背景：

了解到公司内部最近也实现了单元测试集成，我们深圳这边的团队最近也在搞线上代码覆盖率的工程建设，就是内测包中，开发测试根据自测用例和测试用例测试，可以生产代码执行的覆盖率情况。

公司内部其实有团队做了，一是他们不使用ibiu发布，二是不兼容swift和oc混编。我这边最近对可行性方案做了实践。源码自动插桩及覆盖率自动收集也做好了，大部分是自动化的。

工单地址： http://xbp.jd.com/ticket/465423

目标：

有了这些数据，更好的去将发布到线上的代码有更加全面的质量保证，无论是在转测质量、还是线上代码质量都可以量化体现。

现状：

这个是我基于cocoapods和ibiu工程的一个自动脚本：

https://github.com/erduoniba/hdcoverage

这个是实践：

https://github.com/erduoniba/HDOCCoverageDemo

上面是已经实现了oc和swift的代码覆盖率自动化建设。

遇到的问题：

1、代码覆盖率基于源码插桩实现，上面的实践成功也是基于源码（静态库）、非源码（动态库源码编译自己插桩）来实现的。**我们的业务组件都是静态库的方式，所以强依赖源码。我们期望是在bamboo打包的时候，ibiu安装能支持开启源码安装（支持全量、指定模块、正则模块）**

2、ibiu安装开启源码后，需要对这些源码组件的buildsettings插入一些指令，这些指令和插入方式我们已经实践，**但是需要你们在安装的时候，对post_install进行插入**。可以参考：

```
# 需要收集Code Coverage的模块，需要从bamboo配置读取
ntargets = Array['AFNetworking']
# buildType，这里也是从bamboo配置读取，我们会基于 Release 复制一个 Coverage 的 buildType
buildType = "$buildType"

require 'xcodeproj'
post_install do |installer|
  installer.pods_project.targets.each do |target|
    target.build_configurations.each do |config|
```

```
      if(config.name <=> $buildType) == 0
        config.build_settings['OTHER_CFLAGS'] = '$(inherited)'
        config.build_settings['OTHER_SWIFT_FLAGS'] = '$(inherited)'
        config.build_settings['OTHER_LDFLAGS'] = '$(inherited)'
        ntargets.each do |ntarget|
          if(ntarget <=> target.name) == 0
            config.build_settings['OTHER_CFLAGS'] = '$(inherited) -fprofile-instr-
generate -fcoverage-mapping'
            config.build_settings['OTHER_SWIFT_FLAGS'] = '$(inherited) -profile-
generate -profile-coverage-mapping'
            config.build_settings['OTHER_LDFLAGS'] = '$(inherited) -fprofile-instr-
generate'
            break
          end
        end
        else
          config.build_settings['OTHER_CFLAGS'] = '$(inherited)'
          config.build_settings['OTHER_SWIFT_FLAGS'] = '$(inherited)'
          config.build_settings['OTHER_LDFLAGS'] = '$(inherited)'
      end
    end
  end

  # 保存主工程
  project_path = './xxxx.xcodeproj'
  project = Xcodeproj::Project.open(project_path)
  project.save()
end
```

3、其他问题暂时还没有想到会出现什么问题，到时候我们一起**解决优化**，搭建一套自动化、通用性的线上代码覆盖率平台。