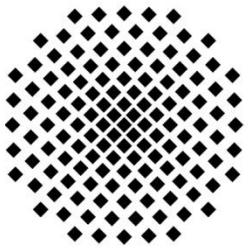
Fault tolerance and live migration in real-time clouds

Sören Berger



University of Stuttgart
Germany

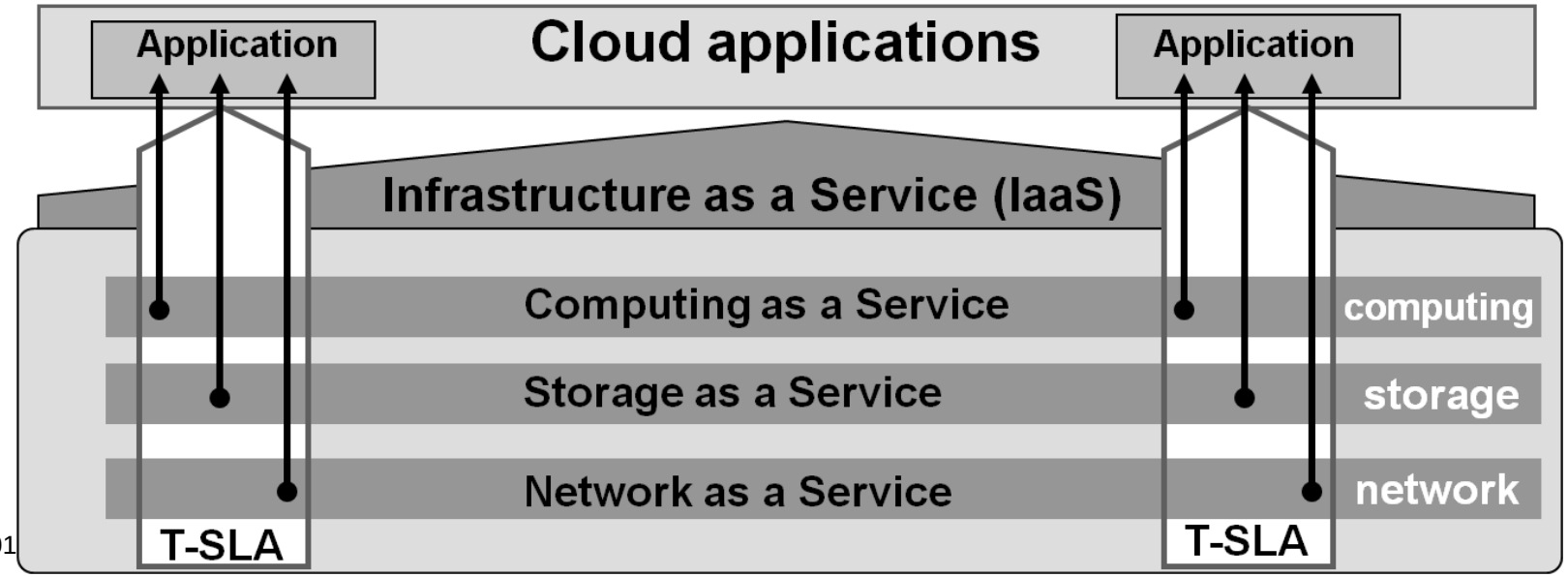
Overview

- ISONI – the IRMOS IaaS approach
- Availability
- Virtualization
- Migration
- The real-time aspect of migration
- Conclusion

ISONI – the IRMOS IaaS approach

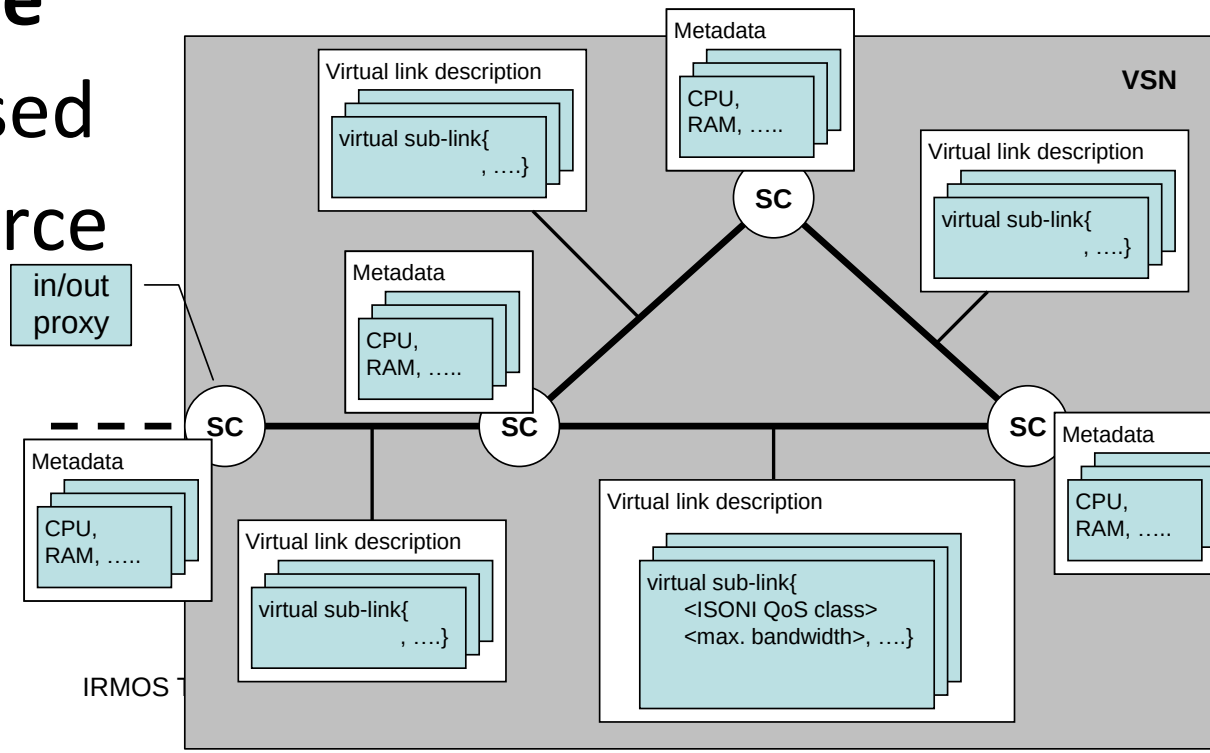
Interactive Realtime Multimedia Applications on Service Oriented Infrastructures

- Provisioning of **fully virtualized network of resources** (Compute, Network, and Storage) with QoS guarantees
- **Automated SLA management** for service instantiation and resource renegotiation
- Resource description through **Virtual Service Networks** (VSNs) with an ontology based graph models (OWL)



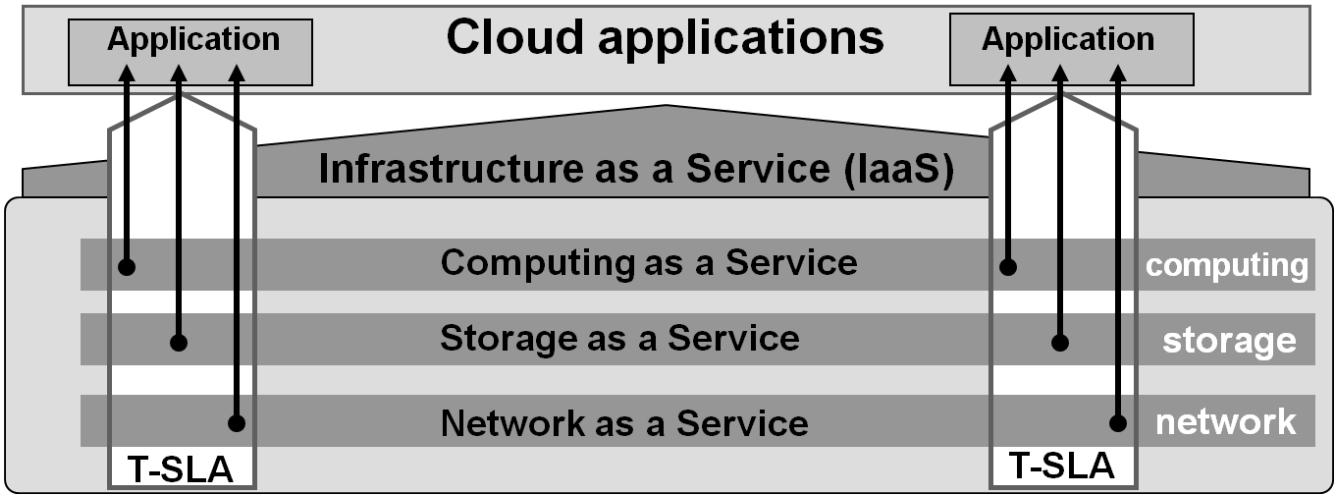
Virtual Service Networks

- Virtual Service Network, defining both computational/storage (vertices) and network (edges) resources
- **VSN is part of the Technical SLA** used during the resource negotiation
- QoS parameters annotated to resources

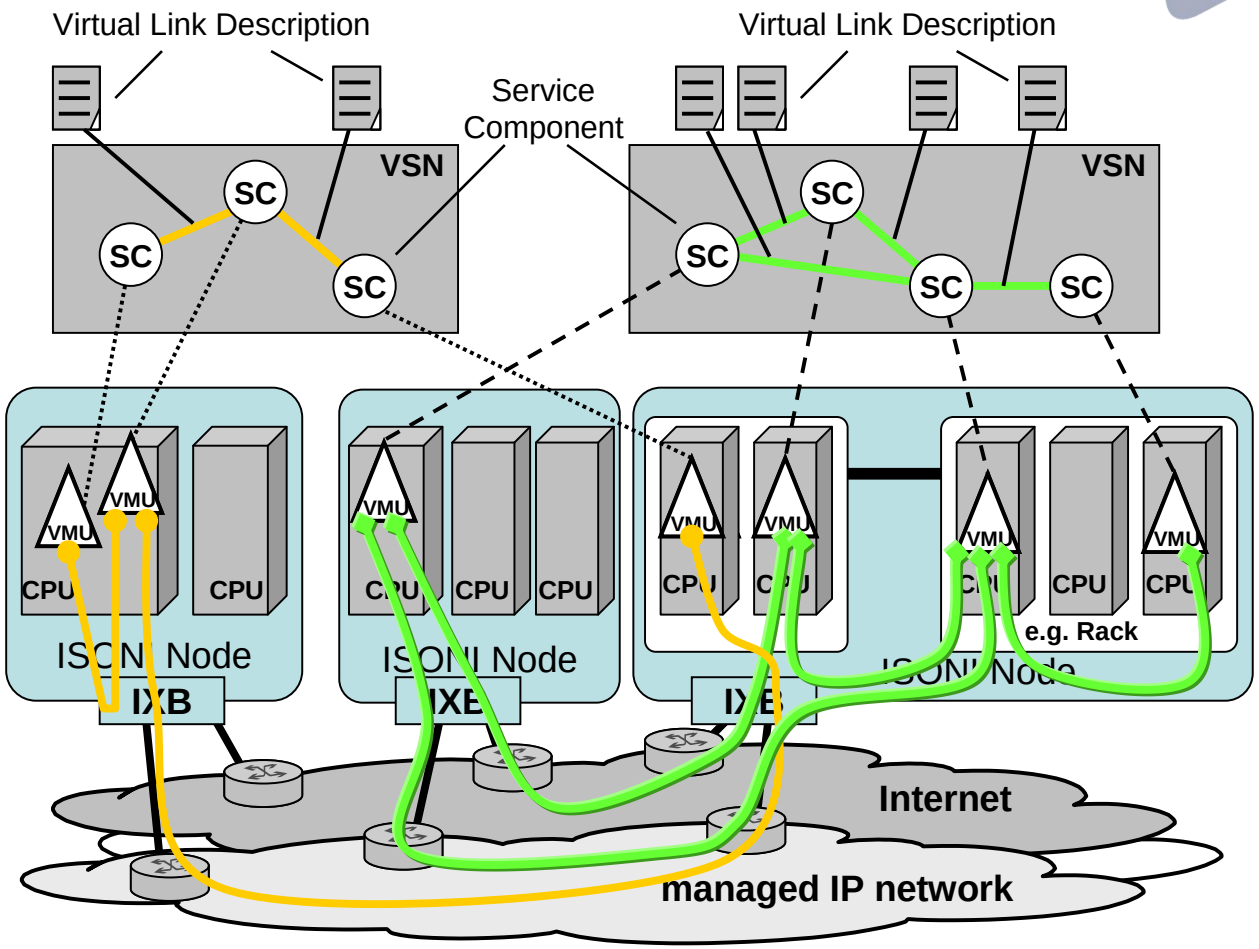


Real-Time enabler

- Compute: hybrid priority/deadline real-time scheduler
- Network: Network traffic flow control, traffic shaping
- Storage: QoS enabled storage
- Management infrastructure for resource planning



Integrated Management of Virtual Environments



Application runs in Virtual Service Networks described in VSNs

Physical Hosts providing computing resources

Network virtualization managed by ISONI

Availability in virtualized infrastructures

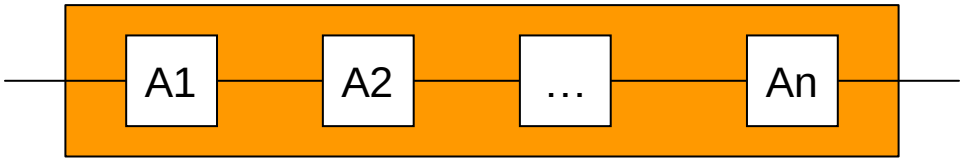
- Myth of virtualization:

  Virtualization increases availability

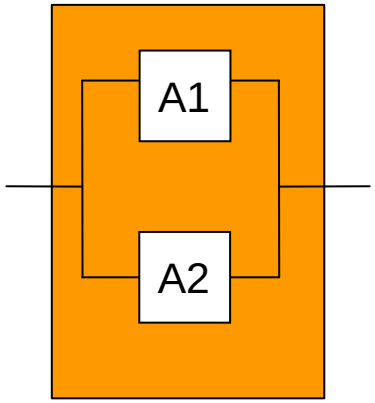
- Example

  3 Components with 95% availability in a row

  → Total 85%

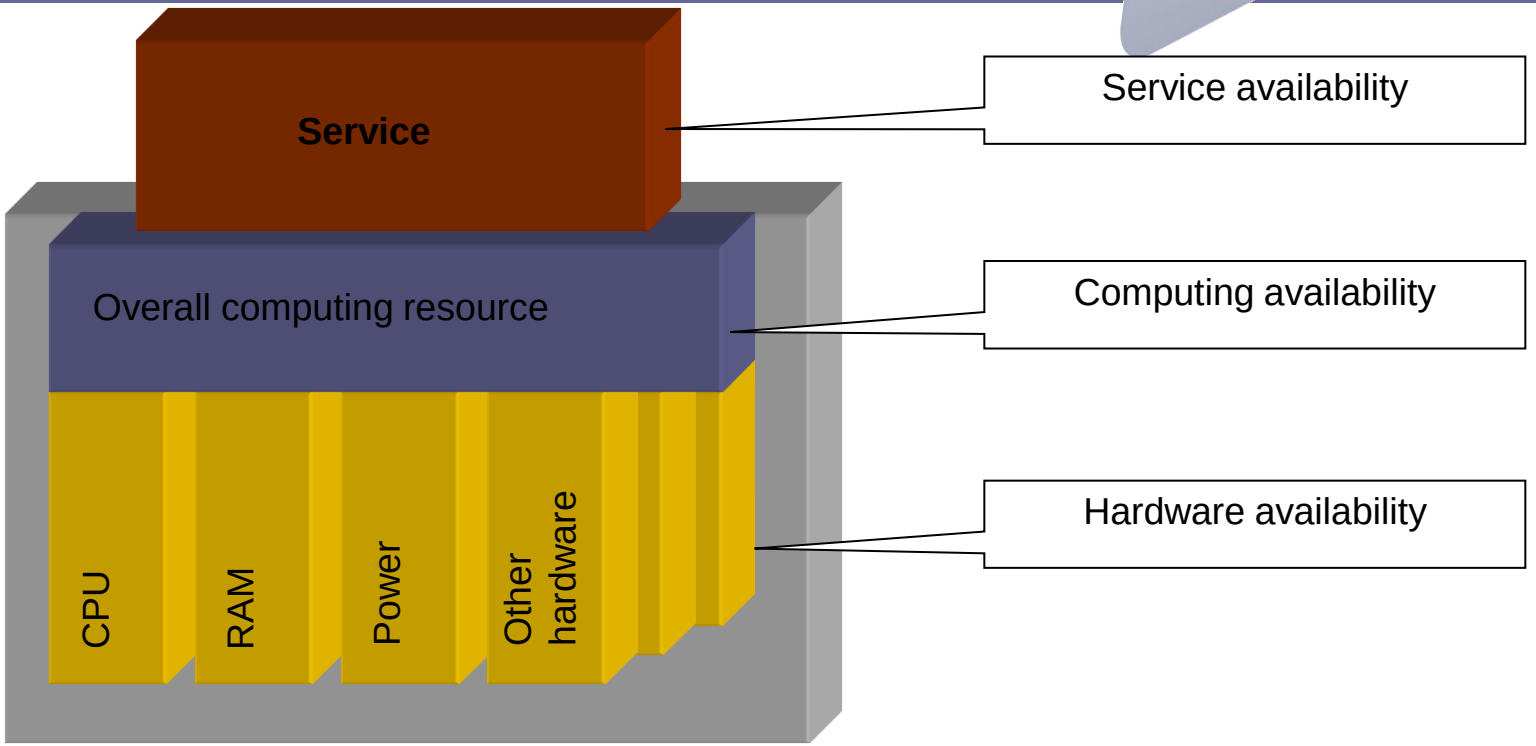


$$A_{total} = A_1 \bullet A_2 \bullet \ldots \bullet A_n$$



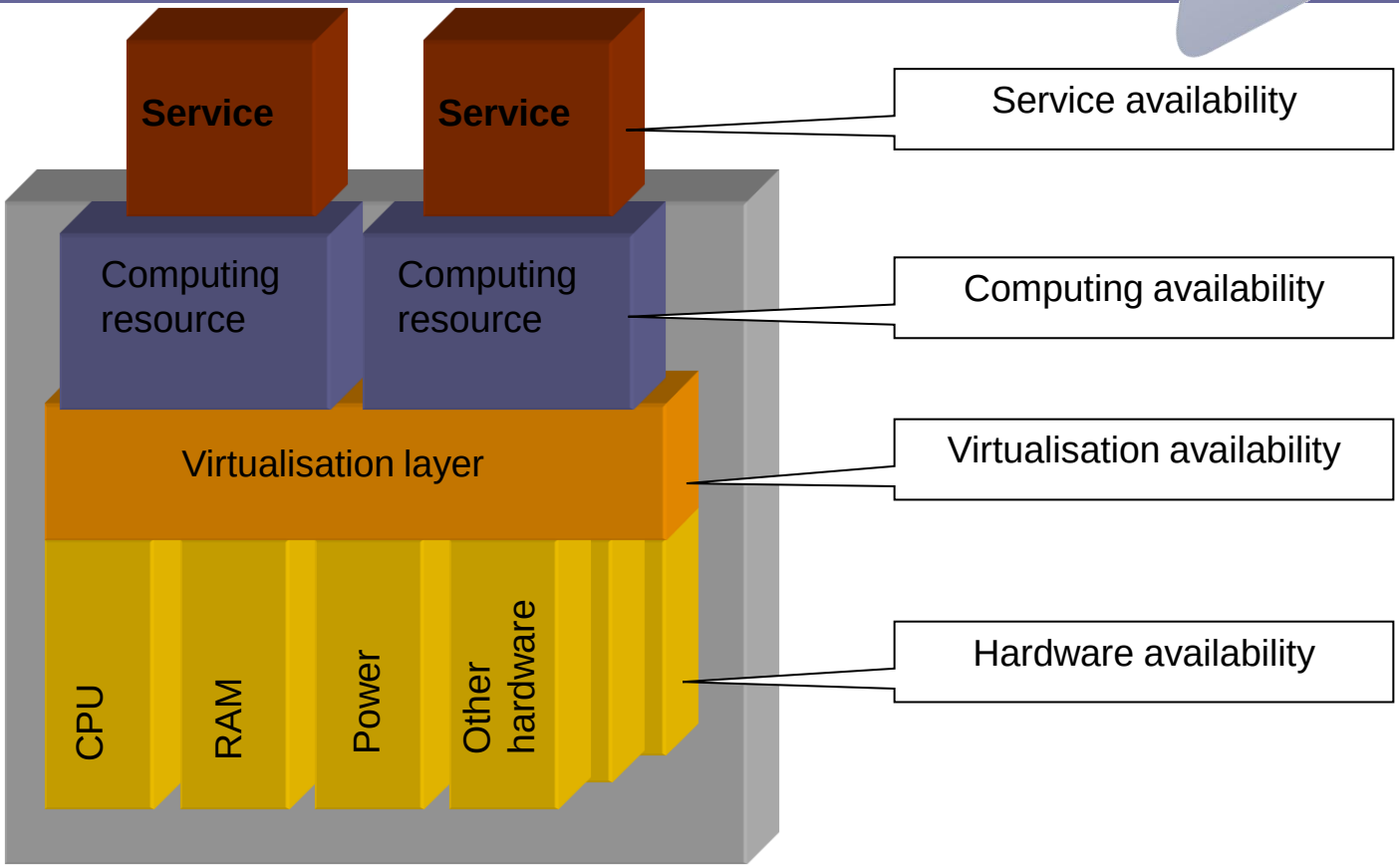
$$A_{total} = 1 - [\,(1 - A_1) \bullet (1 - A_2)\,]$$

Classical view on availability of computing resources



$$A_{total} = A_H * A_C * A_S$$

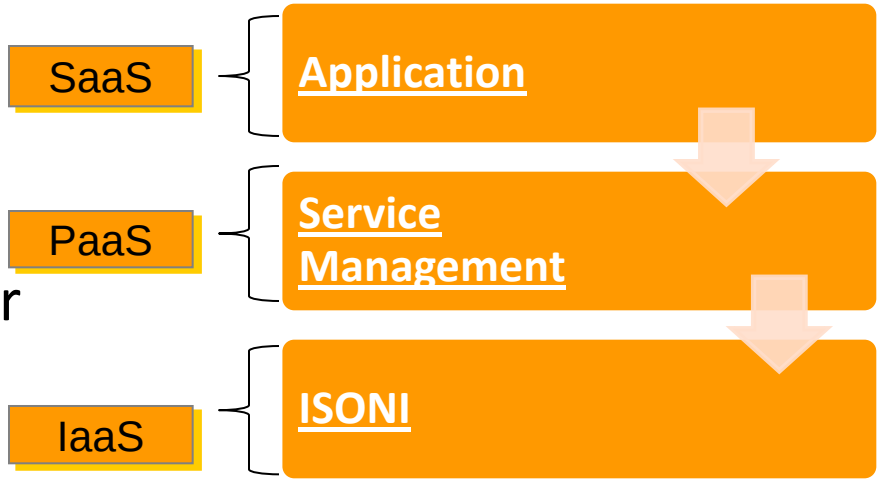
Availability in a virtualized environment



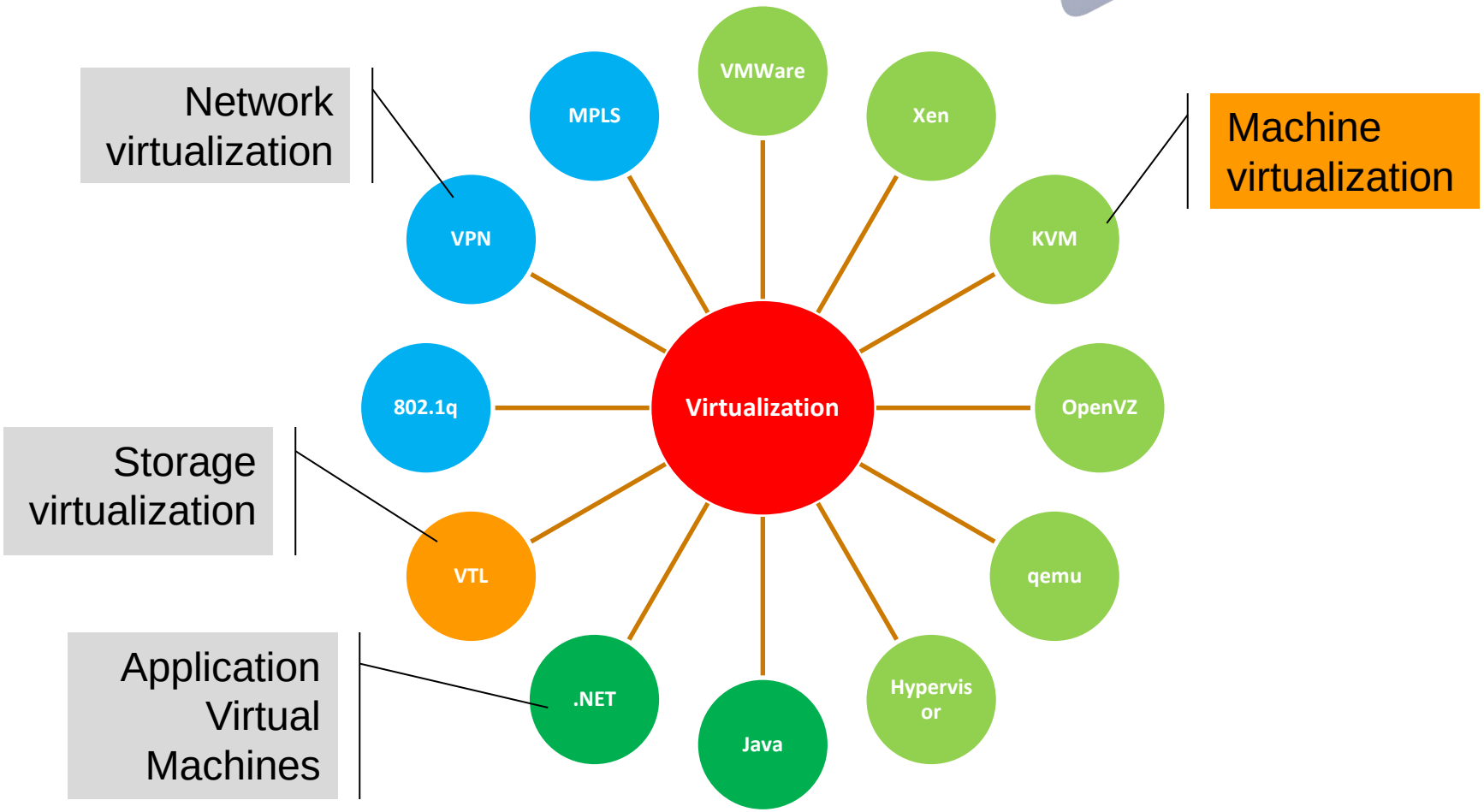
$$A_{total} = A_H * A_V * A_C * A_S$$

Redundancy on different levels

- Application level
  - Use failover server
  - Eg. Client Software connect to another server
- Service level
  - Eg. us a load balancer to cover outage
  - Transparent for the end user
- **Infrastructure level**
  - Use the same virtual machine on another host
  - Transparent for Service Provider



Types of virtualization



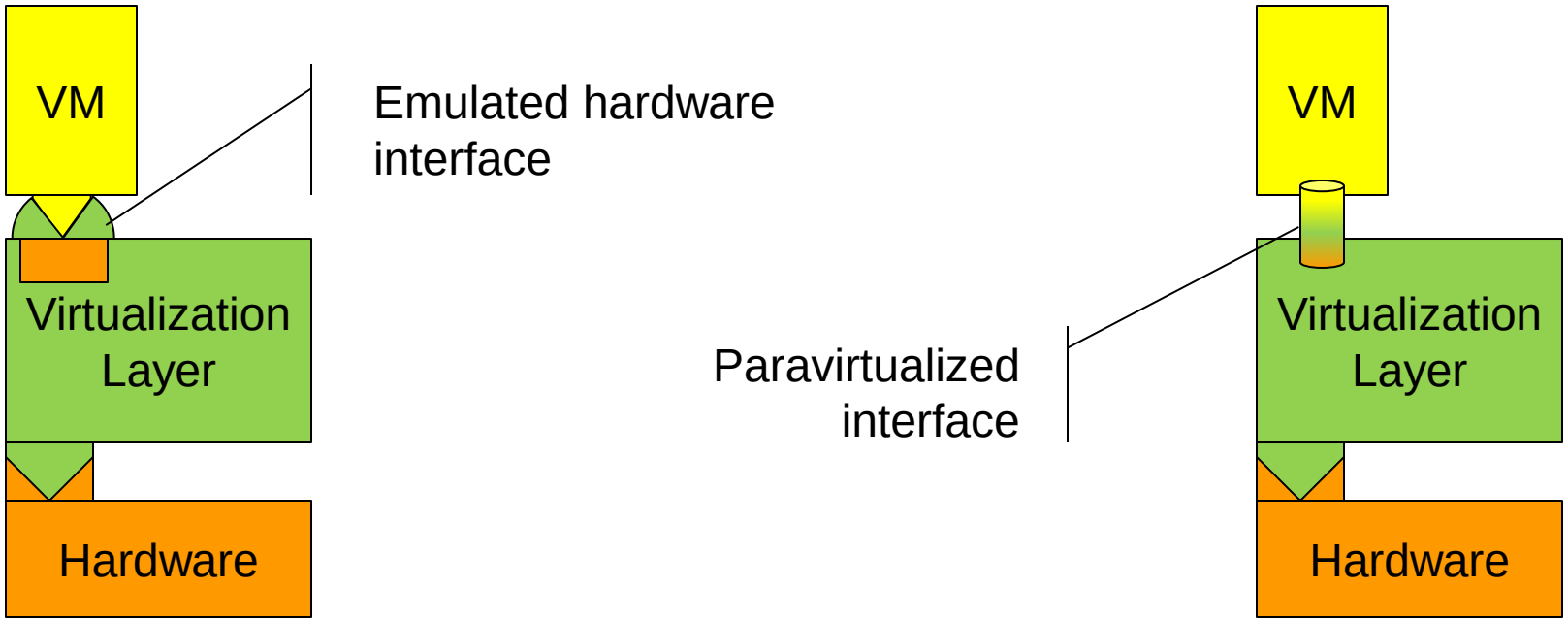
Computer virtualization – a short history

Interactive Realtime Multimedia Applications on Service Oriented Infrastructures

- 1950s Logical CPU (Christopher Strachey)
- 1960s - Memory Virtualization (Atlas Computer)
  - Multiuser Support (IBM System/360)
- 1972 „Architectural Principles for Virtual Computer Systems“ (Robert P. Goldberg)
- 1999 Virtualization of Intel x86 (VMWare)
- 2003 First public release of Xen
- 2005+ Secure Virtual Machine (Intel VT, AMD-V); Intel VT-d, AMD IOMMU

Device access in virtualized environments

- No direct access to hardware
- Performance penalty if hardware is emulated and translated into access to actual hardware
- Making guest aware of virtualization allows use of specialized device drivers → paravirtualization



Machine virtualization: Virtualization vs. emulation

Emulation

- Emulated resource can differ from original resource: x86 -> x86, PowerPC -> x86
- Does not need to be physically present
- Can have significant performance penalty

Virtualization

- Virtualized resource is instance of original resource: x86 -> x86, PowerPC -> PowerPC
- Concurrent access to hardware needs to be controlled
- **Usually fast**

Advantages of virtualization

- More efficient infrastructure
- Flexibility
  - Different guest OS
  - Different performance per virtual machine
  - Server as “throw away” article
- Easier to administrate
  - Can be controlled by software
- Applications are easy to adapt into virtual machines

Migration of virtual machines Overview

- Different kinds of “moving” virtual machines to another hosts
  - Move and restart
  - Migration
  - Live migration
- Different timing behavior

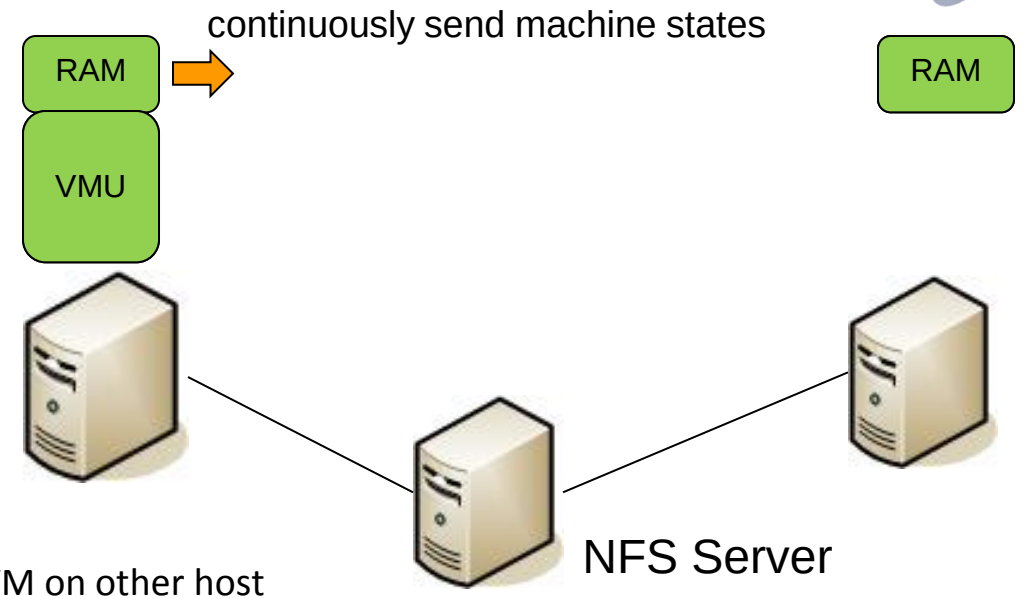
Virtual Machine restart

- Process: Shut down VM and start it on the other host
- Maybe required a restart/configuration of the application
- Not transparent for the User/Application
- Shutdown and startup requires unpredictable time
  - Operating system have to be booted
  - Application have to be started

Virtual Machine migration

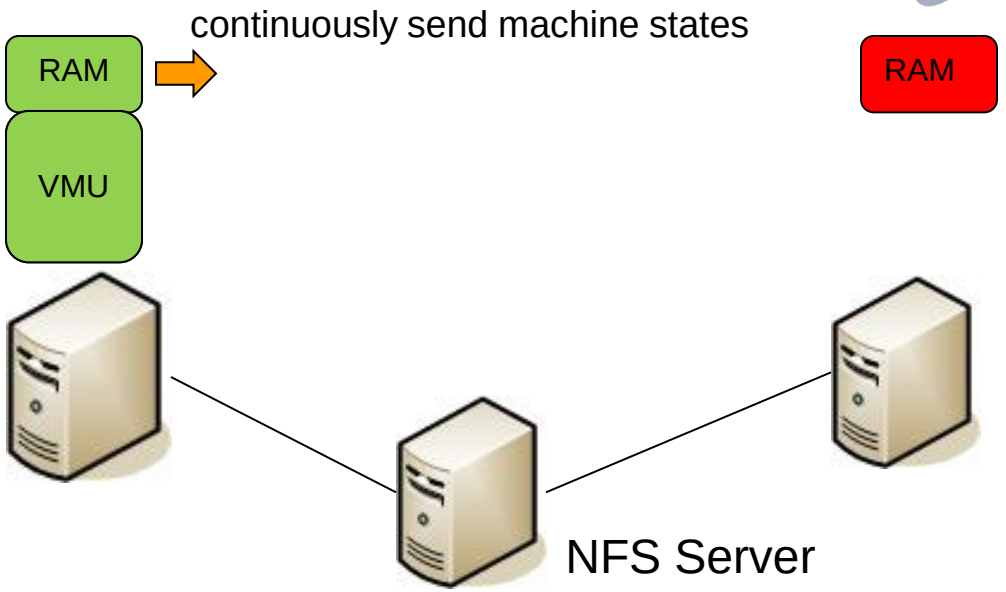
- Process
  - Save machine state to disk
  - Move state and machine to new host
  - Restore machine state
- VM is unavailable during migration process
- Transparent for the User/Application
  - All application and Operating system states are kept
- Shutdown and startup in predictable (Size of VMU, RAM…)
- **Downtime** = Save Time + Transfer Time + Resume Time

Virtual Machine live migration



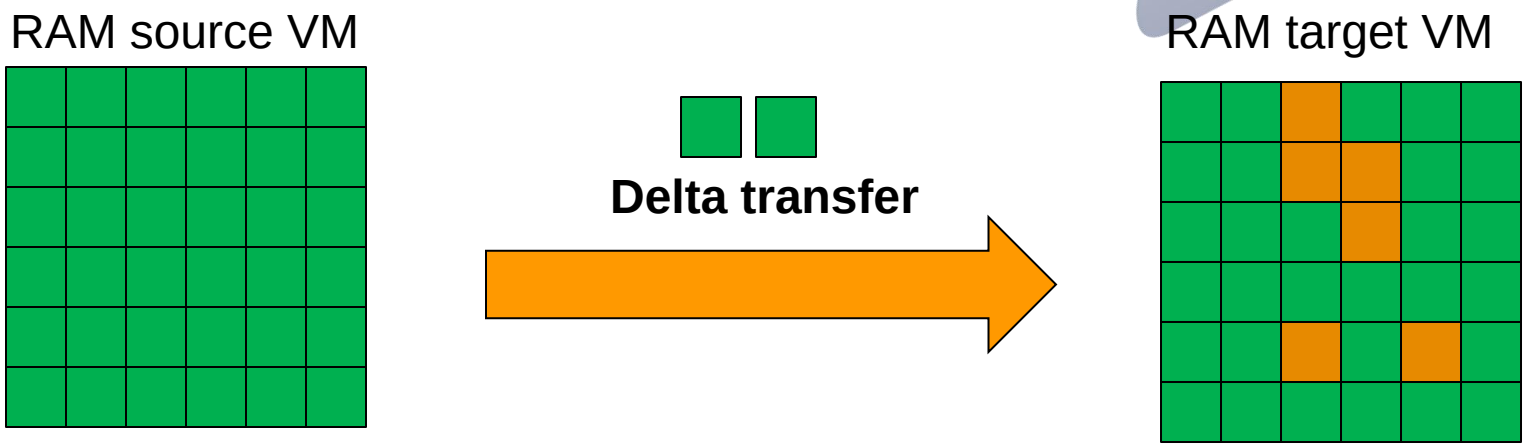
- Process
  - Start VM on other host
  - Send machine states that are not used frequently used
  - Finally stop source VM and send the “last delta”
- VM is **available** during migration process
- Transparent for the User/Application
- Shutdown and startup in predictable (Size of RAM)
- **Downtime** = time to transfer last delta

Why migration is not usable for redundancy?



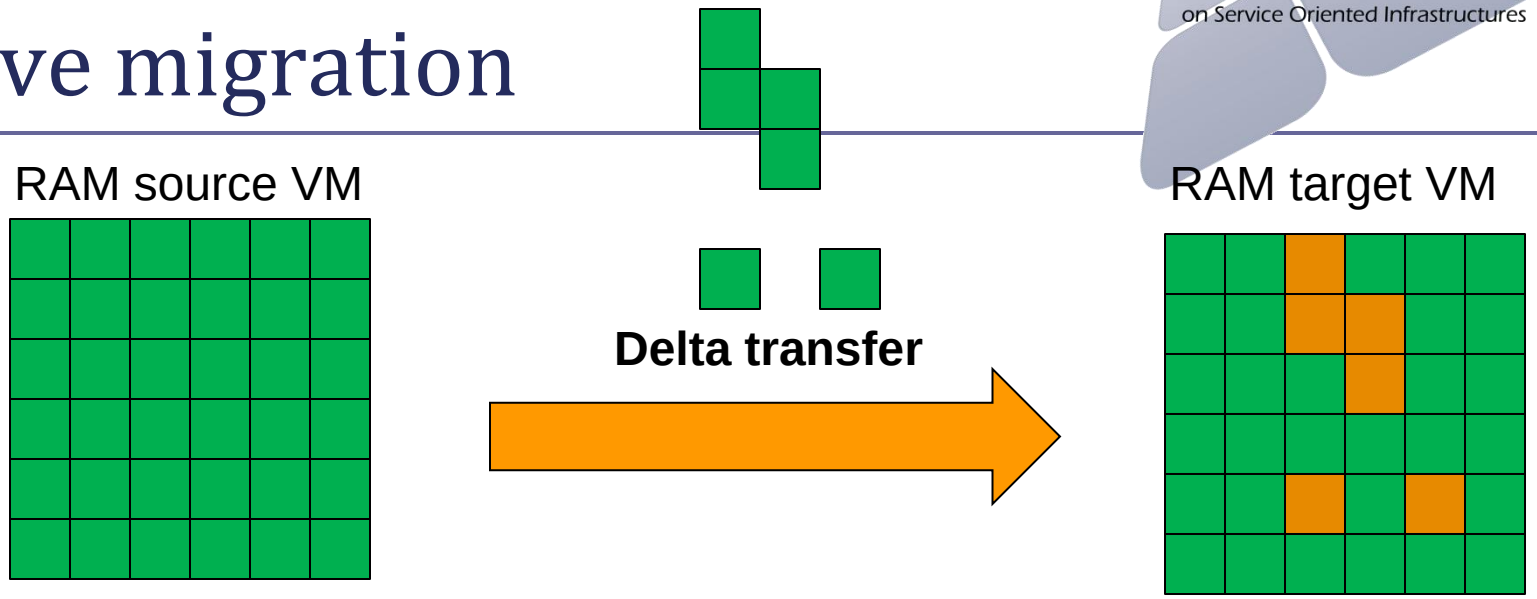
- migration is application unaware!
- During delivery changes of RAM/Cache and CPU states
- Target VMU is always outdated
  - States are lost: data structures in the Memory, TCP connection states
- It would work when:
  - RAM, Cache and CPU states are synchronized
  - But should be: Link throughput = CPU Cache/Register throughput.

Time determination for live migration



- How long does it take to copy the memory
- Time for transferring last delta:
  Dirty memory pages / bandwidth
- During transmission some memory areas will be marked as dirty again
- At some point copy the last missing data

Time determination for live migration



- When to stop the source virtual machine
- Time for transferring last delta:

  #Dirty memory pages / bandwidth
- Condition: Threshold < requested time

Comparison

Migration technique	Usable after crash	Transmission time	VM states	Predictable recovery time
VM restart	No	Long	Lost	No
VM migration	Yes	small	Kept	Yes
VM live migration	yes	tiny	Kept	Yes

Conclusion

- ISONI provides real-time guaranteed resources
- Availability of a service in a cloud is not only realized at the infrastructure level
- Virtualization provides features to increase the availability of services
- Live migration increase and extends the flexibility of a virtualized environment
- Migration has predictable timing

Thank you!

Sören Berger
USTUTT
soeren.berger@rus.uni-stuttgart.de

Further Information
http://www.irmosproject.eu

The research leading to these results has received funding from the EC Seventh Framework Programme FP7/2007-2011 under grant agreement n° 214777