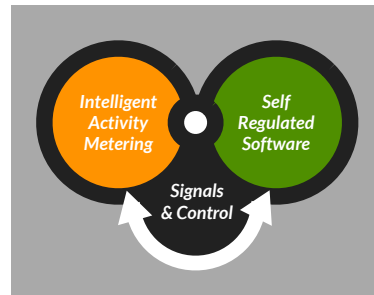


# ***QoS for (Web) Applications***

## ***Velocity EU 2011***



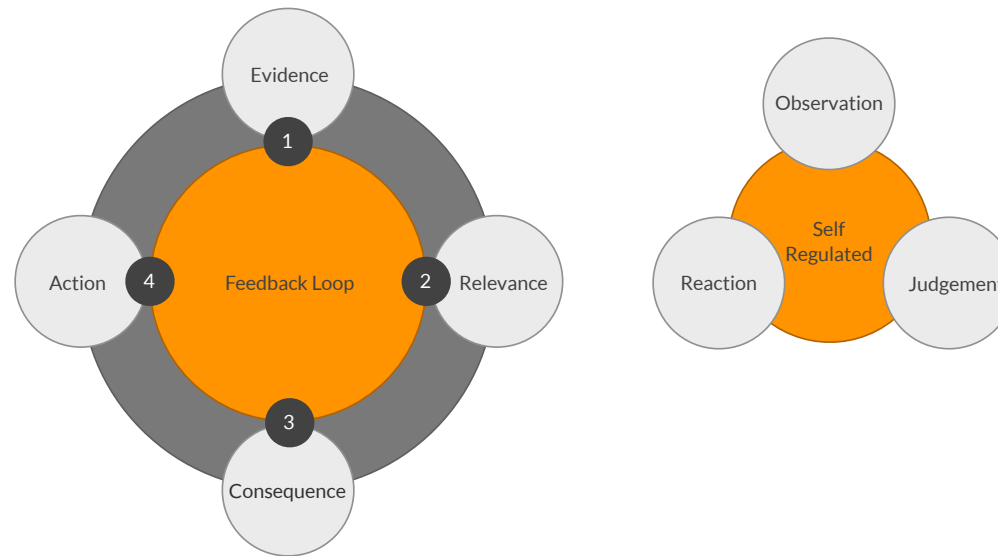
**[william.louth@jinspired.com](mailto:william.louth@jinspired.com)**

# ***Self Adaptive Software***

***“Self Adaptive Software evaluates its own behavior and changes behavior when the evaluation indicates that it is not accomplishing what the software is intended to do, or when better functionality or performance is possible.”***

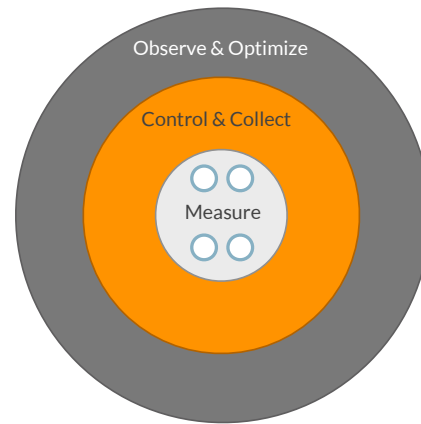
**DARPA**

# ***Feedback Loops***



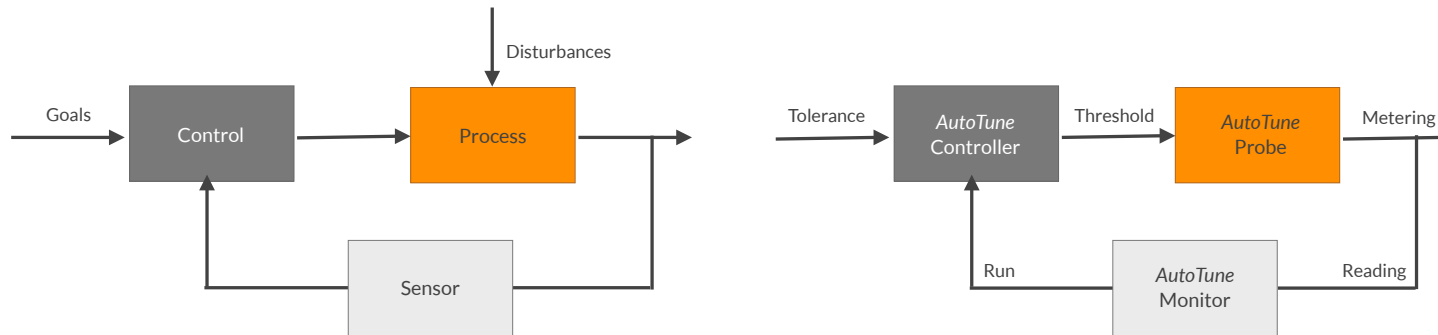
***Evidence, Relevance, Consequence, Action***

# ***Self Observation***



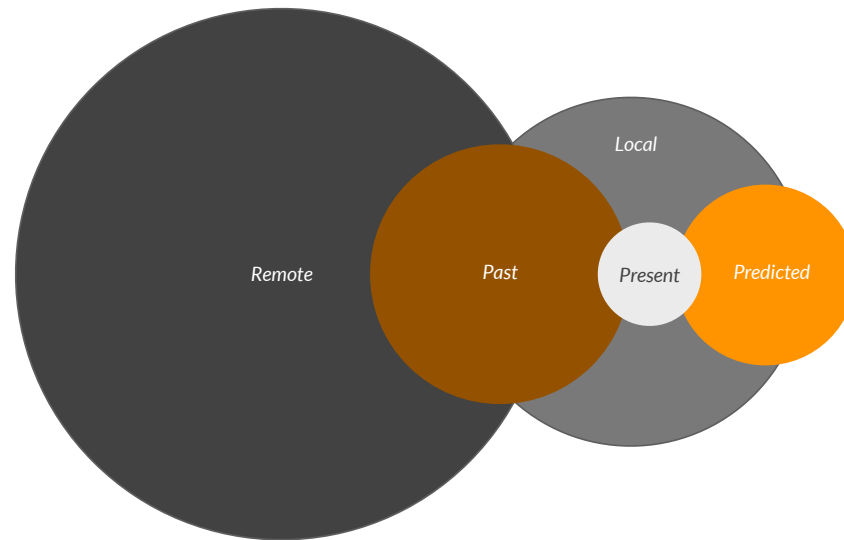
***Observe, Optimize, Control, Collect***

# ***Self Control***



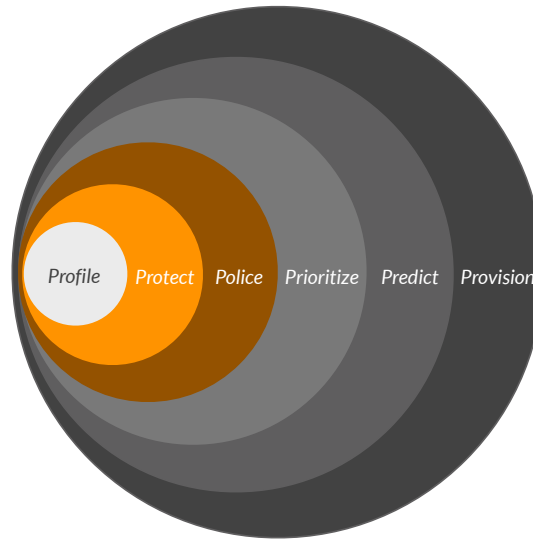
***Control, Process, Sensor***

# ***DIRT - Data in Realtime***



***Remote, Past, Local, Present, Predicted***

**99.9999% => PP.PPPP**



**Profile,Protect,Police,Prioritize,Predict,Provide**

# ***Profile***

***instrument, measure, and collect performance data for a significant portion of an application code base***

# ***Protect***

***supervisory routines that are embedded into the runtime  
to govern the execution flow & consumption of resources***

# ***Police***

***watching for non-compliant behavior & taking corrective measures to enforce call flow contracts and agreements***

# ***Prioritize***

***differentiating the quality of service that is offered to applications, activities and actors***

# ***Predict***

***controllers that use past & current in-flight data & models  
to predict behavior & consumption and then to optimize***

# ***Provide***

***resource capacity planning based on inflight activities & predicted their predicted requirements - in internet time***

# ***QoS for Networks***

***“The ability of the network to provide better or “special” service to a set of packets/dataflows to the detriment of other packets/dataflows”***

# **Traffic Characteristics**

## **Bandwidth**

*the number of bits per second that can be expected to be transmitted*

## **Delay**

*the elapse time between when a packet is first sent and when it arrives*

## **Jitter**

*the variation in the arrival rate, or delay introduced, of packets sent at a uniform rate*

## **Packet loss**

*all routers lose, drop or discard packets for a number of reasons*

# **Traffic Planning**

## **Identify**

*traffic and its requirements in terms of service levels*

## **Divide and Associate**

*traffic into classes of service levels*

## **Define**

*policies for each class of service level which cause some change in the traffic characteristic mentioned previously*

# ***Traffic Service Classification***

***differentiating one packet from another by examining fields inside of the packet's header***

*in some cases this classification is performed automatically and adaptively based on traffic patterns and resource consumption behavior*

# ***Traffic Congestion Management***

***using queues and schedulers which split traffic across queues and schedulers based on service classification associated with a packet or flow***

# ***Traffic Policing and Shaping***

## ***Policing***

*traffic contracts which define how much data can be sent*

## ***Shaping***

*smoothing the peaks & troughs of data transmission in order to optimize or guarantee performance & bandwidth.*

# ***QoS for Applications***

***“The ability of the runtime to provide better or “special” service to a set of calls/threads to the detriment of other calls/threads”.***

# **Call Characteristics**

## **Throughput**

*the number of requests/calls per second that can expect to receive a valid response*

## **Response Time**

*the time between a request/call is sent (received) & when it's response is received (sent)*

## **Response Time Variation**

*the variation in the response time of requests/calls*

## **Exceptions & Errors**

*faults can occur in runtimes such as timeouts caused by contention and corruption*

# **Call Planning**

## **Identify**

*request/call patterns and their requirements in terms of service levels*

## **Divide and Associate**

*request/call patterns into classes of service levels.*

## **Define**

*policies for each class of service level which cause some change in the request/call characteristic mentioned previously*

# ***Call Service Classification***

***differentiation is typically done by examining the execution context associated with a request/call***

# ***Call Congestion Management***

***managed using concurrency and control constructs such as semaphores/latches/locks which are generally backed by implicit or explicit queues consisting of parked threads during contention***

# ***Call Policing & Shaping***

## ***Policing***

*call contracts which define how many calls can be sent*

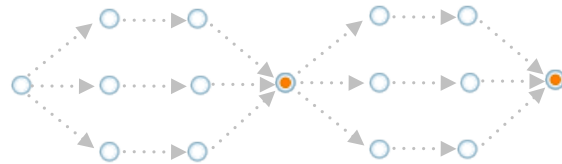
## ***Shaping***

*it tries to conform the call rate to the call contract using some form of delay mechanism such as waiting and parking*

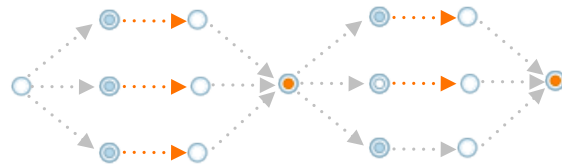
# ***Network-to-Application Mapping***

<b><i>Network</i></b>	<b><i>Application</i></b>
<b><i>Packet</i></b>	<b><i>Call</i></b>
<b><i>Flow</i></b>	<b><i>Thread</i></b>
<b><i>Router</i></b>	<b><i>Method</i></b>
<b><i>Datalink</i></b>	<b><i>Call Site</i></b>
<b><i>Buffer</i></b>	<b><i>Resource</i></b>
<b><i>Queue</i></b>	<b><i>Semaphore</i></b>

# The Application is the Network



- code execution point (method, call site, activity)
- bottleneck point (consumption, coordination, contention)
- code execution flow path



- ⊙ QoS service (re-)classification point
- ⊙ QoS reservation extension or (re-)prioritization point
- QoS queuing, prioritization, rate limiting

# ***Activity Based (Resource) Metering***

***metering has the potential to revolutionize the software design, development, testing, and management***

***a metering standard would define a model, concepts and mechanism which can be used across languages and processing boundaries to control and cost consumption***

# Methods of Measurement

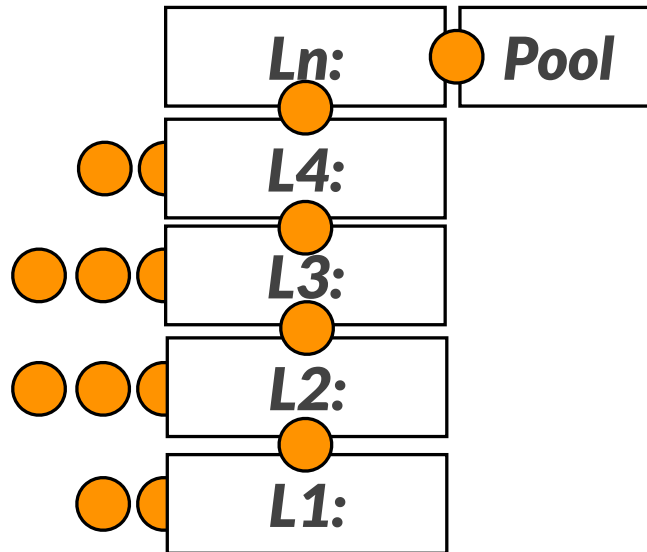
<b>Method</b>	<b>Metering</b>	<b>Metric</b>
<b>Analysis</b>	<b>Causality</b>	<b>Correlation</b>
<b>Accuracy</b>	<b>Event</b>	<b>Sampled</b>
<b>Allocation</b>	<b>Thread</b>	<b>Process</b>
<b>Assignment</b>	<b>Direct</b>	<b>Appportioned</b>

# ***Metering Measurement Model***

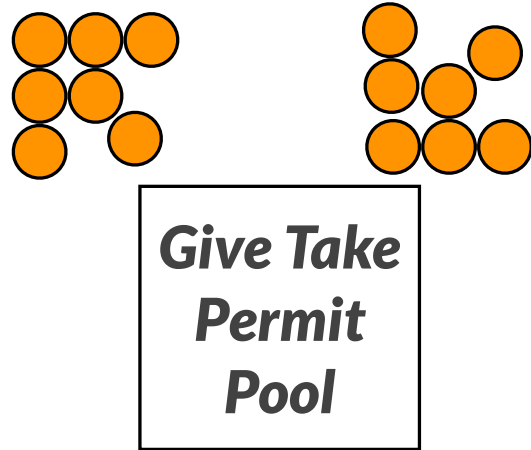
<b><i>Model</i></b>	<b><i>Activity</i></b>	<b><i>Resource</i></b>
<b><i>Device</i></b>	<b><i>Probe</i></b>	<b><i>Meter</i></b>
<b><i>Develop</i></b>	<b><i>Code</i></b>	<b><i>Counter</i></b>
<b><i>Design</i></b>	<b><i>Behavior</i></b>	<b><i>Usage</i></b>
<b><i>Data</i></b>	<b><i>Group</i></b>	<b><i>Metering</i></b>



# ***Demo: Prioritization***



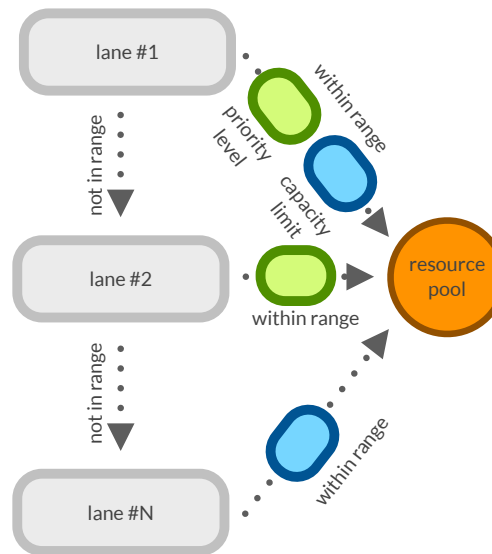
# ***Demo: Resource Reservation***



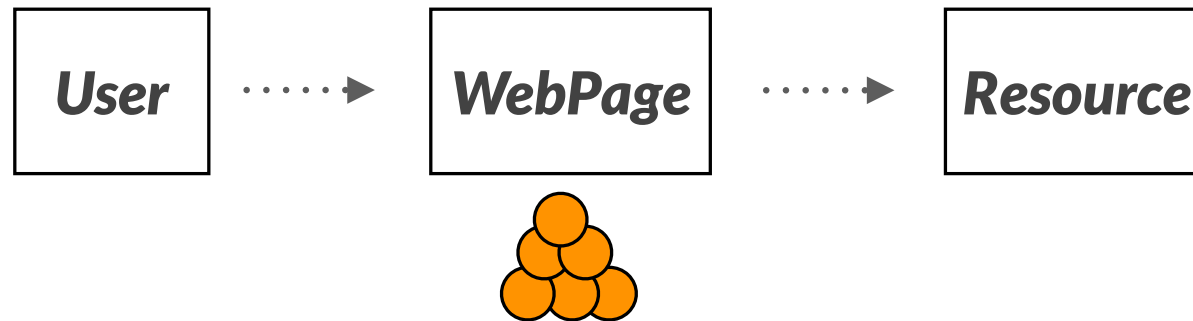
# ***Demo: Rate Limiting***



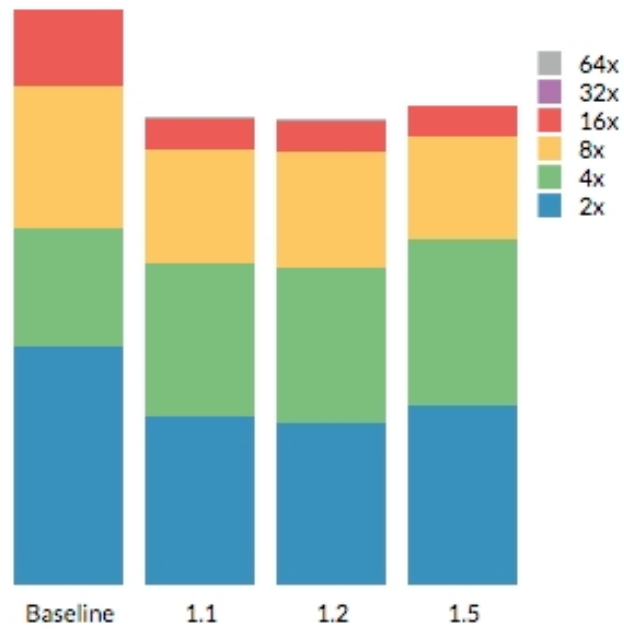
# QoS Reservation Lanes



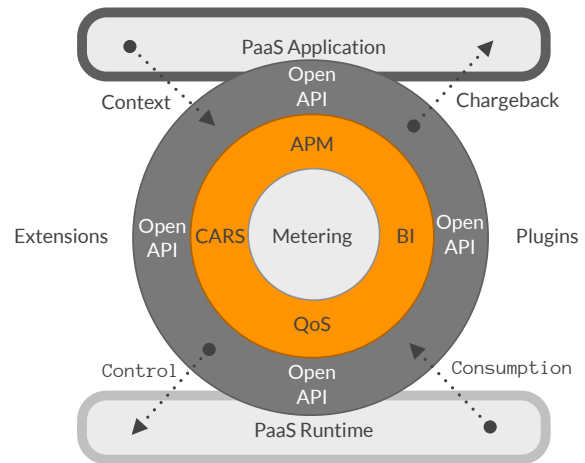
# ***Fairer Web Page Servicing with Performance Credits***



# ***Fairer Web Page Servicing with Performance Credits***

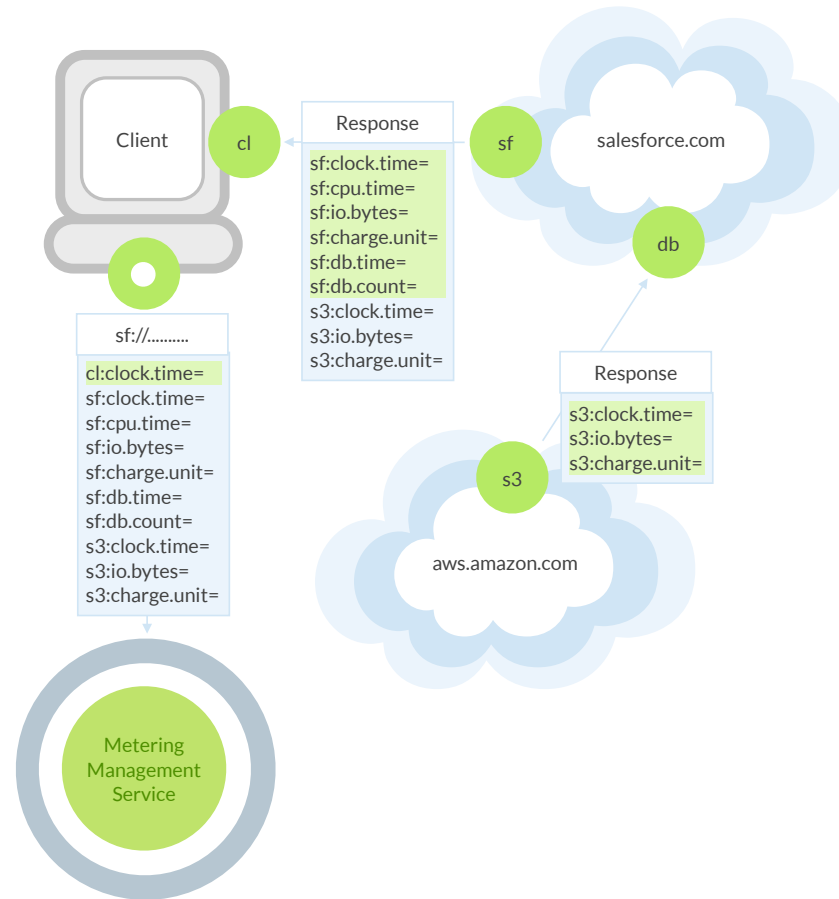


# A Cloud Cortex



APM: Application Performance Management  
CARS: Cost Aware Runtimes & Services  
BI: Business Intelligence  
QoS: Quality of Service

# Cloud Services



# ***More Information***

***<http://opencore.jinspired.com>***