作者：高德地图技术人员

一、为什么要做单元化

- 单机房资源瓶颈

随着业务体量和服务用户群体的增长，单机房或同城双机房无法支持服务的持续扩容。

- 服务异地容灾

异地容灾已经成为核心服务的标配，有的服务虽然进行了多地多机房部署，但数据还是只在中心机房，实现真正意义上的异地多活，就需要对服务进行单元化改造。

二、高德单元化的特点

在做高德单元化项目时，我们首先要考虑的是结合高德的业务特点，看高德的单元化有什么不一样的诉求，这样就清楚哪些经验和方案是可以直接拿来用的，哪些又是需要我们去解决的。

高德业务和传统的在线交易业务还是不太一样，高德为用户提供以导航为代表的出行服务，很多业务场景对服务的RT要求会很高，所以在做单元化方案时，尽可能减少对整体服务RT的影响就是我们需要重点考虑的问题，尽量做到数据离用户近一些。转换到单元化技术层面需要解决两个问题:

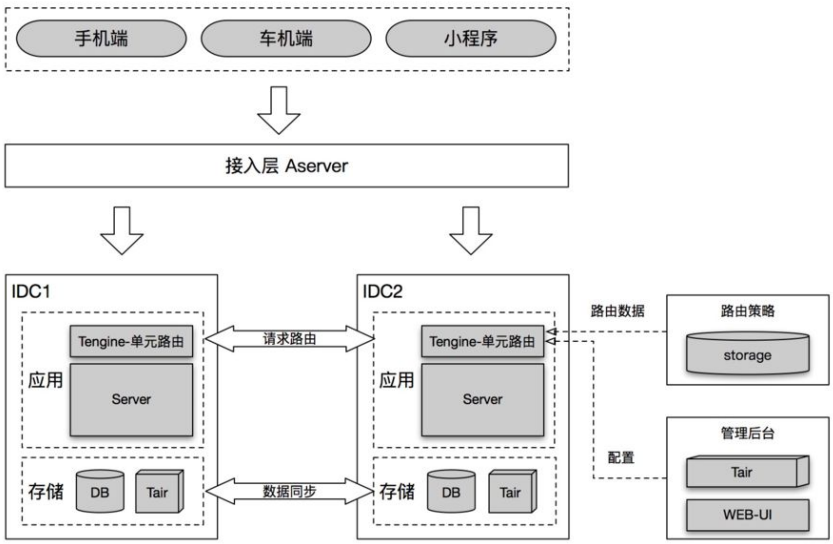
1.用户设备的单元接入需要尽可能的做到就近接入，用户真实地理位置接近哪个单元就接入哪个单元，如华北用户接入到张北，华南接入到深圳。

2.用户的单元划分最好能与就近接入的单元保持一致，减少单元间的跨单元路由。如用户请求从深圳进来，用户的单元划分最好就在深圳单元，如果划到张北单元就会造成跨单元路由。

另外一个区别就是高德很多业务是无须登录的，所以我们的单元化方案除了用户ID也要支持基于设备ID。

三、高德单元化实践

服务的单元化架构改造需要一个至上而下的系统性设计，核心要解决请求路由、单元封闭、数据同步三方面问题。

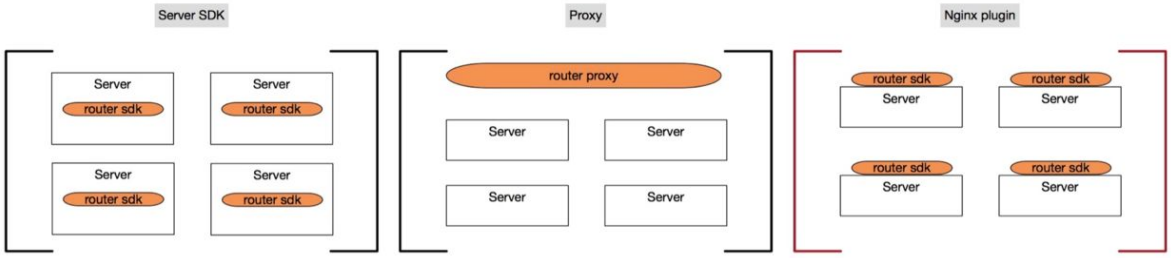


请求路由：根据高德业务的特点，我们提供了取模路由和路由表路由两种策略，目前上线应用使用较多的是路由表路由策略。

单元封闭：得益于集团的基础设施建设，我们使用vipserver、hsf等服务治理能力保证服务同机房调用，从而实现单元封闭(hsf unit模式也是一种可行的方案，但个人认为同机房调用的架构和模式更简洁且易于维护)。

数据同步：数据部分使用的是集团DB产品提供的DRC数据同步。

单元路由服务采用什么样的部署方案是我们另一个要面临的问题，考虑过以下三种方案：

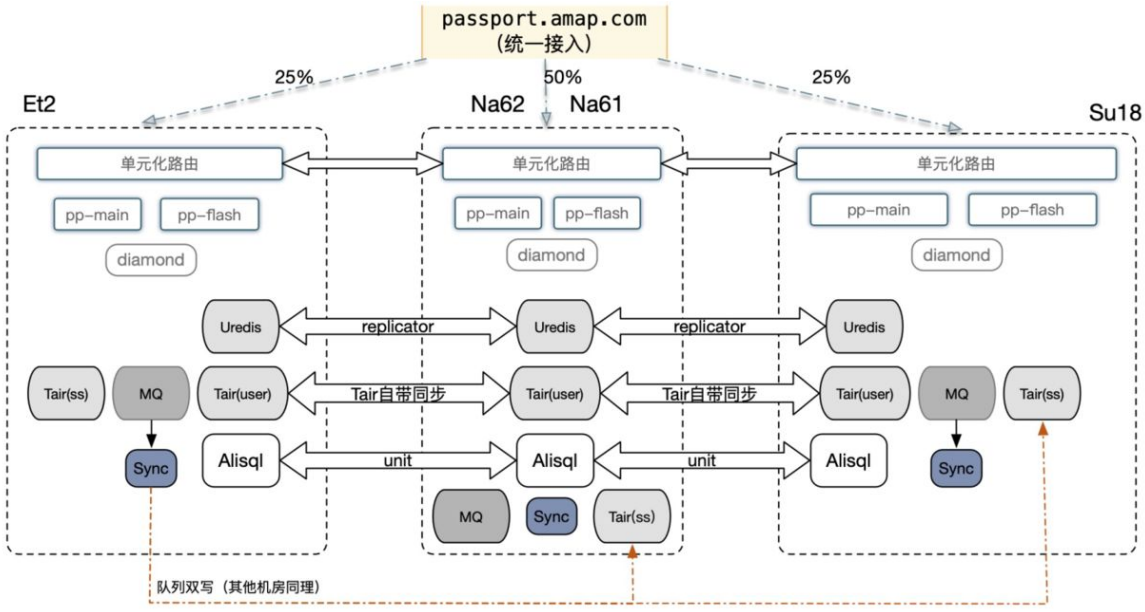


第一种SDK的方式因为对业务的强侵入性是首先被排除的，统一接入层进行代理和去中心化插件集成两种方案各有利弊，但当时首批要接入单元化架构的服务很多都还没有统一接入到gateway，所以基于现状的考虑使用了去中心化插件集成的方式，通过在应用的nginx集成UnitRouter。

服务单元化架构

目前高德账号，云同步、用户评论系统都完成了单元化改造，采用三地四机房部署，写入量较高的云同步服务，单元写高峰能达到数w+QPS (存储是mongodb集群)。

以账号系统为例介绍下高德单元化应用的整体架构。



账号系统服务是三地四机房部署，数据分别存储在tair为代表的缓存和XDB里，数据存储三地集群部署、全量同步。账号系统服务器的Tengine上安装UntiRouter，它请求的负责单元识别和路由，用户单元划分是通过记录用户与单元关系的路由表来控制。

PS：因历史原因缓存使用了tair和自建的uredis(在redis基础上添加了基于log的数据同步功能)，目前已经在逐步统一到tair。数据同步依赖tair和alisql的数据同步方案，以及自建的uredis数据同步能力。

就近接入实现方案

为满足高德业务低延时要求，就要想办法做到数据(单元)离用户更近，其中有两个关键链路，一个是通过aserver接入的外网连接，另一个是服务内部路由(尽可能不产生跨单元路由)。

措施1：客户端的外网接入通过aserver上的配置，将不同地理区域(七个大区)的设备划分到对应近的单元，如华北用户接入张北单元。

措施2：通过记录用户和单元关系的路由表来划分用户所属单元，这个关系是通过系统日志分析出来的，用户经常从哪个单元入口进来，就会把用户划分到哪个单元，从而保证请求入口和单元划分的相对一致，从而减少跨单元路由。

所以，在最终的单元路由实现上我们提供了传统的取模路由，和为降延时而设计的基于路由表路由两种策略。同时，为解无须登录的业务场景问题，上述两种策略除了支持用户ID，我们同时也支持设备ID。

路由表设计

路由表分为两部分，一个是用户-分组的关系映射表，另一个是分组-单元的关系映射表。在使用时，通过路由表查对应的分组，再通过分组看用户所属单元。分组对应中国大陆的七个大区。

先看“用户-(大区)分组”：

路由表是定期通过系统日志分析出来的，看用户最近IP属于哪个大区就划分进哪个分组，同时也对应上了具体单元。当一个北京的用户长期去了深圳，因IP的变化路由表更新后将划进新大区分组，从而完成用户从张北单元到深圳单元的迁移。

再看“分组-单元”：

分组与单元的映射有一个默认关系，这是按地理就近来配置的，比如华南对应深圳。除了默认的映射关系，还有几个用于切流预案的关系映射。

单元分组	默认配置	单元一切流预案	单元二切流预案
东北	单元一	单元二	单元一
华北	单元一	单元三	单元一
西北	单元二	单元二	单元三
.....	单元三	单元三	单元三
模1分组	单元一	单元二	单元二
模2分组	单元二	单元二	单元二
模3分组	单元三	单元三	单元三

老用户可以通过路由表来查找单元，新用户怎么办？对于新用户的处理我们会降级成取模的策略进行单元路由，直至下次路由表的更新。所以整体上看新用户跨单元路由比例肯定是比老用户大的多，但因为新用户是一个相对稳定的增量，所以整体比例在可接受范围内。

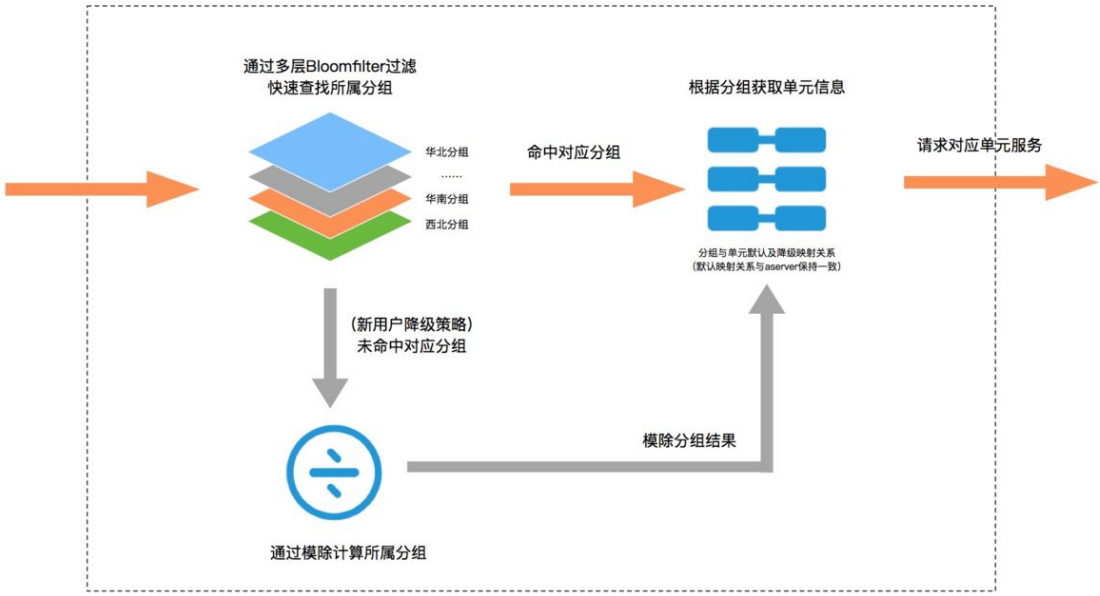
	BitMap	MapDB	Bloom Filter
支持模型	Uid	Uid、Tid、...	Uid、Tid、...
空间占用	小	大	适中
效率	良好	一般	良好
准确率	100%	100%	>99%
实时修改	是	是	否

路由计算

有了路由表，接下来就要解工程化应用的问题，性能、空间、灵活性和准确率，以及对服务稳定性的影响这几个方面是要进行综合考虑的，首先考虑外部存储会增加服务的稳定性风险，后面我们在BloomFilter 、BitMap和MapDB多种方案中选择BloomFilter，万分之几的误命中率导致的跨单元路由在业务可接受范围内。

通过日志分析出用户所属大区后，我们将不同分组做成多个布隆过滤器，计算时逐层过滤。这个计算有两种特殊情况：

1. 因为BloomFilter存在误算率，有可能存在一种情况，华南分组的用户被计算到华北了，这种情况比例在万分之3 (生成BloomFilter时可调整)，它对业务上没有什么影响，这类用户相当于被划分到一个非所在大区的分组里，但这个关系是稳定的，不会影响到业务，只是存在跨单元路由，是可接受的。
2. 新用户不在分组信息里，所以经过逐层的计算也没有匹配到对应大区分组，此时会使用取模进行模除分组的计算。



如果业务使用的是取模路由而非路由表路由策略，则直接根据tid或uid计算对应的模除分组，原理简单不详表了。

单元切流

在发生单元故障进行切流时，主要分为四步骤：

1. 打开单元禁写 (跨单元写不敏感业务可以不配置)
2. 检查业务延时
3. 切换预案
4. 解除单元禁写

PS：更新路由表时，也需要上述操作，只是第3步的切换预案变成切换新版本路由表；单元禁写主要是了等待数据同步，避免数据不一致导致的业务问题。

核心指标

单元计算耗时1~2ms

跨单元路由比例底于5%

除了性能外，因就近接入的诉求，跨单元路由比例也是我们比较关心的重要指标。从线上观察看，路由表策略单元计算基本上在1、2ms内完成，跨单元路由比例3%左右，整体底于5%。

四、后续优化

统一接入集成单元化能力

目前大部分服务都接入了统一接入网关服务，在网关集成单元化能力将大大减少服务单元化部署的成本，通过简单的配置就可以实现单元路由，服务可将更多的精力放在业务的单元封闭和数据同步上。

分组机制的优化

按大区分组存在三个问题：

通过IP计算大区有一定的误算率，会导致部分用户划分错误分组。

分组粒度太大，单元切流时流量不好分配。举例，假如华东是我们用户集中的大区，切流时把这个分组切到任意一个指定单元，都会造成单元服务压力过大。

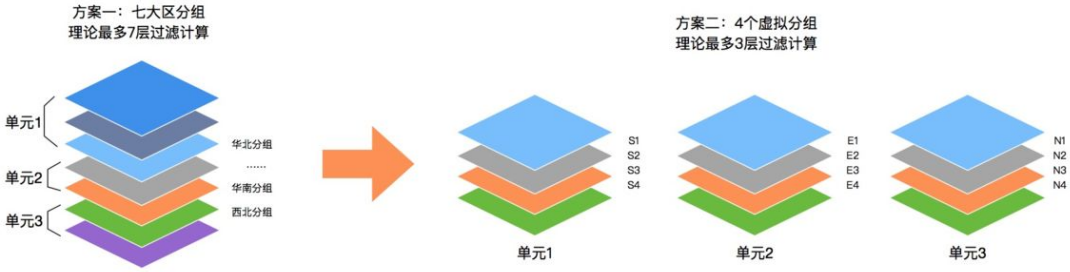
计算次数多，分多少个大区，理论最大计算次数是有多少次，最后采取取模策略。

针对上述几个问题我们计划对分组机制做如下改进

通过用户进入单元的记录来确认用户所属单元，而非根据用户IP所在大区来判断，解上述问题1。

每个单元划分4个虚拟分组，支持更细粒度单元切流，解上述问题2。

用户确实单元后，通过取模来划分到不同的虚拟分组。每个单元只要一次计算就能完成，新用户只需经过3次计算，解上述问题3。



热更时的双表计算

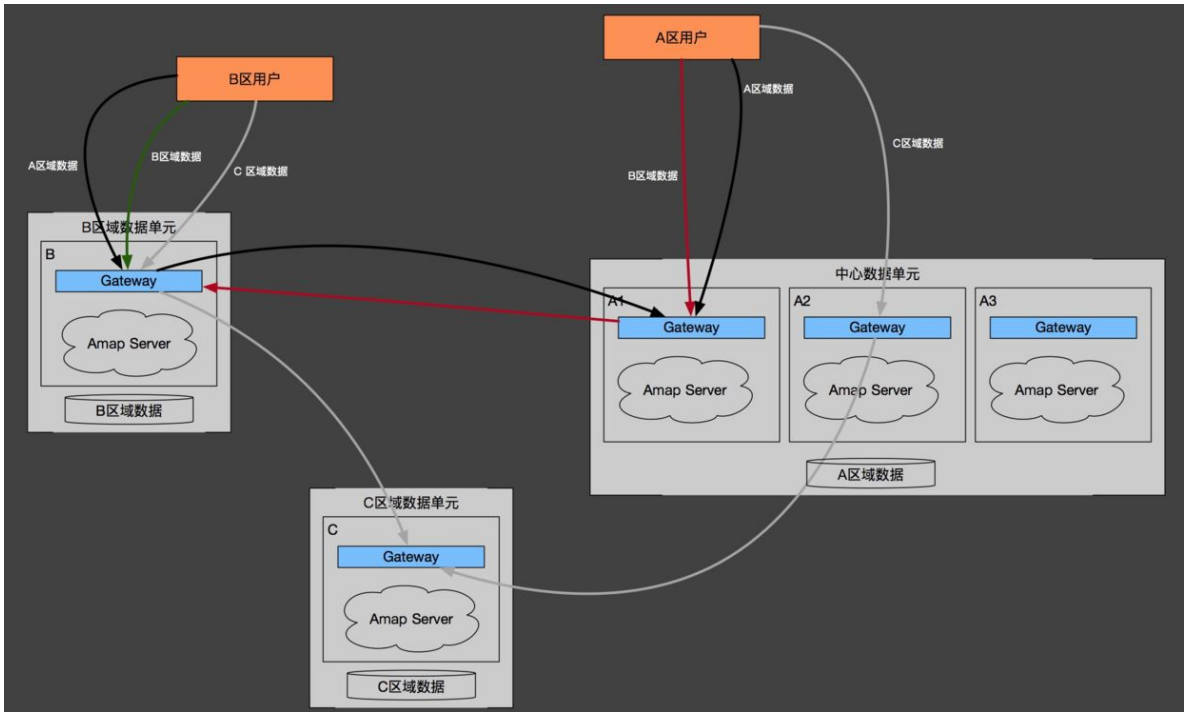
与取模路由策略不同，路由表策略为了把跨单元路由控制在一个较好的水平需要定期更新，目前更新时需要一个短暂的单元禁写，这对于很多业务来说是不太能接受的。

为优化这个问题，系统将在路由表更新时做双(路由)表计算，即将新老路由表同时加载进内存，更新时不再对业务做完全的禁写，我们会分别计算当前用户(或设备)在新老路由表的单元结果，如果单元一致，则说明路由表的更新没有导致该用户(或设备)变更单元，所以请求会被放行，相反如果计算结果是不同单元，说明发生了单元变更，该请求会被拦截，直至到达新路由表的一个完全起用时间。

优化前服务会完全禁写比如10秒(时间取决于数据同步时间)，优化后会变成触发禁写的是这10秒内路由发生变更的用户，这将大大减少对业务的影响。

服务端数据驱动的单元化场景

前面提到高德在路由策略上结合业务的特别设计，但整体单元划分还是以用户(或设备)为维度来进行的，但高德业务还有一个大的场景是我们未来要面对和解决的，就是以数据维度驱动的单元设计，基于终端的服务路由会变成基于数据域的服务路由。



高德很多服务是以服务数据为核心的，像地图数据等它并非由用户直接产生。业务的发展数据存储也将不断增加，包括5G和自动驾驶，对应数据的爆发式增长单点全量存储并不实现，以服务端数据驱动的服务单元化设计，是我们接下来要考虑的重要应用场景。

写在最后

不同的业务场景对单元化会有不同的诉求，我们提供不同的策略和能力供业务进行选择，对于多数据服务我们建议使用业务取模路由，简单且易于维护；对于RT敏感的服务使用路由表的策略来尽可能的降低服务响应时长的影响。另外，要注意的是强依赖性的服务要采用相同的路由策略。