

# The Siamese Twins of IT Infrastructure: Grid and Virtualization

Ravi Subramaniam

*Intel Corporation*

Open Grid Forum

February 26<sup>th</sup>, 2008

***NOTE: This presentation was made earlier at OGF-21 (Seattle)  
on October 16<sup>th</sup>, 2007***

# Agenda

- Discuss virtualization
- Contrast Grids approach to virtualization and “virtualization” approach to virtualization
- Discuss impact to OGF
- Call to action

Finish this proverb

*“Give a man a fish and you feed him for a day;*

*Teach him how to fish ....*

*and you feed him for a lifetime” drink beer  
all day”*

What you choose may be driven by biases (familiarity or culture)?

***More importantly how you formulate is important – embody more concepts***

# What is virtualization?

Some definitions:



Virtualization is the creation of a [virtual](#) (rather than actual) version of something, such as an [operating system](#), a [server](#), a storage device or network resources

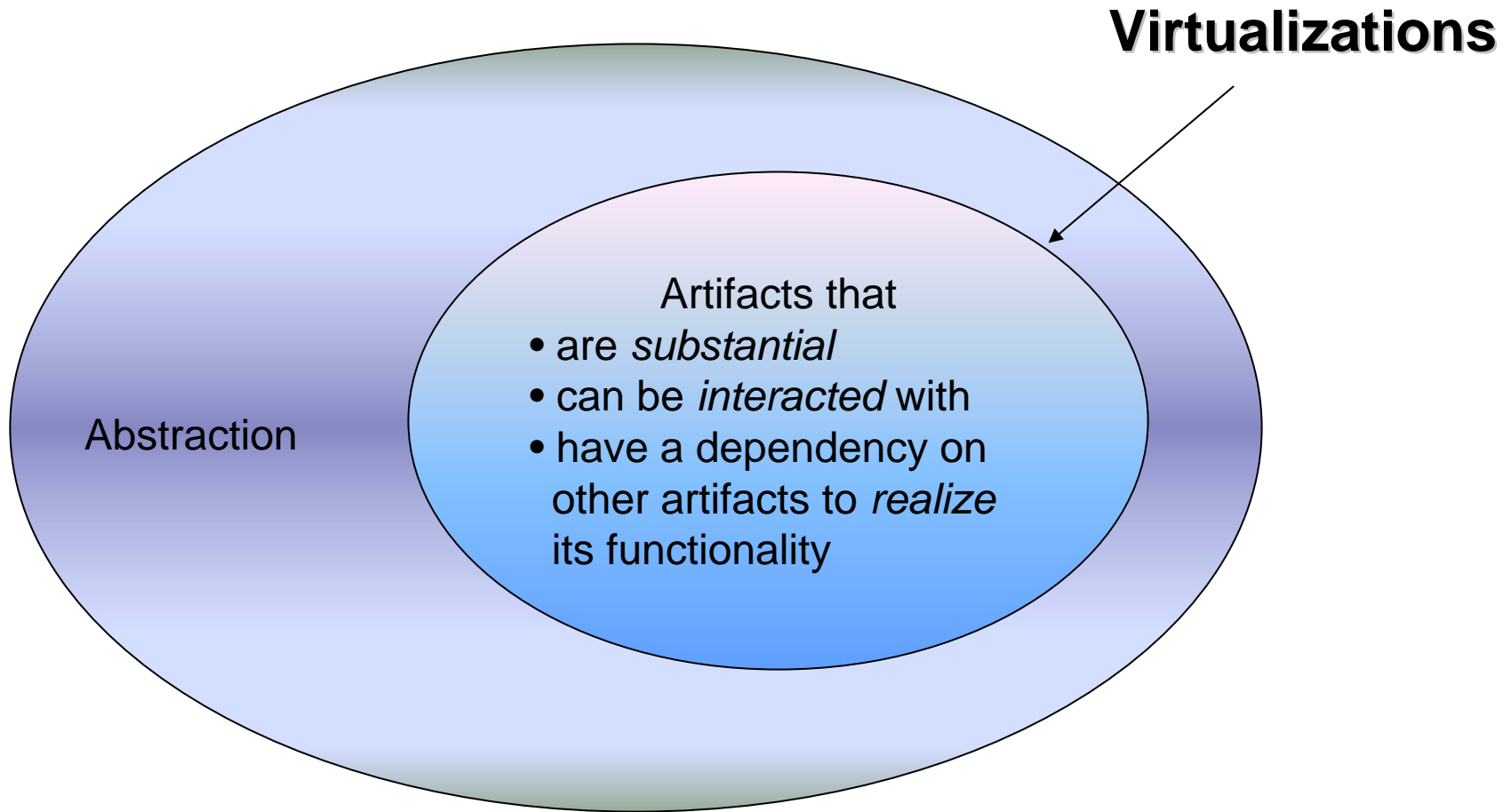


The deployment of a virtual technique

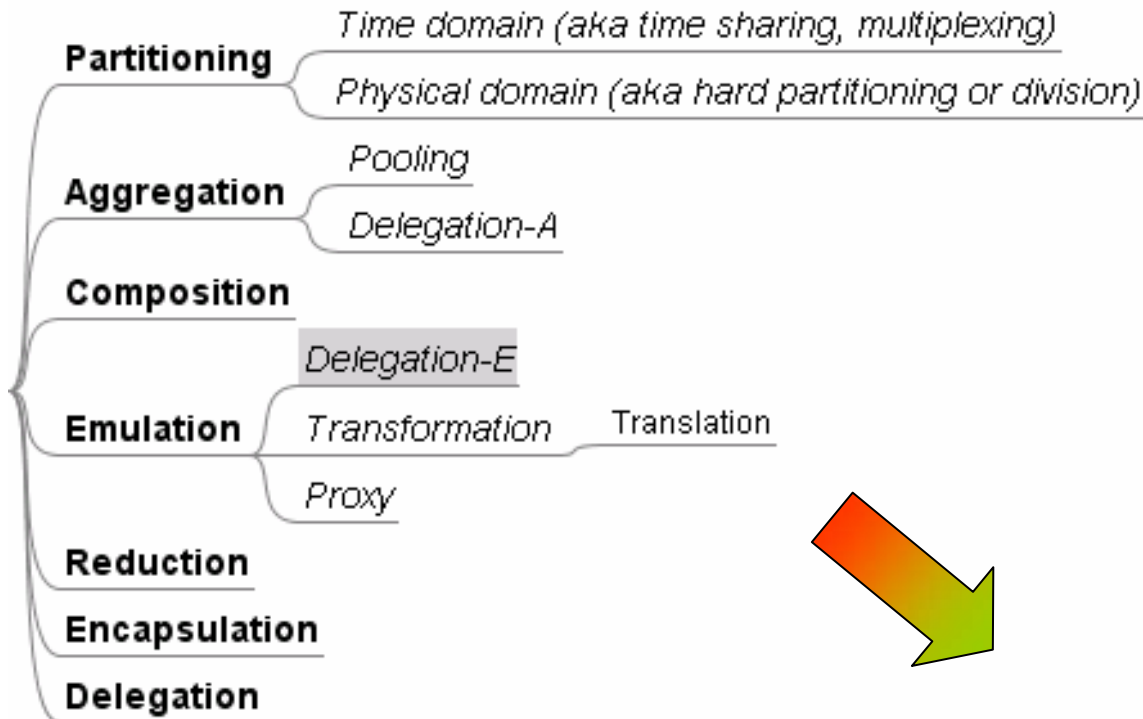


Virtualization is a technique for hiding the physical characteristics of computing resources to simplify the way in which other systems, applications, or end users interact with those resources

# What are the important elements of virtualizations?



# How do I create virtualizations?



- **Machine virtualization** (Partitioning, Composition, Encapsulation)
  - **Application virtualization** (Reduction, Partitioning, Aggregation)
  - **Storage virtualization** (Partitioning, Aggregation)
  - **Interface virtualization** (Reduction) and **Function virtualization** (Encapsulation)
- SOA** {

# Results of virtualization



- De-coupling
- Separation
- Isolation/Insulation
- Simplification
- Encapsulation
- Re-use

**Simplicity at  
component level**

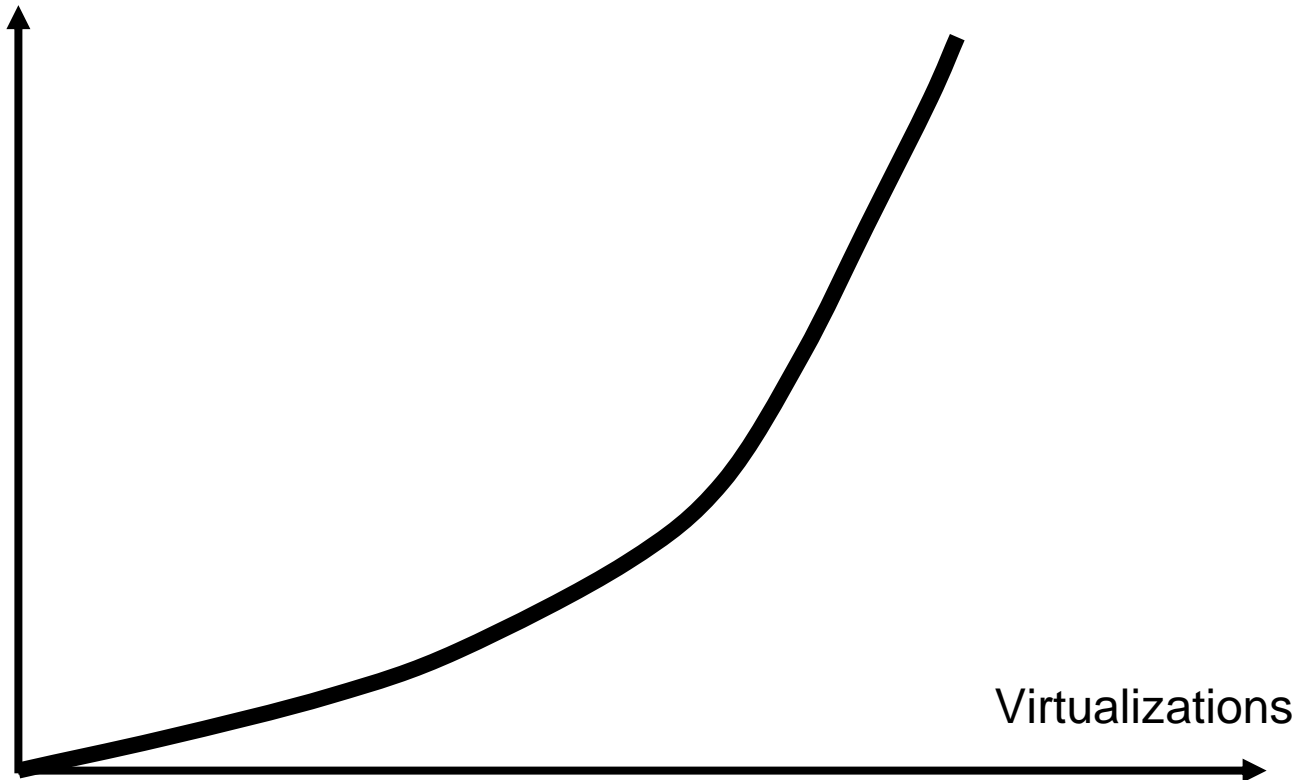


- Increased Associations
- Requires explicit bindings
- Instance and association management
- Performance
- Contextualization

**Complexity at  
system level**

# Virtualization -> Complexity at large scale

Complexity  
(in  
Associations,  
Management  
etc)

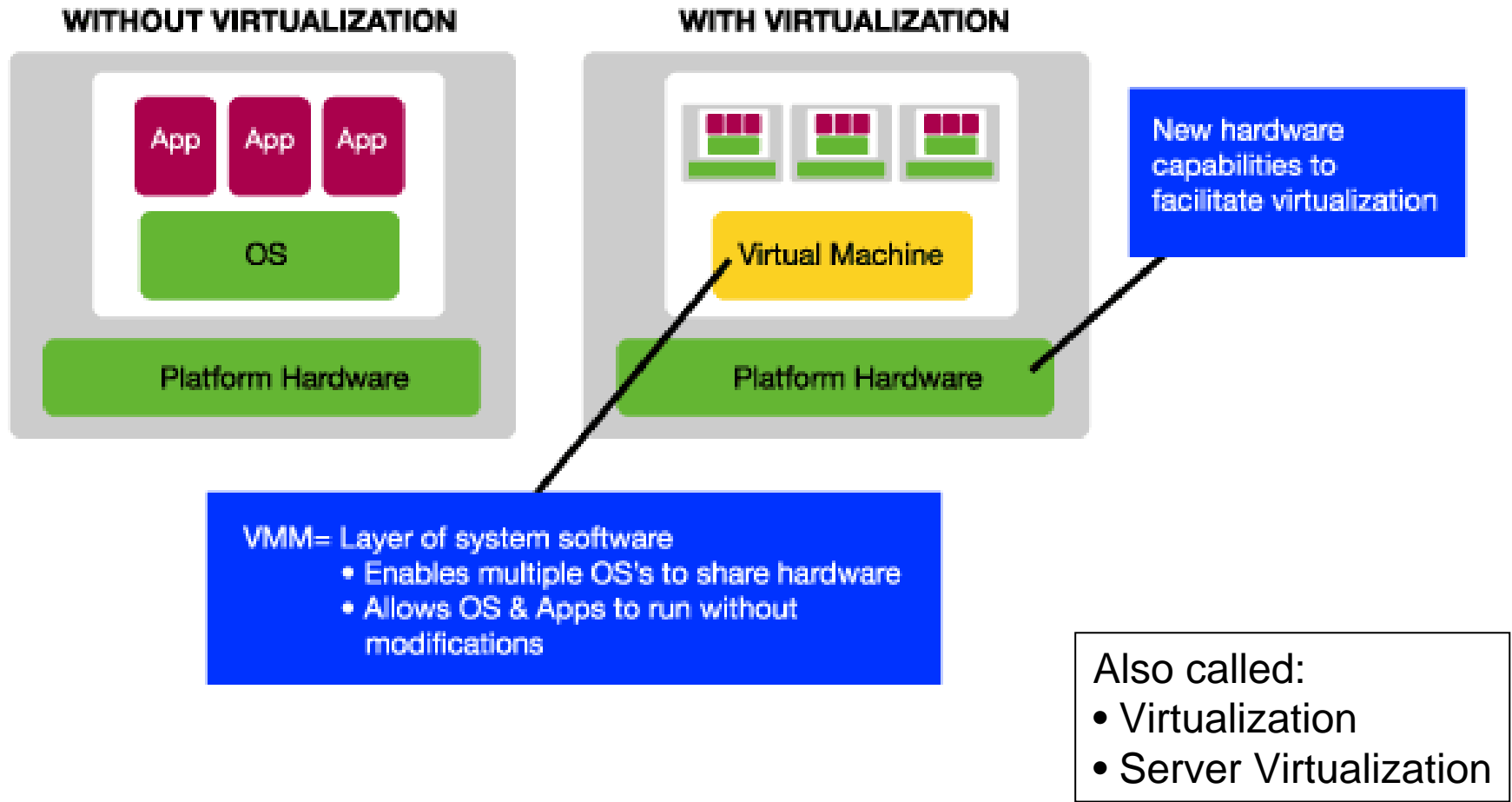


*Disclaimer: A “doodle” for  
Illustration only*

# “Virtualization” and Grids

Using virtualization as found in industry parlance today

# What is (Machine) Virtualization?

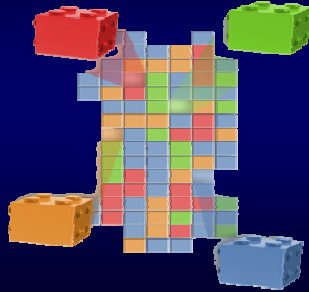


- Ok ... cool I got a virtual machine and a virtualized machine
- So what use is it:
  - I can run workloads in it
  - I can encapsulate an environment in it
  - I can configure, provision and lifecycle manage it....
  - Etc ...
- So .... what did you say one does in a Grid?

Grids are *systems-level virtualizations* but also are infrastructures for managing *workloads* and *resources* (both or one of which can be *physical* or *virtual*)

# Whether broad or narrow, Grids are about scaling IT

**Build *at scale***



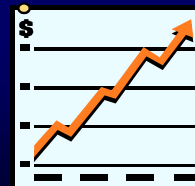
**Operate *at scale***



**Manage *at scale***



**Change *at scale***



Source: Mark Linesch

# Two sides of the same coin?

## Grids

**“How do I *run/manage* my workloads better?”**

**Workloads**



**Resources**

Organize resources to help workloads run better

## **Machine Virtualization**

**“How do I *use/manage* my resources better?”**

**Workloads**



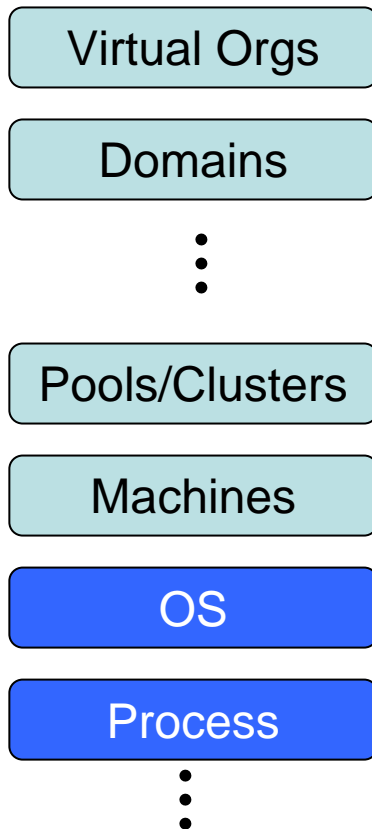
**Resources**

Manipulate workloads to help use resources better

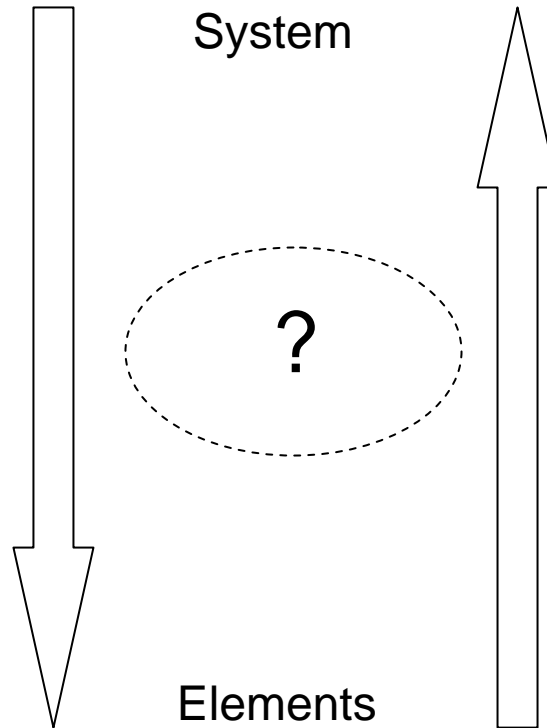
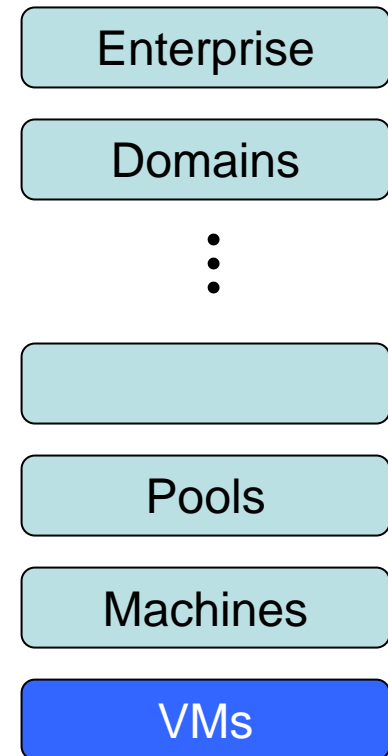
Different starting points but the middleware is very similar (same?) – **need to converge at the middleware**

# Scope inversions?

## Grids



## Machine Virtualization



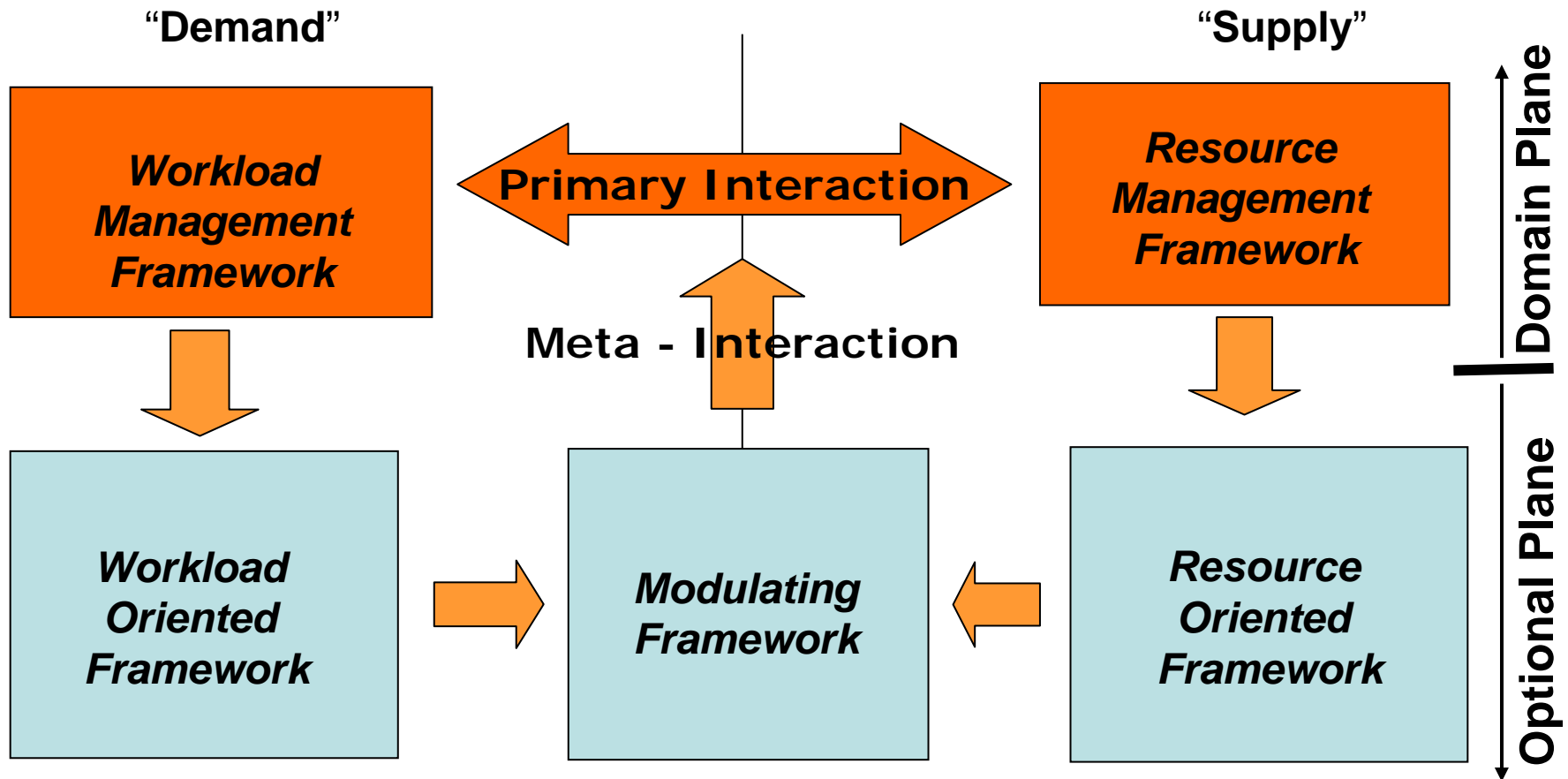
# Grids and (Machine) virtualization

- From a Grid perspective a VM can play two different roles:
  - A container or resource to execute a job in.
  - More importantly for this conversation, a VM is a “workload” that can be scheduled and managed as a job (modulo instantiation)

From this “duality” of VMs that arises from a Grid view, we immediately see that Grids not only can **create/use** VMs then can also **manage** them (without significant modifications and maintaining conceptual purity).

BTW, this duality is true for many virtualizations – think services and SOA

# Unifying Paradigm?



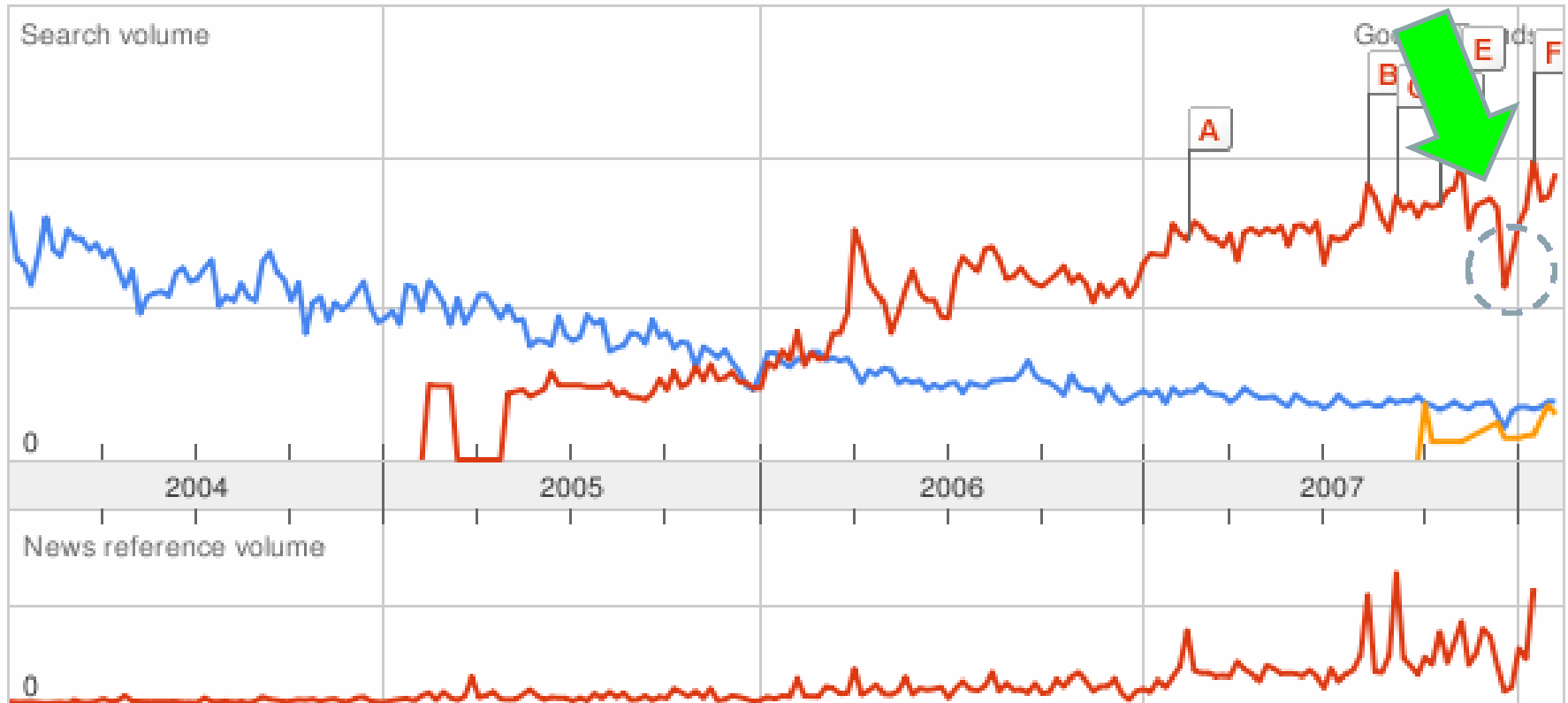
- Demand-Supply model is *not* another way of describing a request/response
- Describes the dynamic nature of complex systems – system equilibrium – involves selective cooperation and competition

# OGF and “Virtualization”

# What's wrong here ....

● "grid computing" ● "virtualization" ● "cloud computing"

**"Subprime crises kills virtualization"**



Source: Google Trends  
"Trends by search volume"

# YAE – Yet another example ...

## Grid Computing

- Coordinates resources that are not subject to centralized control
- Using standard, open, general-purpose protocols and interfaces
- Delivers non-trivial qualities of service

## Cloud Computing

- Constituents/components are opaque to the client i.e. no direct control
- Accessed over the Internet using Internet protocols
- Delivers a service that is implemented on an infrastructure of large scale

These descriptions seem to refer to the same paradigm ... Right?

**I wish ... alas ..**

# Call to Action ...

- OGF needs to recognize and exploit the conceptual similarities between Grids and emerging mainstream paradigms – aggressively educate the lay
- Morph the focus, message and org (?) on usage and value rather than a technology – for example: OGF is about ‘**Scaling IT**’ and not necessarily Grids.
- Extend our standards mindset to rationalize and factor in new models to accommodate adjacent, but emerging, paradigms in one standards framework – (“simplicity is complex”)
- OGF needs to drive to mainstream – rather than niche itself in HPC (relatively little additional work for significant gain)

# To summarize ....

- Grids and “virtualization’ are distinct but joined in the middle – Siamese twins
- Grids (and OGF) are key to realization the full potential of virtualization and vice versa – exploit the duality
- OGF (and its Grids view) has to move mainstream if we are to be relevant in the “virtualization” euphoria and other emerging (but similar) paradigms

Thank you !!