atitit.attilax的软件 架构 理念.docx

[1. Atitit 软件设计的模型 frm lib standard 框架类库标准 FLS模型](#_Toc17245)

[2. fusco](#_Toc25122)

[3. Cirxs融和开发流程](#_Toc12727)

[4. 45个架构与理念方法](#_Toc12926)

[5. 功能（function）方向](#_Toc18435)

[5.1. Mobile 移动化](#_Toc19602)

[5.2. 7个cross](#_Toc2428)

[5.3. Crossplatform跨平台，](#_Toc25900)

[5.4. cross device跨设备cross device，](#_Toc1096)

[5.5. cross screen跨不同屏幕](#_Toc5149)

[6. Scalable可扩展性 可扩展性scalability方向](#_Toc21012)

[6.1. cross lang跨语言 (uapi)](#_Toc28922)

[6.2. Cross db跨数据库](#_Toc26197)

[6.3. component组件化，](#_Toc15059)

[6.4. DNA化预先规划。](#_Toc24475)

[6.5. distributed分布式](#_Toc2733)

[6.6. Service 组件服务化](#_Toc31915)

[6.7. Microservice微服务](#_Toc20162)

[6.8. Multi Lang多语言化](#_Toc6968)

[6.9. H5，ui h5化](#_Toc31958)

[6.10. Hybrid化,](#_Toc5480)

[6.11. Ioc容器化](#_Toc7204)

[6.12. integrate集成化](#_Toc15501)

[7. 易用性（usability）方向](#_Toc2983)

[7.1. cross archi跨架构 (bs cs hybrid)](#_Toc25285)

[7.2. Meta define元定义](#_Toc20438)

[7.3. MultiFun多功能](#_Toc12882)

[7.4. Dsl](#_Toc19206)

[7.5. Deploy friendly部署友好的](#_Toc18660)

[8. 性能kst/udr模型](#_Toc7692)

[9. other](#_Toc32316)

[9.1. Cba ，Cap+base+acid](#_Toc6880)

[9.2. 兼容性compatibility方向](#_Toc14537)

[9.3. secury](#_Toc16708)

[9.4. solid](#_Toc2084)

[9.5. Socail 社区](#_Toc15808)

[9.6. Sd (Software Defined概念。软件定义”(Software Defined](#_Toc15526)

[9.7. All in one全体系化](#_Toc25976)

[9.8. Func base unit 基于功能的单元](#_Toc11365)

[9.9. FURPS模型](#_Toc15776)

[9.10. Green 绿色](#_Toc6491)

[9.11. I18n](#_Toc21400)

[9.12. Open sys开放体系](#_Toc2438)

[9.13. Plugin 插件化](#_Toc4863)

[9.14. platform平台化](#_Toc10557)

[9.15. Process customiz流程自定义](#_Toc28983)

[9.16. Rest 接口rest化](#_Toc20761)

[9.17. Text 协议文本化](#_Toc16947)

[9.18. Third Api](#_Toc21526)

[9.19. Tmork 输入方式](#_Toc26757)

[9.20. Vm化，隔离化 再次封装化](#_Toc8162)

|  |  |
| --- | --- |
|  | [软件即服务理念的提](https://www.baidu.com/s?wd=%E8%BD%AF%E4%BB%B6%E5%8D%B3%E6%9C%8D%E5%8A%A1%E7%90%86%E5%BF%B5%E7%9A%84%E6%8F%90%E5%87%BA&rsf=9&rsp=8&f=1&oq=%E7%90%86%E5%BF%B5 %E8%BD%AF%E4%BB%B6&ie=utf-8&rsv_pq=f10005b100008f39&rsv_t=34fdknBQUNc9d+VgyxCK6ccBF5vko0UEib2FXQfkzX8EYPEH0qPGGQYEhsM&rqlang=cn&rs_src=0&rsv_pq=f10005b100008f39&rsv_t=34fdknBQUNc9d+VgyxCK6ccBF5vko0UEib2FXQfkzX8EYPEH0qPGGQYEhsM) |

# Atitit 软件设计的模型 frm lib standard 框架类库标准 FLS模型

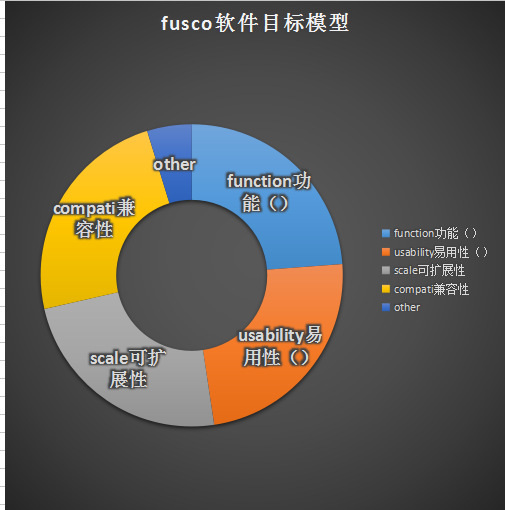
未来对于软件，只有3个终极目标——1.做一个框架，怎么用和你一点关系都没有，就像安卓系统，我只提供树干，爱种成什么样的树，结成什么样的果，我只要有能力在这个框架里完成就可以了，不要成熟系统，不要你管客户的业务。2.做一套标准，既然任何人都不属于一个实际组织了，那怎么来确保秩序和质量呢，特别是服务型的组织，比如物业公司。那就是确保进入你的框架的人，是达到你的起始要求的，可以的技能认证，也可以是公司自己的

# fusco

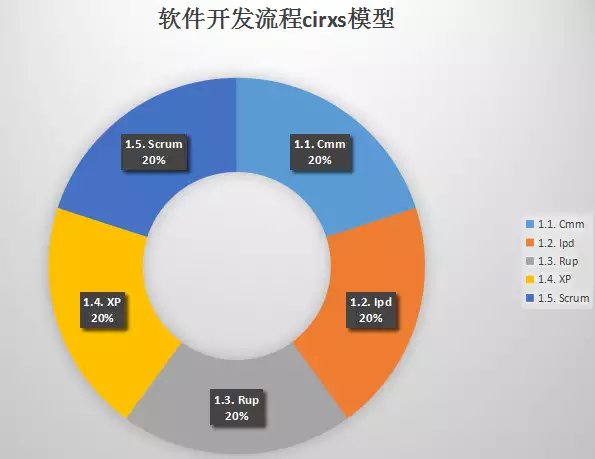
是功能（function）、易用性（usability）、及可扩展性

兼容性 other

Fusco如下图所示



# Cirxs融和开发流程



如上图，融和了cmm ,ipd,rup,xp ,scrum等流程方法

# 45个架构与理念方法

# 功能（function）方向

## Mobile 移动化

## 7个cross

## Crossplatform跨平台，

## cross device跨设备cross device，

## cross screen跨不同屏幕

# Scalable可扩展性 可扩展性[scalability](http://www.baidu.com/link?url=KG68c6-E92kBRBkuSz4wSee71MVBljgoRvvdWhbs3V1yYkp-y1vN3anaQlQYQNrR" \t "https://www.baidu.com/_blank)方向

## cross lang跨语言 (uapi)

## Cross db跨数据库

## component组件化，

二不是模块，粒度更好把控

## DNA化预先规划。

不用功能可隐藏，但要预留

## distributed分布式

## Service 组件**服务化**

## Microservice微服务

配合以前的soa,更好的方法..

## Multi Lang多语言化

## H5，ui h5化

## Hybrid化,

## Ioc容器化

## integrate集成化

通过集成不个

# 易用性（usability）方向

## cross archi跨架构 (bs cs hybrid)

## Meta define元定义

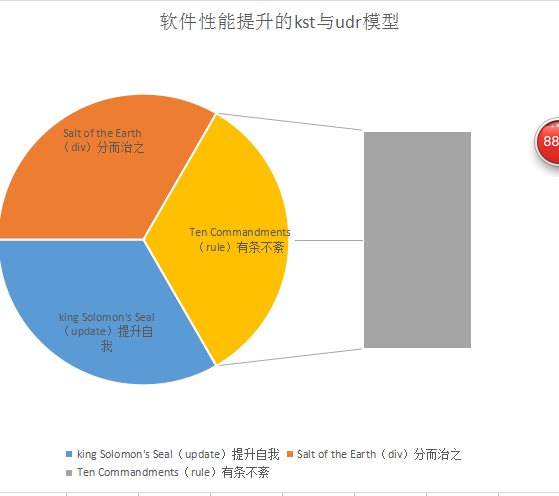
软件界的sd

## MultiFun多功能

## Dsl

## Deploy friendly部署友好的

# 性能kst/udr模型



# other

## Cba ，Cap+base+acid

## 兼容性compatibility方向

6s

## secury

## solid

## Socail 社区

## Sd (Software Defined概念。软件定义”(Software Defined

Sdh sd hardware ,sds sd sys,sdw sd world

Sdn sd network

Sde enterprice 软件定义存储

软件定义世界，软件定义一切！ 说我们为什么需要软件定义？软件定义的强大之处究竟在哪里？

**软件的价值毋庸置疑，如果把硬件比喻成身体，那么软件就是大脑，就是思想，就是灵魂！ 身体可以到达的领域是有限的，但是思想的延伸是无限的**

因为硬件的能力本身有限, 无限的需求。所以需要sd。。

对于软件本身来说，就是需要再次抽象一层，类似元设计meta define

5M

4d

Other化

作者::  ★(attilax)>>>  绰号:老哇的爪子 （ 全名：：Attilax Akbar Al Rapanui 阿提拉克斯 阿克巴 阿尔 拉帕努伊 ） 汉字名：艾龙，  EMAIL:1466519819@qq.com

转载请注明来源： <http://blog.csdn.net/attilax>

## All in one全体系化

。避免某一方面核心技术受制于人..务必做到体系齐全

## Func base unit 基于功能的单元

## FURPS模型

是功能（function）、易用性（usability）、可靠度（reliability）、性能（performance）及可支持性（supportability）

**最早是由惠普公司的罗伯特·格雷迪（Robert Grady）及卡斯威尔（Caswell）提出**

## Green 绿色

## I18n

## Open sys开放体系

## Plugin 插件化

## platform平台化

## Process customiz流程自定义

Workflow等技术实现 决策表

## Rest 接口rest化

Rpc>>rest

## Text 协议文本化

协议文本化

## Third Api

## Tmork 输入方式

Touch mouse remoter key (扫描er,密码读取器,卡片读取器)

## Vm化，隔离化 再次封装化

因为常用的类库为灵活性,所以粒度小的..可以不个常见的api 大的粒度,这样就使用容易的了..

OA系统软件几个热门概念的点评\_协同OA跑跑观察.html

软件公司未来的发展方向、理念和痛点.html