李忠泽

求职岗位：Golang后端工程师

年　　龄：25岁　　性　　别：男

工作年限：3年经验　　电　　话：18941174662

邮　　箱：zonzeeli0523@gmail.com　　博客网站：https://zonzeeli.github.io

教育背景

2016-09 ~ 2020-06　　**大连理工大学**　　**电子信息工程（本科）**

工作经历

2020-08 ~ 至今　　**珠海格力电器股份有限公司**　　**后端开发工程师**

个人工作：

- 负责物理网设备及格力+软件语音模块的后端业务功能开发；
- 负责语音后端的框架设计、业务设计、中间件模块开发；
- 负责语音功能模块与外部合作部分的接口对接。

工作成果：

- 从 0 到 1 负责了任务管理调度的框架设计，为所内及部门内的任务需求提供支持；
- 通过对文本信息进行了预处理，解决了自定义名称导致的语义解析错误的问题；
- 对任务管理调度框架及预处理系统优化，提高了框架的 QPS 从 500 到 2000，预处理系统测试单次耗时从平均 800ms 降低到 100ms 以内，减少响应时间。

实习经验

2019-06 ~ 2019-08　　**东软信息集团（大连）**　　**嵌入式软件工程师**

- 工作总结：完成智能机器车的嵌入式软件开发；
- 学习成果：学习如何用QT语言设计用户界面、使用蓝牙/ZigBee进行模块数据无线透传。

项目经验

2022-08 ~ 2023-02　　**任务管理异步调度框架**　　**主要负责人**

项目内容：对任务进行自动调度、可续做与重试的异步处理框架，可灵活配置任务信息。比如用在管理离线语音的合成任务、设备控制的预设任务、语音入口输入文本数据分析认为的调度执行。

技术栈：gin、mysql、redis、consul、zap等。

个人职责：

- 负责框架的**架构设计**，分成任务管理调度模块、任务治理监控模块和任务执行处理模块；
- 进行性能调优，增大 MySQL 的连接数，**提高了 QPS 从 500 到 2000**；
- 进行架构优化，采用 **Redis 分布式锁**替换 MySQL 本身的行级锁，提高并发处理能力。

技术难点：

- 分布式锁选型：使用 MySQL 实现分布式锁会在多机竞争时，CPU 飙高预警，改用 Redis 实现分布式锁，提高并发性能；
- 分布式锁设计：使用分布式锁避免并发竞争，获取锁的worker节点执行任务，并对锁进行超时释放、续约的设计；
- 任务的执行状态处理设计：对任务的优先级、重试、续做进行设计，并对任务状态进行监控，超时判定失败交由人工手动处理；
- 分表设计：针对数据分析等大数据量场景（每月数据量平均在 400w 以上，高峰值会达到 800 w），采用**分表设计**，按照大小分表，将冷热任务分开。

2021-01 ~ 2021-06　　**自定义名称预处理**　　**主要负责人**

项目内容：对设备名称自定义后的输入话术文本进行替换或删除处理。比如设备“空调”自定义为“小白”，“打开小白”处理为“打开空调”，房间“房间1”自定义为“xxx家”，“打开xxx家的空调”处理为“打开空调”，并携带“xxx家”的信息。

技术栈：grpc、consul、opentelemetry、jarger等。

个人职责：

- 负责文本预处理的方案设计、模块设计、逻辑设计，并针对设备名、房间名进行不同方案的处理；

- 进行性能调优，采用并发处理和优先级的设计，减少响应时间，内部逻辑使用高效数据结构，并用动态规划优化匹配字符串的逻辑，减少时间复杂度，**单次响应从 800ms 降低到 100ms 以内**；
- 增加拼音处理，解决了语音识别错字导致文本匹配失败的问题。

技术难点：

- 减少响应时间方案设计：需要对设备名和房间名及其拼音均进行匹配处理，并涉及到优先级，串行方案会增大耗时，改用并发处理，并用管道信号来通知；
- 减少时间复杂度：字符串匹配及字符串拼音的匹配，采用**动态规划**算法替换暴力循环，并记录替换的索引位置和映射关系，**降低时间复杂度**，从O(n^2)降低到O(m*n)。

2022-04 ~ 2022-07　　语音合成api接口　　主要负责人

项目内容：对第三方语音合成api接口统一接入，整合协议将结果返回。

技术栈：grpc、consul、websocket、zap、viper等。

个人职责：

- 负责对接第三方的语音合成接口并进行统一管理设计，针对不同物联设备可使用不同的第三方的 api；
- 优化传输方案，采用**流式传输**设计，提高音频播放的实时效果；
- **提高服务可用性**，保证非核心模块崩溃不影响主要功能的正常运行。

技术难点：

- 传输方案选型：音频文件数据由 http 单次调用，太大的文件可能响应时间过长，导致播放实时性不好，改用流式传输提高音频播放的效果，改善用户体验；
- 高可用性设计：为保存合成的音频，获取音频存储服务器配置信息和存储音频的功能等非核心功能崩溃采用**熔断式设计**，将异常抛出，并捕获记录，停止调用，不影响合成音频和传输的功能，保证服务的可用性，待异常问题解决后，进行重试恢复；
- 获取参数方案设计：一次合成要上传一份信息，更改原一次合成查询一次存储服务配置为定时拉取，使用心跳机制时间，根据配置信息失效时间设置定时，减少连接数，提高其他服务的并发量。

2021-08 ~ 2022-02　　多轮对话流程调度架构　　主要负责人

项目内容：对语音入口的话术的控制指令进行调度的架构设计，并对多轮语音对话流程进行管理，针对设备控制模块进行方案设计。比如“打开空调”，家庭下有多个“空调”，返回设备信息供用户选择控制，并进行第二轮控制。

技术栈：grpc、gin、redis、consul、opentelemetry、jarger等。

个人职责：

- 负责对话流程的**架构设计**、具有多轮对话流程的进行**方案设计**以及设备功能控制的**模块设计**，解决多设备选择、多参数功能的问题；
- 优化架构设计，**对架构进行拆分解耦，降低模块复杂度**，架构内部采用微服务 gRPC 调用，**减少一次链路调用流程的耗时时间从 2s 级别到 1s 以内**；
- 改用并发请求语义解析，取优先级结果，避免预处理服务误处理导致解析结果有误。

技术难点：

- 文本处理方案设计：一次流程为文本预处理->语义解析-参数判断-设备功能控制，存在误处理问题，改用并发处理及优先级设计，如果原文本处理结果正常则使用原文本结果，否则采用预处理方案或者兜底返回；
- 链路调用设计：对各模块重新设计，将原本的 http 链路调用改成同步 RPC 调用，并通过监控查看调用耗时，协助优化各个模块之间的代码逻辑、调用方案，并采用连接复用，将整体的耗时从 2s 优化到 1s 以内；
- 多轮协议设计：语义解析服务是无状态服务，不保存上轮信息，选用session来判断一套会话流程是否结束，由最后一条链路的服务返回结果更改session，不结束则获取信息缓存拼接，后续计划优化语义解析服务架构；
- 缓存方案设计：原方案由调度模块来控制缓存存储、删除，后改用调度模块控制存储，设备功能控制模块控制删除，减少调度模块本身的复杂设计，后续计划语义解析模块采用数组数据结构对缓存进行控制，过期的会话缓存自动删除，每次语义解析查询判断上轮的缓存信息比对，将完整的语义信息传入下一个服务。

个人项目

秒杀系统：实现简易秒杀系统的后端处理。

- 用mysql对商品信息进行存储，可以查询、添加，并用etcd进行同步，从etcd中加载商品信息，监听数据变化；
- 对库存扣减方案进行优化，不使用etcd，使用redis进行预扣方案，防止超卖，并轮询mysql将失效订单加载回redis，防止少卖；
- 用redis做队列，对创建订单的请求通过redis通知到逻辑处理模块；
- 安全校验，对用户信息进行签名校验，做白名单和黑名单限制；

- 流量限制，对入口及用户ip进行请求频率限制。
- 涉及到的技术栈：gin、mysql、etcd、redis等。
- https://github.com/ZonzeeLi/SecKill
- https://github.com/ZonzeeLi/SpikeSystem

短链接：实现短链的转链、存链、跳转等功能。

- 使用mysql存储短链和长链的映射关系，并加入缓存设计优化查询；
- 使用布隆过滤器防止缓存穿透，并使用singleflight提升高并发性能；
- 基于MySQL主键自增，采用replace替代insert，保证数据量只有一条，实现高可用的发号器，转换成62进制的序列，保证短链尽可能短；
- 考虑分布式部署，对 MySQL 进行主从方案设计。
- 涉及到的技术栈：go-zero、mysql、redis等。
- https://github.com/ZonzeeLi/shortener

个人技能

编程语言：

- Golang ， 使用两年半以上 ， 熟悉 map、slice、channel等底层实现 ， GMP 模型、垃圾回收等机制；

后端组件：

- MySQL ， 熟悉事务、 日志、锁等机制 ， 读写分离、分库分表等分布式场景应用，有 SQL 调优经验；
- Redis ， 熟悉数据结构、线程模型、持久化等机制 ， 高性能、高可用等部署方案，有缓存设计、分布式锁实现经验；
- 消息队列、链路追踪等中间件的使用，熟悉基础知识和实际场景问题的解决方案。

其他：

- 了解 Docker、 Kubernetes 的基础命令和使用；
- 熟悉计算机网络、操作系统、数据结构等计算机基础知识；

证书：大学英语6级证书。

自我评价

1. 工作积极认真 ， 细心负责 ， 会针对架构、业务考虑优化方案 ， 并与团队良好的配合 ， 协调解决项目中的困难点。比如对负责的任务管理调度框架进行性能优化、多轮对话流程进行架构拆分优化、 自定义名称预处理系统服务考虑减少耗时。
2. 热爱学习 ， 经常主动组织培训 ， 与团队成员分享学习知识与新工具 ， 并进行代码评审 ， 帮助团队成员有良好的代码规范。
3. 个人心态乐观 ， 积极向上 ， 期望能增长视野 ， 开阔眼界 ， 会对困难的问题进行钻研。比如对任务调度管理框架的分布式锁 选型设计、 自定义名称预处理服务的并发优先级策略 ， 喜欢阅读技术文章 ， 关注实时热点新闻 ， 热爱旅游健身。