02丨迭代一概述：怎样开启一个麻雀虽小五脏俱全的项目？

2022-12-08 钟敬 来自北京

《手把手教你落地DDD》

课程介绍 >

讲述：钟敬

时长 14:51　大小 13.56M

你好，我是钟敬。今天咱们开始第一个迭代。

在开篇词中我们说过，为了让你更好地掌握 DDD，咱们这门课设置了一个贯穿始终的案例。我们会模仿真实的敏捷开发过程，把案例分成三个迭代，每个迭代的需求规模逐渐扩大，复杂性也逐渐增加。为了满足变化的需求，就会出现新的问题，为了解决这些问题，咱们就会引入新的 DDD 模式和实践，或者深化之前学过的知识。

今天开始，咱们通过迭代一，实现一个“麻雀虽小、五脏俱全”的项目。打通从需求分析，到领域建模，再到架构设计，最后到数据库和代码实现的完整闭环。

学完迭代一，你就可以理解 DDD 实践中那些最基本的套路了，甚至还可以在自己实际工作的项目里，选择一个小的“切片”，开始尝试落地 DDD。

当然了，在实践过程中，你可能也会遇到这样那样的问题，别急，这些问题的答案，很可能就在迭代二和迭代三的内容里。当然，你也可以把问题写在评论区，咱们一起讨论。

现在我们就来看一下这个迭代的具体需求。

这次迭代的需求

假设咱们俩要一起创业，经过了一轮市场调研，我们发现很多中小企业，都有诸如考勤管理、工时管理、项目管理、请假管理等通用的需求。我们姑且把这些应用统称为“企业管理系统”。

现在咱们顺便回顾一下领域驱动设计里“领域”这个词的含义。领域指的是软件要解决的那些业务问题，所以也可以叫业务领域，英文叫 Business Domain。在这个例子里，**“企业管理”就是咱们要处理的领域**。

过去，这些企业只能自建或购买软件系统，安装到公司内部的服务器上，甚至还要有自己的机房，软硬件以及运维成本都比较高。如果咱们能够把这些应用放在云上，那么企业就只需要有选择地购买我们提供的服务，按需付费就好了，不再需要自建机房，也不需要自己运维，可以有效降低企业的成本。

咱们这种系统，其实是基于 SaaS 的应用。所谓 SaaS，英文全称是 Software-as-a-Service，中文译作“软件即服务”。也就是说，软件不再像传统上那样交付并安装在客户本地，而是安装到云上，成为客户可以直接使用的服务。

SaaS 应用往往有个特点，用户五花八门，需求也是复杂多变。有些应用，如果只针对一个企业开发，本来挺简单，一旦放到云上，需求复杂性就会急剧上升。所以这次迭代，我们先做最简单的需求，后面两个迭代中我们再逐渐增加更多的灵活性。

由于云原生是这两年的热点，所以咱们凭这个点子幸运地拉到了投资，成立了一个叫做“卷卷通”的公司，开始着手研发系统的第一个版本。

由于不同行业的需求可能有较大差别，**我们决定先聚焦于一个细分市场，之后再扩大范围**。这个细分市场是我们比较熟悉的软件服务企业。这些企业会向自己的客户提供软件咨询、软件开发等等服务。

我们发现，这些企业的员工一般都工作在各个项目上。企业为了满足管理诉求，希望员工每周在系统中填报自己在哪些项目上花了多少时间，也就是所谓的**报工时**。所以我们第一步先来满足这个功能。但是，**为了完成报工时的功能，我们还要首先对组织、人员、项目等等进行管理**。

这里还要说一下，在领域驱动设计中有一个重要的角色叫做“领域专家”，叫业务专家也可以。领域专家需要对业务有总体性和本质性的把握，同时对业务发展也要有一定前瞻性，也就是说，心里要有一盘棋。

所以领域专家往往不是一线业务操作人员，在很多企业中，是那些干了很多年业务，逐渐成长起来的中级管理干部。还有一些企业，由产品经理担任领域专家。

那么，下面就由我暂时充当一下领域专家，详细说一下需求。

需求一：租户管理

这个系统可以给多家企业使用，每家企业的数据是隔离的。这些作为客户的企业，在云原生应用里，习惯上叫做“租户（tenant）”。**咱们的系统就是要对这些租户进行增删改查等管理**。这样的系统叫多租户系统。

假设我们做市场调研时，研究了一家叫“零零后科技有限公司”的企业，我们这个迭代，就先来满足这家企业的需求。

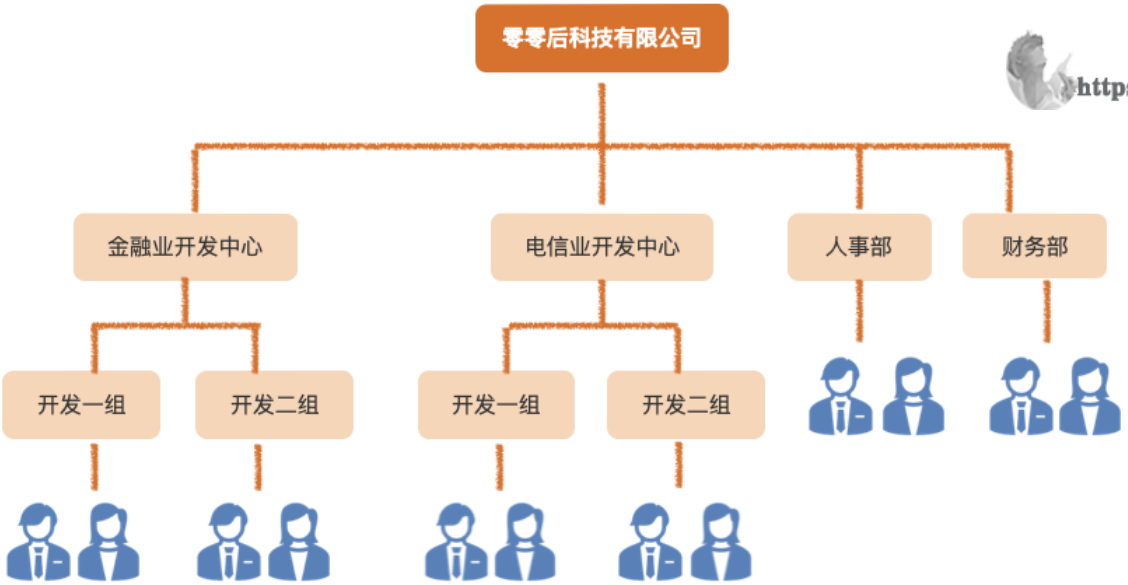
需求二：人员与组织管理

除了对租户进行增删改查等管理，我们还必须还有**人员和组织管理**。

零零后公司有多个开发中心，每个开发中心下有多个开发组。此外，还有公司直属的人事部、财务部等部门，系统要能够管理这些部门。

而且，我们对员工也要进行增删改查，并且把员工分配到部门。每个员工只能属于一个部门，比如王小牛属于开发一组。

下面这个图说明了需求中的人员和组织结构：



需求三：项目管理

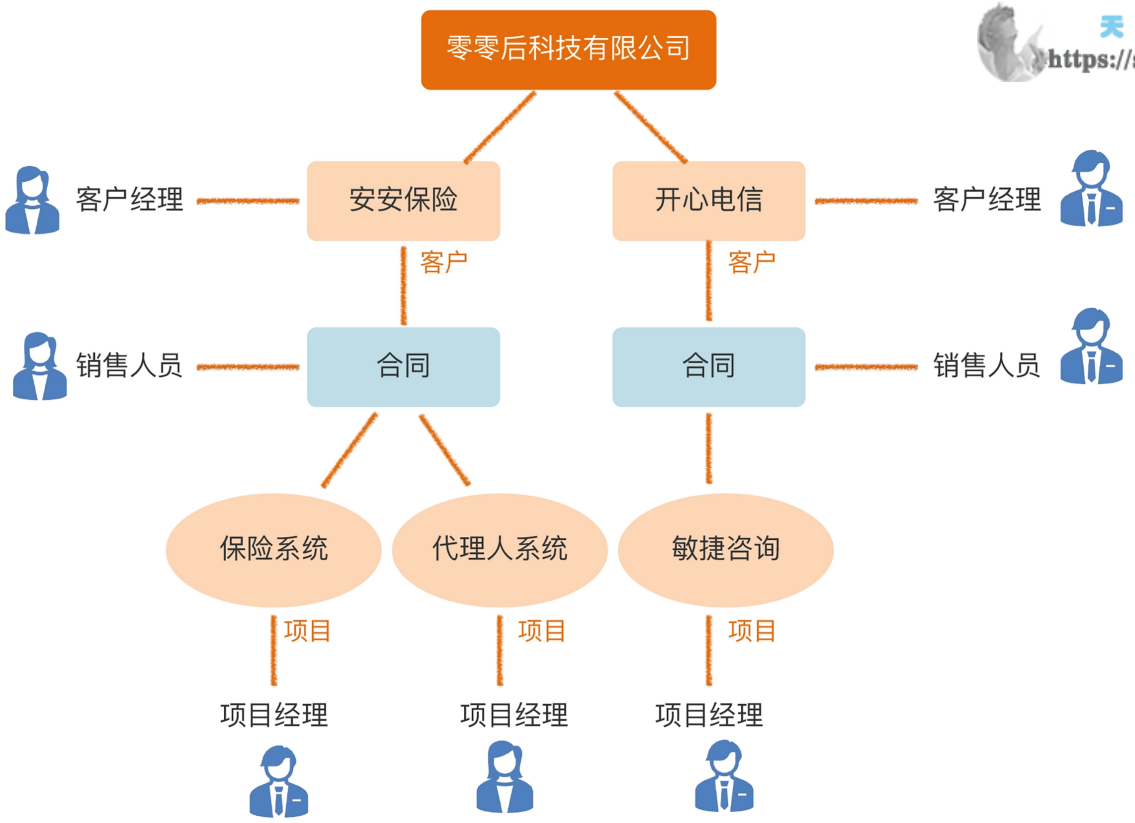
接下来的需求是项目管理。

首先，租户要向它的客户提供服务，就要和客户签订合同，然后完成合同下的项目。比如说，租户零零后公司的客户可能包括安安保险公司、开心电信等等。那么，租户零零后公司就得为安安保险公司，安排一堆程序员帮他们开发核心保险系统项目，也得为开心电信，安排几个咨询师帮他们的敏捷转型做咨询。

这里具体的需求是这样的：

- 一个租户企业可以有多个客户，可以在系统里对客户信息进行增删改查，每个客户都有一个客户经理来跟进；
- 租户可以和它的客户签订多份合同，在系统中对合同信息进行增删改查，每个合同都有一个销售人员负责，可以开始和结束合同；
- 一个合同下可以有多个项目，系统可以对项目进行增删改查，每个项目有一个项目经理；
- 可以在系统中开始和结束项目，需要记录开始和结束时间等信息。项目结束以后，很多事情就不能随便做了。

下图说明了客户、合同、项目等概念的关系：



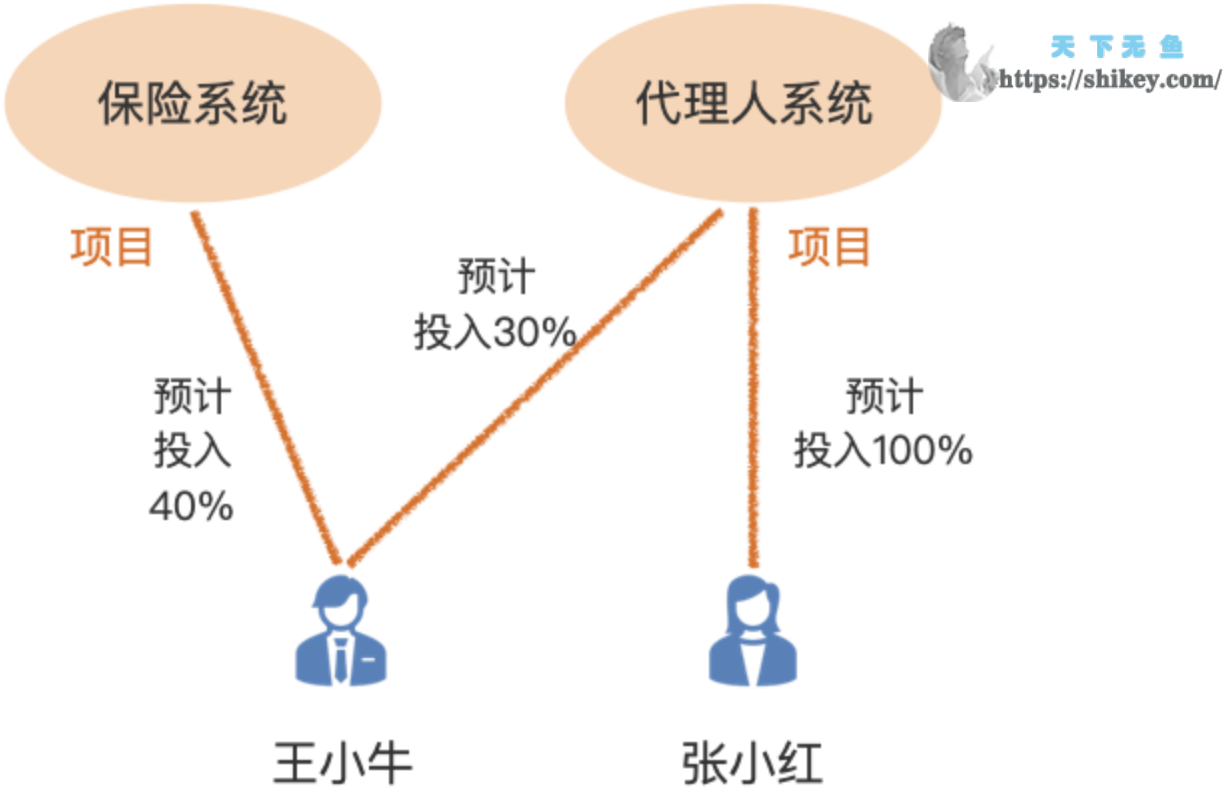
需求四：人员分配

有了项目，我们就可以把员工分配到项目上，也可以让一个人退出项目。而且，一个项目可以有多个人参与，一个人又可以同时参与多个项目。

除了这些，我们在把人分配到项目上的时候，还要记录每个人预计的投入百分比。比如说，王小牛在 A 项目上准备投入自己 30% 的时间，同时在 B 项目准备投入 40% 的时间等等。

之所以要记录预计的投入时间，是为了让企业能够在总体上调配人力资源。比如说管理人员看到王小牛还有 30% 的空余时间，就可以把它同时再安排到另一个项目上，以便实现“内卷最大化”。

下图说明了人员和项目投入的关系：



需求五：工时登记

把人员分配到项目上以后，我们就可以登记工时了。

我们这里规定只有当一个员工分配到一个项目上以后，才能通过这个项目报工时。这样的规定，一来是为了符合公司的管理要求，一个员工不能随意参加一个项目；二来也可以防止员工不小心把工时填到错误的项目上。

而且每个员工每周都需要在系统中填报工时，登记自己哪一天在哪个项目上投入了多少时间。当然，员工可对工时进行查询和修改，还可以为每天的投入的时间填写备注。你可以看一看工时登记的界面原型，进一步理解需求。

工时登记界面原型图

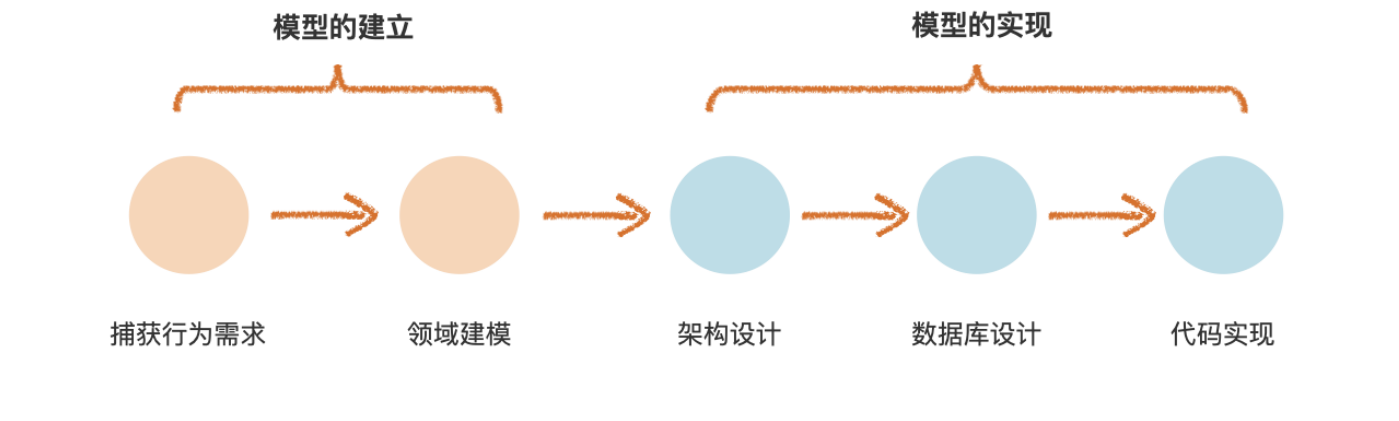
另外，尽管咱们尽量采用接近真实的案例，但也无法面面俱到。为了突出重点，我们这个课程还是省略了一些需求，比如说人员登录和权限控制等等。

现在，第一个迭代的需求基本上就说完了。这些需求看起来简单，其实关键点还是挺多的。而且更重要的是，在实际项目里，这些需求未必会这么清清楚楚、白纸黑字地列出来给你，更多是隐藏在领域专家的脑子里，需要我们去挖掘。

还有一些需求，我故意没有说得很详细，而是要在后续的开发过程中逐步揭示。这也是为了模拟实际项目的场景。

DDD 的基本开发过程

那么，我们怎么用一套系统化的方法，抽丝剥茧、一步一步地把需求落实到代码呢？咱们看看下面这张图，它表示了领域驱动设计中的主要流程。



领域驱动设计主要的开发流程

你可以看到，在整个开发流程中，首先是要**捕获行为需求**，也就是传统软件工程里的“获取需求”。这一步，我们要识别需求里有哪些流程、哪些功能，每个功能由什么人操作，会产生什么结果。

在传统软件工程中，这一步常用的方式是用例建模，也就是用 Use Case 来建模。在这套课程里面，我们会用 DDD 中比较流行的一种方法，叫做“事件风暴”。

接下来，我们就可以**进行领域建模**了，也就是通过建立领域模型，把需求里的主要业务知识描述清楚。DDD 的领域模型，大体上相当于传统软件工程中的分析模型。

基于领域模型，我们就可以**做架构设计**，包括进程间和进程内的架构。比如说微服务设计、中台设计都属于进程间架构。而 DDD 分层架构，通常说的是进程内架构。

然后就可以**根据领域模型进行数据库设计，最后是代码实现**。

这样，就形成了一个基于 DDD 的开发闭环。其实，在实践中，尤其是对敏捷软件开发来说，这些步骤不是线性的，而是反复迭代、互相穿插的。

DDD 是以领域模型为核心的。所以，我们可以把上面说的步骤分成**“模型的建立”**和**“模型的实现”**两部分。

模型的建立阶段，使用的都是业务术语，归根结底来自业务人员，业务人员不仅能听懂，而且负责评价建模的正确性。而模型的实现，则是业务人员不需要理解也不关注的，会包含技术实现方面的内容。这两者的边界很重要，我们在后面还会反复提到。

总结

好，今天就讲到这里，我们来总结一下。

这节课，我们概述了迭代一的目的和需求，并且介绍了 DDD 的基本开发过程。

咱们要开发的是一个基于 SaaS 的企业管理系统，它的需求主要包括租户管理、人员与组织管理、项目管理、人员分配和工时登记这几部分。

在介绍需求的同时，我们也穿插着介绍了领域和领域专家的概念。领域就是软件要解决的业务问题。领域专家则是十分了解业务知识本质的人。

DDD 开发的基本流程是以领域模型为核心的。整个流程可以分为模型的建立和模型的实现两部分。其中模型的建立一定要使用业务语言，而模型的实现则增加了技术语言。模型的建立，是通过领域专家和开发人员的协作共同完成的。

最后，今天讲的所有需求我都汇总到了下面这张表里，你可以看一下，方便后面的实战。

需求一：多租户
系统中可以管理多个租户，每个租户是使用该SaaS服务的一个企业
需求二：人员与组织管理
客户企业有多个开发中心，每个中心下有多个开发组，此外还有公司直属的人事部、财务部等部门
系统要对这些部门进行增删改查
可以对员工进行增删改查
员工可以分配到部门
一个员工只能从属于一个部门
需求三：项目管理
一个租户企业可以有多个客户，可以在系统中对客户信息进行增删改查
每个客户都有一个客户经理来跟进
租户可以和它的客户签订多份合同，在系统中对合同信息进行增删改查
每个合同都有一个销售人员负责
可以开始和结束合同
一个合同下可以有多个项目，系统可以对项目进行增删改查
每个项目有一个项目经理
可以在系统中开始和结束项目，需要记录开始和结束时间等信息
需求四：人员分配
可以为项目分配人员
人员可以退出项目
一个项目可有多个人参与，一个人也可以同时参与多个项目
要记录每个人的投入百分比
需求五：工时登记
员工每周都需要在系统中填报工时，登记自己哪一天在哪个项目上投入了多少时间
可对工时进行查寻
可对工时修改
可为投入的时间填写备注
只有当一个员工分配到一个项目上以后，才能通过这个项目报工时

思考题

1. 如果这是一个真实的项目，你觉得还缺少哪些需求？

2. 希望在你自己的项目里，选择一个不太复杂的部分，然后跟着后续的课程，按照 DDD 的方法尝试实践。

好，今天的课程结束了，有什么问题欢迎在评论区留言，下一节课我们将通过事件风暴的方式，理清系统的行为需求，并为进一步的领域建模打下基础。

上一篇　01丨DDD小传：领域驱动设计为什么这么火？

下一篇　03丨事件风暴（上）：怎样和业务愉快地聊需求？

精选留言 (16)

写留言

吾真本 置顶

2022-12-08 来自广东

钟老师前面把ddd对于复杂系统的分析的价值和开发人员的痛点讲得很透彻，很赞。另外，在“咱们的系统就是要对这些租户进行增删改查等管理”里的“增删改查”的提法，会不会让一部分听众认为crud不算复杂，为何要用ddd? 另外，并不是所有业务场景把crud都用上。比如只有管理员才可以做对租户进行remove，但租户不可以。是否把“增删改查”改为“根据业务需要对租户信息进行管理”好一些？

作者回复: 谢谢道长的建议！这个问题我在写这节课的时候也考虑过，所以您可以看到，目前有些地方用的是“管理”，有些地方用的是“增删改查”。之所以这么写，是考虑两点，第一从课程设计角度出发，第二从实际情况出发。

首先，从课程设计角度，这门课故意模拟实际情况，在真实情况下，业务说需求时可能就是比较随

意，这里缺一点，那里漏一点，这里又不一致一点。所以这节课中的需求就显得没那么完善，在后面的课程会逐渐丰满和严格化。

其次，从实际情况角度，当业务说“增删改查”的时候，一定意味着简单需求吗？可能客户目前只想到增删改查，但深挖下去，其实还有别的，可能有的增删改查背后，有复杂的业务校验逻辑等等。这些都要继续挖掘。这节课只是开始，后面的课会继续深挖。总之，还是希望模拟一种真实的开发过程。

至于这种写法是不是有效，我们还可以听听其他朋友的意见。

共 3 条评论 >

3

业余草 置顶

2022-12-08 来自广东

大家的关注点都在需求，我来总结一下本文的核心知识点。

DDD 是以领域模型为核心的。所以，我们可以把上面说的步骤分成“模型的建立”和“模型的实现”两部分。

模型的建立：事件风暴 -> 领域建模。

模型的实现：架构设计 -> 数据库设计 -> 代码实现。

DDD 的基本开发过程：事件风暴 -> 领域建模 -> 架构设计 -> 数据库设计 -> 代码实现。

后面所有的需求，我相信应该都是围绕整个开发过程来讲述的。

期待后面大家轮流来总结，收获满满！

作者回复: 您的总结很漂亮。提前说一下，等学到后面的课程会发现，事件风暴只是获取行为需求的方法之一，还可以采用别的方法。所以可以把第一步改一下：获取行为需求 -> 领域建模 -> 架构设计 -> 数据库设计 -> 代码实现，这样可能更好些。不过这仍然是为了讲课简化了的流程，有时间我们可以再更全面的讨论一下DDD和开发过程的关系。

2

hk 置顶

2022-12-08 来自广东

终于有门结合案例来进行DDD思想落地的课程了，期待老师的精彩讲解和知识普及，不知道案例代码会结合TDD来实现吗，如果有测试驱动那更好了

作者回复: 您的理解没错，DDD注重演化，所以实践中要配合TDD、重构、持续集成等。不过这门课篇幅所限，倒是不会刻意讲TDD，但后续出了完整的代码例子中会包含。

aoe
2022-12-08 来自广东

现在工作中都使用 MVC，代码结构类似 Controller、Service、DAO。
开发时没有体会过“模型的实现”-架构设计这一步，而是：直接根据需求设计数据库 -> 代码实现。

以我现在的水平，看到文末的需求列表就可以直接实现功能了：
1. 划分功能模块：多租户、人员与组织管理、项目管理、人员分配、工时登记
2. 根据功能描述设计数据库
3. 编码实现

感觉到的问题：
1. 这是一门 DDD 的课程，我这样的写出的代码会无比僵硬，没有《DDD》书中提到的有些地方是柔软的
2. 因为代码僵硬，就算使用 TDD 进行开发，有利于后期重构，但也会因代码简陋（没有设计）给重构工作带来很大的工作量
3. 请教钟老师，推荐哪种方法实践：
A. 是先按自己的想法实现需求汇总里列出的所有需求，再跟着后续课程迭代功能（我能吃苦，但希望不要走弯路）
B. 选一个功能点实现，但是担心选中的这个功能点后续文章直接省略了，重点讲别的功能点（这就有点尴尬了）
C. 或者您有更好的建议

> 作者回复: 每个人都有不同的学习习惯，我只能说一下个人的看法供您参考。建议您先忘记代码，跟着课程尽量先吃透领域建模。到了后面的代码实现部分，我其实只写了一个功能为例，你看了以后，可以用类似的方法尝试写其他功能。然后和自己头脑中原来的做法做比较，应该就可以了。

共 2 条评论 >　　1

骆驼、
2022-12-08 来自广东

还缺少工时统计，包括各个项目下参与人员的投入时长以及每个人员参与项目的投入时间

> 作者回复: 是的，缺少这个。可能在迭代三会加上 ： ）

1

Jxin
2022-12-10 来自湖北

1.有点顺便给自家公司搞个项目管理（包含工时，人员）系统的味道。
2.可以增加的功能 项目工作台，项目仪表盘，项目自动关联，工时缺省填报，工时未填告警等等一系列功能，能做的太多，不罗列了。
3.需要考虑的功能 跨租户业务。 比如商机分销，比如顾问跨租户支援。 这些需要刻画租户模型 租户间协作合同 以及发生跨租户业务后双边的履约刻画以及之间的履约映射 财务结算。

5.少了一个很重要的过程，如何做需求确认。 在未知领域采集业务知识（发散），提炼业务模型/功能点（收敛），评估功能点的必要性（依赖关系）和性价比（按量化价值和成本打分）（发散），制定mvp和演进路线（收敛）。 虽然ddd没写，但这个过程得做，必然可能在开始就凉了。 演进规划一定会有变化，无论是方向还是功能集，可以逐步调整，但不能没有。

Michael
2022-12-10 来自陕西

我比较好奇为什么DDD要有实体和值对象这样的划分？然后就是为什么这样的划分是要通过ID这个听起来很technical的词来划分？

favorlm
2022-12-09 来自广东

钟老师，您好
第一个问题：我感觉需求里缺少“卷卷通”公司的盈利模式，也就是租户管理里缺乏这方面的挖掘。
第二个：我准备选择需求三跟着DDD的流程走一遍。

> 作者回复: 您好呀，你是在从需求的价值层面考虑了，这个意识要赞一个：）
> 选一个需求走一遍的方法也不错。

请叫我和尚
2022-12-09 来自广东

在这里发现一个细节问题，在捕获行为需求阶段，我们梳理了：
1. 需要提供的功能（这里就为后期提供接口功能做铺垫，也算是为功能架构设计做铺垫）
2. 功能下面一些数据之间的关系（这里就为后期做数据库设计做铺垫），比如一个合同下面可以多个项目，这里就是 1：N 的关系
不知道这样理解是否正确

作者回复: 基本是正确的，我再补充一下：

1.关于捕获行为需求，这一课只是开了一个头，下两节课讲事件风暴才是主要部分，在那里，会对后期接口设计起到更具体的作用。

2.按照DDD，这里的需求和数据库设计之间还差了很重要的一步，就是领域建模，领域建模以后，才数据库设计。这是DDD和之前开发方法的主要不同之一。

aoe

2022-12-09 来自广东

刚看到《DDD》第 4 章 分离领域

发现了第一版需求汇总还来得急没分层

作者回复: 分离领域属于“模型的实现”层面，含在分层架构里面，我们后面会专门讲。现在还在需求层面，离后面还差着几步 ： ）

hopez

2022-12-08 来自广东

1、个人觉得还缺少数据统计需求；

2、多租户的数据隔离和权限管理不知道属于业务需求还是技术需求；

作者回复: 没错。

区分业务需求和技术需求，可以这么想，如果没有开发背景的业务人员能明白的就是业务需求，否则就是技术需求。数据隔离和权限，整体而言，业务人员应该能明白，而且开发人员必须和业务达成一致，所以算业务需求。而数据隔离具体采用哪种技术手段，则不是业务需求，是技术决策。

沐瑞Lynn

2022-12-08 来自广东

一、从功能需求和非功能需求的角度讲，目前，还没有描述非功能需求；

二、从用户+场景+行为动机+目标来拆解需求，目前，有些场景也欠缺，譬如：如何分配管理权限等

作者回复: 是的，说的很全面。

老狗

2022-12-08 来自广东

员工提交工时以后干什么用呢？换个提法：作为员工，我如果不按实际工作时间填对公司运营会有什么后果？

> 作者回复: 您这是问到需求的业务价值了，上升了一个层面，在实际工作中，开发人员确实要多问这个层面的问题。我们可以一起想想，比如说我在某个项目上填了2天，而给客户实际上只干了1天，而这是一个按人天收费的合同。那么，我们公司找客户是收1天的钱还是2天的钱呢？这里就出现了不一致。其他情况可以再脑补一下。

飞

2022-12-08 来自广东

可以再补充个服务蓝图，业务有个整体的表达，也可以理清业务角色的关系

> 作者回复: 谢谢，等我考虑下。

leesper

2022-12-08 来自广东

缺少审批功能，真实企业中是需要流程引擎做各种审批的，比如请假，人员进场离场

> 作者回复: 没错，真实情况下一般要有。

bighero

2022-12-08 来自广东

我的思考如下。

需求分为：功能性需求（业务需求），非功能性需求。目前业务功能需求上业务需求上的5w1h都不是太完善，还需要深挖下。而非功能性需求就更少了，比如可扩展性,性能，可用性，可伸缩性，安全性几方面都还需要深挖。

> 作者回复: 您说得非常好

共 5 条评论 >