旅行技术架构

华小明（洪野）

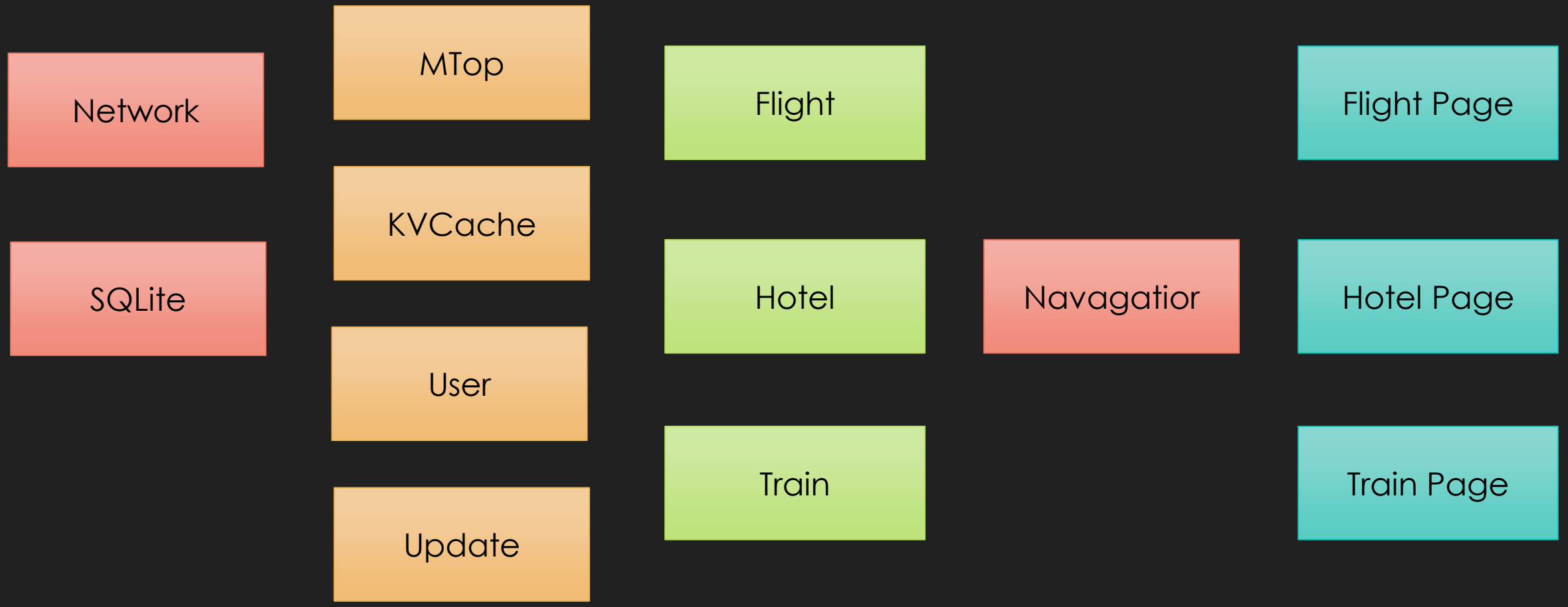
技术目标

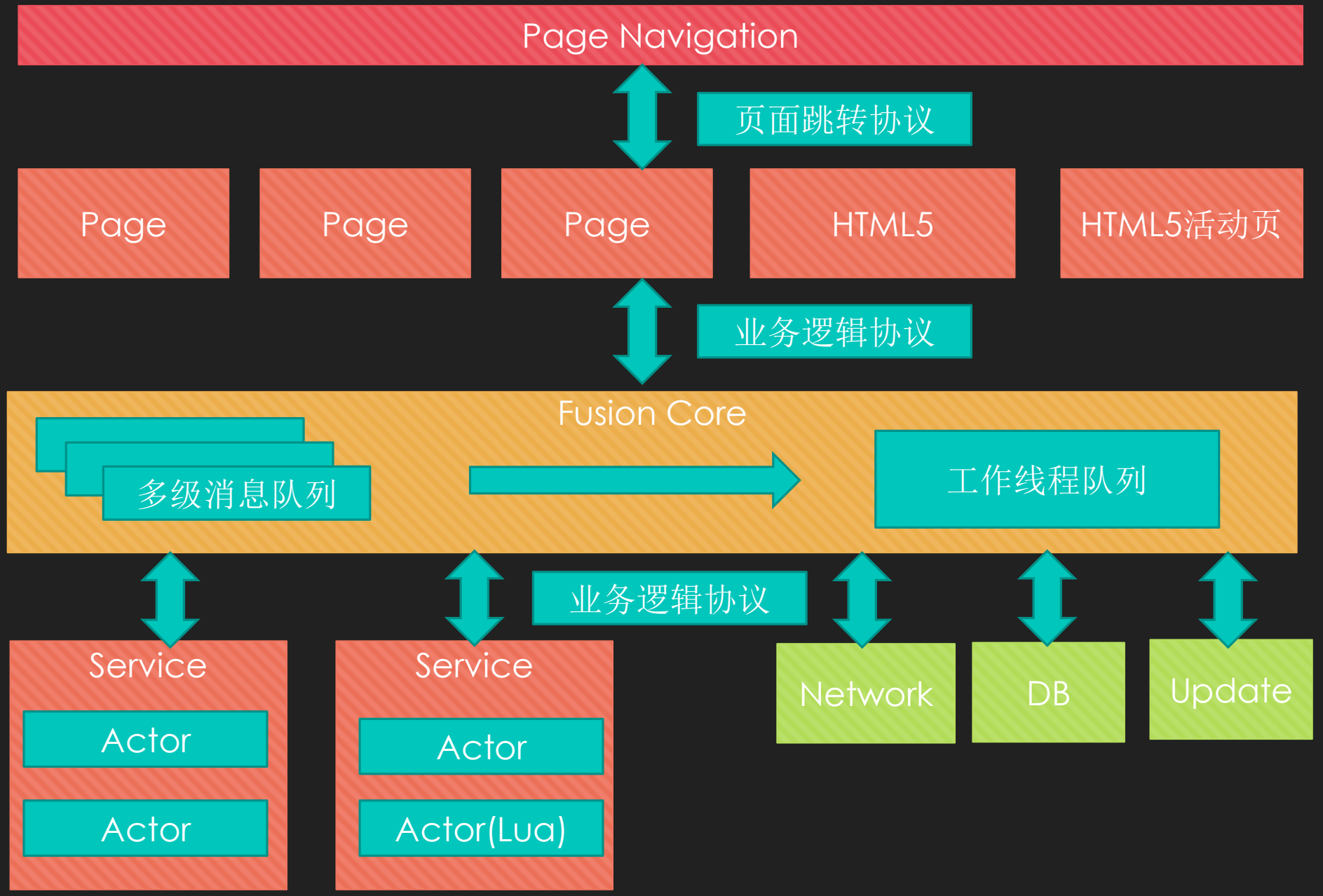
- 降低多业务线并行，单业务线多需求并行开发的风险
- 页面与页面，页面与逻辑解除耦合
- 随着迭代的深入，保证应用稳定性可控
- 页面展示以及业务逻辑的动态更新
- 应用程序crash的日志收集

技术特点

- 统一的协议包括（页面跳转，逻辑调用）
- 灵活的跳转机制，实现Native UI和H5的整合
- 业务逻辑完全模块化，功能细分
- 完全独立的网络层（一般请求与资源下载分离）
- 完善的资源cache管理和过期文件删除机制
- 动态更新机制，可以实现NativeUI的H5替换，数据更新，业务逻辑代码更新

旅行客户端总模块图


Network
SQLite
MTop
KVCache
User
Update
Flight
Hotel
Train
Navagatior
Flight Page
Hotel Page
Train Page


Page Navigation
页面跳转协议
Page
Page
Page
HTML5
HTML5活动页
业务逻辑协议
Fusion Core
多级消息队列
工作线程队列
业务逻辑协议
Service
Actor
Actor
Service
Actor
Actor(Lua)
Network
DB
Update

架构的基础 统一的协议

- 页面跳转协议 page://page_name/cmd?args
- 业务逻辑协议 native://service_name/actor_name?args

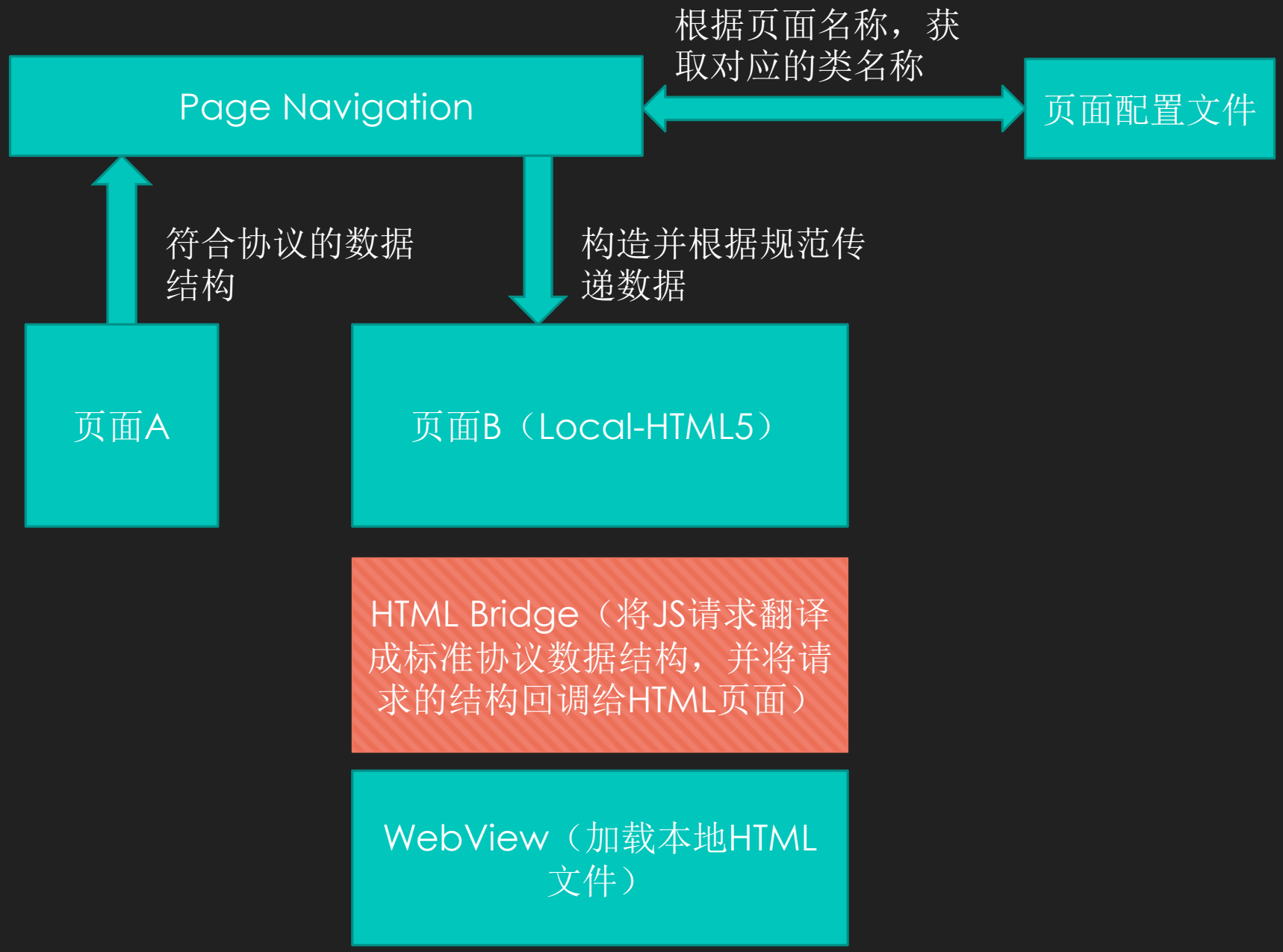
统一协议很好的解除了所有模块之间的依赖关系，3.0中业务模块之间完全用遵循协议的数据结构进行通讯，同时也为跨语言（后端业务逻辑Lua化，前端页面HTML5)提供了基础

灵活的页面跳转

- 灵活的页面跳转，对传统栈进行改造
- 页面之间通过遵循协议的数据结构进行跳转与参数传递
- 所有的页面可以独立测试，页面之间完全解耦
- 支持各种手势的页面跳转操作，大幅提高用户体验
- HTML桥接页面的实现，打通Native UI和HTML5，根据业务协议，HTML5可以随时调用任何客户端的业务逻辑
- 可更新的页面配置，可以在线上随时调整页面

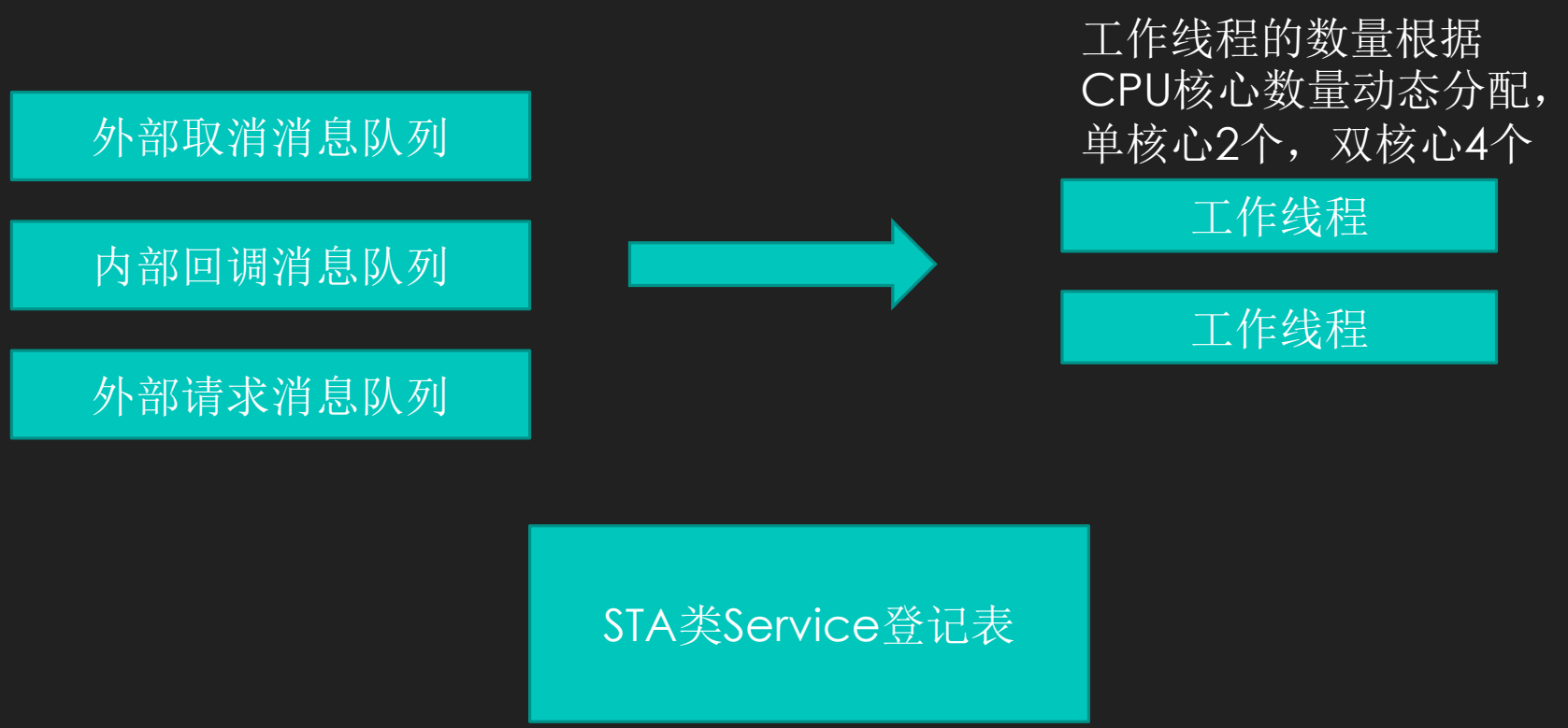
页面的几点注意

- 任何一个页面都是完全独立的
- 页面与页面完全依赖协议来传递数据，数据必须是可JSON化的
- 如果页面与页面需要共享大量数据，建议使用kvcache服务


根据页面名称，获取对应的类名称
Page Navigation
页面配置文件
符合协议的数据结构
构造并根据规范传递数据
页面A
页面B（Local-HTML5）
HTML Bridge（将JS请求翻译成标准协议数据结构，并将请求的结构回调给HTML页面）
WebView（加载本地HTML文件）

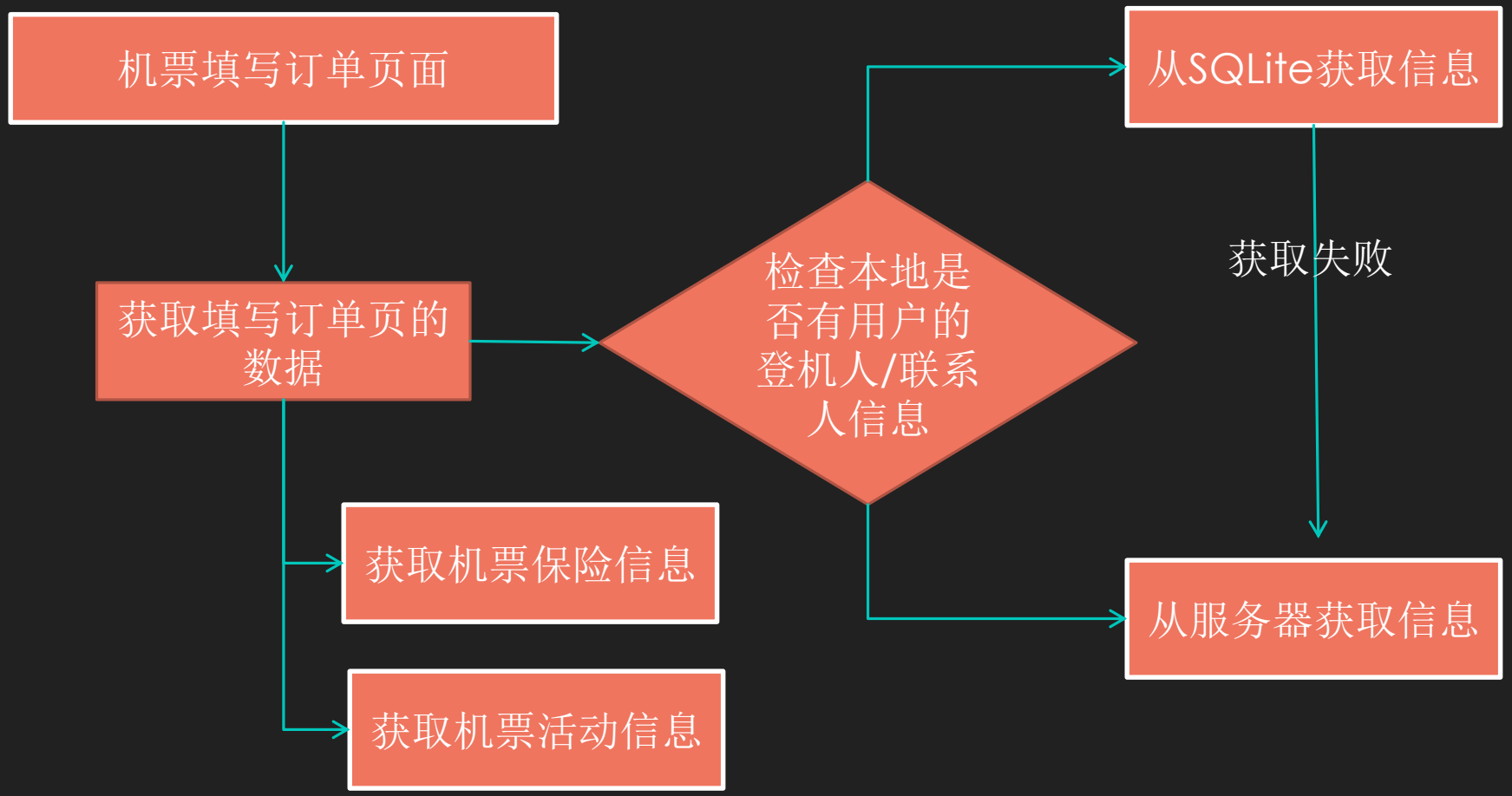
业务逻辑部分的特点

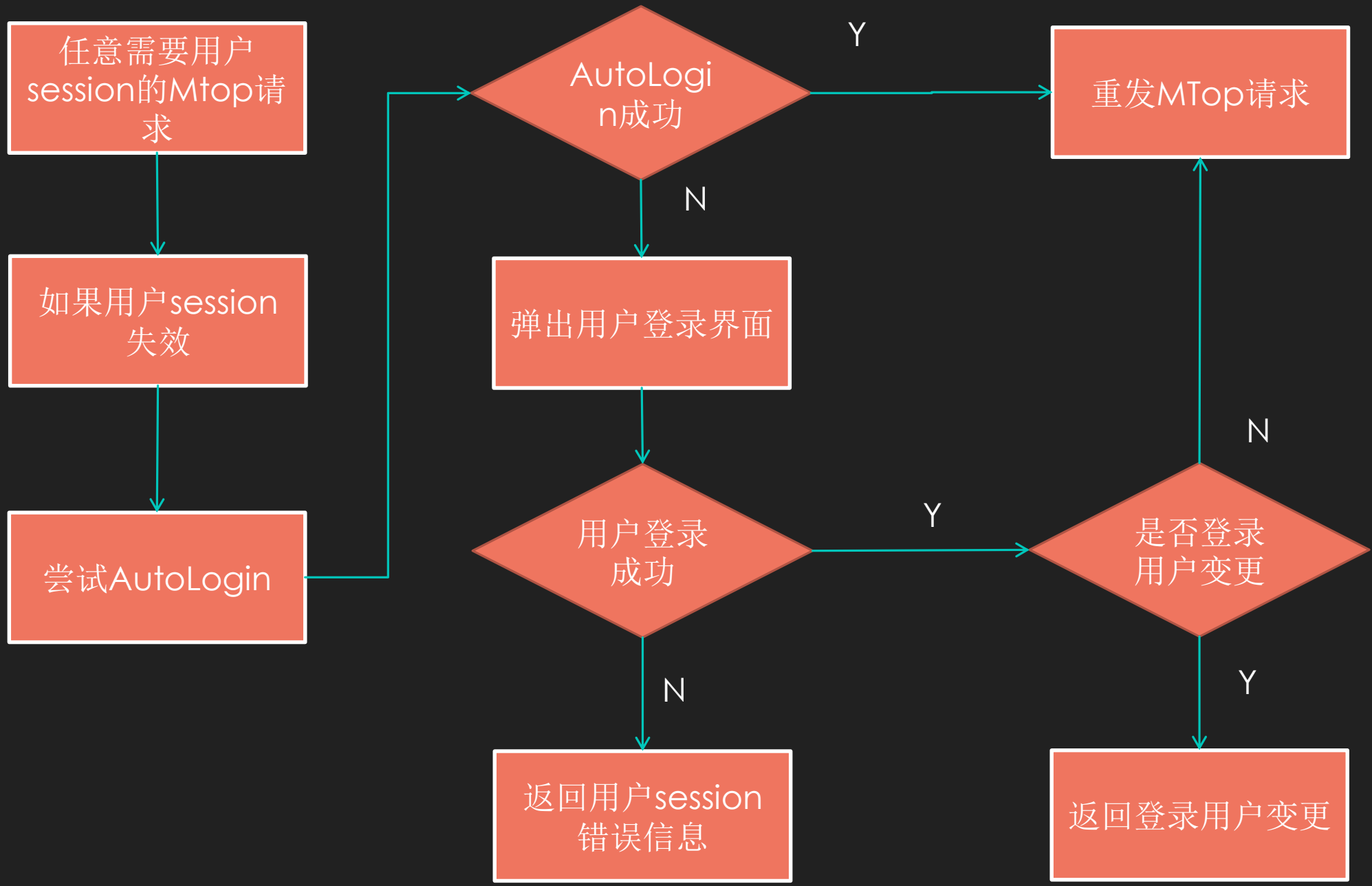
- 每个业务模块作为一个Service
- 每个最细化的功能作为一个Actor
- 所有的Actor之间只能使用消息通讯（遵循统一业务逻辑协议）
- Actor支持级联，一个复杂的功能可以由多个Actor组合
- 每个Actor都是一个状态机模型
- Service支持(MTA和STA双模式)
- 良好的并发性，最大化利用多核心CPU



调度器遵循FIFO的原则从3级消息队列获取消息，然后根据消息所属的Service的类型（MTA/STA）判断是否可以交给空闲的工作线程执行
MTA类Service可以同时被多个工作线程执行
STA类Service同一时间只能在一个工作线程执行，如果可以被执行则会被加入到登记表，执行完成后从登记表删除

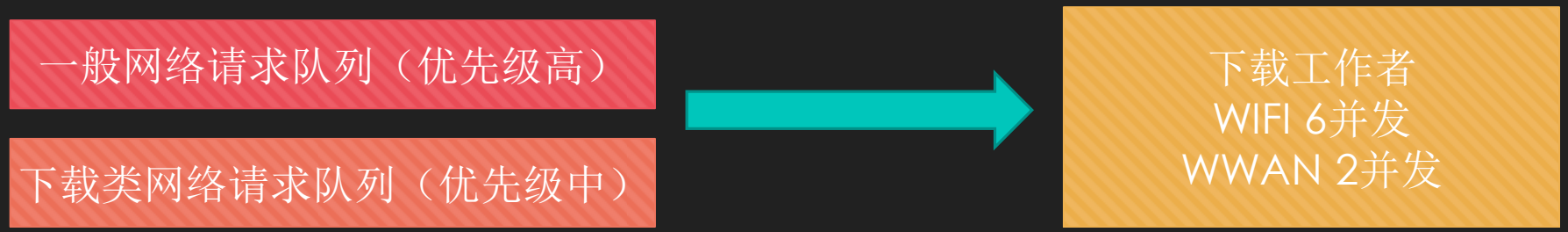
例子：机票下单页面获取准备数据




任意需要用户session的Mtop请求
如果用户session失效
尝试AutoLogin
AutoLogin成功
Y
N
弹出用户登录界面
用户登录成功
Y
N
返回用户session错误信息
是否登录用户变更
N
Y
重发MTop请求
返回登录用户变更

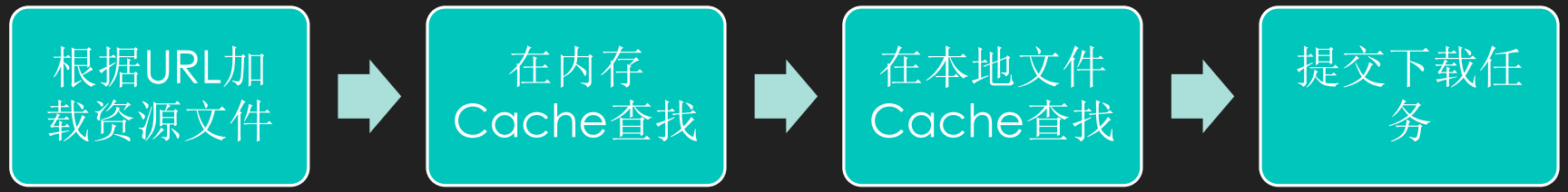
完全独立的网络层

- 对于请求进行分级处理，当前分为2级队列，高优先级针对一般的业务请求，中优先级对应所有的资源文件下载请求（图片等）
- 针对不同网络情况（WIFI/WWAN）来实现并发控制



完善的资源Cache体系

- 内存资源cache和文件cache双层体系
- 唯一的资源文件永远只加载一次，节省内存使用
- 每次加载资源文件都会进行timestamp打标
- 定时任务根据timestamp回收资源文件



更快速的UIImage加载

- iOS借鉴Path的FastImage项目，扩展实现了类似的效果
- 废除了系统的UIImage渲染方式，对于需要加速的Image，可以实现制作一次图片拉伸以及decode的工作，后续所有的操作都会直接使用基于bitmap的cache文件
- 未来准备为iOS扩展.9文件的支持

灵活的在线更新机制

- 定期检查
- 动态替换UI配置文件，以及下载Local HTML5页面（3.0只做支持，后续版本会逐步实施）
- 动态替换后台Service配置文件，以及Actor的Lua脚本（3.0制作支持，后续版本会逐步实施）
- 更新城市列表等数据文件