Enhance stblty 提升稳定性2025总结

单个节点情况下 提升可靠星

Catalog

[1. 本地环境与svr环境的一致性 2](#_Toc27173)

[ 服务端环境尽可能与本地环境尽可能一致 方便排查问题 2](#_Toc15496)

[ 本地测试环境要简单快捷 2](#_Toc13208)

[2. 降低网络组件依赖 2](#_Toc16820)

[ 尽可能优先使用本地组件 2](#_Toc17324)

[ 本地存储来代替网络存储 sqlt 2](#_Toc26189)

[ 文件替代存储 可靠性更高 ini 2](#_Toc15484)

[ 必须使用网络组建的，分离职责简单化，web+cli 3](#_Toc6316)

[3. 释放内存gc 解决gc不及时导致的问题 3](#_Toc23833)

[ 远程api与本地分离 通过cli连接调用，即时释放内存 3](#_Toc25929)

[ 内存持久化到文件，降低内存占用 3](#_Toc27393)

[4. Cpu问题导致的稳定性 4](#_Toc25329)

[5. 简单化 4](#_Toc29708)

[ 减少各种组件的使用，简单化架构 4](#_Toc22892)

[ 适当自研 大力减少对外依赖 4](#_Toc5351)

[ 减少调用链层次 4](#_Toc16020)

[6. 2. 错误处理和异常管理 4](#_Toc4193)

[ 安全api 5](#_Toc17648)

[7. 设计可靠性 5](#_Toc9740)

[ 尽可能自动化 5](#_Toc20556)

[ 自动化索引 字段添加 5](#_Toc19205)

[ 自动化表格 数据库添加 5](#_Toc2018)

[8. Other 5](#_Toc15840)

[. 自动化与监控 5](#_Toc15128)

[5. 备份与恢复 5](#_Toc30472)

[6. 版本控制与部署管理 5](#_Toc13534)

[7. 冗余设计 6](#_Toc16665)

[8. 用户反馈与改进 6](#_Toc15142)

[9. 文档与培训 6](#_Toc24820)

# 本地环境与svr环境的一致性

## 服务端环境尽可能与本地环境尽可能一致 方便排查问题

## 本地测试环境要简单快捷

# 降低网络组件依赖

## 尽可能优先使用本地组件

## 本地存储来代替网络存储 sqlt

Sqlite这类代替 （mysql等

## 文件替代存储 可靠性更高 ini

尽可能使用文件存储，简单方便快捷也更可靠性稳定

## 必须使用网络组建的，分离职责简单化，web+cli

后面通过cli连接业务模块

# 释放内存gc 解决gc不及时导致的问题

## 远程api与本地分离 通过cli连接调用，即时释放内存

## 内存持久化到文件，降低内存占用

# Cpu问题导致的稳定性

# 简单化

## 减少各种组件的使用，简单化架构

## 适当自研 大力减少对外依赖

## 减少调用链层次

Fp代替oop aop

# 2. 错误处理和异常管理

* **健壮的错误处理**：捕获和处理所有可能的异常，避免系统崩溃。
* **日志记录**：记录详细的错误和警告日志，便于后期分析和排查问题。

## 安全api

# 设计可靠性

## 尽可能自动化

## 自动化索引 字段添加

## 自动化表格 数据库添加

# Other

**文件和网络连接管理**：确保及时关闭文件句柄和网络连接，避免资源枯竭。

### . ****自动化与监控****

* **健康检查**：实现定期的系统健康检查，及时发现和处理异常情况。
* **性能监控**：使用监控工具跟踪系统性能，发现潜在的瓶颈或故障。

### 5. ****备份与恢复****

* **数据备份**：定期备份重要数据，确保在系统故障时能够快速恢复。
* **故障恢复计划**：制定并测试故障恢复流程，确保在出现问题时能快速恢复服务。

### 6. ****版本控制与部署管理****

* **使用版本控制**：使用 Git 等版本控制工具管理代码，确保可追溯性。
* **持续集成/持续部署（CI/CD）**：自动化构建和测试流程，减少人为错误。

### 7. ****冗余设计****

* **服务冗余**：在可能的情况下，使用冗余服务或组件，以避免单点故障。
* **分布式架构**：虽然是在单机环境下，考虑将系统拆分为模块，增强灵活性。

### 8. ****用户反馈与改进****

* **用户反馈机制**：建立用户反馈渠道，收集和处理用户遇到的问题。
* **迭代改进**：根据用户反馈和监控数据，不断迭代和优化系统。

### 9. ****文档与培训****

* **完善文档**：编写详细的系统文档，确保开发和运维人员能够理解和维护系统。
* **团队培训**：定期对团队进行培训，提高他们的技术水平和对系统的理解。

通过上述措施，可以有效提升单机软件系统的可靠性，降低故障率，提高用户满意度。如果您有特定的场景或问题，欢迎进一步讨论！