

# JINSPIRED Developer

Activity Metering, Metric Monitoring and Event Signaling for Enterprise Java Applications, Cloud Services and Platforms

[Builds](#) [Usage](#) [Docs](#) [Repo](#) [API](#) [Install](#) [Config](#) [Console](#) [Site](#)

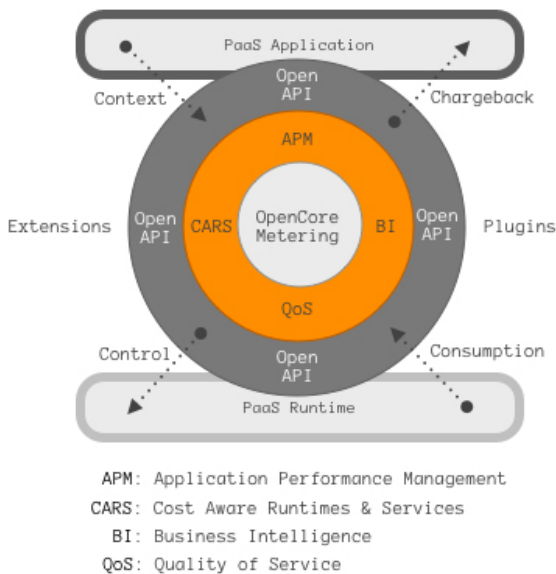
## OpenCore – A Cloud Cortex for PaaS

by Architect on May 12, 2011

Over the last few weeks we've witnessed a daily flurry of news and tweets related to upcoming PaaS platform offerings. Whilst some of these have been in the works for sometime what has been surprising is the lack of imagination and technical vision with the approach taken. It would seem that PaaS vendors have inhibited innovation in the cloud in order to ensure that existing and possibly under-development enterprise applications are managed at arms length by treating them as black box processes to be started and stopped from the command line or a web console. This is unfortunate because there is so much potential benefit in offering application execution runtimes that are much more aware of the business activities that drive the execution behavior and resource consumption of applications. It is this same context which can be used to profile (*cost & performance*), persuade (*tracking & analytics*), predict (*intelligence*), provision (*reservation & leasing*), protect (*self-regulation & supervision*) and prioritize (*classify & coordinate*) runtimes and resources.

Such things can be achieved, even today, with the help of OpenCore's activity based costing and resource metering technology acting as a cerebral cortex that abstracts and models the problem domain in terms of named activities, their associated groupings, and resource meters tracking consumption mapped to latency, (*cost*) liability and (*resource*) leasing measures. Most importantly it can be achieved in real-time, local and immediate, with very little human intervention and no dependency on another external service and its availability.

Below is a graphic that depicts the typical contextual and control flows that we foresee in more ambitious and future proof PaaS environments in which the runtime plays a more active and transparent role.



1. Applications provide context (*beyond code namespaces*) to the execution that is performed in a PaaS runtime in firing contextual named (*activity*) probes via the [Open API](#).
2. PaaS runtimes register request worker (*thread*) specific resource measures with the metering engine which are then tracked in terms of consumption and in turn traced backed to one or more activities in a processing chain or flow.
3. Applications then use the [Open API](#) to access the metering (*tallied consumption records*) for each activity the metering engine and runtime were made aware of. This metering information can be used to reconcile billing, report & control costs as well as chargeback to users of the application and its exposed metered services.
4. Paas runtimes use the application & activity context as a means to distinguish different cost and consumption profiles and to predict consumption, protect resources and prioritize worker activity executions in real-time.

FROM: CARS, CLOUD COMPUTING, COSTICITY, OPEN, QOS