支付宝日志规范

文件编号	
文件版本	
编　　制	
批　　准	
负 责 人	
读　　者	
保密级别	Internal

修订记录

版本号	修订日期	修订概述	修订人	审核人	批准人	备注
1.0	/	1.0版本	SQPG			
beta2.0	2014.05.23	增加1.5.5 日志版本规范 增加1.6 安全日志 增加1.7 资损日志 增加1.10 框架日志	相如			
beta2.0	2014.05.28	增加1.3.2，集成《支付宝日志存储规范V1.0》	相如			
beta2.1	2014.08.13	增加1.11 无线日志	浪翻云			
2.0	2014.09.03	修订版本号为 2.0	SQPG			

分发/访问列表

姓名	角色	访问类型 (只读/编辑)

1.1 概述

日志信息是开发人员排查线上问题最主要的手段之一，形成一个规范性的日志记录规则，对于线上问题排查定位，形成有效监控规则，降低排查时间具有重大意义，同时日志信息也是大数据的一部分，也需要遵循数据的规范(含安全)。

系统的 log4j.xml/log4j.xml.vm 配置文件的维护权限属于该系统 owner，他人不得随意修改配置文件。系统 owner 有义务针对日志规范对开发人员进行培训，并有 review 的权利和职责。新系统上线前系统 owner 必须 review 日志配置文件是否符合规范。

1.2 基本原则

- 日志输出不允许影响系统正常运行；
- 日志输出不允许产生安全问题；
- 日志不允许输出公司机密信息；
- 日志可供开发人员定位问题的真正原因；
- 日志可供监控系统自动监控与分析；
- 日志使用符合安全审计要求。

1.3 日志文件规范

1.3.1 日志文件定义

系统中定义日志文件信息，主要分两种：在 webx 系统中，使用 log4j.xml.vm 定义日志文件信息；在 sofa 系统中，使用 log4j.xml 定义日志文件信息。对于 log4j.xml.vm 和 log4j.xml 实际编写规则是一样的，webx 中 log4j.xml.vm 文件在系统编译完成的时候，最终会自动生成 log4j.xml 文件。

一个最简单的 log4j.xml

```
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">
    <appender name="DEFAULT-APPENDER" class="com.alibaba.common.logging.spi.log4j.DailyRollingFileAppender">
        <param name="file" value="${loggingRoot}/${sys_host_name}/common-default.log"/>
        <param name="append" value="true"/>
        <param name="encoding" value="GBK"/>
        <layout class="org.apache.log4j.PatternLayout">
            <param name="ConversionPattern" value="%d %m%n"/>
        </layout>
    </appender>

    <logger name="com.alipay" additivity="false">
        <level value="${settlecore_loggingLevel}"/>
        <appender-ref ref="SOFA-DEFAULT-APPENDER"/>
      <appender-ref ref="ERROR-APPENDER"/>
    </logger>

    <root>
      <level value="${settlecore_loggingLevel}"/>
      <appender-ref ref="DEFAULT-APPENDER"/>
      <appender-ref ref="ERROR-APPENDER"/>
    </root>
</log4j:configuration>
```

Log4j.xml 由于 APACHE LOG4J 进行解析，各个元素及参数的含义这里不做解释，仅说明如何定义一个日志文件。

1.3.1.1 定义 logger 元素

- Logger 元素命名方式主要有两种，一种是包结构命名，如：logger name="com.alipay.settlecore.core.deposit"；另一种是变量命名，如：logger name="BATCH_ARCHIVE"，一般选择第一种定义方式。命名完成后，设置日志级别并关联到相应 appender 元素即可。

- Logger 元素使用 level 标签定义日志级别，对于应用日志使用${系统名_loggingLevel_info}设置，对于框架及第三方包日志使用${系统名_loggingLevel}设置。这些变量最终由 antx.properties 定制级别。
- Logger 元素使用 appender-ref 标签与 Appender 元素进行关联，一般关联自定义的应用日志及 ERROR 日志。

1.3.1.2 定义 appender 元素

- Appender 元素命名规则为 xxx-APPENDER，对应 class 为：com.alibaba.common.logging.spi.log4j.DailyRollingFileAppender，这种 appender 每天生成一个新的日志文件。日志文件名如：${loggingRoot}/${sys_host_name}/common-default.log"，文件名尽量体现具体业务含义，如 settlecore-deposit.log 表述清算层充值日志文件。再设置一下 appender 元素的编码集、和日志样式即可。
- 生产环境下的文件日志必须使用 DailyFileAppender
    - datePattern:’.’yyyy-MM-dd (默认)
    - encoding:GB18030 或者 GBK
    - append:默认(true)
    - immediateFlush: 默认(true)
    - bufferedIO: 默认(false)，对非关键海量型日志允许设为 true 以提高性能(同时自动设置 immediateFlush 为 false)
- 生产环境下的 Jboss 日志必须使用 ConsoleAppender
    - org.jboss：输出到 System.out

- STDOUT：输出到 System.out
- STDERR：输出到 System.err

- 使用 PatternLayout 格式化日志信息
  - 对于 WEB 层应用推荐使用

    %d

    [%X{loginUserEmail}/%X{loginUserID}/%X{remoteAddr}/%X{clientId}

    - %X{requestURIWithQueryString}] %-5p %c{2} - %m%n
  - 对于后台应用程序推荐使用

    %d [%t] %-5p %c{2} - %m%n
  - 除非必要，不使用以下格式化描述符: %C、%M、%F、%L、%l。位置信息可通过 logger 名称、异常栈或者消息体间接获知

1.3.1.3 日志文件名

- 日志文件名称定义应简明，看到日志文件就知道里面是哪方面的日志内容：如 xxx-integration.log 是本系统与外界系统交互的日志信息;xxxx-deposit.log 表示充值相关日志等等。目前 sofa2.0 基本按照工程结构打印日志。
- 所有错误日志必须输出到 error.log(webx 系统)或 common-error.log(sofa 系统)，目前线上所有的错误日志会自动生成报告，限期安排人员跟踪解决。

1.3.1.4 注意事项

- 涉及到数据丢失会造成资金损失的业务。必须记录远程灾备日志，保证灾难发生时我们的资金来往有据可查并能快速恢复。

- 不要将签名记录日志与其它非关键应用日志混合记录在同一个日志文件中，应该分别记录日志，独备份签名日志文件。
- 不同的Appender之间不允许共享文件，Log4j不会控制不同Appender间的同步，同时写一个文件会产生并发写问题

1.3.2 日志文件保存

原始日志在产生日志设备默认保存 14 天，在磁盘空间不足的情况下，系统管理员根据日志生成时间的先后，删除较早的日志。

1.3.2.1 日志统一存储

系统管理员需要把原始日志收集到日志统一存储系统，进行集中管理。

日志统一存储的形式有两种：

- 在线存储，可以立即查询与分析，适用于访问频率高的日志。
- 离线存储，查询与分析的需要较长的准备时间（磁带存储），适用于访问频率较低的日志。

日志存储时间的配置规则有两种：默认存储计划和特殊存储计划。

- 特殊存储计划中有规定的日志，按照特殊存储计划保存；
- 特殊存储计划中对同一日志存储时间要求不同时，以存储时间长的为准。
- 其他的日志按默认存储计划进行保存。

1.3.2.2 默认存储计划

1）应用日志

应用日志的保存时间由其包含的业务数据决定。如果一个日志文件中包括多种业务数据，保存时间以时间要求较高的为准。应用日志的保存计划见表 1。应用日志的保存时间将在日志名中体现。

表 1 应用日志保存计划

应用日志业务类型	在线保存时间	总保存时间
支付清算日志	1 年	5 年
客户身份验证日志	1 年	5 年
交易日志	1 年	3 年
关键数据修改日志	1 年	3 年
错误日志	1 年	3 年
信用卡业务日志	1 年	1 年
其他日志	3 月	3 月

- **支付清算日志**，日志中包含用户支付指令，支付宝向银行发出的支付清算指令等数据。
- **客户身份验证日志**，日志中包含用户身份验证、商户身份验证、登录密码验证、支付密码验证、数据签名验证、手机验证码验证等数据。
- **交易日志**,日志中包含交易创建、交易付款等交易过程中各个阶段产生的数据。
- **关键数据修改日志**，日志中包含客户身份信息、联系方式、安全认证信息、资产信息等数据的修改操作。
- **信用卡业务日志**，日志中包含信用卡还款、信用卡快捷、信用卡绑定等信用卡相关业务数据。
- **错误日志**，日志中包含应用或业务的错误信息。

2）操作系统及组件日志

操作系统及组件日志存储计划见表 2。

表 2 操作系统及组件日志存储计划

日志分类	在线保存时间	总保存时间
系统日志	1 年	1 年
Apache 日志	3 月	3 月
Jboss 日志	3 月	3 月

1.3.2.3 特殊存储计划

对于默认存储计划不能满足的日志存储需求，可以建立特殊存储计划，系统管理员须维护特殊存储计划表，并根据存储计划进行日志的管理。

特殊存储计划至少应该包含以下的要素：

- 计划名称
- 需求的提出部门
- 建立计划的依据
- 日志的类型
- 目标设备与系统
- 日志保存时间
- 失效日期

特殊存储计划可以进行添加，修改与删除。特殊存储计划的变更须需求方提出申请，系统管理员评估通过，则添加新的特殊存储计划。计划失效后，须删除该计划。

日志特殊存储需求的申请流程：

- 需求方填写日志特殊存储需求申请表（见表 3）；
- 系统管理员评估；
- 建立特殊存储计划。

表 3 日志特殊存储需求申请表

提出部门	需求依据	日志名	目标设备与系统	日志量/天	在线保存时间/总保存时间	特殊计划失效日期

1.3.2.4 日志的命名规则

1) 应用日志命名规则

文件名格式：{日志原名}.{原始日志保存时间}.{在线保存时间}.{总保存时间} .{日志业务类型标识}.log.{日期}

- 原始日志保存时间格式：{Num}dt
  - 代表原始日志在生产集群上保存的时间。
  - Num 是数字，d 表示天。
  - 缺省则按原始日志默认计划最多保存 14 天。
- 在线保存时间格式：{Num}{(d/m/y)}o
  - 代表日志集中在线存储的时间。
  - Num 是数字，d 表示天，m 表示月，y 表示年。
  - 缺省则按照该日志业务类型的默认保存时间保存。
- 总保存时间格式：{Num}{(d/m/y)}e

- Num 是数字，d 表示天，m 表示月，y 表示年。
- 缺省则按照该日志业务类型的默认保存时间保存。

- 日志业务类型是日志保存时间的依据。如果一个日志文件中包括多种业务数据，保存时间以时间要求高的为准。日志业务类型标识参考表 4。
  - 缺省则默认为其他日志。
- 日期：yyyy-mm-dd

表 4 日志业务类型标识

应用日志分类	日志类型标识
支付清算日志	p
客户身份验证日志	v
交易日志	t
关键数据修改日志	a
错误日志	e
信用卡相关日志	c
其他日志	d(可选项)

示例：

trade.14dt.1yo.3ye.t.log 表示这个日志是交易业务日志，总保存时间为 3 年，在线保存 1 年，原始日志在生产集群上保存 14 天。

trade.t.log 表示这个日志是交易日志，保存时间按照交易业务日志保存，原始日志在生产集群上保存 14 天。

cache-common.log 表示这个日志按照默认保存时间保存。

2）其他日志命名规则

日志分类	命名规则
系统日志	系统默认
Apache 日志	文件名格式：{日期}-{日志原名}.log ■ 日期：yyyy-mm-dd 举例： 2011-03-30-access.log
Jboss 日志	文件名格式：{日志原名}.log.{日期} ■ 日期：yyyymmdd 举例： stdout.log.20110323

1.4 日志定义规范

- 基础库系统直接使用 apache log4j 的 Logger 进行日志定义。业务系统使用 Alibaba 封装后的 Logger 进行日志定义。
- 日志定义和日志文件中 logger 元素的定义有很大的关联性，主要有两种方式：

  private static final Logger logger = LoggerFactory.getLogger(XXX.class);

  private static final Logger logger = LoggerFactory.getLogger(“XXX”);
- 第一种方式中的 XXX 为本类类名，定义时请留意类名，系统中存在 copay 代码的现象，logger 定义的类名非本类类名，导致日志输出有误。在打印日志时 log4j 框架将根据类名对应的包路径在 log4j.xml 中进行匹配，根据匹配结果，将日志内容输出

到对应的日志文件中。

- 第二种方式中的 XXX 于 log4j.xml 文件中的 logger 元素名对应。在打印日志时，日志内容将直接输出到 logger 元素名为 XXX 所指定的 appender 中。
- 注 :Logger 定义一般情况不允许使用非 private 修饰符之外的其他修饰符进行修饰。

1.5 日志内容规范

1.5.1 日志级别

Log4j 共六个日志级别：TRACE、DEBUG、INFO、WARN、ERROR、FATAL，常用的有 4 个 DEBUG、INFO、WARN、ERROR，公司生产系统一般只打印 INFO 级别以上的日志信息，对于 DEBUG 级别的日志，只在测试环境中打印。

1.5.2 日志格式

[线程 ID:]业务描述:方法描述:[业务流水号:]参数内容

例: logger.info(Thread.currentThread().getId() + ":卡通协议:接收到卡通协议签约请求:" + signInfo);

- 线程 ID：可选，建议在接口服务层及外部系统调用层使用。
- 业务描述：简明描述一下当前业务。
- 方法描述：简明描述一下当前操作。
- 业务流水号：可选，如果能安全获取到，建议打印出来，安全获取的意思是不能为了获取业务流水号产生异常。
- 参数内容：业务操作时所用到的参数信息。
- 分隔符：暂定为冒号，（经常看到有使用 [] 或 【】这种的，个人觉得编写起来

不太顺手，对排查问题也没有太大的帮助）。

1.5.3 日志安全

- 应用中绝对不允许直接使用 System.out 或 System.err 输出日志或使用 e.printStackTrace()打印异常堆栈，也不允许使用 STDOUT、STDERR 作为 Logger 名字。由于标准日志输出与标准错误输出文件每次 Jboss 重启时才滚动，因此，如果大量输出送往这两个文件，容易造成文件大小超过操作系统大小限制。
- 日志消息中不允许出现用户密码、短信验证码、支付验证码、身份证号、银行卡号等敏感信息。
- 日志消息中不允许出现银行密钥，商户密钥、密码等秘密信息。
- 凡记录与外部系统通过公钥机制进行报文签名的日志，必须与运维部门确认进行备份与长期保存。
- 打印日志的代码任何情况下都不允许失败。

详见《1.6 安全日志》

1.5.4 日志规则

- 打印 INFO 及 INFO 级以下的日志时，必须使用对应的 isxxxEnable()方法进行判断。
- 系统中供外部调用的接口，与外部系统交互，定时任务入口，类间公用（public）方法调用必须打印 INFO 级别日志。其中供外部调用的接口在进入和返回时都需要打印 INFO 级别日志。
- 对于复杂方法体中的主要分支，建议在分支入口处打印 debug 日志，以方便跟踪程序流程。

- 方法体内的 if...else...判断 ,对于 else 是非正常的情况 ,需要根据情况选择打印 warn 或 error 日志，对于只有 if 没有 else 的地方，如果 else 的路径是不可能的，应当加上 else 语句，并打印 error 日志。
- 日志内容中所打印的对象必须实现 toString()方法，如果对象中包含敏感信息，应当进行过滤，一般使用如下方式实现 toString()方法：

```
public String toString() {
        return
  ToStringBuilder.reflectionToString(this,ToStringStyle.SHORT_PREFIX_STYLE
  );
}
```

- 循环体内禁止打印 INFO 级别日志，如果数据量过大会造成日志文件过大，导致排查问题不方便。
- 构造字符串消息，不要使用 concat()方法，使用 + 运算符，使用 concat 容易出现异常导致业务失败。
- 不要在不同地方重复记录针对同一事件的日志消息。
- Catch 块中的异常记录必须堆栈信息打印出来。
- 不记录对于排查故障毫无意义的日志信息，如：logger.error(“系统未知错误”,e); logger.info(“开始发送请求…”);日志信息一定要带有业务信息。
- 打印错误日志时，需要区分是业务异常(如:用户名不能为空)还是系统异常(如:调用会员核心异常)，业务异常使用 warn 级别记录，系统异常使用 error 记录。避免 error 日志过大，影响紧急故障排查。

1.5.5 日志版本

目前日志文件的使用场景逐渐扩大，日志文件的稳定需要关注，本节规范为**试行规范。**

- 发布需要评估新增或变更日志的兼容性
  - 是否涉及已有字段定义、字段位置的改变？
  - 是否采用行末追加字段方式？
  - 是否涉及动态字段（字段数会动态变化）打印？
  - 是否涉及不规范分隔符使用？
- 如果涉及日志文件或日志内容较大变更，需要记录变更。
- 如果涉及日志文件或日志内容较大变更，没有记录变更，通过 alipay monitor 的配合发布变更提示(警告)。

1.5.6 日志打印例子

- DEBUG 级日志:

```
if (logger.isDebugEnabled()) {
    logger.debug("准备获取文件信息:bankId=" + bankId + ":bizDate=" + bizDate);
}
if (logger.isDebugEnabled()) {
     logger.debug("准备获取文件信息:"+fileObject);
}
```

- INFO 级日志:

logger.info(Thread.currentThread().getId() + ":卡通协议:接收到卡通协议签约请求:" +

signInfo);

logger.info("开始处理卡通协议签约请求:" + signInfo);

- WARN 级日志:

logger.warn("非法卡通协议操作类型: status =" + status);

- ERROR 级日志:

logger.error("获取不到对应的卡通协议校验器:" + agreementInfo);

错误形式

```
try {
    ……
} catch (Exception e) {
    logger.error("清算层调用异常", e);
    return false;
}
try {
    ……
} catch (Exception e) {
    logger.error("清算层调用异常"+e);
    return false;
```

```
}
```

正确形式

```
try {
      ……
} catch (Exception e) {
      logger.error("清算层调用异常
                  channelApi="+channelApi+”:serialNO=”+serialNo, e);
    return false;
}
```

1.6 安全日志

1.6.1 内容完整规范

根据安全需求，业务日志需要尽量完整，考虑到全站性要求，由框架统一负责输出，包括 sofa mvc/rpc/msgbroker/httpclient/zdal 等等，保证日志的完整性和一致性，其中针对 web 类日志输出(安全需求)，框架提供：

1、 请求参数日志 filter：支持增加请求参数，可关；

2、 业务描述 reportAPI：业务描述与应用相关，需要应用主动 report 业务类型；

3、 异常描述 reportAPI：业务异常与应用相关，需要应用主动 report 业务异常；

具体框架和容器升级详见文档。

1.6.2 内容脱敏规范

1. 高敏感度信息禁止打印

在日志中禁止打印密码/口令、密钥等高度敏感度的信息。

如：登录/支付密码、手机校验码、密码保护问题答案、加密或签名密钥、sessionId 等。

2. 部分打印

对于会员个人信息和银行卡信息，在打印日志时应按照规范只打印部分内容。

3. 支付宝敏感信息按照内容分为五类， 具体范围与打印规则如下：

类型	信息范围	规范	备注
密码/口令及相关	1) 登录密码、 2) 支付密码、 3) 手机校验码、 4) 密码保护问题答案、 5) 支付盾 PIN 码、 6) 宝令动态密码、 7) 汇票密码、 8) 3D 密码、 9) 银行卡 PIN、 10) sessionId 等	禁止打印	登录/支付密码的 MD5 摘要值也不能打印。
密钥	1) 数据加密密钥、 2) 签名私钥、 3) Md5/HMAC 消息认证密钥等	禁止打印	
信用卡信息	1) 信用卡卡号	前 6 和后 4 位	PCI-DSS 要求， 任何情况都不能打印完整信用卡卡号。 （索引卡号可以打印）
	2) 信用卡 CVV2/CVC2	禁止打印	PCI-DSS 要求。
	3) 信用卡有效期	禁止打印	PCI-DSS 要求。 例外：虚拟信用卡如支付宝境外发卡，可打印，但必须不与卡号同时出现。
借记卡信息	借记卡卡号	前 6 和后 4 位	新项目应按此规范，旧系统正在逐步改造。
个人信息	1) 身份证号	前 6 和后 4 位	新项目应按此规范，旧系统正在逐步改造。
	2) 手机号	暂无要求	
	3) 固定电话号码	暂无要求	
	4) 邮箱	暂无要求	
	5) 地址信息	暂无要求	

4. 脱敏暂行处理机制

考虑到框架脱敏带来的额外性能消耗，本规范采用强推规范事后检查的模式，检查结果定期报告，相关应用务必根据报告整改。

1.7 资损日志

任何有资金流转的业务操作必须记录资损日志。

日志文件建议系统名 xxx 开始,yyy 表示业务名，比如 xxx-check-yyy-7dt.log，格式基本参 sofa-service-digest 日志，每行表示一笔业务记录，各要素分割以“,”为准，其中无此要素则“-”替代，各要素位置需要固定,个性化核对扩展要素方在行末，日志默认保留 7 天，如果量大的系统可以适当减少，各要素说明见下表。

也可以按照业务类型区分日志文件，以保险为例，如：baoxiancore-check-claim.log(理赔)，baoxiancore-check-surrender(退保)等，各应用可根据实际来定，但各文件内容务必遵守下表约定。

字段要素	描述	备注
业务单据号	业务唯一单号	
操作类型	支付\退款\冻结\转账	pay\back\freeze\unfreeze
资金单据号(根据实际情况)		
全局号	uuid	
金额		
付款账号(2088)		
收款账号(2088)		
其他信息	各平台自定义需要核对元素	

1.8 监控日志

对于重要的业务需要记录 digest 日志，要求打出内容：日期，业务标识，执行业务所花时间，本次业务成功还是失败，业务名称。 每一条日志的格式都要一样，如：2009-04-16 00:00:00,807 [batchAccountTrans,Y, 9ms,...,...,...]，并加入支付宝业务监控系统进行监控。

监控日志打印可以采用 Annotation 结合拦截器的方式完成，不影响业务执行。例如：

```
public interface ApplyEBankService {

    /**
     * 申请网银充值协议
     *
     *  <br>1、完成充值协议数据登记;
     *  <br>2、完成分流任务数据登记(可选);
     *  <br>3、报送清算层并获取供页面跳转的表单对象;
     *  <br>4、完成协议数据变更登记并返回申请处理结果。
     *
     * @param order 网银充值申请专用单据
     * @param operationContext 充值行为操作上下文
     * @return 充值协议申请结果
     */
    @DigestLogAnnotated(name = DepositDigestLogConstant.DEPOSIT_DIGEST_LOG, level = LoggerLevel.INFO)
    public ContractApplyResult apply(EBankOrder order, OperationContext operationContext);
}
```

配置文件为：

```
<!-- 异步网关型充值协议申请服务AOP代理 -->
<bean id="applyEBankService"
    class="org.springframework.aop.framework.ProxyFactoryBean">
    <property name="proxyInterfaces"
        value="com.alipay.paycore.service.deposit.api.ApplyEBankService"/>
    <property name="target" ref="applyEBankServiceTarget"/>
    <property name="interceptorNames">
        <list>
            <value>depositDigestLogInterceptor</value>
        </list>
    </property>
</bean>

<!-- 异步网关型充值协议申请服务  -->
<bean id="applyEBankServiceTarget"
    class="com.alipay.paycore.ext.service.deposit.impl.ApplyEBankServiceImpl"
    parent="abstractApplyService">
</bean>
```

1.9 性能日志

系统中提供的服务如果业务逻辑比较复杂的接口,或业务逻辑可能存在性能问题的方法，我们需要在业务处理过程中添加性能日志。公司统一使用阿里巴巴性能日志工具类：com.alibaba.common.lang.diagnostic.Profiler 来记录性能日志。性能日志记录的信息可以在系统的 common-perf.log 日志中进行查看,Profiler 将把处理时间超过 300 毫秒的逻辑分支记录下来。记录形式如下：

```
2010-08-04 21:31:55,968 [/// - ] WARN  PERF -
[服务请求处理时间] 0 [668ms (39ms), 100%] - 调用SOFA服务：调用的方法为：apply 参数个数为：1
        +---1 [629ms (6ms), 94%, 94%] - 卡通协议处理开始
        |   +---1 [17ms, 3%, 3%] - 设置用户信息
        |   |   `---1 [17ms, 100%, 3%] - 调用HSF服务：com.alipay.cif.user.UserInfoQueryService:1.0@XFIRE 调用的方法为：findUserInfoByLogonId 参数个数为：1
        |   +---18 [587ms (1ms), 93%, 88%] - 卡通协议校验
        |   |   +---18 [5ms, 1%, 1%] - 调用HSF服务：com.alipay.cif.facade.UserQueryService:1.0@XFIRE 调用的方法为：isQregistMobileUser 参数个数为：1
        |   |   +---23 [5ms, 1%, 1%] - 调用HSF服务：com.alipay.cif.facade.CardService:1.0@XFIRE 调用的方法为：findChildAccountNosByCardNo 参数个数为：1
        |   |   `---29 [576ms, 98%, 86%] - 调用HSF服务：com.alipay.certify.service.facade.CertifyGonganCheckFacade:1.0@XFIRE 调用的方法为：gonganCheckByNameA
ndNo 参数个数为：2
        |   +---605 [0ms] - 卡通协议领域构建
        |   +---605 [0ms] - 协议领域预激活处理
        |   +---605 [16ms, 3%, 2%] - 协议领域激活处理
        |   +---621 [0ms] - 协议领域激活后置处理
        |   `---621 [3ms, 0%, 0%] - 发送协议统一事件
        `---631 [0ms] - 将cardSignDO转成KatongCardInfo
```

从上面这个性能日志可以看出，这是一个卡通协议申请操作，总共消耗 668 毫秒。总时间是各分支处理时间之和,你可能发现存在一些差异,这是由于系统中没有对所有的处理都加上性能日志的缘故。这上面这个例子也能看出，卡通协议申请业务，消耗时间较长的大都是远程 WS 调用。

性能日志使用方法如下，在使用性能日志时一定要检查 Profiler.enter()与 Profiler.release()成对出现，不然可能出现内存泄露问题。一般而言，在方法调用处记录 Profiler 性能日志，而不是在方法体内进行。

```
Profiler.enter("卡通协议处理开始");
{
    serviceResult = katongAgreementOperator.operateRequest(agreementInfo, agreementInfo
        .getOperateType());
}
Profiler.release();
```

1.10 框架日志

关于 PE+容器+监控三合一配合推行一套行之有效的框架日志规范的目标，有必要对框架日志进行更好的规范，框架日志规范暂试行。

1.11 无线日志

日志格式：

| 日期 | 时间| 时区| 请求方法| rpc_api | rpc_version | referrer | UserAgent | 源 ip | 用户 id | 用户账号| traceid| 返回 status | 返回数据大小 |业务处理耗时| location|请求参数 | 业务处理结果| 业务处理返回数据| 设备是否越狱 | 是否模拟器 | Mac 地址 | IMEI | IMSI | utdid | tid | apdid | umid | 设备品牌 | 设备型号 | 屏幕宽*高 | 屏幕分辨率 | 终端经纬度 | 操作系统 | 操作系统版本号 | 客户端版本 | 安装包渠道 |Wifi 信息 bssid | Wifi 信息 ssid| Wifi 信息 active | wifi 信号强度 | 网络类型| 基站信息 Mcc | 基站信息 Mnc | 基站信息 CellID| 基站信息 Lac

日志解释：

字段要素	**描述**	**备注**
日期		2014-08-08
时间		00:00:04
时区		钱包推向国际后的预留字段
请求方法	Get/post/rpc	
rpc_api	Rpc 请求的具体接口命名	
rpc_version	Rpc 请求的具体接口版本	

referrer	来源 URL	H5 专用
UserAgent	浏览器 UA 信息	H5 专用
源 ip	请求的来源 IP	
用户 id (2088)		
用户账号(logonid)	手机号或邮箱	
traceid	SOFA 框架追踪 ID	M+md5(actionToken+random)+inc
返回 status	Rpc 请求返回的状态码	
返回数据大小	Rpc 请求返回的数据包的大小	
业务处理耗时	Rpc 请求处理的耗时	
location		H5 专用
请求参数	Rpc 请求的参数	每个参数可配置以下 3 种处理方式之一进行处理：urlencode 后打印、urlencode 后部分屏蔽打印、不打印
业务处理结果	Rpc 请求的业务处理结果	例如登录成功
业务处理返回数据	Rpc 请求的业务处理返回数据	返回数据内容中的敏感信息，包含敏感信息的类别（手机号、证件号、银行卡号），敏感信息 MD5 值，敏感内容部分屏蔽方式打印
设备是否越狱/root		
是否模拟器		
Mac 地址	Media Access Control 地址，用来定义网络设备的位置。	

IMEI	移动设备国际身份码	
IMSI	移动用户国际识别码	
utdid	淘宝 usertrack 设备 ID	
tid	支付宝客户端设备 ID	
apdid	支付宝设备指纹 ID	
umid	唯一机器标识	
设备品牌		
设备型号		
屏幕宽*高		
屏幕分辨率		
终端经纬度		
操作系统	Ios/android	
操作系统版本号		
客户端版本		
安装包渠道	客户端投放的下载渠道	例如豌豆荚、91 市场等
Wifi 信息 bssid	wifi_mac 地址	
Wifi 信息 ssid	wifi 名称	
Wifi 信息 active	wifi 是否已经连接上	
wifi 信号强度		
网络类型	GPRS/CDMA/EDGE/HSDPA	
基站信息 Mcc	Mobile Country Code，移动国家码，三位数，如中国为 460	

基站信息 Mnc	Mobile Network Code，移动网络号，两位数	
基站信息 CellID	Cell Identity 小区码，是一个 2 个字节长的十六进制 BCD 码	
基站信息 Lac	Location Area Code，是一个 2 个字节长的十六进制 BCD 码	