Dynamic Deployment and Scalability for the Cloud

Jerome Bernard
Director, EMEA Operations
Elastic Grid, LLC.

Speaker's qualifications

- Jerome Bernard is a committer on Rio, Typica, JiBX and co-founder of Elastic Grid, LLC.
- Jerome Bernard speaks frequently on Cloud Computing
  - Recently: Devoxx, JavaZone, JavaOne, and the Open World Forum
- Jerome Bernard is working with many clients using EC2, from TV channels to specialized media processing companies.

Agenda

Introduction to Cloud Computing
Introduction to Amazon EC2
Introduction to Elastic Grid
Systems that never stop...

Introduction to Cloud Computing

Virtualization is used for consolidation.

Cloud Computing allow you to rent resources when they are needed.

- Why Cloud Computing?
  - Next logical step after virtualization
    - Better usage of your IT infrastructure
    - Cost Savings
  - Can your traditional hosting scale to thousands of machines in a week?
  - Can you afford spending huge amounts buying hardware if you only need it for a week?

Animoto Use Case
Company (US startup) creating cool videos based on a bunch of uploaded pictures. Really CPU intensive.
Went from dozen of servers up to 3500 servers in a few days when their application was released on Facebook. But went down to a few hundreds after another week.
How would you cope with that situation in a few days? Would you be able to raise money from VCs, buy the hardware, have the dealer send you the machine, install them and put them in a datacenter in just a few days?
What would do a week after with all the servers you don't need anymore?

Introduction to Cloud Computing

- Which Cloud Computing flavor?
  - Software as a Service (SaaS)
  - Platform as a Service (PaaS)
  - Infrastructure as a Service (IaaS)

IaaS: you rent some infrastructure -> some servers

PaaS: you rent access to a platform hosting your applications.

- References
  - SaaS: SalesForce, Facebook, LinkedIn
  - PaaS: SalesForce (EC2), Google App Engine, Microsoft Azure
  - IaaS: Amazon EC2, GoGrid, Flexiscale

Introduction to Cloud Computing

- Google App Engine
  - Make use of BigTable and Memcache
  - Integrate with Google Accounts
  - But in Python only...
- Microsoft Azure
  - Mostly for Windows and .Net solutions
  - Pricing model yet unclear

PaaS vs IaaS

- PaaS Pros
  - Usually easier to use than IaaS
  - Integrate with specific environments (Google, Microsoft Live, SalesForce, etc.)

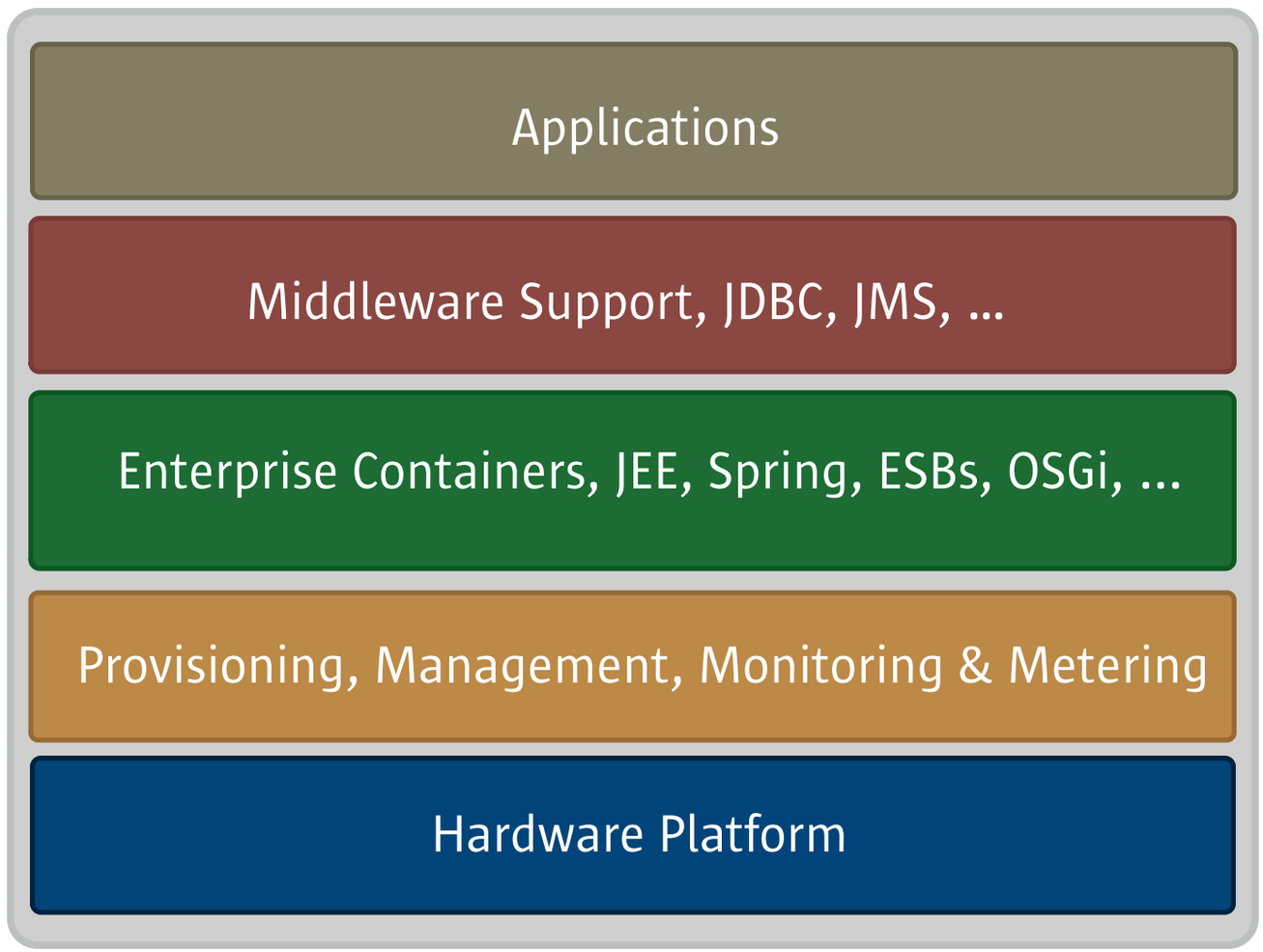
- PaaS Cons
  - Less/No control over the Infrastructure
  - Languages/Services chosen by the provider
  - Vendor Lock-in

Introduction to Amazon EC2

- Amazon EC2 is Infrastructure as a Service (IaaS)
  - Rent a server on a per hour base (from $.10 to $.80)
  - Many Operating Systems (Linux, Solaris, Windows)
- EC2 Amazon Machine Image (AMI)
  - Operating and system stack
  - Deployed to Amazon S3 (cheap storage)
- EC2 instances
  - Virtual machines that run AMI

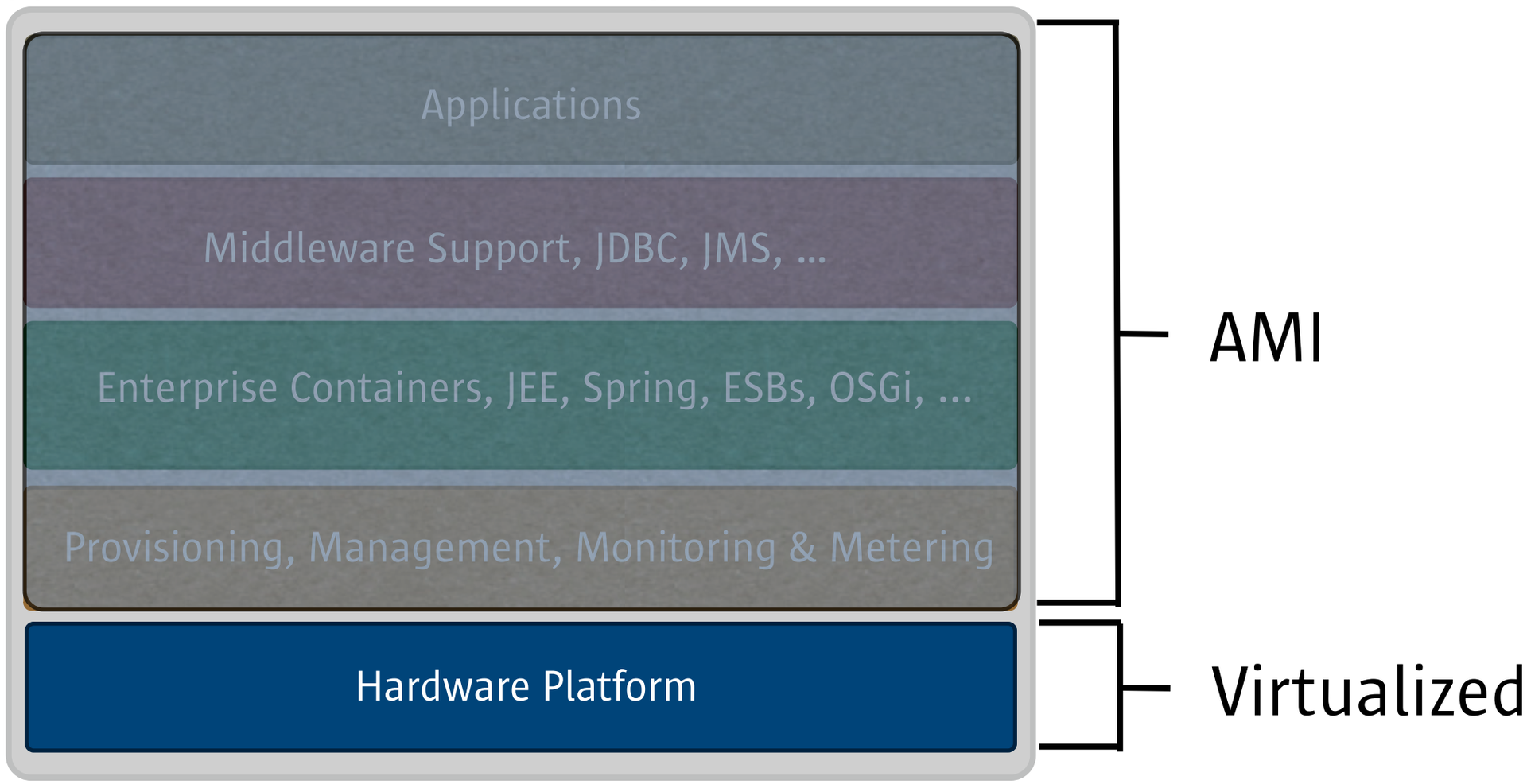
Introduction to Amazon EC2

- Typical Architecture Taxonomy



Introduction to Amazon EC2

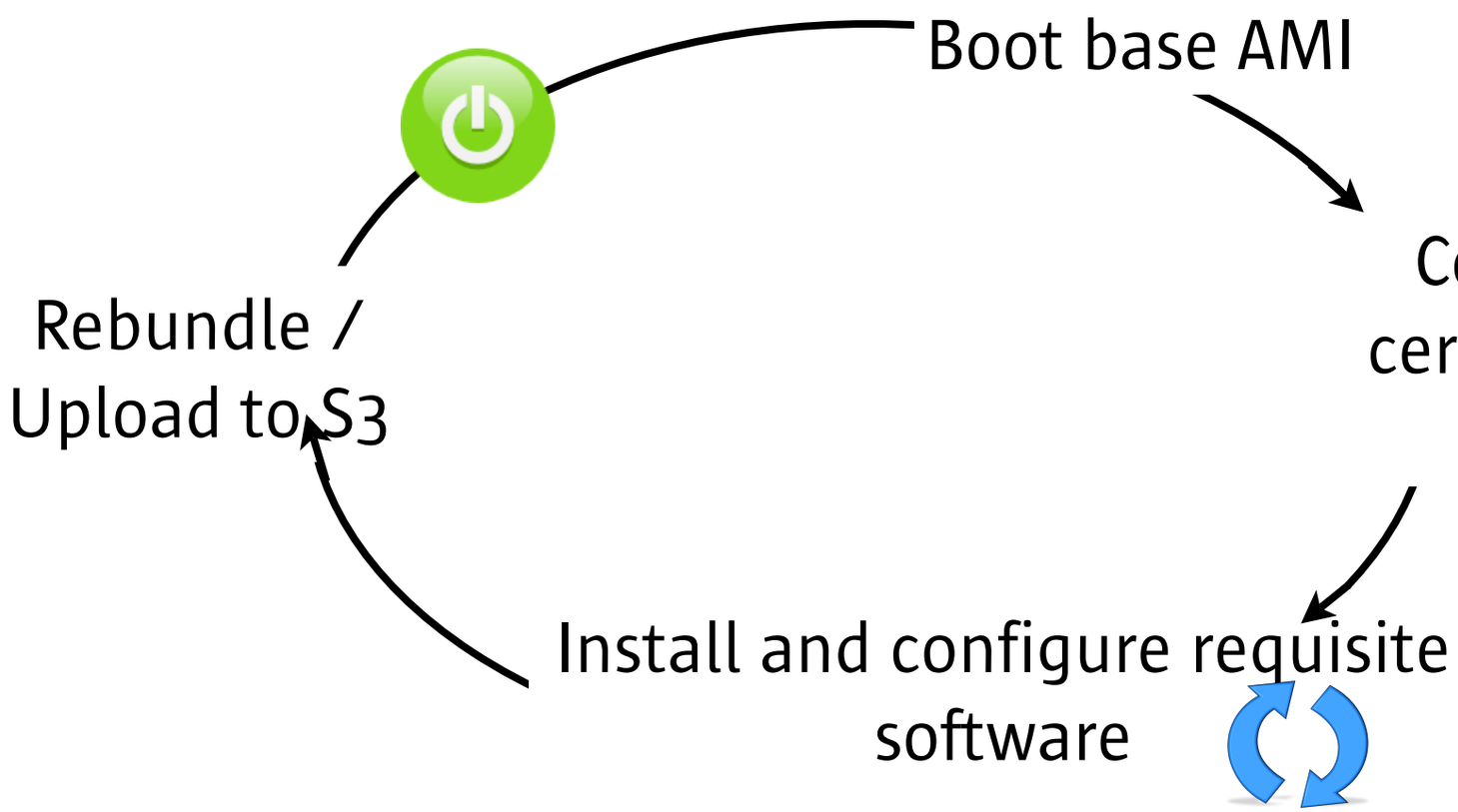
- Typical Architecture Taxonomy



Amazon EC2 pitfalls

- EC2 AMI Challenges
  - The EC2 AMI is a boot image, requires substantial system administrator knowledge
  - As application code changes, AMIs typically need to change / be re-bundled

Not focused on developer productivity. And actually you need to do this work twice: once for 32bits AMI and once for 64 bits.



Amazon EC2 pitfalls (continued)

- Infrastructure challenges
  - Networking: no multicast but this is what most Java framework uses for clustering (JGroups, Shoal, etc.)
  - Backup: the local filesystem has no durability guarantee
  - Significant boot latencies of EC2 instances (can be several minutes)
  - Failures: you have to design your application to be resilient to EC2 instance failures. Anyway you should always do so :-)

Amazon EC2 Advice

I/O are way better and you benefit from durability, snapshot supports, etc.

- Some AWS Advice
  - I/O: prefer an Elastic Block Storage (EBS) volume to a local filesystem
  - Snapshot EBS volumes periodically (incremental backup) but export to S3 for complete backups
  - Choose the right instance type
    - Don't use Small for production!
    - Don't choose based on disk space (think EBS)
    - Choose based on available memory and CPU virtual cores

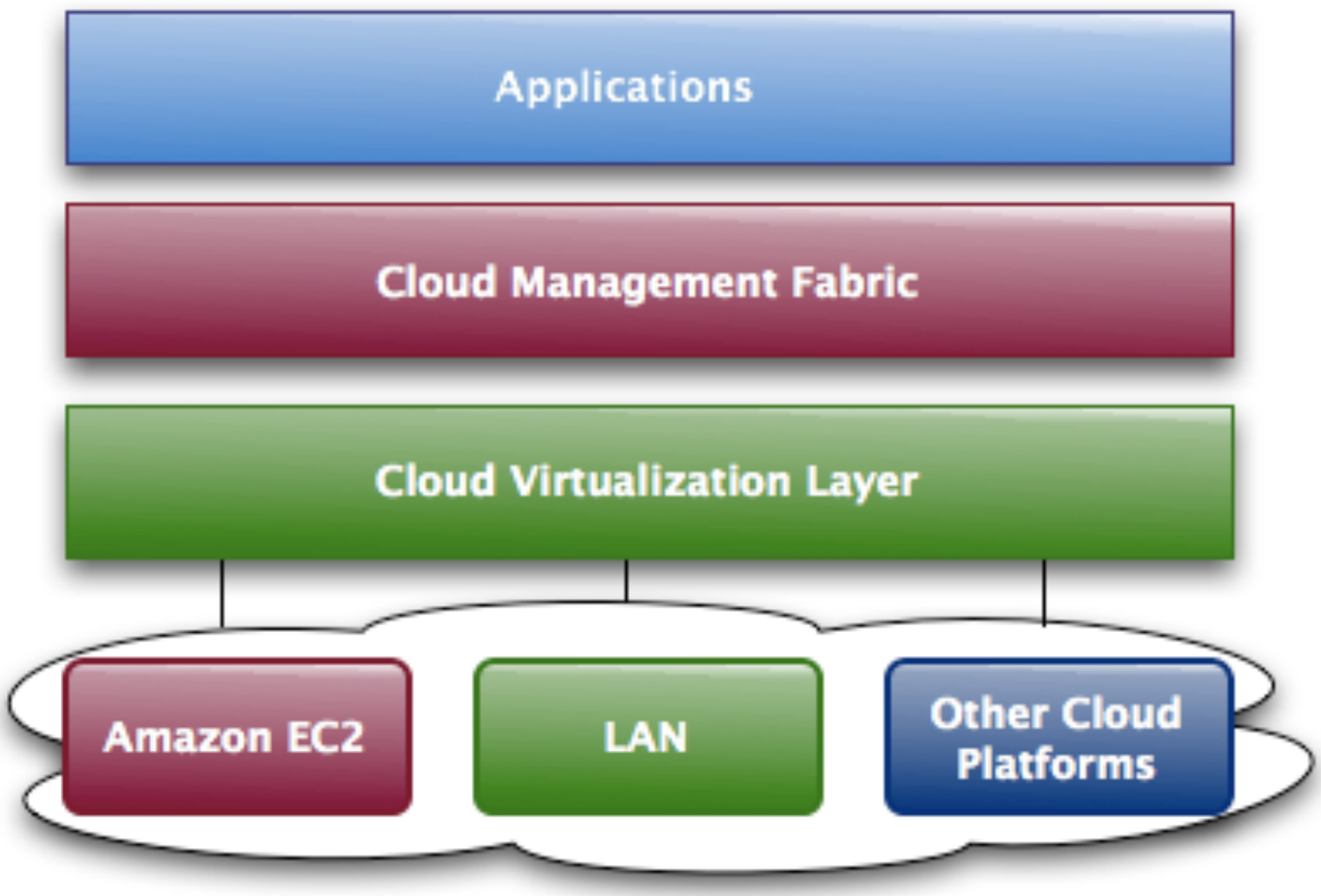
High CPU Medium is the best tradeoff usually unless you need a lot of memory.

Introduction to Elastic Grid

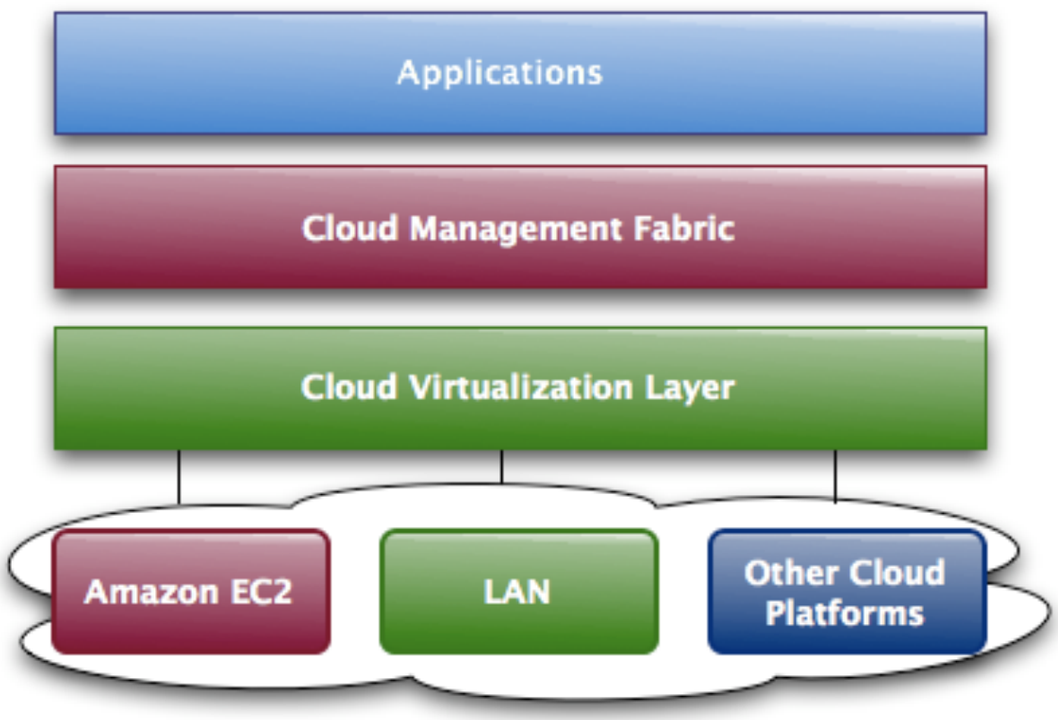
- Elastic Grid (abbreviated as EG)
  - Project initiated in early '08
  - AGPLv3 license
  - Part of the OW2 community

- Elastic Grid, LLC. founded in May '08
  - Team:
    - Dennis Reedy: Director US Operations
    - Jerome Bernard: we already went through this :-)

Introduction to Elastic Grid



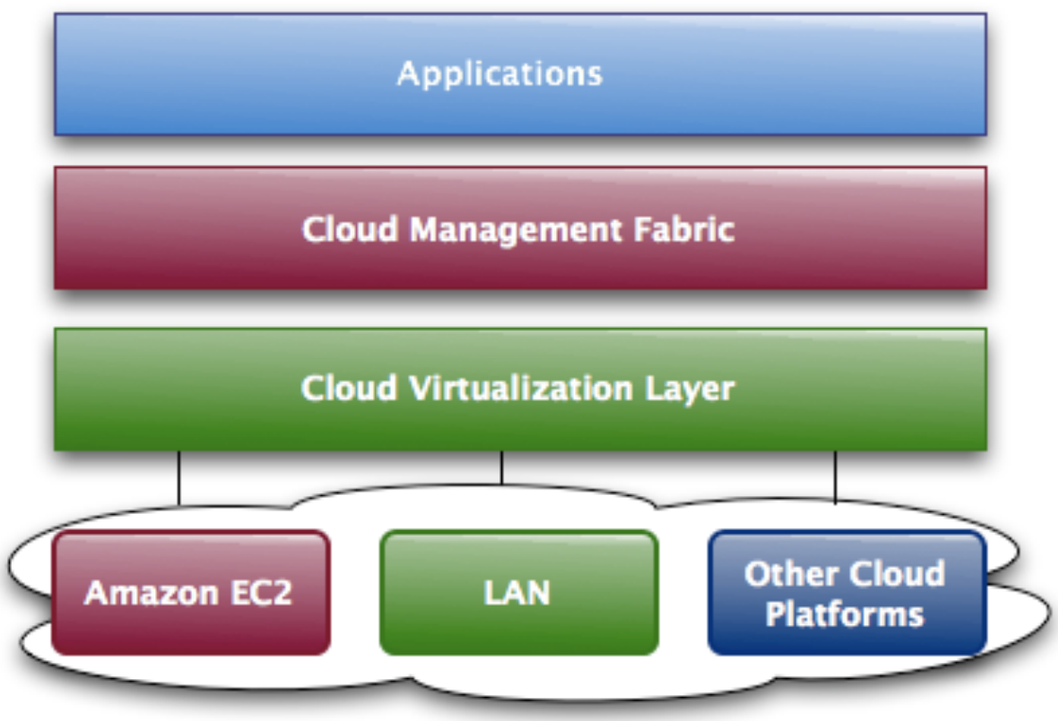
Introduction to Elastic Grid



Cloud Management Fabric

- Provides an adaptive capability to dynamically instantiate, monitor & manage application components
- The deployment provides context on service requirements, dependencies, associations and operational parameters
- Provisioning services additionally provides pluggable download distribution and resource

Introduction to Elastic Grid



Cloud Virtualization Layer

- Abstracts specific Cloud Computing provider technology
- Allows portability across specific implementations

- You can deploy on:
  - Private Cloud
  - Amazon EC2
  - More to come soon...

Sound Familiar?

- The network is reliable
- Latency is zero
- Bandwidth is infinite
- The network is secure
- Topology doesn't change
- There is one competent administrator
- Transport costs are zero
- The network is homogenous

"*Essentially everyone, when they first build a distributed application, makes the following assumptions. All prove to be false in the long-run and all cause big trouble and painful learning experiences*".

Peter Deutsch - "Deutsch's 8 Fallacies of Networking"

Service Level Agreements in the Cloud

- The SLA that Cloud Computing Providers enable
  - Machine availability
  - Disk
  - Network...

- The SLA(s) that you must provide for your application
  - Meeting performance objectives
  - Adapting to failure
  - Deployment of new features
  - Application fault detection and recovery

Application SLAs

- Enable visibility of critical metrics
  - System
    - CPU, Memory, Disk...
  - Infrastructure
    - Threads, Heap, Garbage Collection
    - Queue depths, Pool sizes, ...
  - Application
    - Response times
    - Wait times
    - Others...

EG Focus: Application SLAs

- Visibility is a start, but behavior is key!
- EG focuses on a policy based approach
  - Deployment Policies
    - Whether a compute resource can support the requirements of a service
  - Behavioral Policies
    - Whether a service is operating to specified objective(s)
    - SLA Management, Service Associations, Dynamic system state
  - Reachability Policies
    - Heuristics determining whether a service is available on the network

EG Focus: Application SLAs

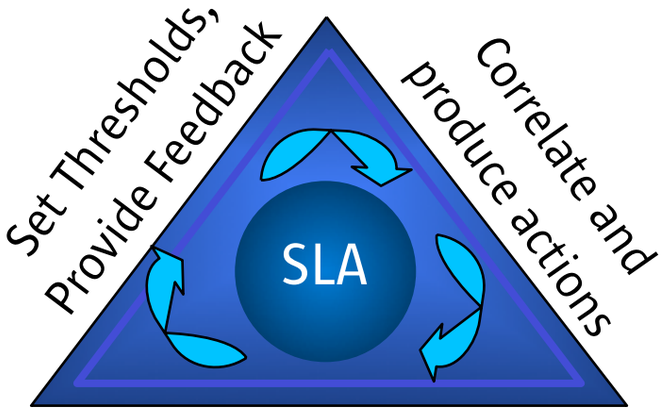
- Specified in Elastic Grid Deployment Descriptors: non-intrusive with your code
- Provides selection of the best machine where to deploy the services based on your requirements
- Provides active monitoring of SLAs with many strategies like service relocation, provisioning of additional EC2 instances, provisioning of additional service instances, etc.

EG Focus: Application SLAs

- Dynamic Deployment
  - Push application resources to the cloud and dynamically deploy, or ...
  - Can be into CI based approach, or ...
  - Cloud burst, or ...
- Green Computing and Cost Savings 
  - When the load decreases, EG will unprovision your unneeded services instances and servers

Behavioral Policies

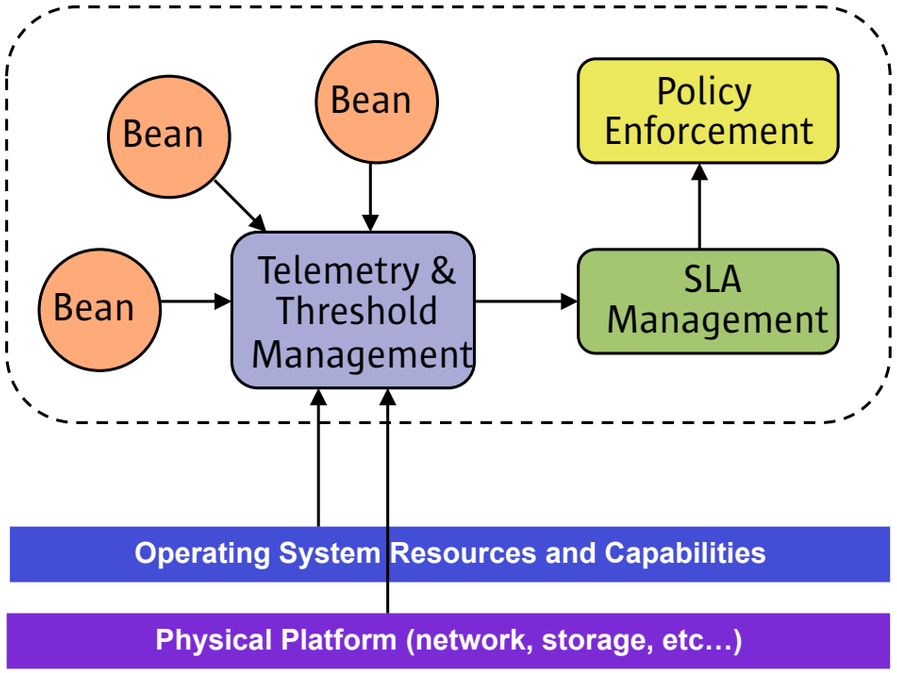
- Compute resources have capabilities
  - CPU, Disk, Connectivity, Bandwidth, etc...
  - Software components need to run on most appropriate compute resource based on definable criteria
  - Feedback mechanism to subscribe to changes to quantitative QoS mechanisms
  - Provide a resource utilization approach to measure compute resource



Measure, Collect &
Respond to administrative control & actions

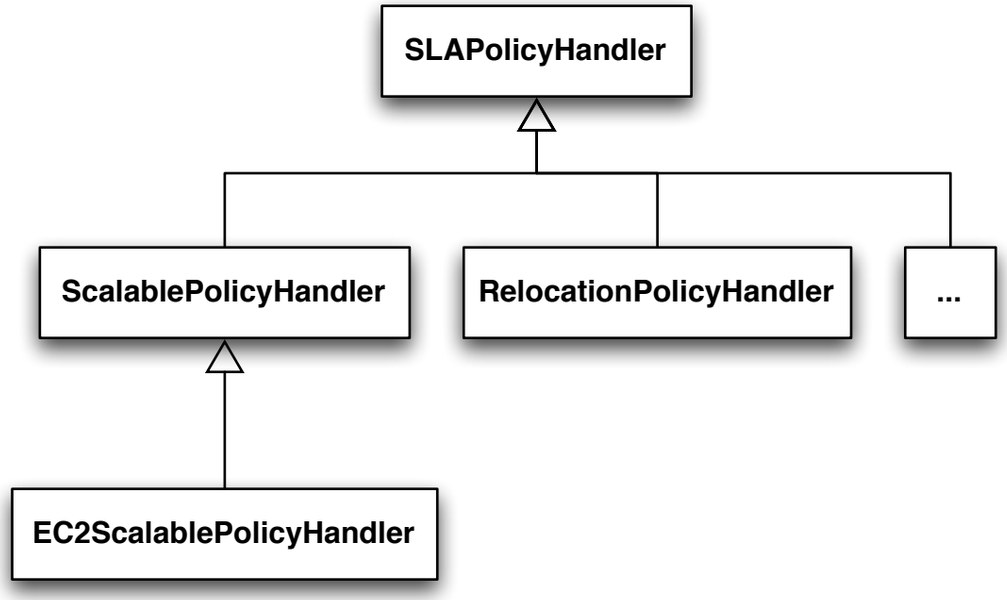
Autonomic SLA

- Sensor-effect pattern
- Data is observed from applications, OS, hardware, etc. and measured against declared thresholds
- Policy enforcement can happen locally, distributed or hierarchically
- SLA threshold events are fired to registered consumers
- Each SLA is Autonomic



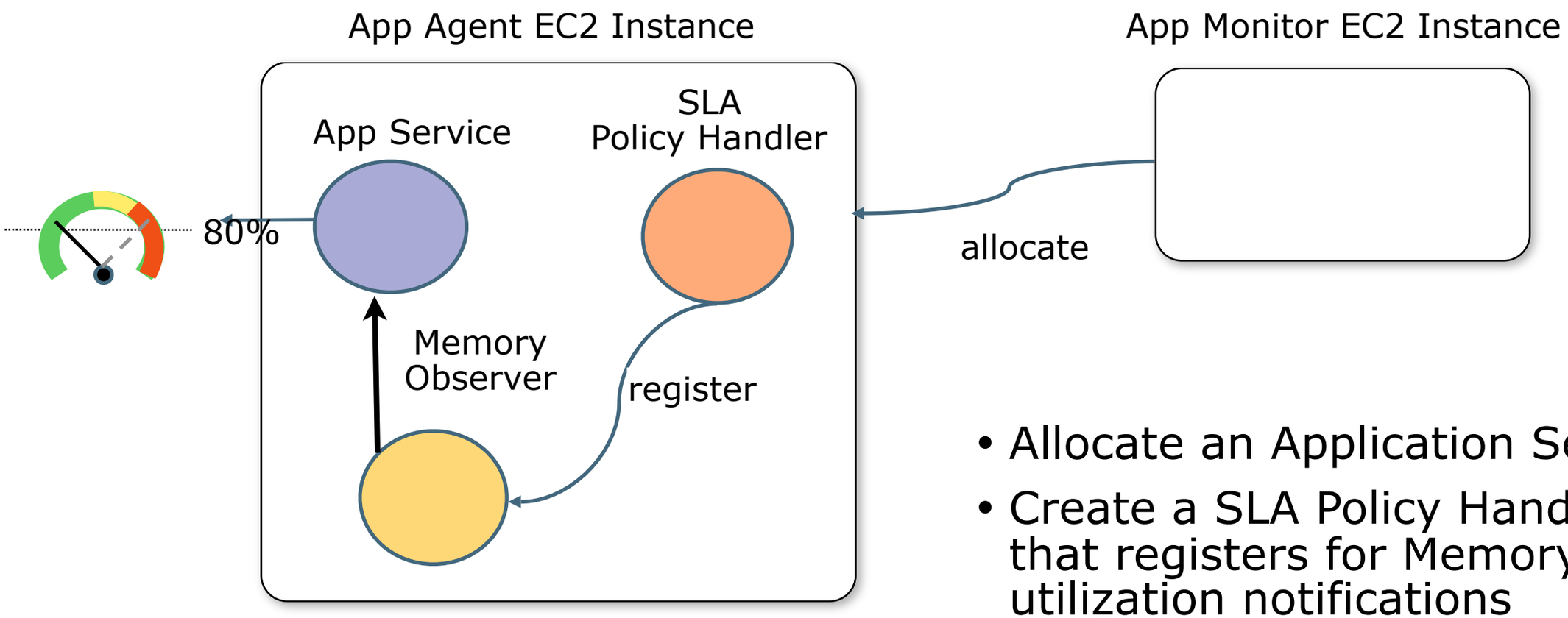
SLA Policy Handlers

- SLA Policy Handlers are policy enforcement points, providing if-then-else logic required to deal with the problem set they know about
- Are associated to Watches, and are notified of out-of-bound conditions
- Extendable model, development of more sophisticated handlers is encouraged



Elastic Grid Scalability on EC2

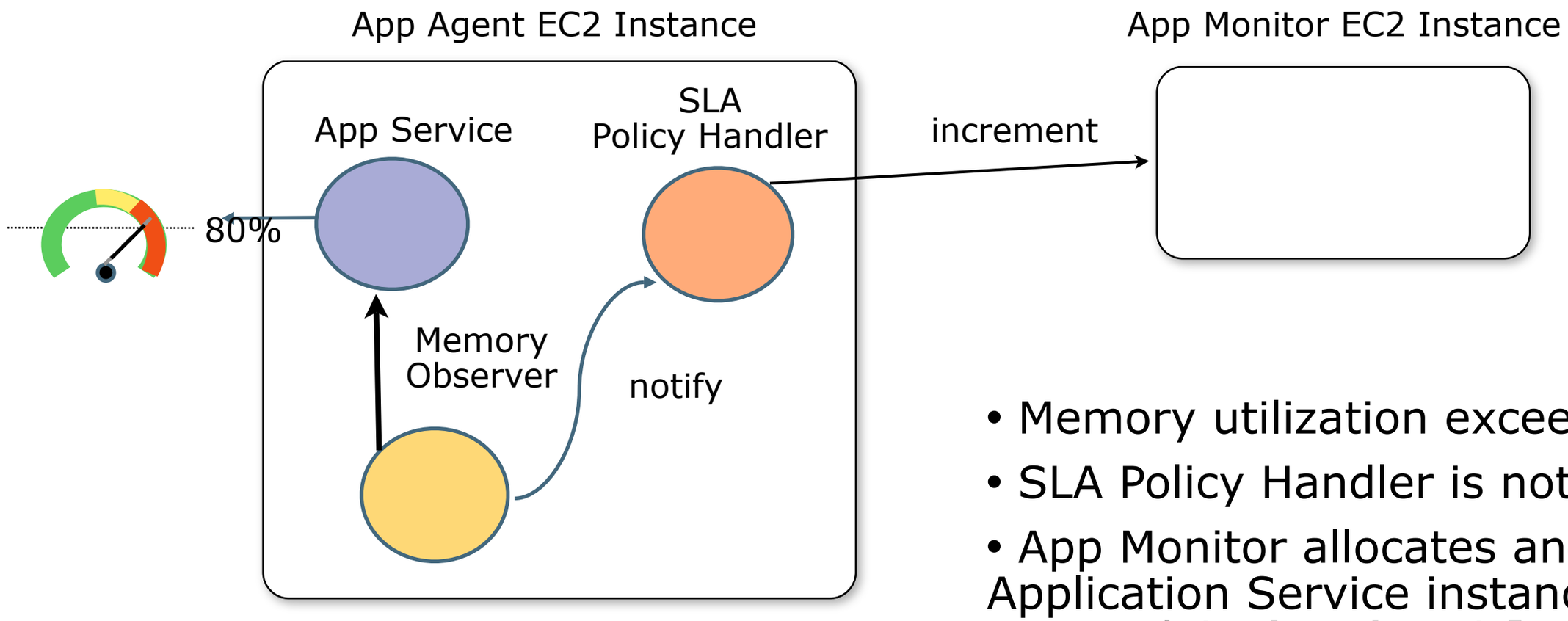
- Across existing EC2 instances



- Allocate an Application Service
- Create a SLA Policy Handler that registers for Memory utilization notifications
- SLA has upper limit set to 80%

Elastic Grid Scalability on EC2

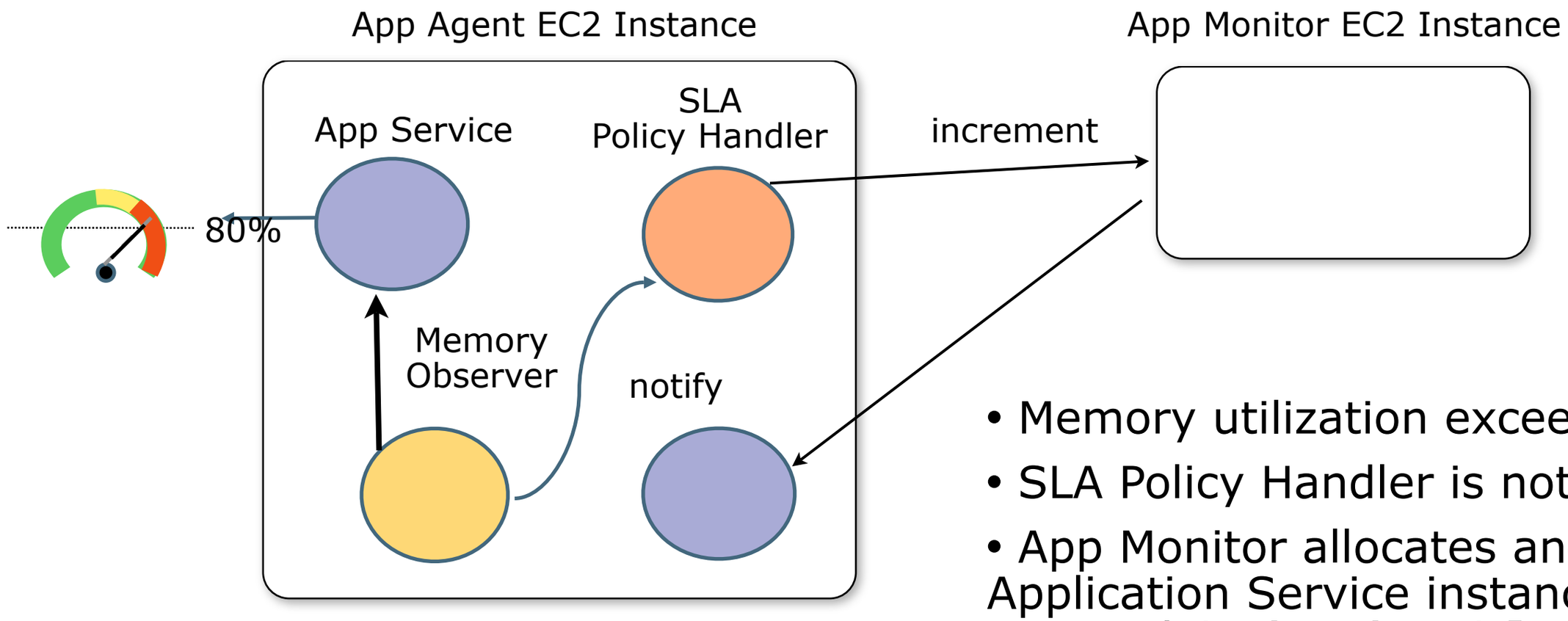
- Across existing EC2 instances



- Memory utilization exceeds 80%
- SLA Policy Handler is notified
- App Monitor allocates another Application Service instance to appropriate App Agent Instance

Elastic Grid Scalability on EC2

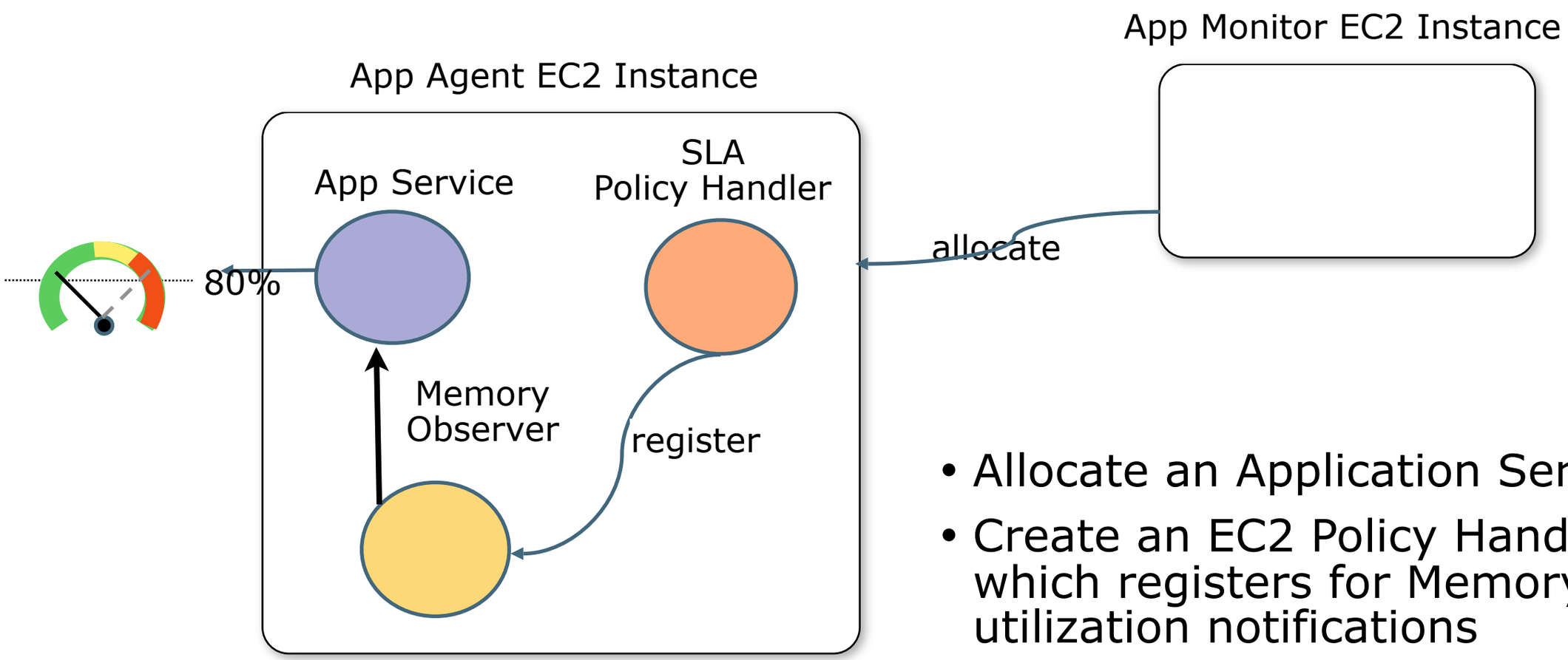
- Across existing EC2 instances



- Memory utilization exceeds 80%
- SLA Policy Handler is notified
- App Monitor allocates another Application Service instance to appropriate App Agent Instance

Elastic Grid Scalability on EC2

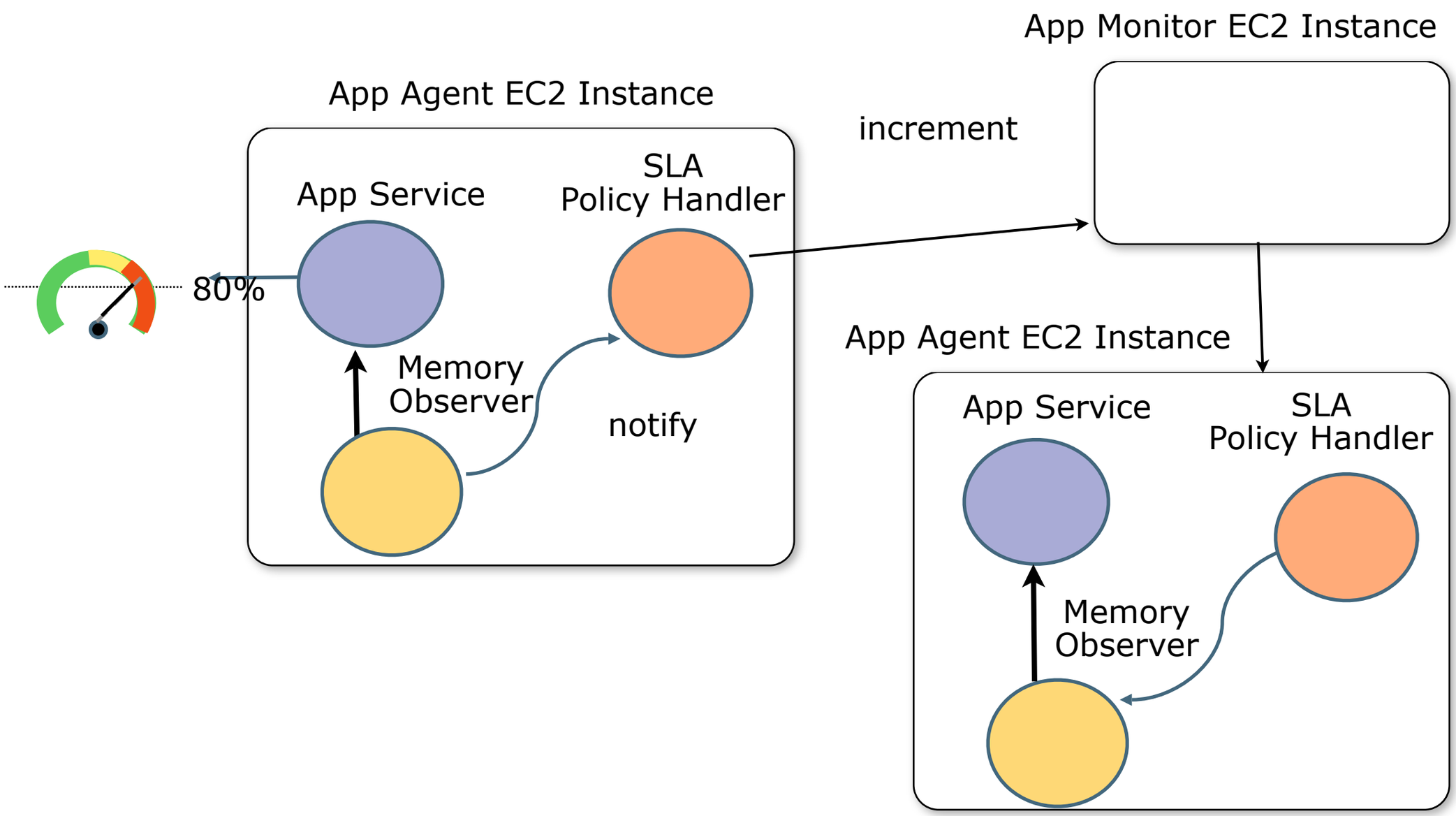
- New EC2 instance



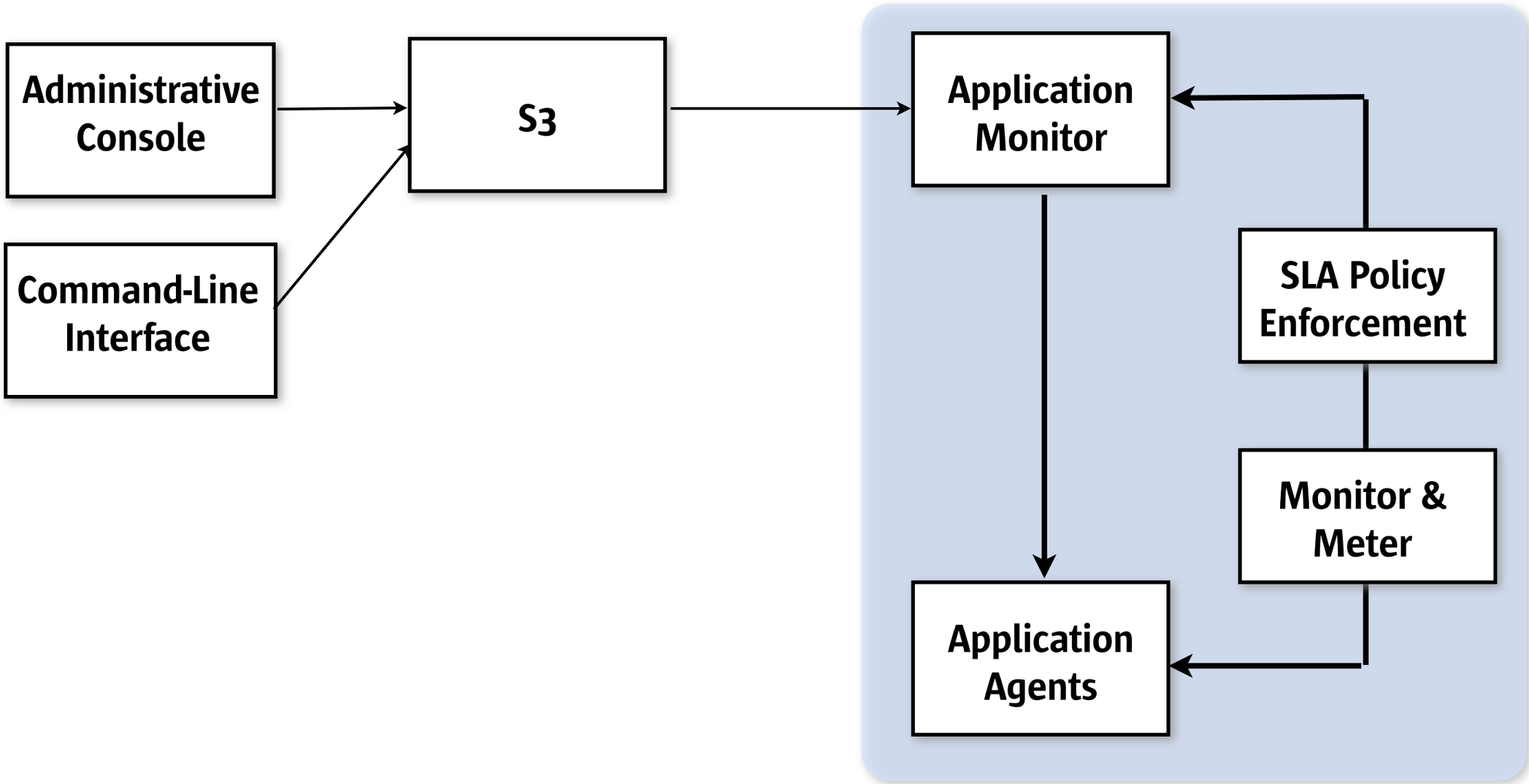
- Allocate an Application Service
- Create an EC2 Policy Handler which registers for Memory utilization notifications
- SLA has upper limit set to 80%

Elastic Grid Scalability on EC2

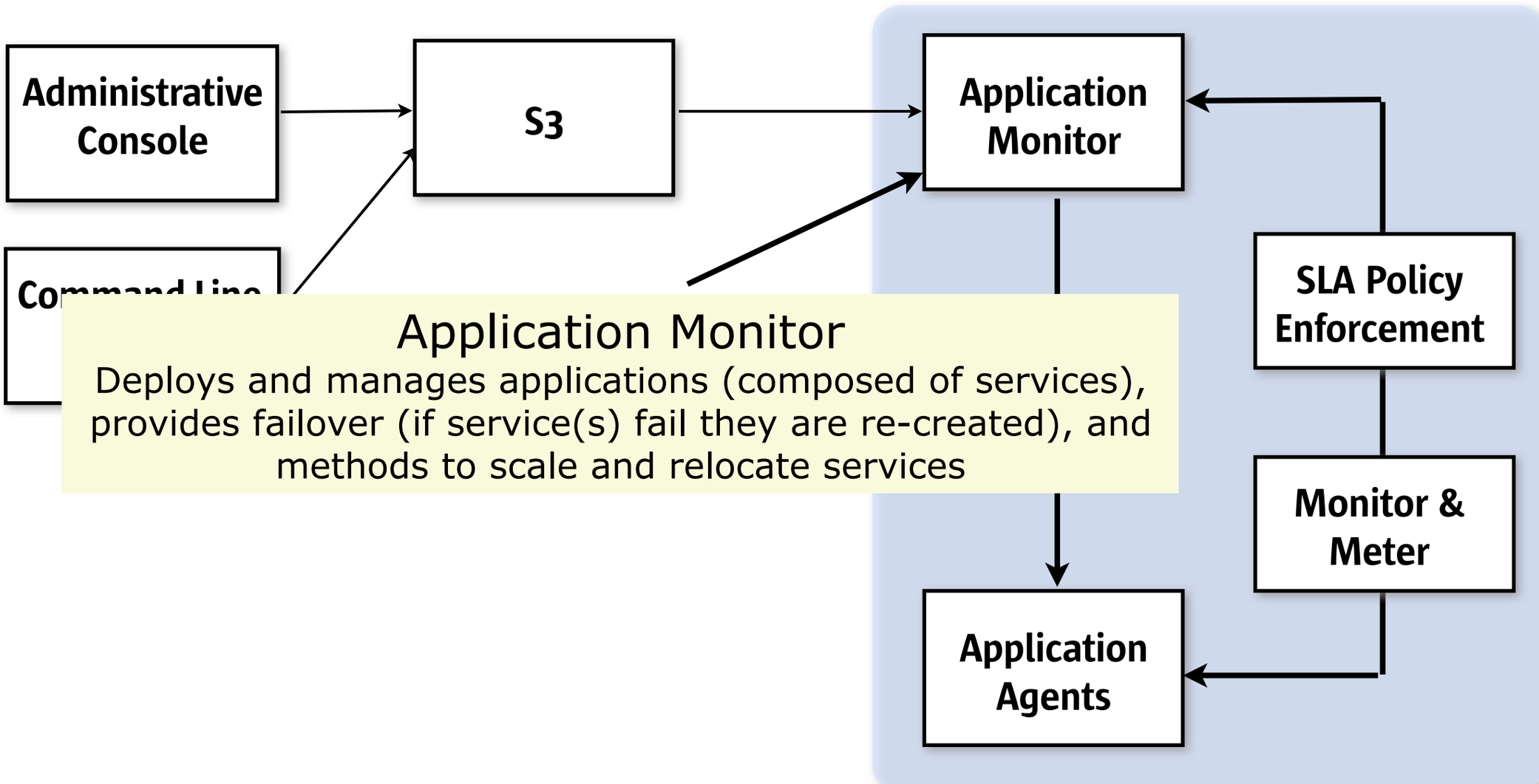
- New EC2 instance



Elastic Grid Architecture



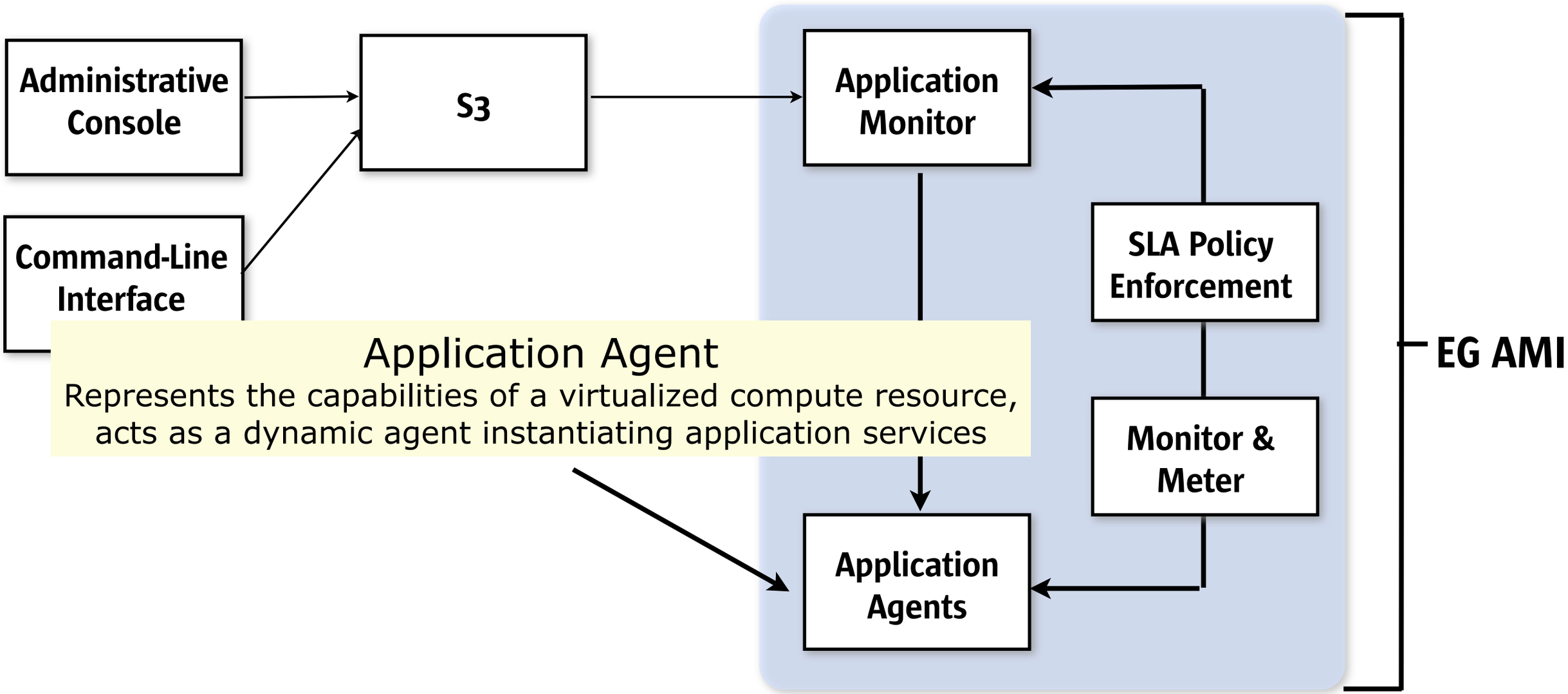
Elastic Grid Architecture



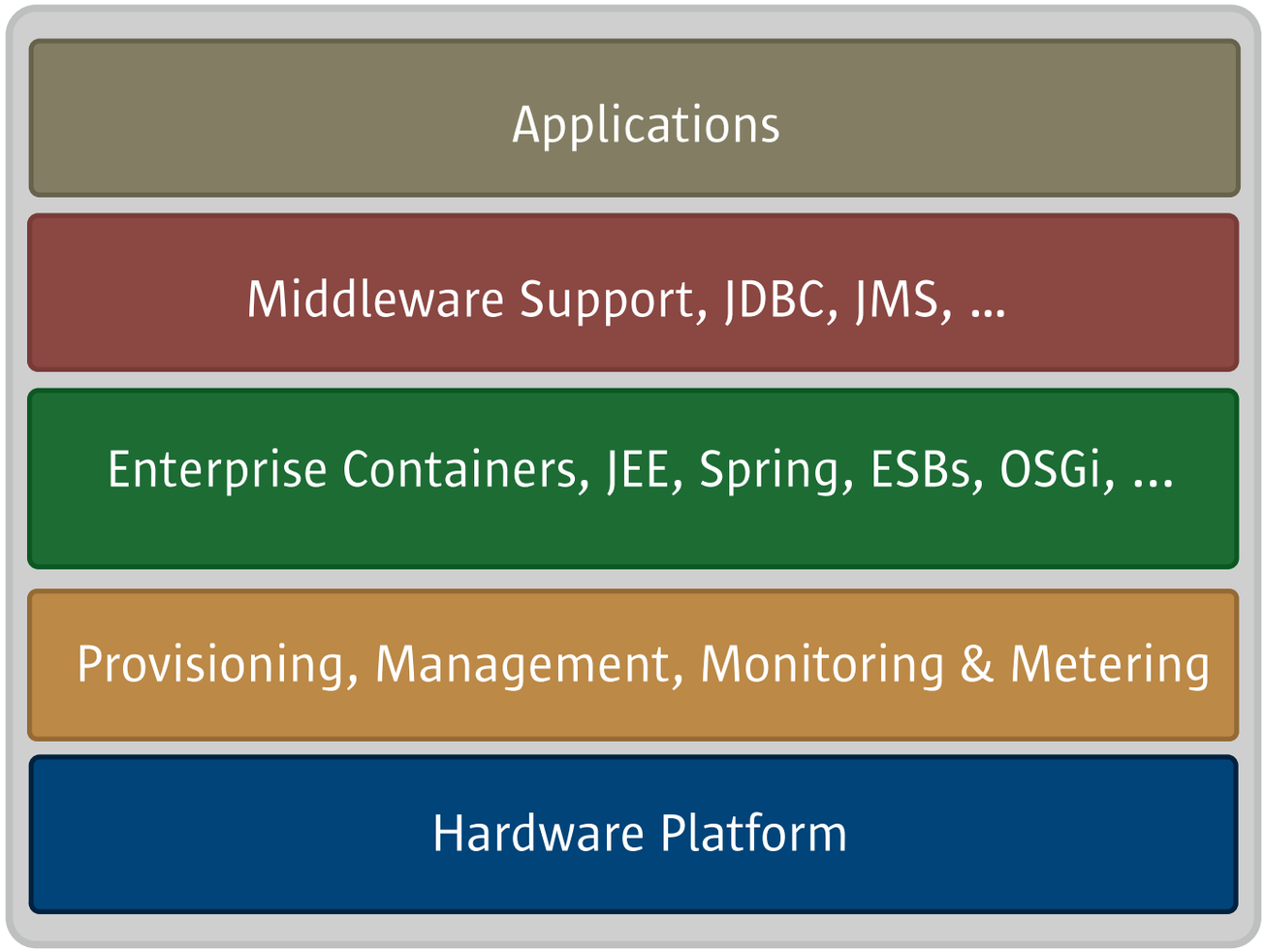
Application Monitor

Deploys and manages applications (composed of services), provides failover (if service(s) fail they are re-created), and methods to scale and relocate services

Elastic Grid Architecture



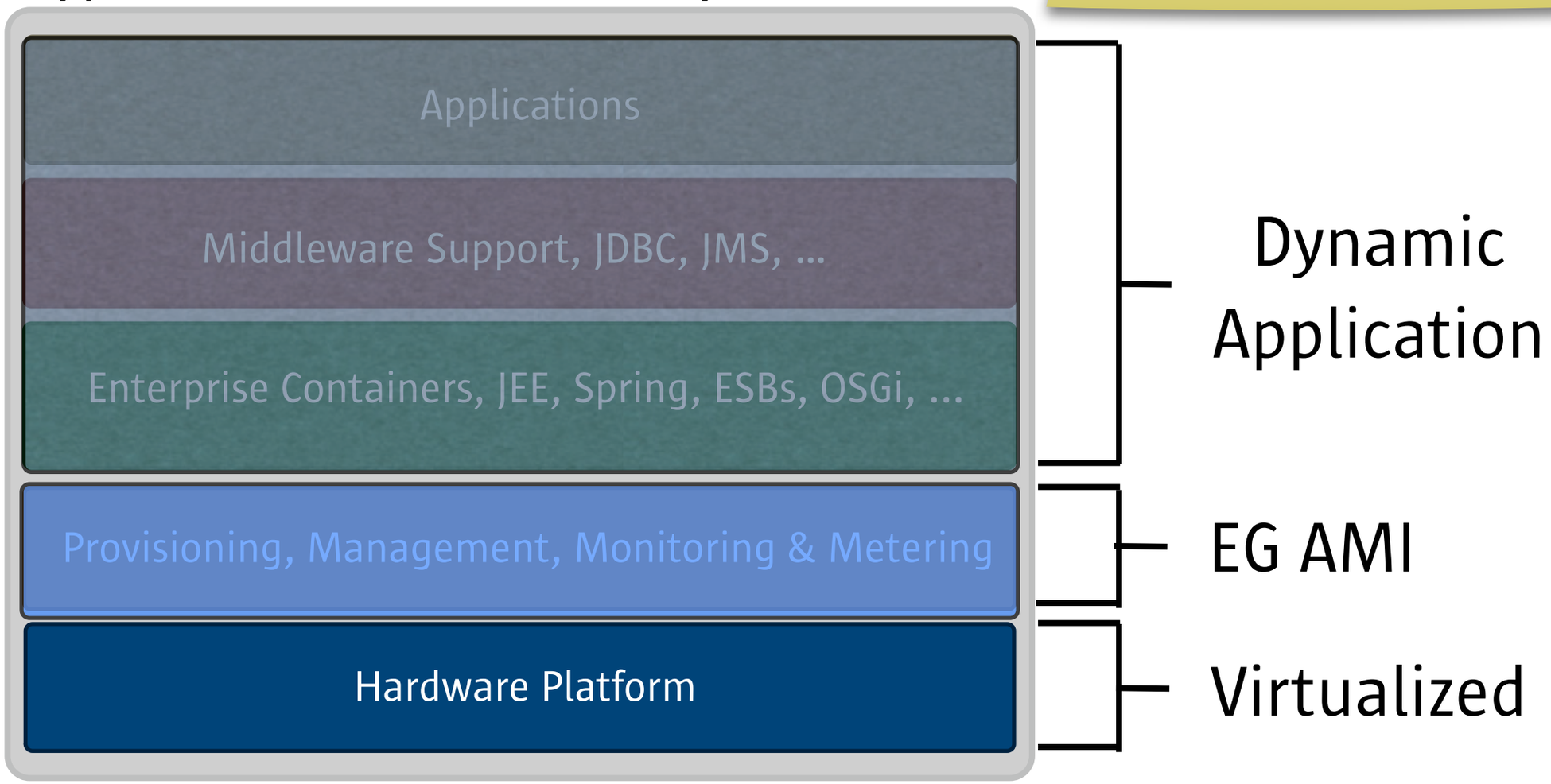
- Typical Architecture Taxonomy



elasticgrid

You won't have to create AMIs anymore and of course, you won't have to update an AMI when your application code changes!

- Typical Architecture Taxonomy

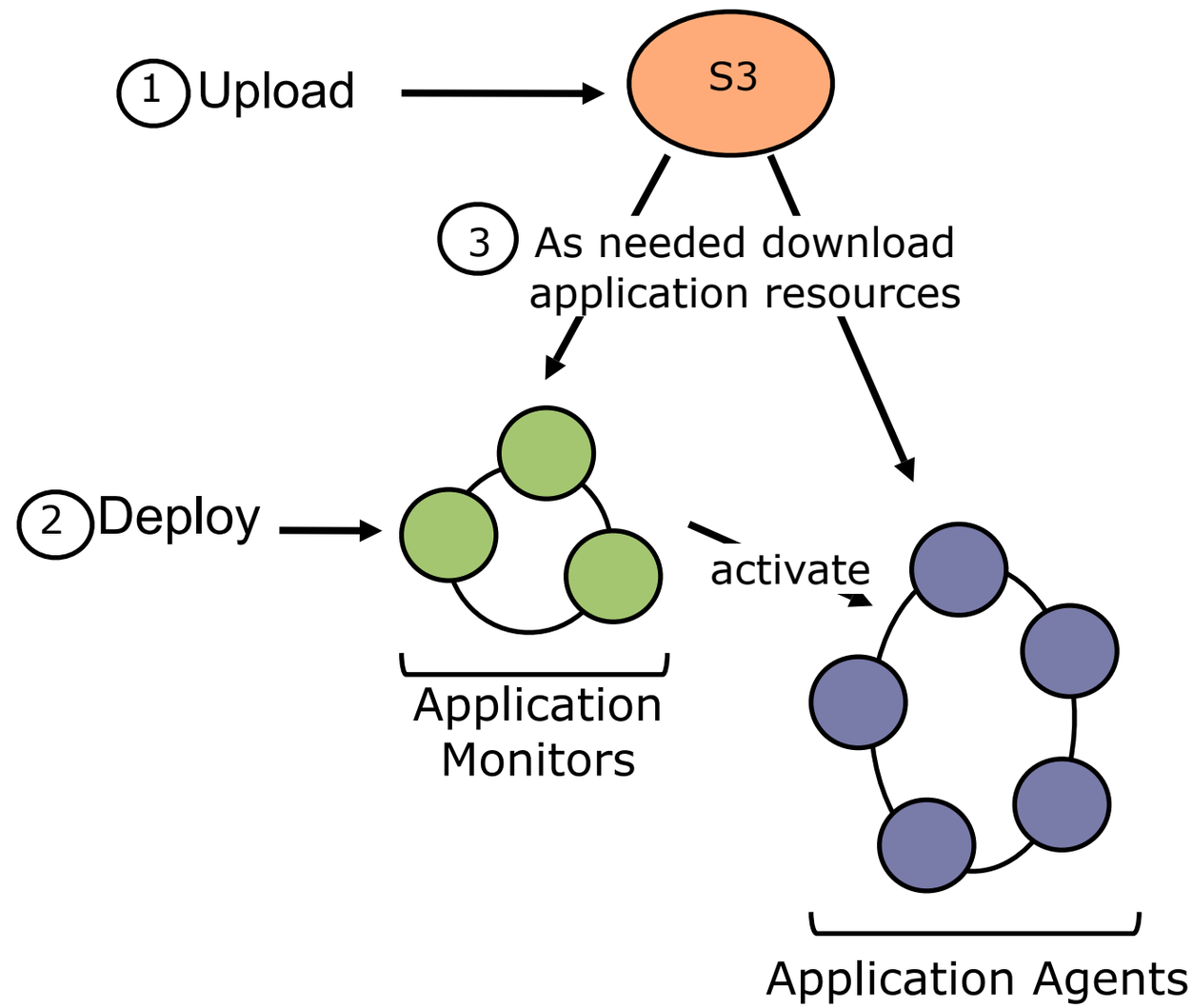


Deployments with Elastic Grid

- EG AMIs are pre-set, no need to (re-)bundle
- As application code changes, upload to S3 and deploy
- Focuses on developer productivity

Elastic Grid Deployments

- Deploy application code to S3
- Run the deploy command
- All code is dynamically served and instantiated
- Application is monitored and managed across EC2 instances



Back to our QCon track...

- How do you develop, test, update, maintain, and reason about systems without borders?

Systems Without Borders

- How do you **Develop**, Test, Update, Maintain, and Reason about systems without borders?
- With the Elastic Grid Cloud Management Fabric, you can simply develop on your machine or a LAN
- Elastic Grid provides virtualization layer allowing LAN deployment to behave in a similar way that Cloud Deployment behaves.

Systems Without Borders

- How do you Develop, **Test**, Update, Maintain, and Reason about systems without borders?
- Elastic Grid provides a mocked agent you can use for tests
- It's easier with the Cloud: you simply start a bunch of servers and qualify your app on it
  - We are actually working on a solution with would simply run a bunch of JUnit/TestNG tests on the Cloud for you: start some servers, run the tests, collect the results, stop the servers.

Systems Without Borders

- How do you Develop, **Test**, Update, Maintain, and Reason about systems without borders?

- Case Study:
  - US Army Research Labs uses Elastic Grid and EC2 to test the concurrency of a highly parallelized distributed system in the cloud.
  - EG provisions EC2 instances, dynamically deploys and scales the system across the cloud, providing the continuous deployment needed for assurance evaluation.

Systems Without Borders

- How do you Develop, Test, **Update**, Maintain, and Reason about systems without borders?

- Did I say that if you ask Elastic Grid to deploy an updated deployment descriptor of a currently running application, that EG will figure out what should be started and/or stopped and won't touch the running instances if you still need them?

Systems Without Borders

- How do you Develop, Test, Update, **Maintain**, and Reason about systems without borders?
- SLAs are Elastic Grid answer to this issue
- It's all a matter of declaring how your application/ system should react!

Systems Without Borders

- How do you Develop, Test, Update, Maintain, and **Reason** about systems without borders?
- Elastic Grid provides hook that you can use so that you keep a record of everything
- We are working on integration with rules engine / CEP engine which would allow to fine-tune the if-then-else behavior depending on what happened before

Summary

- So what does Elastic Grid do for the app?
  - Ease development and deployment of Java applications using Amazon services
  - Provides automated management, fault detection and scalability for the application
  - To be available soon: Cloud Bursting!
- Why you should use Elastic Grid?
  - Avoid Cloud Computing platforms pitfalls
  - Focus on development, not infrastructure

Q&A

Thanks for your attention!

Elastic Grid Website: http://www.elastic-grid.com
Elastic Grid Blog: http://blog.elastic-grid.com
Elastic Grid Wiki: http://wiki.elastic-grid.com