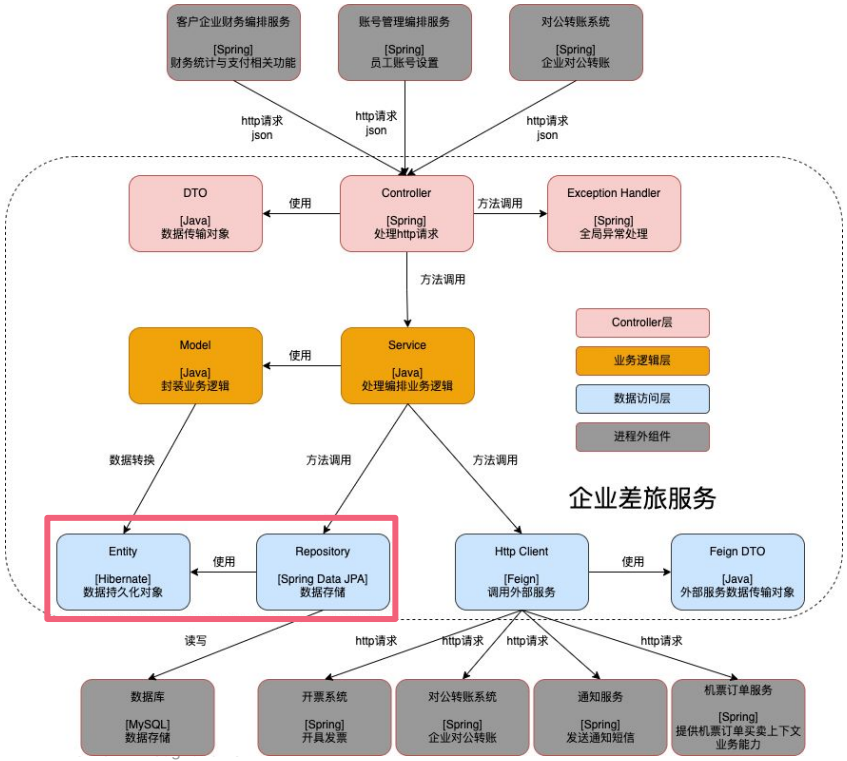
测试这些组件是因为Service需要按照业务逻辑编排业务顺序，同时调用Model来实现业务功能。

测试时，会依赖:

- 进程内组件Repository，采用Stub或Spy来替代该组件，因为Service不关注数据的存储。
- 进程内组件Http Client，采用Stub或Spy来替代该组件。首先需要状态验证，因为不需要调用真实的外部服务，只需要外部服务按要求返回的结果；其次需要行为验证，需要验证有没有调用过外部服务。

测试策略 - 3 - Repository+Entity

进程内架构图



被测组件:

Repository+Entity

采用组件测试，属于第1象限的测试，

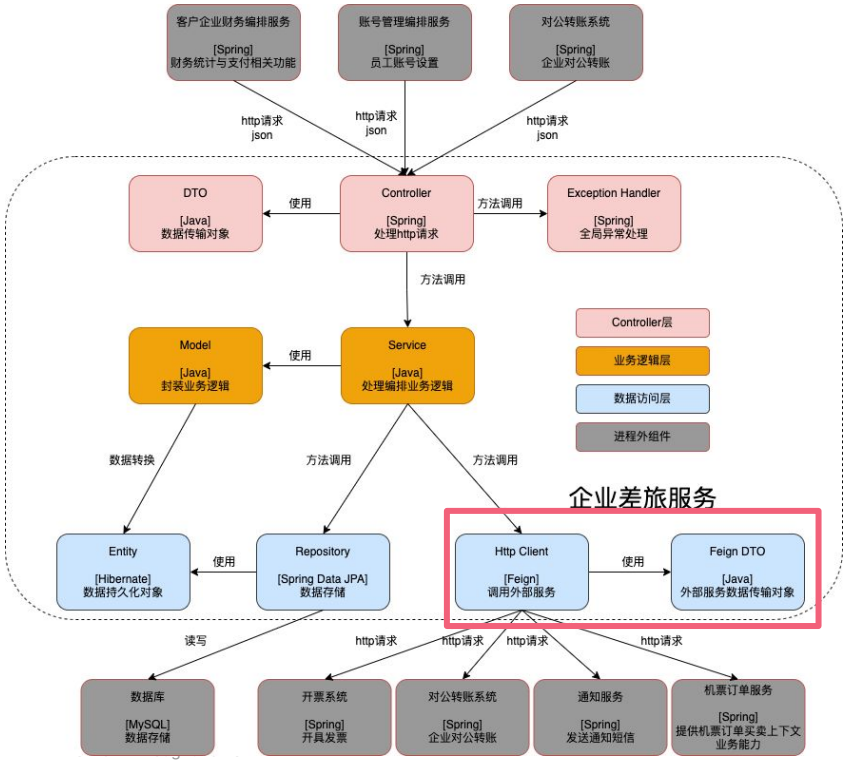
测试这些组件是因为Repository会调用数据库进行读写，而Entity是数据持久对象，所以需要一起测。

测试时，会依赖:

- 进程外组件数据库，采用Fake Object - H2来替代该组件，因为它位于进程边界，需要验证进程间功能有效性。同时使用Fake Object，通过使用内存数据库可以避免搭建真实数据库，提升测试速度增加可操作性，降低测试成本。

测试策略 - 4 - Http Client+Feign DTO

进程内架构图



被测组件:

Http Client+Feign DTO

采用单元测试，属于第1象限的测试，

测试这些组件是因为Controller需要将Http请求以及对应的响应进行反序列化以及序列化，同时也需要测试能否正确处理外部服务按契约返回的数据。

测试时，会依赖:

- 进程外组件外部服务，采用stub来替代该组件，因为它位于进程边界，需要验证进程间功能有效性。同时使用stub可以避免调用真实外部服务，提升测试速度与稳定性。

4 效能管理

/thoughtworks

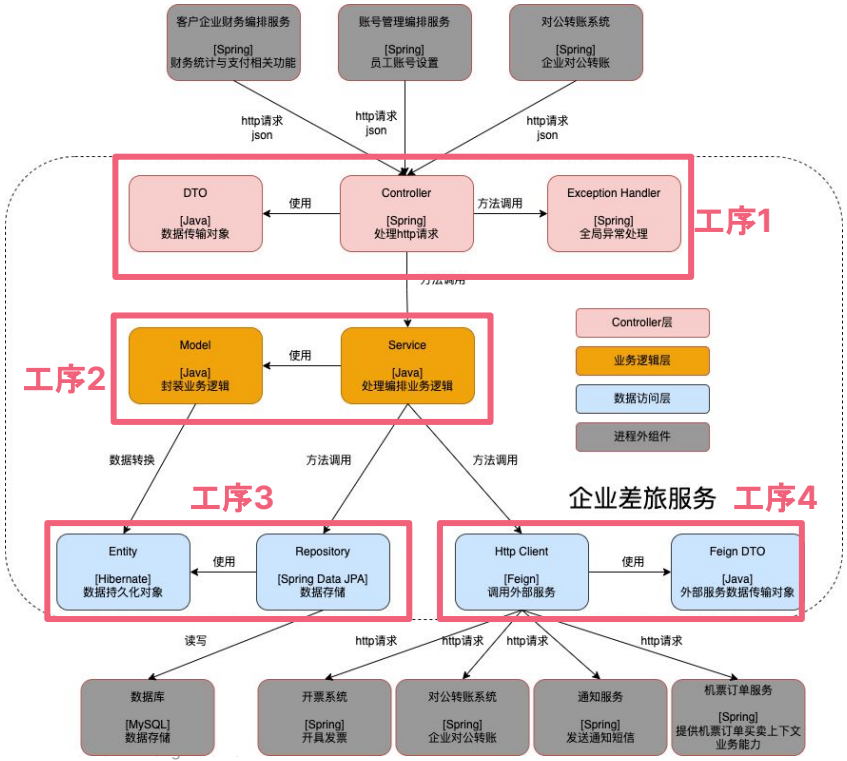
效能管理

工序设计与任务拆解

/thoughtworks

工序设计

进程内架构图



针对当前架构，设计以下工序来实现架构，覆盖所有组件并应用此前设计的测试策略：

Controller+DTO+Exception Handler

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。

Service+Model

- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。

Repository+Entity

- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。

Http Client+Feign DTO

- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

Story1 描述及AC - 准确性业务场景

Story:

作为任你行平台财务部门员工, **我想要**请求企业支付固定服务费用, **以便于**收取固定服务费用并开展差旅管理服务。

相关API定义:如右图所示

ACs:

- AC1: 业务成功场景-当客户企业财务编排服务传入合法协议id(cid), 去请求支付固定服务费用时, 成功发起固定服务费用支付请求。
- AC2: 业务异常场景-当客户企业财务编排服务传入不合法协议id(cid), 去请求支付固定服务费用时, 发起固定服务费用支付请求失败。
- AC3: 准确性的分区异常场景-客户企业财务编排服务传入合法协议id(cid), 但对公转账系统不可用, 去请求支付固定服务费用时, 发起固定服务费用支付请求失败。

1.请求支付固定服务费用

```
POST /travel-contracts/{cid}/fixd-fee
Request
    Body:
        {
            "destinationCardNumber": "cardNumber"
        }
Response (happy path):
    Code: 201
    Body:
        {
            "requestId": "requestId"
        }
Response (unhappy path):
    Code: 400, 404, 500
    Body:
        {
            "code": "000001",
            "msg": "input param invalid",
            "detail": "missing field",
            "timestamp": 1654672132
        }
```

2.调用对公转账系统API

```
POST /union-pay/payments
Request
    Body:
        {
            "destinationCardNumber": "cardNumber",
            "amount": 1000.00,
            "requestId": "requestId"
        }
Response (happy path):
    Code: 201
    Body:
        {
            "paymentId": "paymentId"
        }
Response (unhappy path):
    Code: 5xx
```

3.查询对公转账系统请求状态API

```
GET /union-pay/payments/{requestId}
Response (happy path):
    Code: 200
    Body:
        {
            "paymentId": "paymentId"
        }
Response (unhappy path):
    Code: 404, 5xx
```

Story 1 任务拆解 - AC1-业务成功场景

当客户企业财务编排服务传入合法协议id(cid)，去请求支付固定服务费用时，成功发起固定服务费用支付请求。

- EXAMPLE 1：传入合法的cid=123，请求支付固定服务费用，成功发起固定服务费用支付请求，返回response http code=201，requestId=1-2-3。
  - 工序2，stub TravelContractRepository，当调用 TravelContractRepository.getContract时返回未完成固定服务费用支付的contract；stub BusinessPaymentClient，当调用BusinessPaymentClient.requestPayment时返回paymentId。实现TravelContractService.requestFixdFee，使其返回requestId。
  - 工序3，fake 数据库，当调用数据库的select cid=123方法的时候返回cid=123的contract，实现TravelContractRepository.getContract。
  - 工序4，stub 对公转账系统，当调用POST /union-pay/payments返回http code=201（参考Story1相关API定义图2-response happy path），实现BusinessPaymentClient.requestPayment。
  - 工序1，stub TravelContractService，当调用TravelContractService.requestFixdFee时返回requestId=1-2-3，实现TravelContractController的/travel-contracts/{cid}/fixd-fee，得到response的http code=201（参考Story1相关API定义图1-response happy path）

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- **工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。**
- **工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。**

Story 1 任务拆解 - AC1-业务成功场景

当客户企业财务编排服务传入合法协议id(cid)，去请求支付固定服务费用时，成功发起固定服务费用支付请求。

- EXAMPLE 2：传入合法的cid=123，请求支付固定服务费用，调用对公转账系统处理请求成功但返回异常，经过重试查询请求状态后，成功发起固定服务费用支付请求，返回response http code=201，requestId=1-2-3。
  - 工序2，stub TravelContractRepository，当调用 TravelContractRepository.getContract时返回未完成固定服务费用支付的contract；stub BusinessPaymentClient，当调用 BusinessPaymentClient.requestPayment时返回http code=504；当调用 BusinessPaymentClient.getPayment时返回paymentId。实现 TravelContractService.requestFixdFee，使其返回requestId。
  - 工序4，[仅测试] stub 对公转账系统，当调用POST /union-pay/payments返回http code=504（参考Story1相关API定义图2-response unhappy path），验证BusinessPaymentClient.requestPayment抛出504异常；[需实现] stub 对公转账系统，当调用GET /union-pay/payments/{requestId}返回paymentId（参考Story1相关API定义图3-response happy path），实现BusinessPaymentClient.getPayment。

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- **工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。**
- **工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。**

Story 1 任务拆解 - AC2-业务异常场景

当客户企业财务编排服务传入不合法协议id(cid)，去请求支付固定服务费用时，发起固定服务费用支付请求失败。

- EXAMPLE 1：传入不存在的cid=321，请求支付固定服务费用，发起固定服务费用支付请求失败，返回response http code=404，msg=data not found
  - 工序2，stub TravelContractRepository，当调用TravelContractRepository.getContract时返回null，实现TravelContractService.requestFixdFee，抛出DataNotFoundException。
  - [仅测试]工序3，fake 数据库，当调用数据库的select cid=321方法的时候返回空，验证TravelContractRepository.getContract返回null
  - [仅测试]工序1，stub TravelContractService，当调用TravelContractService.requestFixdFee时抛出DataNotFound异常，验证调用TravelContractController的/travel-contracts/{cid}/fixd-fee，得到response的http code=404，msg=data not found（参考Story1相关API定义图1-response unhappy path）

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。
- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

Story 1 任务拆解 - AC2-业务异常场景

当客户企业财务编排服务传入不合法协议id(cid)，去请求支付固定服务费用时，发起固定服务费用支付请求失败。

- EXAMPLE 2：传入存在的cid=123但对应的contract已完成支付固定服务费用，发起固定服务费用支付请求失败，返回response http code=400，msg=input param invalid
  - 工序2，stub TravelContractRepository，当调用 TravelContractRepository.getContract时返回已完成支付固定服务费用的contract，实现TravelContractService.requestFixdFee，抛出BadRequestException。
  - [仅测试]工序3，fake 数据库，当调用数据库的select cid=123方法的时候返回已完成支付固定服务费用的contract，验证TravelContractRepository.getContract返回已完成支付固定服务费用的contract
  - [仅测试]工序1，stub TravelContractService，当调用TravelContractService.requestFixdFee时抛出BadRequestException异常，验证调用TravelContractController的/travel-contracts/{cid}/fixd-fee，得到response的http code=400，msg=input param invalid（参考Story1相关API定义图1-response unhappy path）

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。
- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

Story 1 任务拆解 - AC3-分区异常场景

客户企业财务编排服务传入合法协议id(cid)，但对公转账系统不可用，去请求支付固定服务费用时，发起固定服务费用支付请求失败。

- EXAMPLE 1：传入合法的cid=123，请求支付固定服务费用，调用对公转账系统处理请求失败，经过重试查询请求状态后，确认发起固定服务费用支付请求失败，返回response http code=500，msg=please retry later。
  - 工序2，stub TravelContractRepository，当调用 TravelContractRepository.getContract时返回未完成固定服务费用支付的contract；stub BusinessPaymentClient，当调用BusinessPaymentClient.requestPayment时返回http code=503；当调用BusinessPaymentClient.getPayment时返回http code=404。实现TravelContractService.requestFixdFee，使其抛出ExternalServerException。
  - [仅测试]工序4，stub 对公转账系统，当调用POST /union-pay/payments返回http code=503（参考Story1相关API定义图2-response unhappy path），验证BusinessPaymentClient.requestPayment抛出503异常；stub 对公转账系统，当调用GET /union-pay/payments/{requestId}抛出404异常（参考Story1相关API定义图3-response happy path），验证BusinessPaymentClient.getPayment抛出404异常。
  - [仅测试]工序1，stub TravelContractService，当调用TravelContractService.requestFixdFee时抛出ExternalServerException异常，验证调用TravelContractController的/travel-contracts/{cid}/fixd-fee，得到response的http code=500，msg=please retry later（参考Story1相关API定义图1-response unhappy path）

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。
- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

Story 1 任务拆解 - AC3-分区异常场景

客户企业财务编排服务传入合法协议id(cid)，但对公转账系统不可用，去请求支付固定服务费用时，发起固定服务费用支付请求失败。

- EXAMPLE 2：传入合法的cid=123，请求支付固定服务费用，调用对公转账系统处理请求失败，重试查询请求状态3次均返回503，确认发起固定服务费用支付请求失败，返回response http code=500，msg=please contact with IT。
  - 工序2，stub TravelContractRepository，当调用 TravelContractRepository.getContract时返回未完成固定服务费用支付的contract；stub BusinessPaymentClient，当调用BusinessPaymentClient.requestPayment时返回http code=503；当调用BusinessPaymentClient.getPayment时返回http code=503。实现TravelContractService.requestFixdFee，使其抛出ExternalServerException。
  - [仅测试]工序4，stub 对公转账系统，当调用GET /union-pay/payments/{requestId}抛出503异常（参考Story1相关API定义图3-response happy path），验证BusinessPaymentClient.getPayment抛出503异常。
  - [仅测试]工序1，stub TravelContractService，当调用TravelContractService.requestFixdFee时抛出ExternalServerException异常，验证调用TravelContractController的/travel-contracts/{cid}/fixd-fee，得到response的http code=500，msg=please contact with IT（参考Story1相关API定义图1-response unhappy path）

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- **工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。**
- **工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。**

Story 2 - Story 描述及AC - 可用性业务场景

Story:

作为企业财务，**我想要**请求任你行平台开具固定服 务费用发票，**以便于**企业进行核算。

相关 API 定义：如右图所示

ACs:

- AC1: 业务成功场景-当客户企业财务编排服务传入合法协议id(cid)，去请求开具固定服 务费用发票时，成功创建固定服务费用发票请求，返回发票开具状态。
- AC2: 业务异常场景-当客户企业财务编排服务传入不合法协议id(cid)，去请求开具固定服 务费用发票时，创建固定服务费用发票请求失败。
- AC3: 可用性的分区异常 场景-客户企业财务编排服务传入合法协议id(cid)，但发票系统不可用，去请求开具固定服 务费用发票时，创建固定服务费用发票请求成功，返回 发票开具状态。

1.请求开具固定服务费用发票

```
POST /travel-contracts/{cid}/fixed-fee-invoice
Request
    Body:
        {
            "identifier": "identifier"
        }
Response (happy path):
    Code: 201
    Body:
        {
            "requestId": "requestId"
            "status": "PROCESSING"
        }
Response (unhappy path):
    Code: 400, 404, 500
    Body:
        {
            "code": "000001",
            "msg": "input param invalid",
            "detail": "missing field",
            "timestamp": 1654672132
        }
```

2.调用开票系统API

```
POST /invoices
Request
    Body:
        {
            "identifier": "identifier",
            "amount": 1000.00,
            "requestId": "requestId"
        }
Response (happy path):
    Code: 201
    Body:
        {
            "invoiceId": "invoiceId"
        }
Response (unhappy path):
    Code: 5xx
```

Story 2 任务拆解 - AC1-业务成功场景

当客户企业财务编排服务传入合法协议id(cid)，去请求开具固定服务费用发票时，成功创建固定服务费用发票请求，返回发票开具状态。

- EXAMPLE 1：传入合法的cid=123，第一次请求开具固定服务费用发票，成功创建固定服务费用发票请求，返回response http code=201，requestId=1-2-3，status=PROCESSING。
  - 工序2 spy TravelContractRepository，当调用 TravelContractRepository.getContract时返回未创建过固定服务费用发票请求的contract，当调用TravelContractRepository.save时保存固定服务费用发票请求；stub InvoiceClient，当调用InvoiceClient.requestInvoice时返回invoiceId。实现TravelContractService.requestFixdFeeInvoice，使其返回固定服务费用发票请求。
  - 工序3，fake数据库，当调用数据库的insert的时候插入cid=123的创建过固定服务费用发票请求的协议，实现TravelContractRepository.save。

续下页

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- **工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。**
- **工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。**

Story 2 任务拆解 - AC1-业务成功场景

当客户企业财务编排服务传入合法协议id(cid)，去请求开具固定服务费用发票时，成功创建固定服务费用发票请求，返回发票开具状态。

接上页

- 工序4，stub 发票系统，当调用POST /invoices返回http code=201（参考Story2相关API定义图2-response happy path），实现InvoiceClient.requestInvoice。
- 工序1，stub TravelContractService，当调用TravelContractService.requestFixdFeeInvoice时返回requestId=1-2-3且没有收到固定服务费用发票确认的固定服务费用发票请求，实现TravelContractController的/travel-contracts/{cid}/fixd-fee-invoice，得到response的http code=201，requestId=1-2-3，status=PROCESSING（参考Story2相关API定义图1-response happy path）。

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。
- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

Story 2 任务拆解 - AC1-业务成功场景

当客户企业财务编排服务传入合法协议id(cid)，去请求开具固定服务费用发票时，成功创建固定服务费用发票请求，返回发票开具状态。

- EXAMPLE 2：传入合法的cid=123且对应的contract已完成开具固定服务费用发票，第二次请求开具固定服务费用发票，返回response http code=201，requestId=1-2-3，status=COMPLETED。
  - 工序2，spy TravelContractRepository，当调用 TravelContractRepository.getContract时返回已完成开具固定服务费用发票的contract，同时验证不调用TravelContractRepository.save；spy InvoiceClient，验证不调用InvoiceClient.requestInvoice。实现TravelContractService.requestFixdFeeInvoice，使其直接返回固定服务费用发票请求。
  - [仅测试]工序1，stub TravelContractService，当调用TravelContractService.requestFixdFeeInvoice时返回requestId=1-2-3且收到固定服务费用发票确认的固定服务费用发票请求，实现TravelContractController的/travel-contracts/{cid}/fixd-fee-invoice，得到response的http code=201，requestId=1-2-3，status=COMPLETED（参考Story2相关API定义图1-response happy path）。

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- **工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。**
- **工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。**

Story 2 任务拆解 - AC2-业务异常场景

业务异常场景-当客户企业财务编排服务传入不合法协议id(cid)，去请求开具固定服务费用发票时，创建固定服务费用发票请求失败。

- EXAMPLE 1：传入不存在的cid=321，请求开具固定服务费用发票，创建固定服务费用发票请求失败，返回response http code=404，msg=data not found
  - 工序2，stub TravelContractRepository，当调用TravelContractRepository.getContract时返回null，实现TravelContractService.requestFixdFeeInvoice，抛出DataNotFoundException。
  - [仅测试]工序1，stub TravelContractService，当调用TravelContractService.requestFixdFeeInvoice时抛出DataNotFound异常，验证调用TravelContractController的/travel-contracts/{cid}/fixd-fee-invoice，得到response的http code=404，msg=data not found（参考Story2相关API定义图1-response unhappy path）

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。
- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

Story 2 任务拆解 - AC2-业务异常场景

业务异常场景-当客户企业财务编排服务传入不合法协议id(cid)，去请求开具固定服务费用发票时，创建固定服务费用发票请求失败。

- EXAMPLE 2：传入存在的cid=123但该contract未完成固定服务费用支付，请求开具固定服务费用发票，创建固定服务费用发票请求失败，返回response http code=400，msg=input param invalid
  - 工序2，stub TravelContractRepository，当调用TravelContractRepository.getContract时返回未完成固定服务费用支付的contract，实现TravelContractService.requestFixdFeeInvoice，抛出BadRequestException。
  - [仅测试]工序1，stub TravelContractService，当调用TravelContractService.requestFixdFeeInvoice时抛出BadRequestException异常，验证调用TravelContractController的/travel-contracts/{cid}/fixd-fee-invoice，得到response的http code=400，msg=input param invalid（参考Story2相关API定义图1-response unhappy path）

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。
- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

Story 2 任务拆解 - AC3-分区异常场景

可用性的分区异常 场景-客户企业财务编排服务传入合法协议id(cid)，但发票系统不可用，去请求开具固定服务费用发票时，创建固定服务费用发票请求成功，返回发票开具状态。

- EXAMPLE 1：传入合法的cid=123，第一次请求开具固定服务费用发票，调用开票系统处理请求失败，返回response http code=201，status=PROCESSING。但经过重试调用开票系统后，最终成功。
  - 工序2，spy TravelContractRepository，当调用 TravelContractRepository.getContract时返回未创建过固定服务费用发票请求的contract，当调用TravelContractRepository.save时保存固定服务费用发票请求，当调用TravelContractRepository.findAll时返回所有协议；stub InvoiceClient，当第一次调用InvoiceClient.requestInvoice时抛出504异常，第二次调用时返回invoiceId。实现①TravelContractService.requestFixdFeeInvoice，使其返回固定服务费用发票请求；②TravelContractService.retryRequestFixdFeeInvoice，使其重试调用开票系统
  - 工序3，fake数据库，当调用数据库的select all contract的时候返回所有协议，实现TravelContractRepository.findAll。
  - [仅测试]工序4，stub 发票系统，当调用POST /invoices返回http code=504（参考Story2相关API定义图2-response happy path），验证InvoiceClient.requestInvoice抛出504异常。

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。
- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

Story 2 任务拆解 - AC3-分区异常场景

可用性的分区异常 场景-客户企业财务编排服务传入合法协议id(cid)，但发票系统不可用，去请求开具固定服务费用发票时，创建固定服务费用发票请求成功，返回发票开具状态。

- EXAMPLE 2：传入合法的cid=123，请求开具固定服务费用，调用开票系统处理请求一直失败，返回response http code=201，status=FAILED。且经过12次重试调用开票系统后，依然失败。
  - 工序2，spy TravelContractRepository，当调用 TravelContractRepository.getContract时返回开具固定服务费用发票失败的contract，同时验证不调用TravelContractRepository.save；spy InvoiceClient，验证不调用InvoiceClient.requestInvoice。实现TravelContractService.requestFixdFeeInvoice，使其直接返回固定服务费用发票请求。
  - [仅测试]工序1，stub TravelContractService，当调用TravelContractService.requestFixdFeeInvoice时返回requestId=1-2-3且超过12次重试都没收到确认的固定服务费用发票请求，实现TravelContractController的/travel-contracts/{cid}/fixd-fee-invoice，得到response的http code=201，requestId=1-2-3，status=FAILED（参考Story2相关API定义图1-response happy path）。

附：工序列表及效能基线

- 工序1：stub Service，实现Controller调用Service获取业务处理结果并返回response DTO，30分钟。
- 工序2：stub/spy Repository+Http Client，实现Service与Model的业务逻辑，2小时。
- 工序3：fake 数据库，实现Repository通过Entity对数据库进行读写，1小时。
- 工序4：stub 外部服务，实现Http Client通过Feign DTO对外部服务的访问，30分钟。

效能管理

带领团队落实

/thoughtworks

代码结构展示

代码结构/目录截图

代码链接: https://github.com/deesyx/heima

Q & A

【姓名】
【业务线名称，如华北、华南】

/thoughtworks