目标

- 实现性能分析的相关功能可以在 OpenJ9 上正常使用
- 平台OS：优先支持 Linux x86_64，再适配 AIX 平台；

规约

1. 尽全力做到，仅仅改变底层数据的采集方式，不改变数据模型；
2. 对于实在无法兼容之前的数据模型的，需要考虑上层展现层做相关调整，包括 UI、后端数据模型兼容等。

涉及模块和功能

几乎所有的分析模块都涉及改动。肯定需要整体修改的是：attach 包需要和 openJDK 的 attach 包区分开来。

XLion

涉及功能

- 热点栈采集，基于 Hotspot JVM 的实现需要调整方案，该方案的实现可以参考 JMC、YouKit 的采样方式来实现，但是，需要详细了解其原理和评估其性能；
- 进程 CPU 使用率采集，当前为把热点栈和 CPU 的使用率完美的集合，该功能的实现放到了 CPU 的 JVMTI Agent 里实现，而且是基于 linux 平台来实现的，即读取的是 proc 下的数据。所以涉及两方面的改造；
  1. 涉及 linux proc 的，如果移植到 aix，这部分可能需要修改，需要调研下 aix 下是否有同样的机制；
  2. 热点栈的采集若采用 JMC、YouKit 的采样方式，很可能会用 java 实现，那么上述 CPU 使用率的实现也最好是放到其中实现；

改动模块

JVMTI-Agent、Xowl-module

XSheepDog

分为实时分析和离线分析

实时分析

- 线程的 CPU 使用率问题，就目前 OpenJ9 的 thread-dump 文件来看，其中只有 java-thread-id，并没有本地 LWP 线程的 id，也就无法在 openJ9 上使用 OPEN JDK 上同样的手段来爬去线程的 CPU 使用率，只能使用 JMX(获取 CPU 使用率这是 Java 语言包下的 API，原则上，OpenJ9 应该也是支持的，需要验证下) 的方式来实现了。
- 由于无法对 java-thread-id 和 LWP id 做对应，CPU 负责的视图可能就无法呈现数据了。如果要做，估计只能通过将线程的 Agent 作为 on_load 的机制加载上去，以此来对应 pid 和 lwp_id，从而到达此目的，这个方案还有待验证；但是，这样一样来，底层的挂载行为就需要做变动；
- 线程创建和删除视图，需要验证 OpenJ9 是否有相关的 JVMTI 事件，若有则最好，否则将比较麻烦，原则上应该有，Open J9 JVMTI；
- 线程池视图依赖线程的创建和删除；
- 线程状态一栏，目前是采用 lwp 的状态来映射的，这个结果可能和 JConsole、JMC 等不一致，该特性依赖是否可以对应 java-thread-id 和 LWP id；否则只能通过采用 JConsole 和 JMC 的形式来获取相关视图的信息；

> 综上所述，核心需要讨论或验证的地方包括两点：
> 是否采用 thread-id 和 lwp-id 对应的方式继续来实现 open j9 的相关功能；从性能的角度来考虑，最好是这样；
> Open j9 的 JVMTI 是否支持 Thread-Start 和 Thread-end 事件；

离线分析

- J9 dump 文件中不包含 lwp id；
- J9 dump 文件最后没有 openJdk 的死锁分析，需要自己在上层建模分析；

> 更详细的区别，需要在仔细对比一下

改动模块

JVMTI-Agent、Module、离线解析，可能涉及上层模型和 UI 的改造；

XElephant

实时分析

- 内存区域视图，GC 监控，目前都是通过 JMX 获取的，需要验证，自动模式下，GC 的数据还从 PerfData 中拿；
- mini dump 是采用 JVMTI Agent 实现，这里的实现可以测试一下，看是否可以用，如果不行，就需要参考 J9 的 jmap 实现；
- dump 文件的生成和 OpenJDK 不一样，需要在启动进程中加上 JVM 参数`j9_heap_opt="-Xdump:heap:events=user+abort,label=/home/admin/heapdump.%Y%m%d.%H%M%S.%pid.%seq.phd,range=1..4,priority=500,request=exclusive+compact+prepwalk,opts=PHD"`，然后使用 kill -3 $pid 来生成内存 dump 文件，且文件格式也不是 OpenJDK 的 hprof 格式，而是 J9 的独有格式 phd。

离线分析

- 如上说述，pdh 文件和 hprof 文件完全不一样，需要重新定制，这部分在 MAT 中已经有了，需要移植。
- 需要验证模型是否是一样的？这个可能需要和前端做适配

改动模块

xowl module、common-agent、mem-agent，离线解析引擎，UI

X 分析

验证一下，理论上不需要做修改

初步分工

	XSheepDog	XLion	XElephant	X 分析	Agent	UI	设计
负责人	涂改	东山	心弦	西罗	西罗	江城	玉衡

规划

为了提高并发性，计划分三步走

1. 各负责人先把相关方案的可行性分析做完
2. 由西罗牵头商讨和确定整体兼容方案，做到更优雅的兼容
3. 编码、测试