技术研究与提升之道

> 写在前面：
>
> 日常的需求分析、系统设计、代码编写、运维管理等工作容易形成“技术陷阱”，即止步于相对稳定的手头工作，什么需求都能做，做出来样子都差不多，此时在技术上需要有一些 “刺激” 才能进一步提升，跳出思维定势，激发思考和总结，从而不断升级技术品味，让每次做出的工作都不一样(更好)。

1. 初识达成

技术成长和提升本质上是提升自身的认知纬度，可从如下三个维度来切入：

- 输入阶段：掌握某个概念、知识点，了解它是什么及其背后的原理，需伴随一段时间的强记和实践

> 例：Rust ownership/borrow 和 C++11 的 move ，理解其语法合理应用它们写代码并解决问题

- 输出阶段：发现多个知识点或技术点的内在关联，理解各自的应用背景和本质区别，能讲解清楚

> 例：栈变量自动分配与回收，内存拷贝，堆变量动态分配与回收，变量与值的关系

- 融汇贯通阶段：自身形成一个知识体系的循环引擎，能吸引任意知识点，完成快速输入和输出

> 例：操作系统内存管理，段式页式各自优缺点，二进制ELF文件格式与排布，代码段、全局数据段、栈、大页、堆，内存分配器，垃圾回收（引用计数/三色标记），手动new/free，编译器自动等

2. 输入之道

来源	工程界	学术界
宏观层面	经典成熟的软件系统的架构设计和思路 经典工程实践产品的设计和思想	业界前沿的研究热点、研究方向 把握和洞察未来中长期技术演进方向
微观层面	通用代码能力：从变量命名技巧、函数设计、参数传递、接口定义、类交互、设计模式合理应用、数据结构设计等 代码架构：代码的精巧性、稳定性、高性能、可扩展性、可维护性、可观测性、可测试性等	

宏观层面切入点：

- 结合技术热点、工程实践痛点、开发经验等角度，挖掘和总结富有价值的技术研究 topic
- 为 topic 构建知识图谱，组织全员技术交流会，打平对应 topic 基础知识壁垒

- 针对每个 topic 从多个纬度进行拆分，拆解不同难易程度的入手点，细化工作
- 将抽象的 topic 问题 具象化 为一件件可入手的任务，便于各成员分工合作
- 定期组织技术研讨交流会，讨论和思考各自学习探索的技术点并讨论应用价值
- 避免落入单纯理论和学术层面研究，面向具备实践价值的 topic，以务实的精神争取能产生实际价值

微观层面切入点：

发现和理解复杂问题的本质

- 业务开发过程中遇到的某个复杂问题，与现实世界中众多复杂问题本质是一样的
- 选择某一个通用的复杂问题，寻找和调研该问题的产生背景、复杂性所在，理解问题的本质

学习和思考优秀的解决方案

- 面临复杂问题时所需的思维是可复用的，对个人来说是不断积累和思考，形成技术素养的过程
- 选择典型的复杂问题及其优秀的解决方案，学习和思考背后的精巧设计和解决思路，形成提升

严格遵循技术标准并构建优秀代码片段集

- 结合LBA 基本功规范项目，落地严格的技术标准和规范，需进行强制推行
- 建立优秀经典代码实现集，介绍和说明其中一点或多点存在值得学习的地方（公司有一个迷惑代码大赏群，大家分享的是特别不好不可取的代码片段）

经典工程系统

- LevelDB：https://github.com/google/leveldb
- Memcache：https://memcached.org/，https://github.com/memcached/memcached
- Redis：https://redis.io/，https://github.com/antirez/redis
- MySQL：https://www.mysql.com/
- Etcd：https://etcd.io/，https://github.com/etcd-io/etcd
- ZooKeeper：https://zookeeper.apache.org/，https://github.com/apache/zookeeper
- TiKV / TiDB：https://pingcap.com/，https://github.com/pingcap/tidb
- InfluxDB：https://github.com/influxdata/influxdb
- Ceph：https://ceph.readthedocs.io/en/latest/start/intro/
- Doker：https://www.docker.com/
- Kubernetes：https://kubernetes.io/
- OpenStack：https://www.openstack.org/
  - Nova；Neturon
  - Swift； Cinder
- PyTorch：https://pytorch.org/，https://github.com/pytorch/pytorch
- Tensorflow：https://www.tensorflow.org/，https://github.com/tensorflow

> 更多内容可扩展～

优秀论文源

- USENIX：https://www.usenix.org/ （优先推荐）
  - 1975年成立的Unix用户群，后发展成一个紧密联合工业界的学术组织
  - 2008年起，由USENIX发起的会议全程提供论文、演讲幻灯片和演讲视频的免费下载
  - 工业界会把很多经验总结成论文发布在这里，如Google的MapReduce、BigTable等
  - 主要以一些重磅顶级会议为载体
    - USENIX Symposium on Operating Systems Design and Implementation (OSDI)
    - USENIX Symposium on Networked Systems Design and Implementation (NSDI)
    - USENIX Conference on File and Storage Technologies (FAST)
    - USENIX Annual Technical Conference
- 算法和理论
  - ESA – European Symposium on Algorithms
  - WADS – Algorithms and Data Structures Symposium
  - STOC – ACM Symposium on Theory of Computing
- 分布式和并行计算
  - ICDCS - IEEE International Conference on Distributed Computing Systems
  - ICPADS - IEEE International Conference on Parallel and Distributed Systems
  - IPDPS - IEEE International Parallel and Distributed Processing Symposium
  - FOCS – IEEE Symposium on Foundations of Computer Science
  - DISC - International Symposium on Distributed Computing
  - PPoPP - ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming
  - SPAA - ACM Symposium on Parallelism in Algorithms and Architectures
  - SOSP - Symposium on Operating Systems Principles
- 分布式与数据存储
  - ICDT - International Conference on Database Theory
  - PODS - ACM Symposium on Principles of Database Systems
  - SIGMOD - ACM SIGMOD Conference
  - VLDB - International Conference on Very Large Data Bases
  - SYSTOR - ACM International Systems and Storage Conference

论文获取方法：选择一个上述会议期刊名直接Google搜索，可找到该会议对应年份的首页，下载方式：

- Google学术搜索
- Sci-hub

3. 输出之道

- 交流讲解与分享
  - 一个技术 Topic 进行多维度拆分后，需明确分工各个部分的 owner，用排期去督促自己学习
  - owner 完成输入阶段的基本过程，并形成自己的理解，能在交流讲解过程中给他人讲清楚
- 思考提问
  - 经常陷入“还有没有其他什么问题” 却无人问津的境地
  - 不提问题就不知道是否理解，同时也容易忘记，需制定强制机制，必须每人至少提一个问题
- 关联知识发散讨论
  - 相似的知识点是什么？
  - 本质是什么？
  - 在知识体系中的位置是哪里？
  - 带来有什么启示？

思考：三色标记清理垃圾回收 <=> Redis rehash 操作，设计思路的相似性