30 | 异地多活设计4步走

2018-07-05 李运华

30 | 异地多活设计4步走
朗读人：黄洲君　12′27″ | 5.70M

上一期，基于异地多活架构设计复杂度最高的“跨城异地”，我结合自己的经验总结了异地多活设计的 4 个技巧及其核心思想，我认为掌握这些技巧是进入具体设计步骤的前提。

今天，在掌握这 4 大技巧的基础上，我来讲讲跨城异地多活架构设计的 4 个步骤。

第 1 步：业务分级

按照一定的标准将业务进行分级，挑选出核心的业务，只为核心业务设计异地多活，降低方案整体复杂度和实现成本。

常见的分级标准有下面几种：

- 访问量大的业务

以用户管理系统为例，业务包括登录、注册、用户信息管理，其中登录的访问量肯定是最大的。

- 核心业务

以 QQ 为例，QQ 的主场景是聊天，QQ 空间虽然也是重要业务，但和聊天相比，重要性就会低一些，如果要从聊天和 QQ 空间两个业务里面挑选一个做异地多活，那明显聊天要更重要（当然，此类公司如腾讯，应该是两个都实现了异地多活的）。

- 产生大量收入的业务

同样以 QQ 为例，聊天可能很难为腾讯带来收益，因为聊天没法插入广告；而 QQ 空间反而可能带来更多收益，因为 QQ 空间可以插入很多广告，因此如果从收入的角度来看，QQ 空间做异地多活的优先级反而高于 QQ 聊天了。

以我们一直在举例的用户管理系统为例，“登录”业务符合“访问量大的业务”和“核心业务”这两条标准，因此我们将登录业务作为核心业务。

第 2 步：数据分类

挑选出核心业务后，需要对核心业务相关的数据进一步分析，目的在于识别所有的数据及数据特征，这些数据特征会影响后面的方案设计。

常见的数据特征分析维度有：

- 数据量

这里的数据量包括总的数据量和新增、修改、删除的量。对异地多活架构来说，新增、修改、删除的数据就是可能要同步的数据，数据量越大，同步延迟的几率越高，同步方案需要考虑相应的解决方案。

- 唯一性

唯一性指数据是否要求多个异地机房产生的同类数据必须保证唯一。例如用户 ID，如果两个机房的两个不同用户注册后生成了一样的用户 ID，这样业务上就出错了。

数据的唯一性影响业务的多活设计，如果数据不需要唯一，那就说明两个地方都产生同类数据是可能的；如果数据要求必须唯一，要么只能一个中心点产生数据，要么需要设计一个数据唯一生成的算法。

- 实时性

实时性指如果在 A 机房修改了数据，要求多长时间必须同步到 B 机房，实时性要求越高，对同步的要求越高，方案越复杂。

- 可丢失性

可丢失性指数据是否可以丢失。例如，写入 A 机房的数据还没有同步到 B 机房，此时 A 机房机器宕机会导致数据丢失，那这部分丢失的数据是否对业务会产生重大影响。

例如，登录过程中产生的 session 数据就是可丢失的，因为用户只要重新登录就可以生成新的 session；而用户 ID 数据是不可丢失的，丢失后用户就会失去所有和用户 ID 相关的数据，例如用户的好友、用户的钱等。

- 可恢复性

可恢复性指数据丢失后，是否可以通过某种手段进行恢复，如果数据可以恢复，至少说明对业务的影响不会那么大，这样可以相应地降低异地多活架构设计的复杂度。

例如，用户的微博丢失后，用户重新发一篇一模一样的微博，这个就是可恢复的；或者用户密码丢失，用户可以通过找回密码来重新设置一个新密码，这也算是可以恢复的；而用户账号如果丢失，用户无法登录系统，系统也无法通过其他途径来恢复这个账号，这就是不可恢复的数据。

我们同样以用户管理系统的登录业务为例，简单分析如下表所示。

数据	数据量	唯一性	实时性	可丢失性	可恢复性
用户ID	每天新增1万注册用户	全局唯一	5秒内同步	不可丢失	不可恢复
用户密码	每天1千用户修改密码	用户唯一	5秒内同步	可丢失	可重置密码恢复
登录session	每天1000万	全局唯一	无须同步	可丢失	可重复生成

第 3 步：数据同步

确定数据的特点后，我们可以根据不同的数据设计不同的同步方案。常见的数据同步方案有：

- 存储系统同步

这是最常用也是最简单的同步方式。例如，使用 MySQL 的数据主从数据同步、主主数据同步。

这类数据同步的优点是使用简单，因为几乎主流的存储系统都会有自己的同步方案；缺点是这类同步方案都是通用的，无法针对业务数据特点做定制化的控制。例如，无论需要同步的数据量有多大，MySQL 都只有一个同步通道。因为要保证事务性，一旦数据量比较大，或者网络有延迟，则同步延迟就会比较严重。

- 消息队列同步

采用独立消息队列进行数据同步，常见的消息队列有 Kafka、ActiveMQ、RocketMQ 等。

消息队列同步适合无事务性或者无时序性要求的数据。例如，用户账号，两个用户先后注册了账号 A 和 B，如果同步时先把 B 同步到异地机房，再同步 A 到异地机房，业务上是没有问题的。而如果是用户密码，用户先改了密码为 m，然后改了密码为 n，同步时必须先保证同步 m 到异地机房，再同步 n 到异地机房；如果反过来，同步后用户的密码就不对了。因此，对于新注册的用户账号，我们可以采用消息队列同步了；而对于用户密码，就不能采用消息队列同步了。

- 重复生成

数据不同步到异地机房，每个机房都可以生成数据，这个方案适合于可以重复生成的数据。例如，登录产生的 cookie、session 数据、缓存数据等。

我们同样以用户管理系统的登录业务为例，针对不同的数据特点设计不同的同步方案，如下表所示。

数据	数据量	唯一性	实时性	可丢失性	可恢复性	同步方案
用户ID	每天新增1万注册用户	全局唯一	5秒内同步	不可丢失	不可恢复	消息队列同步
用户密码	每天1千用户修改密码	用户唯一	5秒内同步	可丢失	可重置密码恢复	MySQL同步
登录session	每天1000万	全局唯一	无须同步	可丢失	可重复生成	重复生成

第 4 步：异常处理

无论数据同步方案如何设计，一旦出现极端异常的情况，总是会有部分数据出现异常的。例如，同步延迟、数据丢失、数据不一致等。异常处理就是假设在出现这些问题时，系统将采取什么措施来应对。异常处理主要有以下几个目的：

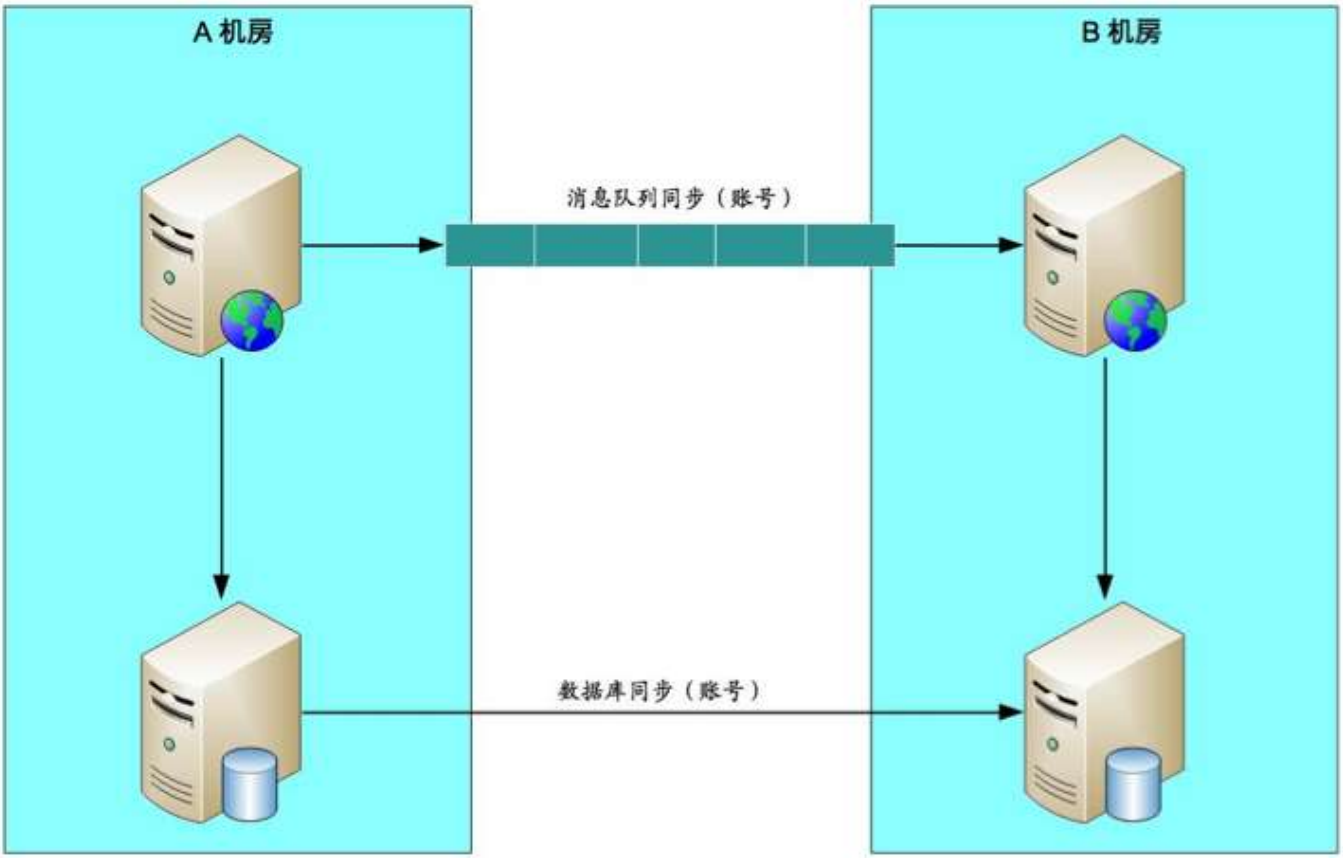
- 问题发生时，避免少量数据异常导致整体业务不可用。
- 问题恢复后，将异常的数据进行修正。
- 对用户进行安抚，弥补用户损失。

常见的异常处理措施有这几类：

1. 多通道同步

多通道同步的含义是采取多种方式来进行数据同步，其中某条通道故障的情况下，系统可以通过其他方式来进行同步，这种方式可以应对同步通道处故障的情况。

以用户管理系统中的用户账号数据为例，我们的设计方案一开始挑选了消息队列的方式进行同步，考虑异常情况下，消息队列同步通道可能中断，也可能延迟很严重；为了保证新注册账号能够快速同步到异地机房，我们再增加一种 MySQL 同步这种方式作为备份。这样针对用户账号数据同步，系统就有两种同步方式：MySQL 主从同步和消息队列同步。除非两个通道同时故障，否则用户账号数据在其中一个通道异常的情况下，能够通过另外一个通道继续同步到异地机房，如下图所示。

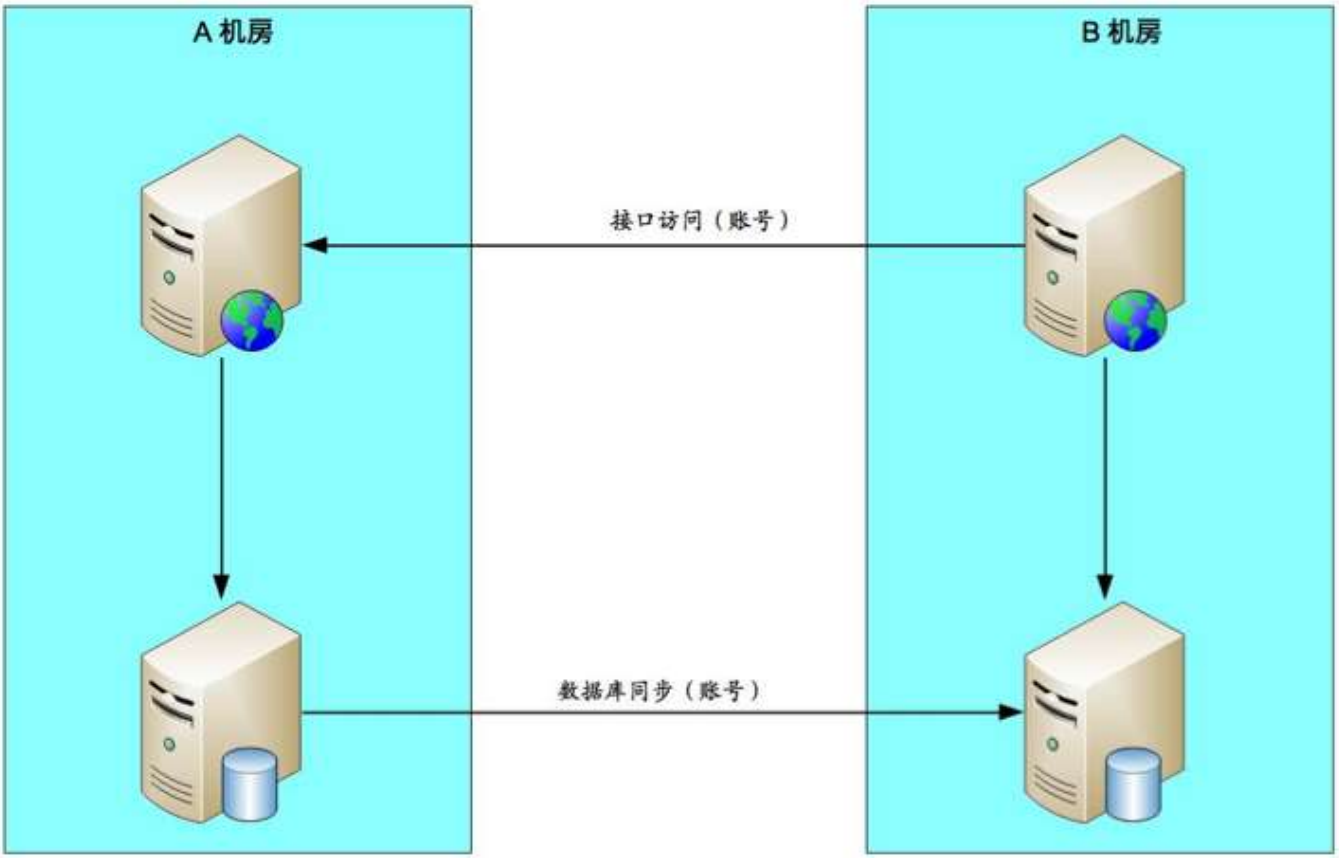


多通道同步设计的方案关键点有：

- 一般情况下，采取两通道即可，采取更多通道理论上能够降低风险，但付出的成本也会增加很多。
- 数据库同步通道和消息队列同步通道不能采用相同的网络连接，否则一旦网络故障，两个通道都同时故障；可以一个走公网连接，一个走内网连接。
- 需要数据是可以重复覆盖的，即无论哪个通道先到哪个通道后到，最终结果是一样的。例如，新建账号数据就符合这个标准，而密码数据则不符合这个标准。

2. 同步和访问结合

这里的访问指异地机房通过系统的接口来进行数据访问。例如业务部署在异地两个机房 A 和 B，B 机房的业务系统通过接口来访问 A 机房的系统获取账号信息，如下图所示。



同步和访问结合方案的设计关键点有：

- 接口访问通道和数据库同步通道不能采用相同的网络连接，不能让数据库同步和接口访问都走同一条网络通道，可以采用接口访问走公网连接，数据库同步走内网连接这种方式。
- 数据有路由规则，可以根据数据来推断应该访问哪个机房的接口来读取数据。例如，有 3 个机房 A、B、C，B 机房拿到一个不属于 B 机房的数据后，需要根据路由规则判断是访问 A 机房接口，还是访问 C 机房接口。
- 由于有同步通道，优先读取本地数据，本地数据无法读取到再通过接口去访问，这样可以大大降低跨机房的异地接口访问数量，适合于实时性要求非常高的数据。

3. 日志记录

日志记录主要用于用户故障恢复后对数据进行恢复，其主要方式是每个关键操作前后都记录相关一条日志，然后将日志保存在一个独立的地方，当故障恢复后，拿出日志跟数据进行对比，对数据进行修复。

为了应对不同级别的故障，日志保存的要求也不一样，常见的日志保存方式有：

- 服务器上保存日志，数据库中保存数据，这种方式可以应对单台数据库服务器故障或者宕机的情况。

- 本地独立系统保存日志，这种方式可以应对某业务服务器和数据库同时宕机的情况。例如，服务器和数据库部署在同一个机架，或者同一个电源线路上，就会出现服务器和数据库同时宕机的情况。

- 日志异地保存，这种方式可以应对机房宕机的情况。

上面不同的日志保存方式，应对的故障越严重，方案本身的复杂度和成本就会越高，实际选择时需要综合考虑成本和收益情况。

4. 用户补偿

无论采用什么样的异常处理措施，都只能最大限度地降低受到影响的范围和程度，无法完全做到没有任何影响。例如，双同步通道有可能同时出现故障、日志记录方案本身日志也可能丢失。因此，无论多么完美的方案，故障的场景下总是可能有一小部分用户业务上出问题，系统无法弥补这部分用户的损失。但我们可以采用人工的方式对用户进行补偿，弥补用户损失，培养用户的忠诚度。简单来说，系统的方案是为了保证 99.99% 的用户在故障的场景下业务不受影响，人工的补偿是为了弥补 0.01% 的用户的损失。

常见的补偿措施有送用户代金券、礼包、礼品、红包等，有时为了赢得用户口碑，付出的成本可能还会比较大，但综合最终的收益来看还是很值得的。例如暴雪《炉石传说》2017 年回档故障，暴雪给每个用户大约价值人民币 200 元的补偿，结果玩家都求暴雪再来一次回档，形象地说明了玩家对暴雪补偿的充分认可。

> 只要在 2017 年 1 月 18 日 18 点之前登录过国服《炉石传说》的玩家，均可获得与 25 卡牌包等值的补偿，具体如下：
> 1000 游戏金币；
> 15 个卡牌包：经典卡牌包 x5、上古之神的低语卡牌包 x5、龙争虎斗加基森卡牌包 x5。

小结

今天我为你讲了异地多活设计的具体步骤，分别是业务分级、数据分类、数据同步和异常处理，希望对你有所帮助。

这就是今天的全部内容，留一道思考题给你吧，业务分级讨论的时候，产品说 A 也很重要，因为影响用户使用；B 也很重要，因为影响公司收入；C 也很重要，因为会导致客户投诉……这种情况下我们该如何处理业务分级？

欢迎你把答案写到留言区，和我一起讨论。相信经过深度思考的回答，也会让你对知识的理解更加深刻。（编辑乱入：精彩的留言有机会获得丰厚福利哦！）

精选留言

小浩子 3

之前学到过一个应对这种多因素筛选的方法，不知道对不对：列出所有参与讨论的业务，以及考虑到的影响因素（访问量、影响使用、收益……），每个因素设定一个权值，每个业务针对每个因素进行打分（权值和打分均不允许有重复值），最后计算每个业务的加权和，得出核心业务

2018-07-05

作者回复

打分最后会变成打架 参考前面备选方案选择中介绍的内容

2018-07-05

Enjoystudy 2

我们如果通过某种路由策略，比如用户id保证同一个用户的请求都路由到同一个机房就可以解决同步延时的问题

2018-07-12

作者回复

这个机房挂了怎么办？

2018-07-13

summer 2

业务分级讨论的时候，产品说 A 也很重要，因为影响用户使用；B 也很重要，因为影响公司收入；C 也很重要，因为会导致客户投诉……这种情况下我们该如何处理业务分级？
了解产品挣钱的底层商业逻辑，就可以知道怎么分级了。不是每一单都要赚钱的，在这里亏的，在别处或下次再挣回来。比如微信聊天业务不挣钱，但是影响面大，坏了口碑，大家不来玩了，商家凭什么来投广告。减少影响面，然后跪求被影响用户的原谅

2018-07-10

作者回复

挺好的思路，根据商业模式的核心来判断

2018-07-11

xuan 2

业务分级我觉得得从如下方面来分析：
1.部门或者公司当期的发力点，如果着重新用户的增长 我觉得顺序应该是ACB，如果着重现有客户满意度，我觉得顺序应该是CBA，如果看重看重收益，那B的优先级最高
2.可通过预期货币价值分析，进行业务分级；大致维度如下：风险发生概率 风险损耗成本 技术改造陈本 技术改造时长（月） 改造后成本节省（月）

2018-07-05

作者回复

很赞👍

2018-07-05

但莫 2

多个场景都重要的情况可以进行层次分析，首先俩俩比较，得到分析矩阵，进行归一化处理，经过计算可以得到每种场景的重要性权重。

当然，俩俩比较的时候判断哪个更重要就看评判人的喜好了，这一步可以通过头脑风暴的到一个平均值。

记得之前的课程中介绍过另一种方法，一时没想起来名字。

2018-07-05

作者回复

前面在评估备选方案的时候，我给出了"360度评估"和"按优先级选择"的两个技巧

2018-07-05

空档滑行 1

通常来说toC的应用，影响用户使用的应该是比较靠前的，尤其是产品还处于抢市场的阶段。to B的应用客户买的是专业性，相对来说用户体验可以往后排，用户能使用和公司收入还是比较重要的。客户相对固定，可以通过客服主动联系客户的方式来做部分挽回

2018-07-05

作者回复

赞同，根据业务和公司判断

2018-07-09

成功 0

B优先级最高，A第二，C第三。B要考虑跨城多活，A做同城多活，C不同机房就行

2018-07-09

作者回复

如果是阿里，“用户第一”应该选A，所以不同公司有不同的标准，没有统一答案😁

2018-07-09

凡凡 0

对于业务分级，往往不会有固定的优先级，首先最重要的是明确各种情况的没在逻辑，如何影响用户，如何影响公司收入，如何导致客诉。明确内在逻辑之后，客诉一般要优先处理，不然对公司的声誉影响很大，也会造成公司业务部门对技术部门信任的崩塌。另外，外讨论的时候，需要看各种情况是否能够以迭代的方式，同步开展，小步幅行进，。现实情况，这往往也是居多的一种结果。

2018-07-08

作者回复

是的，没有固定分级，和具体公司具体业务有关

2018-07-09

刹那间的永恒 0

关于最后的思考题我认为首先确保用户体验才能扩大用户群体，同时一定程度上减少用户投诉以及间接性为公司增加收入；其次处理客户投诉效率同样也能为公司带来口碑，创造粉丝用户，所以我认为a>c>b

2018-07-05

作者回复

没有绝对的答案，和公司的业务相关

2018-07-09

XNorth 0

a>c>b

保证用户正常使用最重要，然后是可能导致用户投诉的业务，最后是公司利益。

业务稳定，利益最后肯定会上来，不然只是眼前的利益。

2018-07-05

作者回复

不一定，不同发展阶段和规模要求不一样，不过通常情况下，客户第一总是没错的😂

2018-07-05

王维 0

另外，想问下华仔，消息队列在分布式系统中比较重要，您能写一篇消息队列的文章吗？例如什么时候用消息队列，用消息队列要注意什么？万分期待！谢谢！

2018-07-05

作者回复

网上的资料已经很多了，可以尝试按照专栏的内容的指导，去理解这些案例，看看会不会有什么不同的收获

2018-07-05

王维 0

是不是可以这样理解:凡是唯一性的数据，如果发生了丢失，都是不可以恢复的？

2018-07-05

作者回复

不一定，有备份就可以恢复

2018-07-05

Seven4X 0

没有实现细节，看完只记住了几个名词概念，是要课上建索引，课下再用功，不然怕是会成为ppt架构师

2018-07-05

作者回复

找个自己的业务按照步骤模拟操作一下

2018-07-05

feifei 0

我觉得是不同类型的公司分类是不一样的，公司阶段的不同也不一样
比如新闻的初创公司，第一步就是积累用户，那么久是用户投诉最高，体验第二，最后是收入。
公司新推出了一个业务，目的是提高用户的粘性，这类用户体验首先解决，然后投诉，最后收入。
在比如转账收费类业务，那优先保证的就是收入了，其次就是投诉，最后是体验

不同类型的业务，在不同场景下，就有不同的选择，这就需要根据实际情况来选择了

老师不知道你是否有一个标准呢？

2018-07-05

作者回复

你已经回答很好了，按照业务最需要的就是标准

2018-07-05

浪子恒心 0

异地多活的一个关键就是要把用户锁定在一个机房，避免在多机房飘来飘去。

2018-07-05

作者回复

这种做法只是某些业务场景要求，例如网购；而有的业务不要求这样做，例如微博

2018-07-05

浪子恒心 0

能讲讲阿里的异地多活是怎么做的吗？

2018-07-05

作者回复

网上资料很多

2018-07-05

阳光灿烂的日子

0

华仔，otter是不是可以保证数据实时同步？

2018-07-05

作者回复

官方介绍都是用“准实时”😄

2018-07-05