01 | 架构的本质：如何打造一个有序的系统？

2020-02-19 王庆友

讲述：王庆友
时长 17:48 大小 14.27M

你好，我是王庆友，今天是专栏的第一讲，我想先和你聊聊架构的本质。

我们知道，现在的软件系统越来越复杂，当然相应地，架构的作用也越来越明显。作为开发人员，我们每天都在和架构打交道，在这个过程中，对于架构也经常会产生各种各样的问题：

什么是架构？架构都有哪些分类，分别解决什么问题呢？

怎样才是一个好的架构设计？我怎么才能成长为一名优秀的架构师呢？

这些问题涉及我们对架构的认识，也是学习和运用架构的开始。所以，今天，我们就来深入地分析架构的实质，让你能够透彻地理解它。

作为专栏的第一讲，我希望先和你讨论架构中理念性的部分，就是所谓架构的道，这样可以指导你学习后续的实操层面的内容，也就是架构的术。

接下来，我们就正式开始吧，先说下我对架构本质的理解。

架构的本质

物理学中有个很著名的“熵增定律”：一个封闭系统，都是从有序到无序，也就是它的熵（即混乱程度）会不断地增加，最终系统会彻底变得无序。

这个理论放在软件系统的演化上，也是非常适用的。

一方面，随着业务需求的增加，我们会往系统里不停地添加业务功能；另一方面，随着访问量的不断增加，我们会不断通过技术手段来加强系统非业务性功能。如果事先不做良好的设计，随着时间的推进，整个系统野蛮生长，就会逐渐碎片化，越来越无序，最终被推倒重来。

不过，自然界中的生物可以通过和外界交互，主动进行新陈代谢，制造“负熵”，也就是降低混乱程度，来保证自身的有序性，继续生存。比如，植物通过光合作用，把光能、二氧化碳和水合成有机物，以此滋养自己，延续生命。对于软件系统，我们也可以主动地调整系统各个部分的关系，保证系统整体的有序性，来更好地适应不断增长的业务和技术变化。这种系统内部关系的调整就是通过架构实现的，所以，架构的本质就是：

> **通过合理的内部编排，保证系统高度有序，能够不断扩展，满足业务和技术的变化。**

这里包含两层意思，我们具体展开说下：

首先，架构的出发点是业务和技术在不断复杂化，引起系统混乱，需要通过架构来保证有序。我们知道架构这个词来源于建筑行业，那为什么建筑行业需要“架构”呢？

搭一个草房子很简单，可以直接上手；盖一个 2 层楼房，稍微复杂一些，但在工匠的经验指导下，问题也不大；而盖一座高楼，复杂性就大不一样了，我们需要考虑内部结构、承重、采光、排水、防雷抗震等，这就需要专业人员事先做好整体的架构设计，并严格地按照设计来施工。

这里，你可以看到，建筑里的架构不是天然就有的，而是因为建筑越来越复杂，我们需要通过架构来管理这种复杂性，避免建造过程的失控。

软件系统也是如此，从简单的桌面应用发展到现在的大型互联网平台，这个过程中，系统规模越来越大，业务和技术也越来越复杂。我们同样需要通过架构设计，消化复杂性带来的混乱，使系统始终处于一个有序状态，能够应对现有和将来的需求变化。

其次，架构实现从无序到有序，是通过合理的内部编排实现的，基本的手段，就是“分”与“合”，先把系统打散，然后将它们重新组合，形成更合理的关系。

具体地说，**“分”就是把系统拆分为各个子系统、模块、组件。**拆分的时候，首先要解决每个部分的定位问题，然后根据定位，划分彼此的边界，最后实现合理的拆分，我们比较熟悉的微服务架构，就是一种典型的拆分做法。

“合”就是基于业务流程和技术手段，把各个组件有机整合在一起。比如说在微服务架构中，拆分为具体微服务后，我们需要对这些服务进行归类和分层，有些属于底层基础服务，有些属于上层聚合服务，还要尽可能地实现服务的平台化，比如我们最近说的中台，这些都是合的思想体现。

这个分与合的过程将系统的复杂性分解为两个层次：

首先，各个子系统承担独立的职责，内部包含了自身的复杂性。子系统的复杂性对外部是透明的，外部不用关心。

其次，子系统通过封装后，简化为职责明确的一个点，因此，我们只需要在合的过程中，解决各个点之间的依赖关系，这样就可以定义出系统整体。

举个例子，我们都知道 GoF 的 23 个设计模式，在 Builder 模式中，它的主逻辑只需要给出各个部件的组装关系即可，它不关心创建某个具体部件的内部逻辑，这个可以交给工厂模式去实现。这里，Builder 模式负责粗粒度的组装逻辑，它承担的是合的部分；工厂模式负责细粒度的构造逻辑，承担的是分的部分，大家各自管理自己的复杂性。

通过合理的“分”与“合”，系统不是回到了原点，而是把原先铁板一块的系统变成一个富有弹性的结构化系统。这样，系统的复杂性有效地分解了，系统的有序度大幅度地提升了。

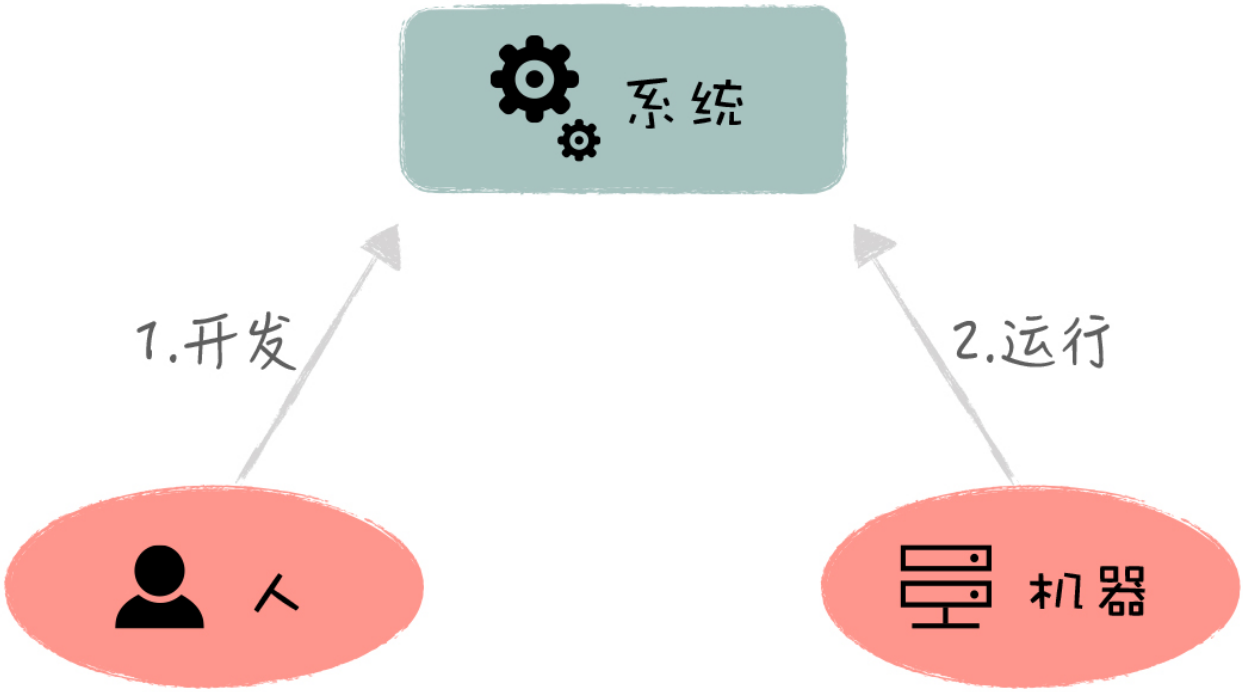
当然，系统的复杂性是多方面的，有技术上和业务上的，架构也是一个体系，会有多种架构一起来应对这些复杂性挑战。那么接下来，我们就来具体看下。

架构的分类

按照不同的角度，架构可以有很多分类，但一般来说，主要分为**业务架构**、**应用架构**和**技术架构**。那么，这些架构分别为谁服务，解决什么问题，相互之间是什么关系呢？

回答这些问题前，我们先来看下系统的落地过程。

系统首先由人来开发，然后由机器来运行，人和机器共同参与一个系统的落地。



对于负责开发的人来说，比较头疼的是，业务太复杂，脑子想不清楚，即使当前勉强把业务逻辑转化为代码，系统后续的维护也是问题。因此，开发人员的要求是系统概念清晰，业务逻辑容易理解，可以直观地进行代码开发。

对于负责运行的机器来说，比较头疼的是，外部请求并发量太大，导致机器扛不住，有的时候，硬件还会出问题。因此，它的要求是系统能够水平扩展，支持硬件容错，保证系统的高性能和高可用。

这里，**开发的痛点主要由业务架构和应用架构来解决，机器的痛点主要由技术架构来解决。**

为什么这么说呢？我们看下，这些架构具体都是做什么用的。

简单来说，业务架构就是讲清楚核心业务的处理过程，定义各个业务模块的相互关系，它从概念层面帮助我们理解系统面临哪些问题以及如何处理；而应用架构就是讲清楚系统内部是怎么组织的，有哪些应用，相互间是怎么调用的，它从逻辑层面帮助我们理解系统内部是如何分工与协作的。

技术架构就是讲清楚系统由哪些硬件、操作系统和中间件组成，它们是如何和我们开发的应用一起配合，应对各种异常情况，保持系统的稳定可用。所以，技术架构从物理层面帮助我们理解系统是如何构造的，以及如何解决稳定性的问题。

这里你可以看到，业务架构、应用架构和技术架构，分别从概念、逻辑和物理层面定义一个系统。业务架构给出了业务模块的划分和依赖关系，这也大致决定了应用系统如何分工和协作，当然这不需要严格地一一对应，比如一个商品业务，可能对应 3 个应用，一个前台商品展示应用、一个后台商品管理应用，以及一个商品基础服务，但这不影响我们从逻辑上理解，一个业务场景，有哪些应用参与，并且它们是如何协作的。

而技术架构呢，通过保障应用的稳定运行，最终保证业务不出问题。比如在大促的时候，多个应用可能会受大流量冲击，技术架构就要考虑怎么通过技术手段，保障相关的应用能够处理高并发，从而保证大促顺利进行。

这里，我举个拍电影的例子，来帮助你更直观地理解这三种架构的关系：业务架构定义了这个电影的故事情节和场景安排；应用架构进一步定义有哪些角色，每个角色有哪些职责，并且在每个场景中，这些角色是如何互动的；技术架构最后确定这些角色由谁来表演，物理场景上是怎么布置的，以此保证整个拍摄能够顺利完成。

最后，我想强调一下：系统是人的系统，架构首先是为人服务的。因此，业务概念清晰、应用分工合理、人好理解是第一位的。然后，我们再考虑技术选型的问题，保证系统非功能性目标的实现。**所以做架构设计时，一般是先考虑业务架构，再应用架构，最后是技术架构。**

什么是好的架构?

从上面的内容，我们不难看出，一个好的架构必须满足两方面挑战：业务复杂性和技术复杂性。

1. 业务复杂性

系统首先要满足当前的业务需求，在此基础上，还要满足将来的业务需求，因此系统要能不断地扩展变化，包括调整现有功能，以及增加新功能。

而且，系统的功能变化不能影响现有业务，不要一修改，就牵一发动全身，到处出问题。因此，在架构设计上，要做到系统的柔性可扩展，能够根据业务变化做灵活的调整。

此外，市场不等人，上新业务要快，之前花了半年上了个业务，这回再上个类似的新业务，需要短时间就能落地。因此，架构设计上，还要做到系统功能的可重用，这样才能通过快速复用，实现业务敏捷和创新。

2. 技术复杂性

要保证一个业务能正常运行，除了满足业务功能之外，还要保证这个系统稳定可用。

一个复杂系统是由很多部分组成的，如应用程序、服务器、数据库、网络、中间件等，都可能会出问题。那怎么在出问题时，能够快速恢复系统或者让备用系统顶上去呢？

还有流量问题，平时流量不大，少量机器就可以处理，但在大促的时候，大量流量进来，系统是不是能够通过简单地加机器方式就能支持呢？

此外还有低成本的问题，系统能否做到，使用廉价设备而不是高大上的 IOE 设备，使用免费的开源组件而不是昂贵的商业套件，使用虚拟化技术而不是物理机，并且在流量低谷和高峰的不同时期，让系统能够弹性缩容和扩容呢？

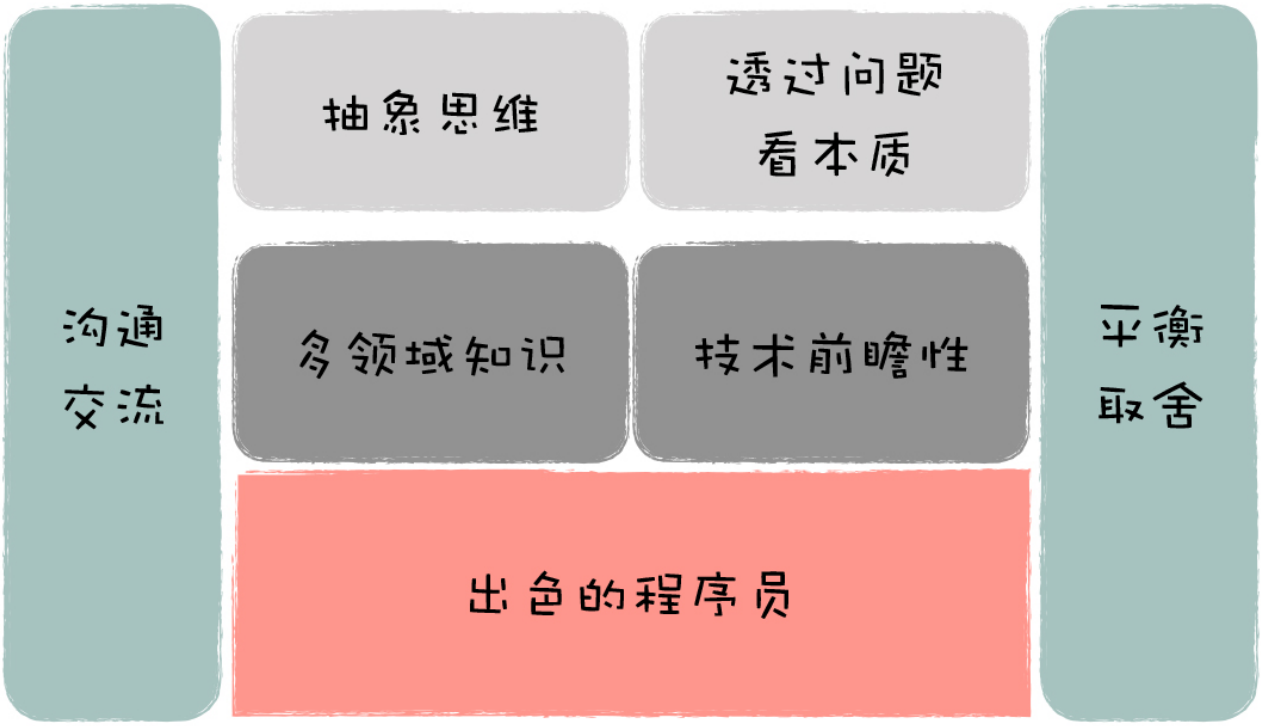
这些都属于技术性的挑战，解决的是系统的非业务性功能，也都是架构设计要支持的。

因此，一个好的架构设计既要满足业务的可扩展、可复用；也要满足系统的高可用、高性能和可伸缩，并尽量采用低成本的方式落地。所以，对架构设计来说，技术和业务两手都要抓，两手都要硬。

那么，一个优秀的架构师需要具备什么样的能力，才能设计一个好的架构呢？

什么是好的架构师？

一个优秀的架构师，应具备很强的综合能力，要内外兼修，“下得厨房，上得厅堂”，下面我来通过典型的架构方式，来介绍一名优秀架构师应该具备的能力：



一个驾校教练，必定开车技术好；一个游泳教练，必定游泳水平好，因为这些都是实践性很强的工作。架构师也是一样，TA 必定是一个**出色的程序员**，写的一手好代码。

在此基础上，**架构师要有技术的广度（多领域知识）和深度（技术前瞻）**。对主流公司的系统设计非常了解，知道优劣长短，碰到实际问题，很快就能提供多种方案供评估。

此外，架构师还需要有**思维的高度，具备抽象思维能力**。抽象思维是架构师最重要的能力，架构师要善于把实物概念化并归类。比如，面对一个大型的 B2C 网站，能够迅速抽象为采购 -> 运营 -> 前台搜索 -> 下单 -> 履单这几大模块，对系统分而治之。

架构师还需要有**思维的深度，能够透过问题看本质**。透过问题看本质是由事物的表象到实质，往深层次挖掘。比如，看到一段 Java 代码，知道它在 JVM（Java Virtual Machine，Java 虚拟机）中如何执行；一个跨网络调用，知道数据是如何通过各种介质（比如网卡端口）到达目标位置。透过问题看本质，可以使架构师能够敏锐地发现底层的真实情况，以端到端闭环的方式去思考问题，能够识别系统的短板并解决它。

还有很重要的一点，能落地的架构才是好架构，所以架构师还需要具备**良好的沟通能力（感性）**，能确保各方对架构达成共识，愿意采取一致的行动；而**良好的平衡取舍能力（理性）**，可以确保架构在现有资源约束下是最合理的，能让理想最终照进现实。

总结

我今天和你分享了架构的本质，架构的终极目标是保证系统的有序，通过拆分和整合，使系统具有柔性，能够进化，从而可以满足现有的和将来的各种变化。

如果你能深入地理解架构的这些本质和手段，就可以不用照搬某某大厂的方案了，而是能够根据实际情况，以最合理的方式来解决系统面临的问题。

这里呢，我也分享了架构的三种典型分类，包括它们各自的定位和相互关系，相信你现在对架构整体有了一个简明的框架，知道架构设计都要做哪些事情了。

最后，我还为你提供了高标准的架构师能力模型，这样，你能比较清楚自己的努力方向是什么，这些要求很高，但你也不要有任何的畏难情绪，你可以在架构实践中，逐步地往这个目标上靠近，通过本专栏后续的学习，相信你也可以更快地达到这个目标。

最后，给你留一道思考题：除了本文提到的三种架构，你还知道有哪些架构分类，它们分别做什么用？

我是王庆友，欢迎在留言区和我互动，我会第一时间给你反馈。如果觉得有收获，也欢迎你把这篇文章分享给你的朋友。感谢收听，我们下期再见。

上一篇　开篇词 | 想吃透架构？你得看看真实、接地气的架构案例

下一篇　02 | 业务架构：作为开发，你真的了解业务吗？

精选留言 (21)

写留言

秋天 置顶

2020-02-22

系统架构？有时候让别人了解我们现有的系统会让提供这个架构图，这属于应用架构吗？

作者回复: 系统架构是一个比较含糊的概念，实践中，也经常被人提到，如果按照业务架构，应用架构，技术架构的分类，系统架构实际上更偏向于技术架构。
专栏里说三种架构分别从概念，逻辑和物理层面定义系统，那我们如何判断一个实际的架构图，它是业务架构，还是应用架构，技术架构呢？
业务架构关注系统大的业务流，数据流和资金流，举个例子，假设A要通知B，在业务架构里，我们简单标识A"通知"B，过程中传递了什么信息即可；在应用架构图里，这个具体的通知方式要给出，也就是从逻辑上理解这个过程是怎么回事的，可能是A同步调用B；也可能是B定时轮询A的接口获取信息，还有可能是A先发送消息到MQ，然后有个监听应用负责接收消息，然后调用B的接口实现。
而如果是技术架构，还要把具体的MQ技术选型给出来，是RabbitMQ呢，还是Kafaka。

当然在技术的架构图里，并没有很明确地按照这个来给出，很可能有些地方按照业务架构简化表达，有些部分给出具体的技术选型，这个我们清楚就可以，不必太纠结，不是每个系统都要给出3个架构图，那太麻烦。

3　2

小伟 置顶

2020-02-20

我很赞同老师从三个维度来、从上至下来理解架构，但相对技术架构，业务架构和应用架构都比较难积累。比如一个公司的业务线往往不会很多，能都参与的机会就更少。而每个行业或多或少都有一些成熟的业务模型，不了解的话会走很多弯路。有什么好的办法能更快接触、积累吗？

展开

作者回复: 架构处理的原则是相同的，类似你会了一门语音，再学新语言就非常快。你可以对照自己熟悉行业的业务系统，去理解它们的系统设计。一般来说，电商行业交易相关的系统设计具有比较好的典型性，其他行业对核心的业务模型了解下就可以。

1 1

Better me
2020-02-23

老师您好，在平常项目开发过程中架构属于全局性的概念，而现阶段纯粹的业务开发导致并没有什么机会去接触架构实践，这种情况我们应该如何在业务开发中深入架构并学习架构相关的东西

作者回复: 业务系统开发也有架构在里面，你需要跳出来，关注系统整体的结构是怎么样，而不仅仅是具体的业务功能逻辑。

1

淘淘
2020-02-20

没有思考就直接购买了

展开

1

Tony
2020-02-20

牛逼，必须支持庆友哥

展开

1

刘杨
2020-02-25

业务架构师应该多数是都是领域专家，用于衔接产品经理，技术架构师，程序员，并整理出团队沟通中的统一语言。只是多数团队中可能就是项目经理或者技术架构师兼职。

Luke
2020-02-25

老师的理解和4+1架构视图模型不谋而合，准备好这几种结构视图，整体的轮廓就清晰了

杨先森
2020-02-24

架构分业务架构 应用架构 技术架构这三类，感觉挺有层次感的，各司其职。之前总是想的是技术架构，其实业务决定了技术的走向。老师解释的非常好，谢谢老师

展开

探索无止境
2020-02-24

老师您好，我看很多文章说分布式是术，微服务架构是道，但是从我的理解是分布式是道，微服务架构是术，不知道老师是怎么理解的?

展开

作者回复: 分布式概念更大，微服务属于分布式的一种。分布式概念太大了，微服务作为一种架构理念和方式，更明确和有针对性。我觉得，这两个更类似是马和白马的关系，谈不上道与术的关系。

约书亚
2020-02-23

熵增这个类比挺有意思

展开

Y
2020-02-23

子系统的复杂性对外部是透明的，外部不用关心。老师，这个描述不对吧

作者回复: 对外部透明的，意思是对外部来说，子系统就是黑盒子，透明是啥都看不到，不是啥都看到

1

孙泽勇
2020-02-22

https://www.processon.com/view/link/5e51378ce4b0c037b5f9d1e3 把内容整理了一

下 方便快速回顾

作者回复: 强，知识点整理得很好，继续啊，一图在手，别无所求。

卫江
2020-02-22

老师，我说一下我的感受不知道对不对，在业务架构设计中，核心是分和和，根据不同的抽象层级，最后形成树的结构，父节点不关心太多的细节而是交给子节点，并负责子节点粒度的业务逻辑整合，这样一来在分层的时候可以通过增加层来降低各个节点的复杂度。同时通过树结构避免循环依赖来使得模块之间关系更加清晰和简单，不知道这么描述对不对？

展开

作者回复: 分解为不同抽象层次是对的，但组合时，一般我们不说树的结构，而是层次化结构，你看具体的业务架构，都是层次结构，一个层可以把多个定位相同的模块包含在一起，简化整体依赖关系，当然也不会有循环依赖的问题。
下一讲就会深入地讲业务架构。

卫江
2020-02-22

之前看了很多架构相关的知识，但是很少能很完整的描述的，比如有介绍设计原则，设计模式，重构，分布式架构，大部分书都是讲某一方面让我很疑惑，很难形成对于架构的全貌，但是通过老师的讲解，学习到其实它们都是架构的一部分并能把它们整合起来，感谢。

展开

作者回复: 这就是我在开篇词讲的架构学习比较碎片化，没有形成体系。

乐乐
2020-02-21

我就是做B端和G端业务的同学，业务复杂度很高，不过确实不太会表达出技术难度

Kǎfkā²⁰²⁰

2020-02-21

对于内部系统来说，业务架构再往上还有一层，可以叫管理架构。管理架构决定业务应该怎么走，将来可能怎么走，内部怎么协作。深入思考和洞察这些，有利于提前对业务可能发生的变化做一些预判

作者回复: 按照比较规范的TOGAF架构分法，上面确实还有企业架构，不过这个是老板定，架构师到不了这个层面。

天灾

2020-02-20

必须赞一个

展开

Rory

2020-02-20

子系统的复杂性对外部是透明的，外部不用关心

——————

这里是 复杂性对外部不用透明，外部不关心。

我这理解正确么?

展开

作者回复: 透明这个词大家都在用，但是容易误解。

A对B透明，实际上表达的是，对于B来说，把A当作黑盒子看就行，透明指的是你啥都看不到。

1

一步

2020-02-20

架构的本质: 通过分，合让系统从无序变的有序

展开

Mandalorian

2020-02-20

来得太及时了。

展开