Utilification *redux*

Middleware 2006

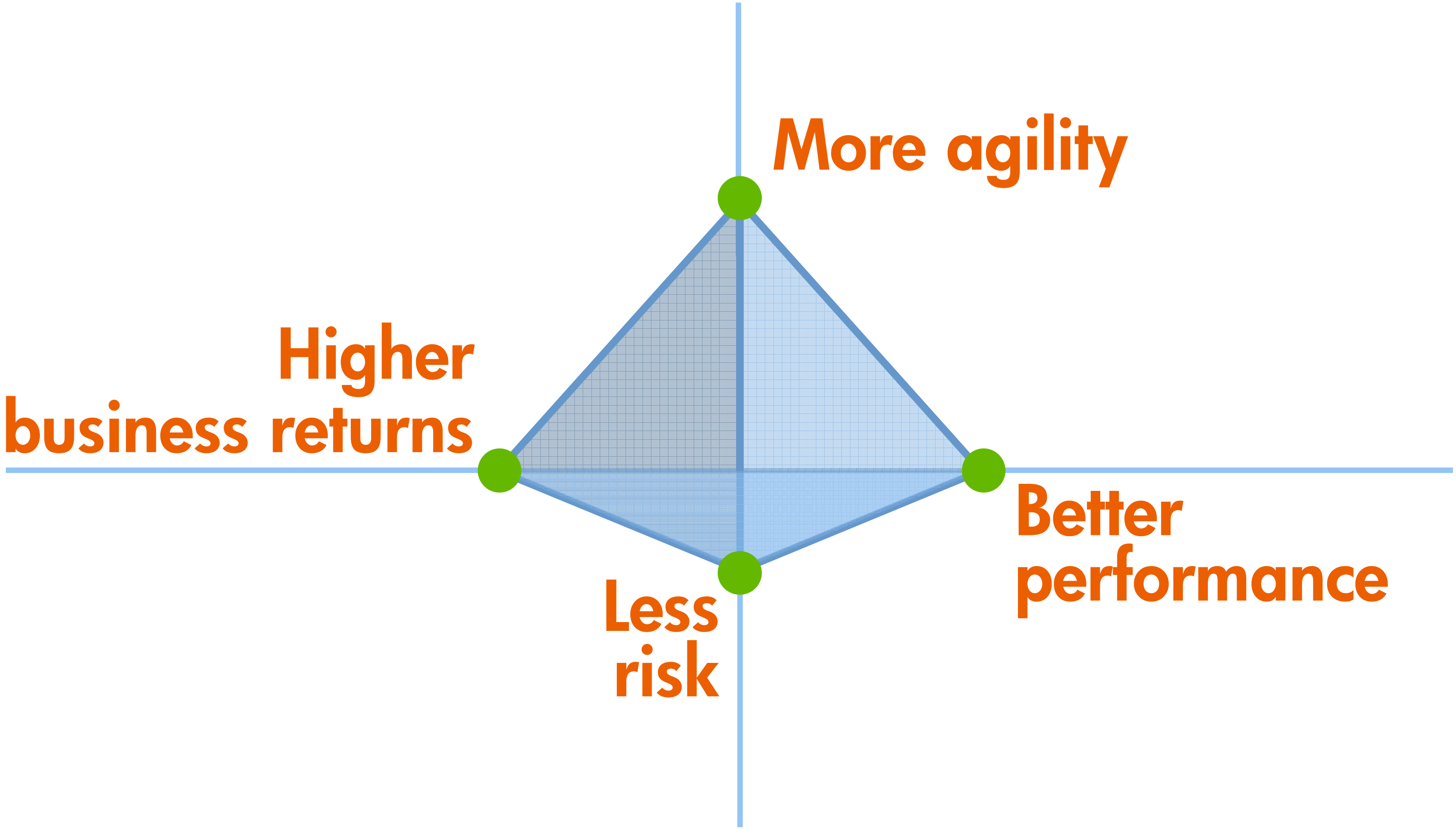
John Wilkes
HP Labs, Palo Alto, California

Utilification

European SIGOPS workshop 2004

John Wilkes, Jeff Mogul, Jaap Suermondt
HP Labs, Palo Alto, California

Q: what does enterprise IT need?



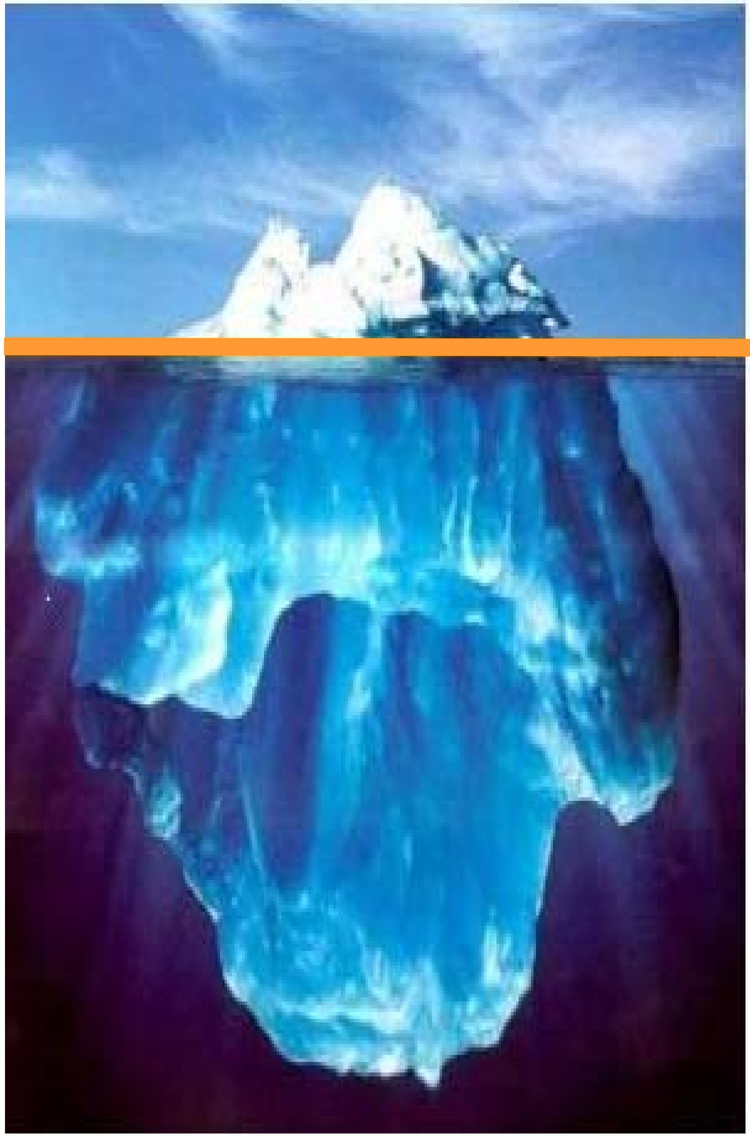
Q: what does enterprise IT need?
A: utility computing!

Flexible, scalable provisioning of computer-based services

- on demand: as and when needed
- agile: in response to events

- without all the hassle

“Utility computing” is not enough



utility computing

getting to
utility computing

Q: what does enterprise IT need?
A: utility computing!

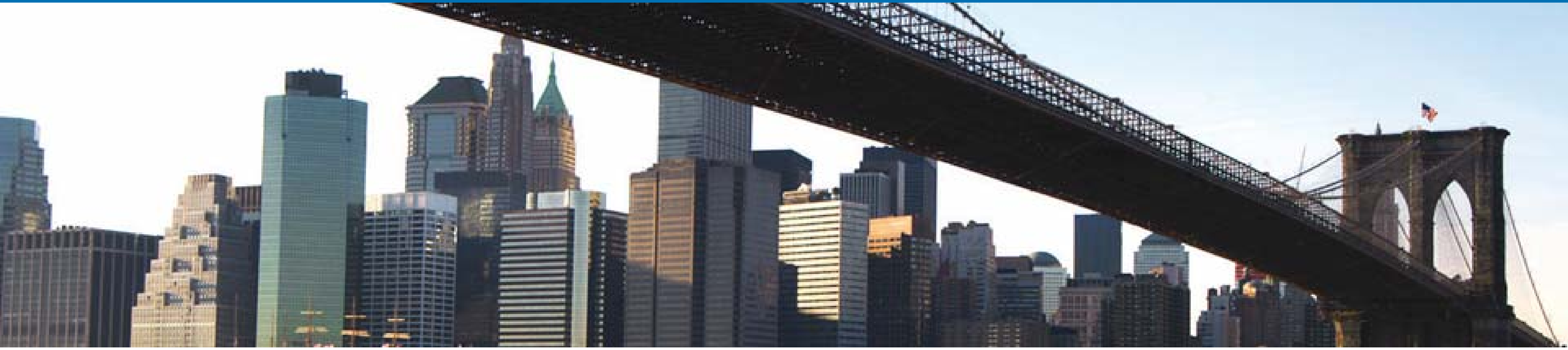
OK then. How do you get there?

Transform applications from their standalone version into a utility-computing one

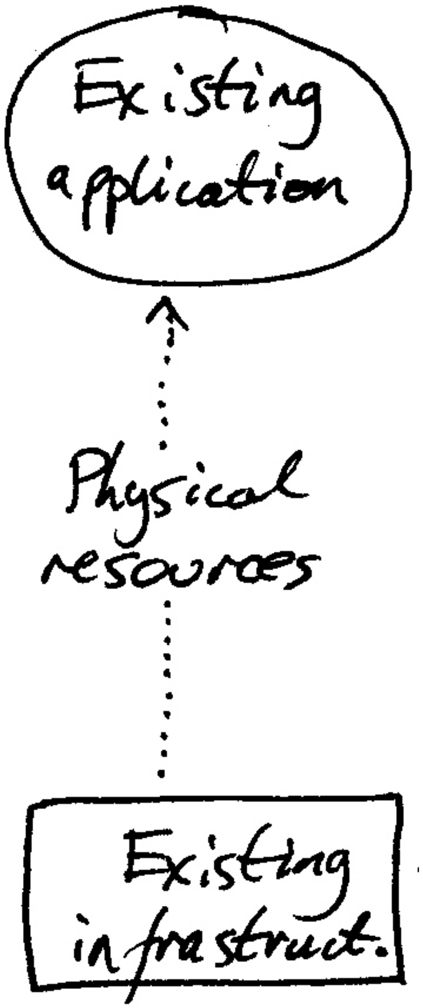
→ **Utilification**

"I'm all for progress. It's change I don't like."

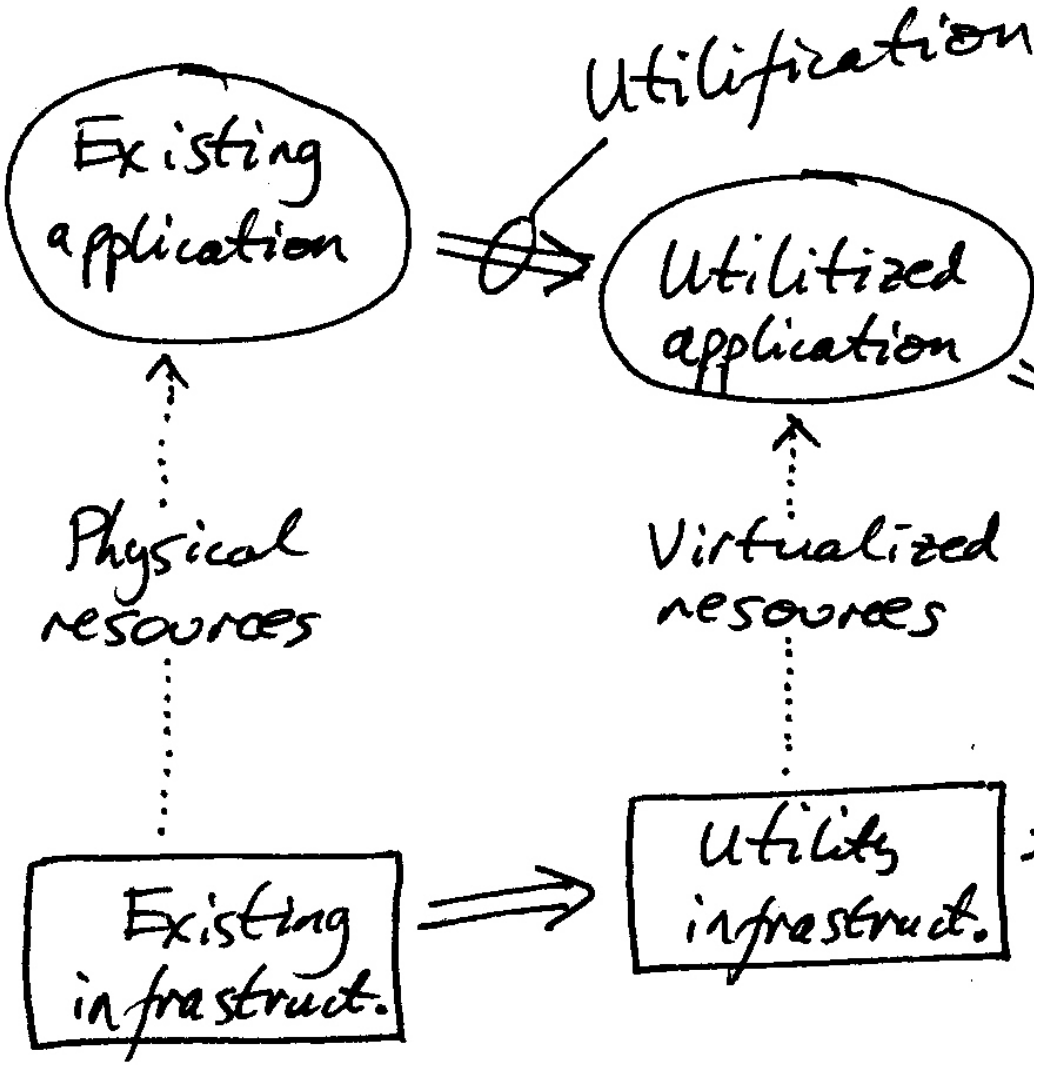
– Mark Twain



Utilification: the process
Isn't it simple?



Utilification: the process
Isn't it simple?

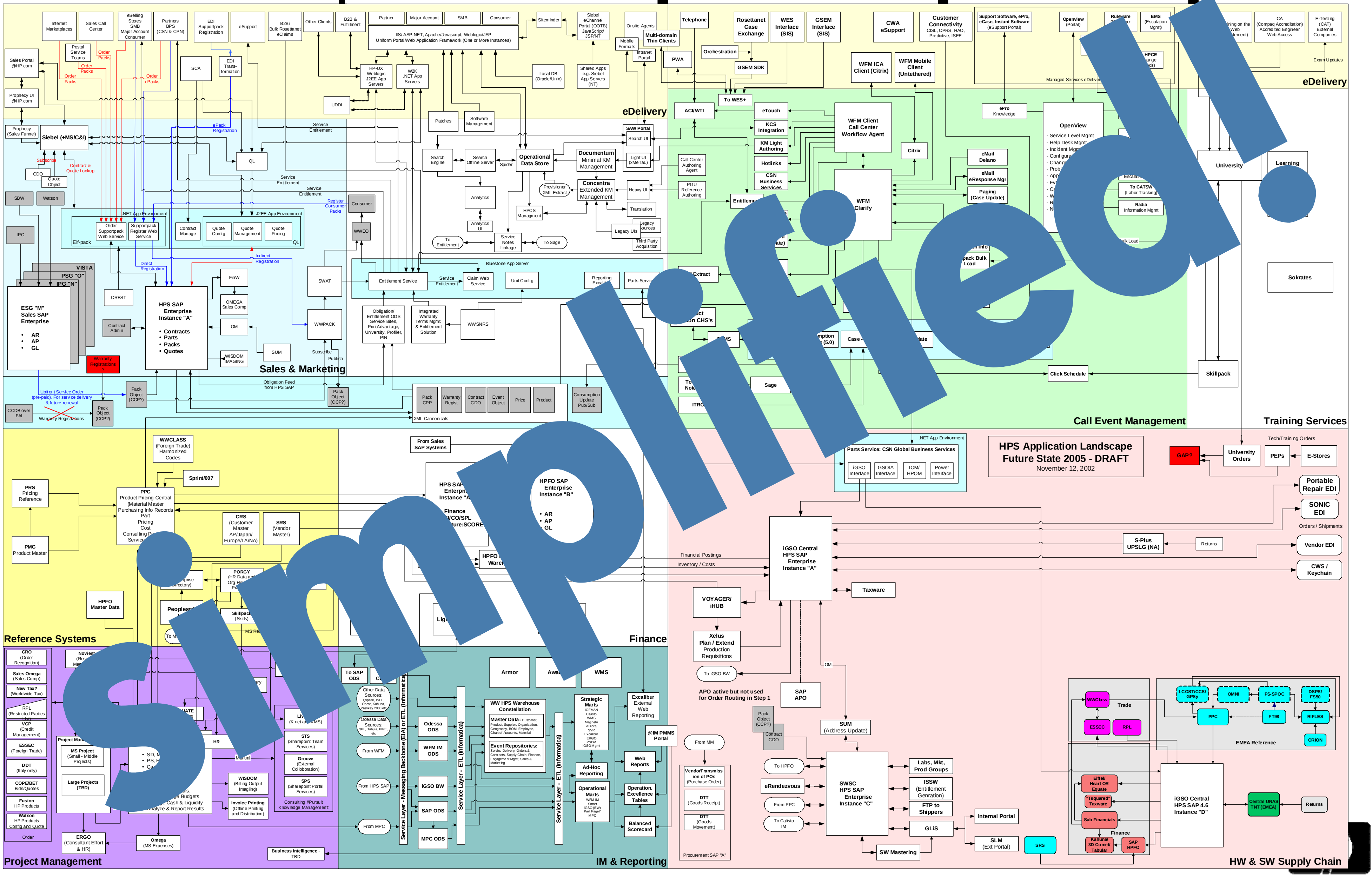


Utilification: the process
Isn’t it simple?

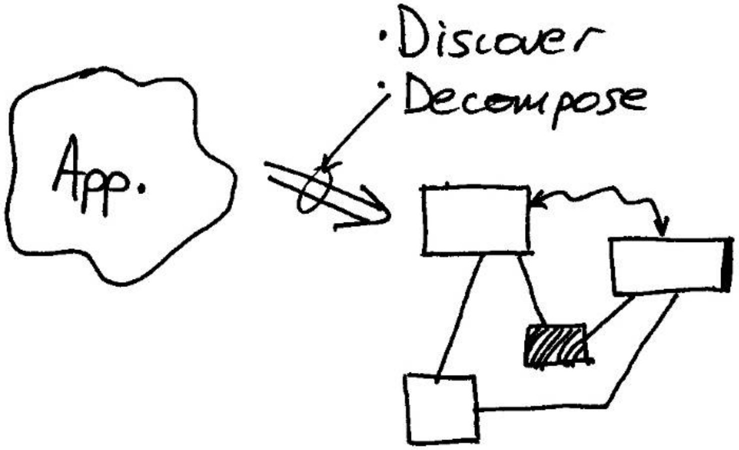
- Pick the application to migrate
- [Shut it down]
- Bring it up in the new environment
  - pick throughput and response times
  - work out how many resources the app needs
  - tune things a bit
- Maybe wrap some resource-management stuff around it
- Basically straightforward, even if effort-intensive
- **Right?**

Utilification: the process

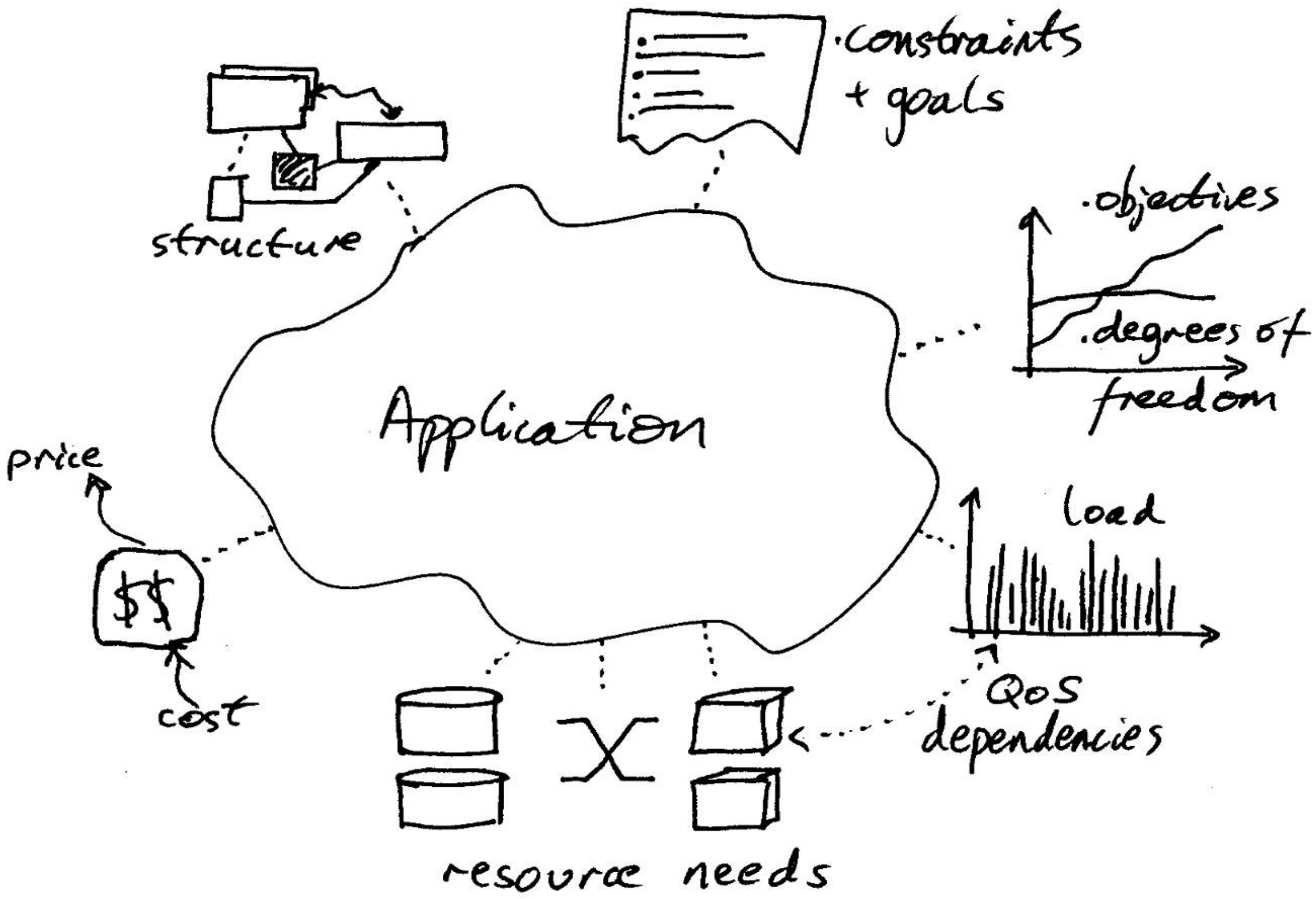
Isn't it simple? **Sample enterprise IT plan**



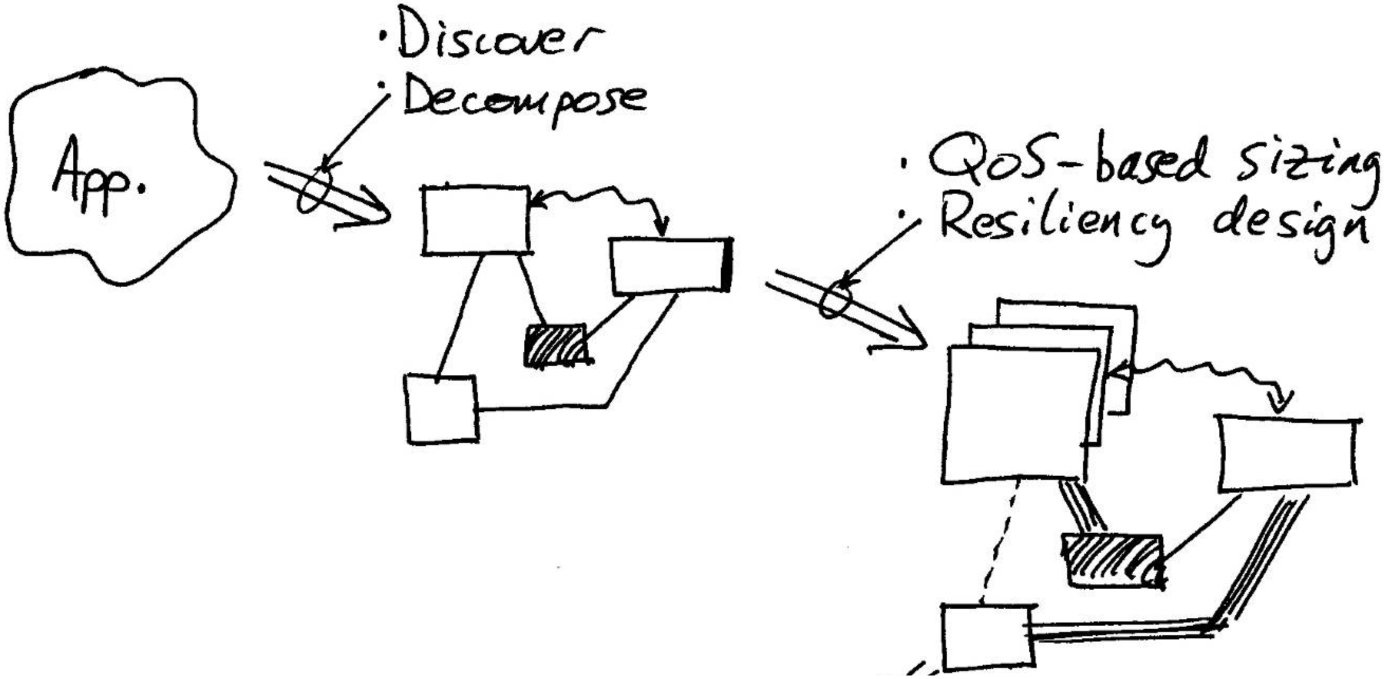
Utilification: the process
Assessment



Utilification: the process
Assessment (aka blueprinting)



Utilification: the process
QoS-based sizing



BELLE-VUE

INTERBREW
20 L
2004
2003
0035648
0034965

Utilification: the process
QoS-based sizing

- Requires mappings from:
  - offered load + QoS needs → resource needs
  - offered load + resources → predicted QoS needs

- This is hard!!
  - even for the single-element application
    - typically lots of knobs and settings
  - now add many moving parts
    - multiple control parameters, which interact
  - now add operating in a new environment

Utilification: the process
QoS-based sizing

- Sample question: what's the "QoS budget" for each component?
  - How should a 100ms response-time be split between two components?
  - What if the resource demands of these two alternatives lead to very different costs?
  - What if the cheapest solution is the most susceptible to mis-estimations of the load?

Utilification: the process
QoS: **resiliency**

- Maybe now is the time to increase this?
  - add redundancy and replication
  - add better predicting, detecting, recovering from failures

- How much application-level resiliency is needed?
  - *availability* (percentage uptime)
  - *reliability* (resistance to data loss or corruption)
  - *performability* (probability of achieving a given performance level)

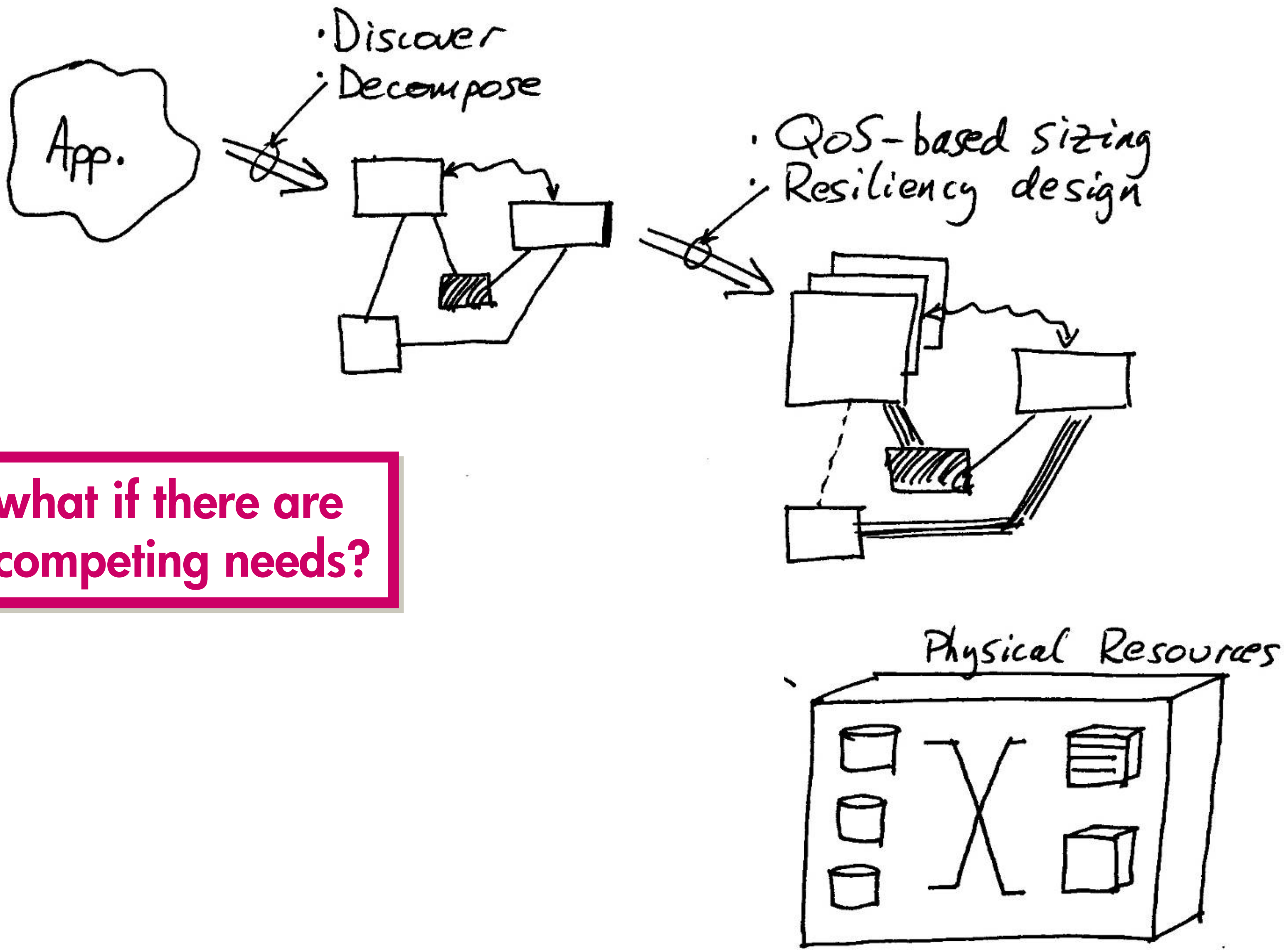
Utilification: the process
QoS: **security**

- Utility computing ➔ shared platform
  - across mutually-distrusting customers?
  - *not on my watch!!*

- How to write a security QoS specification?
  - probably not just: “time to apply virus patches”

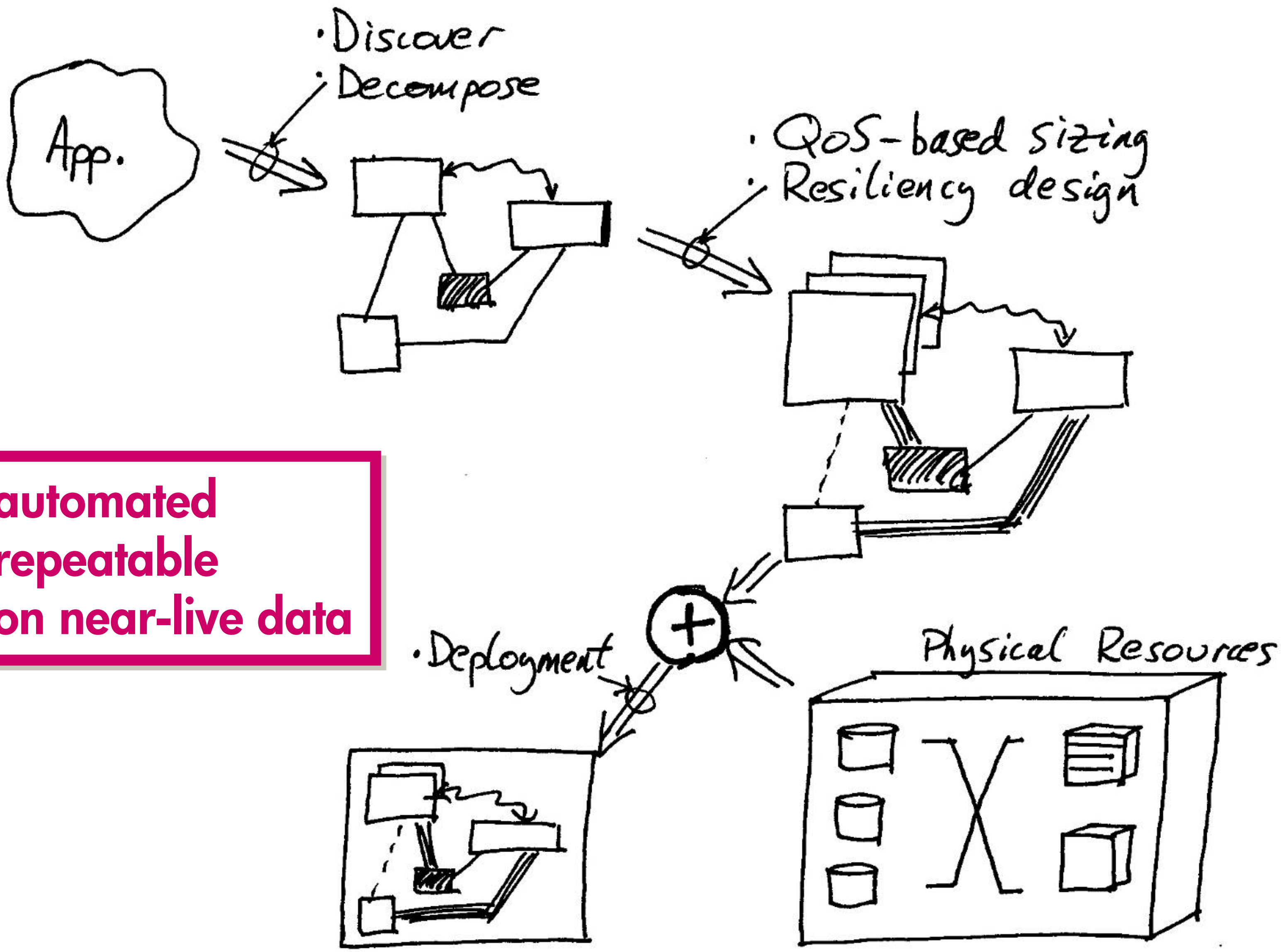
- How to pick the right mechanisms?
  - predicted efficacy? cost? ROI?

Utilification: the process
allocating/assigning resources



- **what if there are competing needs?**

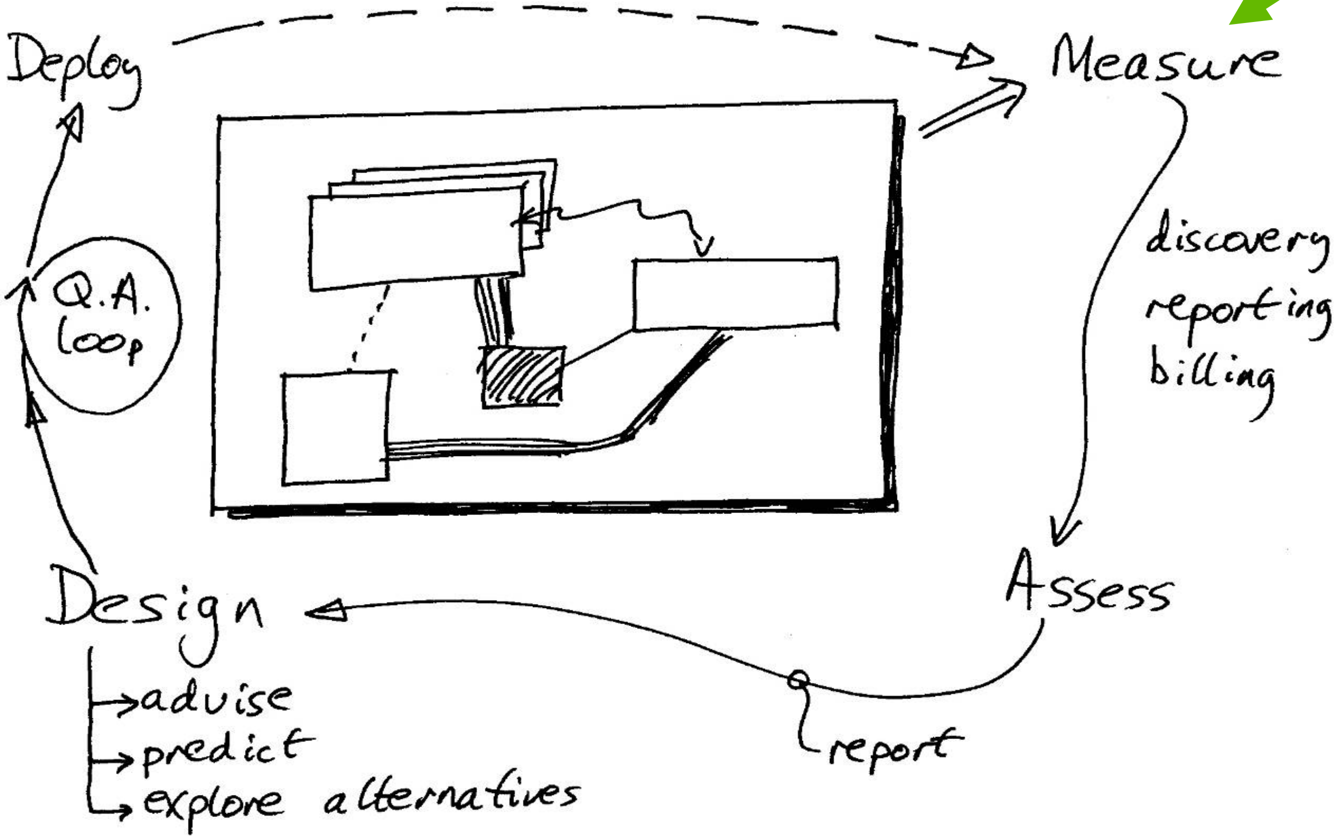
Utilification: the process
test + deploy



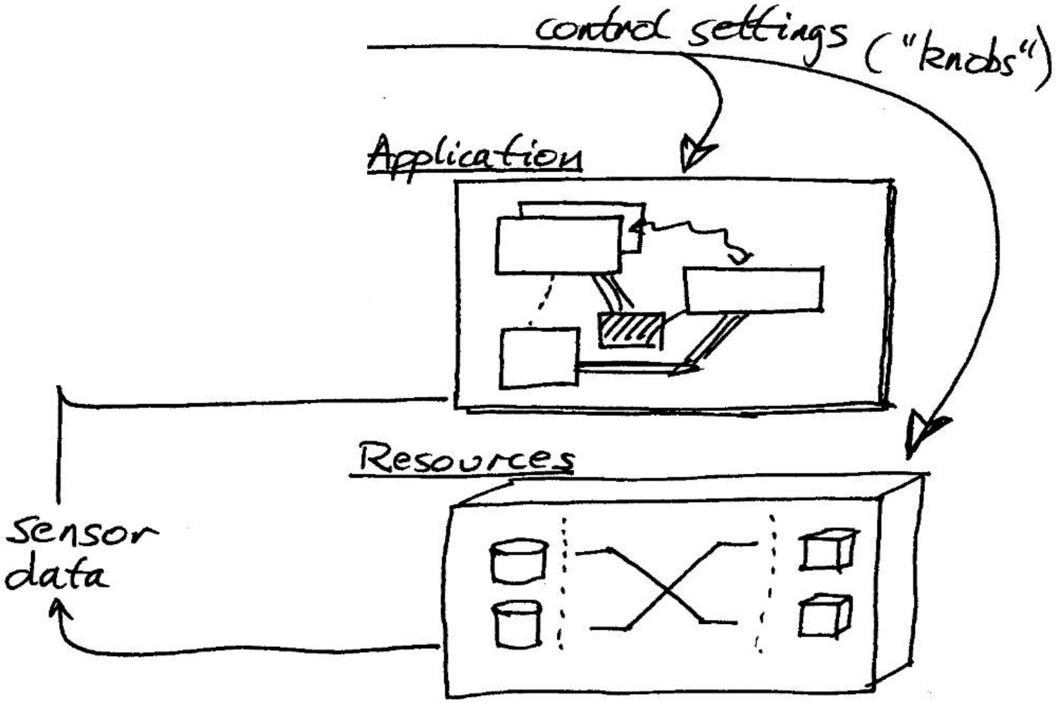
- **automated**
- **repeatable**
- **on near-live data**

Utilification: the process
QoS enforcement

start here



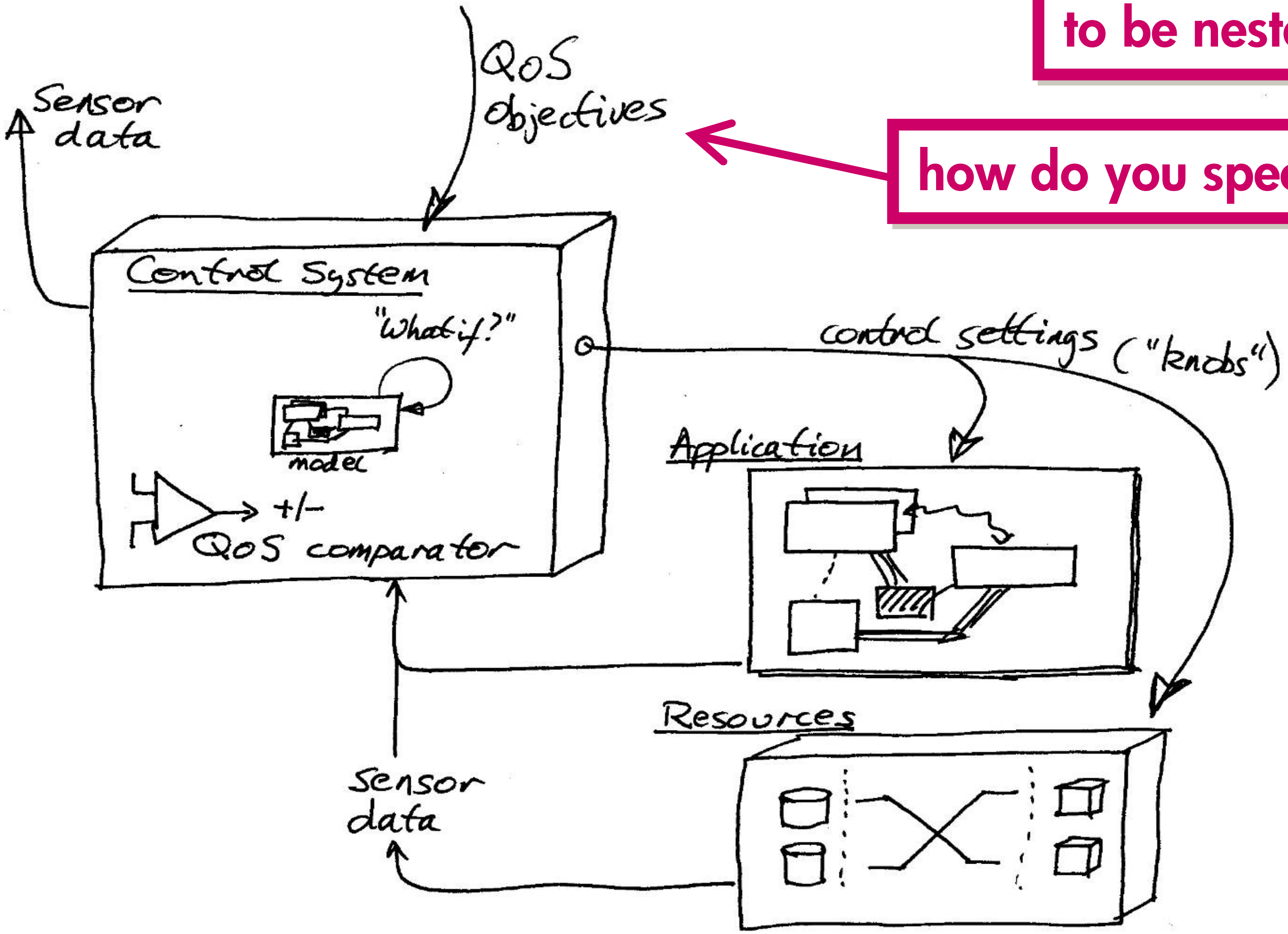
Utilification: the process
QoS enforcement



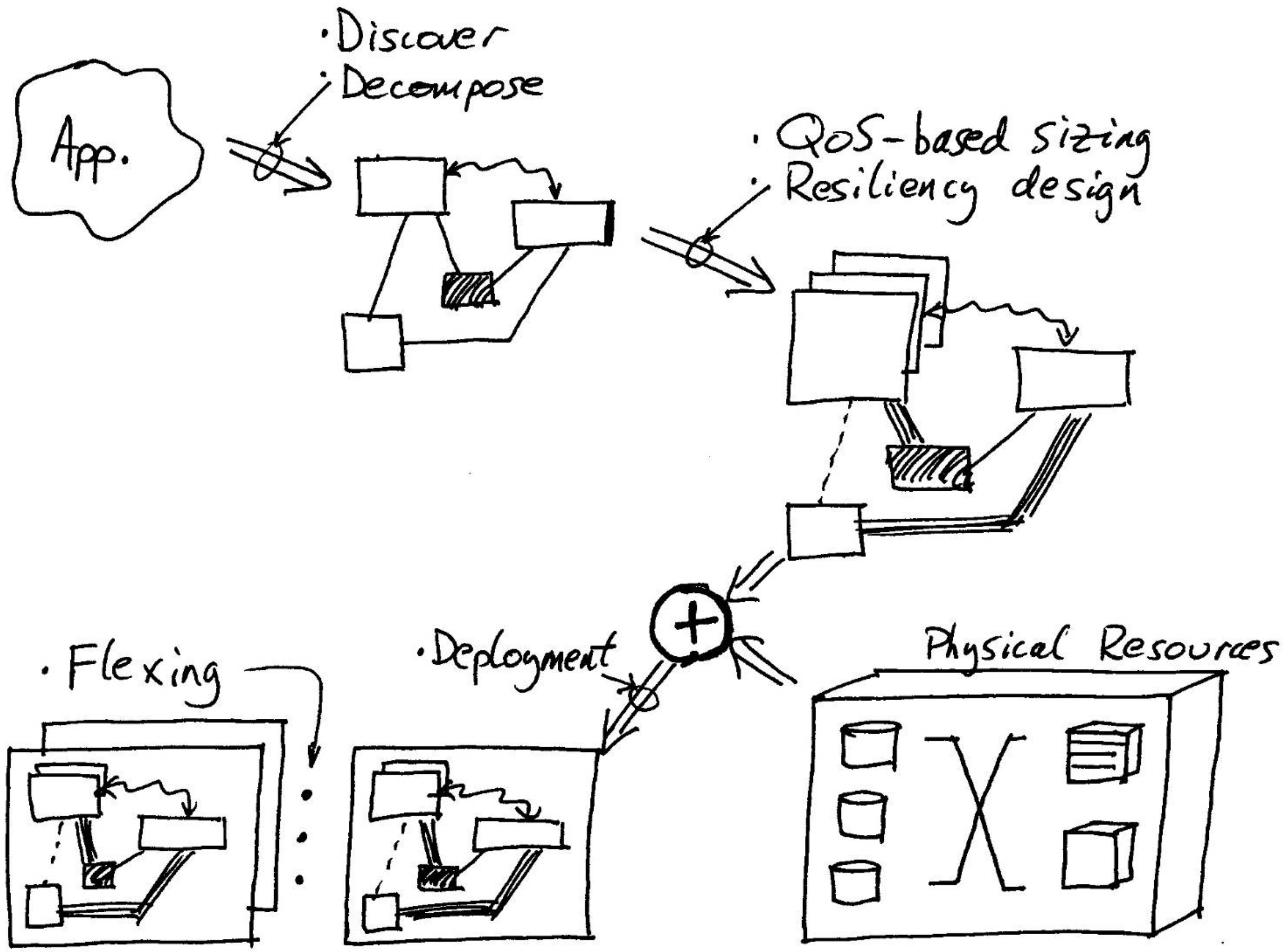
Utilification: the process
QoS enforcement

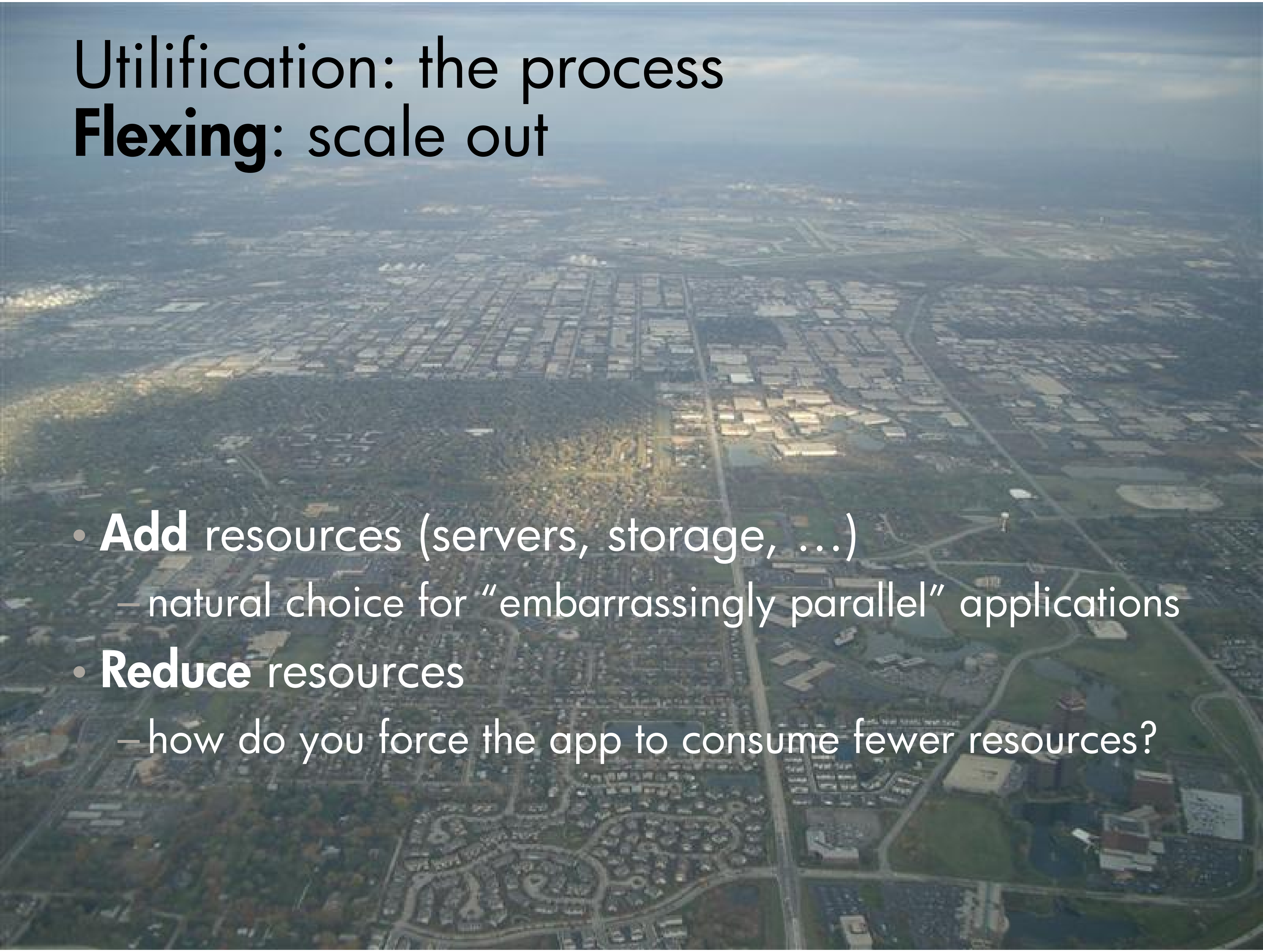
expect control loops to be nested!

how do you specify these?



Utilification: the process
Flexing





Utilification: the process
Flexing: scale out

- **Add** resources (servers, storage, …)
  - natural choice for “embarrassingly parallel” applications
- **Reduce** resources
  - how do you force the app to consume fewer resources?

Utilification
Flexing: scale up

- necessary if application can’t scale out
- **Migrate** application to faster system(s)
  - what if the app is still running?
  - what if the target configuration needs to be different?

Utilification: the process
Trust

- Requires belief in performance, resiliency, and security properties + the systems that provide them
  - technical solutions exist: these are not the hard part

- Opportunity: methods to build **trust**
  - will the proposed design work? → design audit
  - has the design been deployed? → deployment audit
  - has the design been altered? → runtime audit
  - was it adequate? → runtime audit

Call to action:
utilification needs you!

- Utility computing is coming
- The process of getting there is harder than the end point
- Help make it possible!

2004 version!

What do I think now?

What do I think now?

much the same, only more so ...

What's changed?

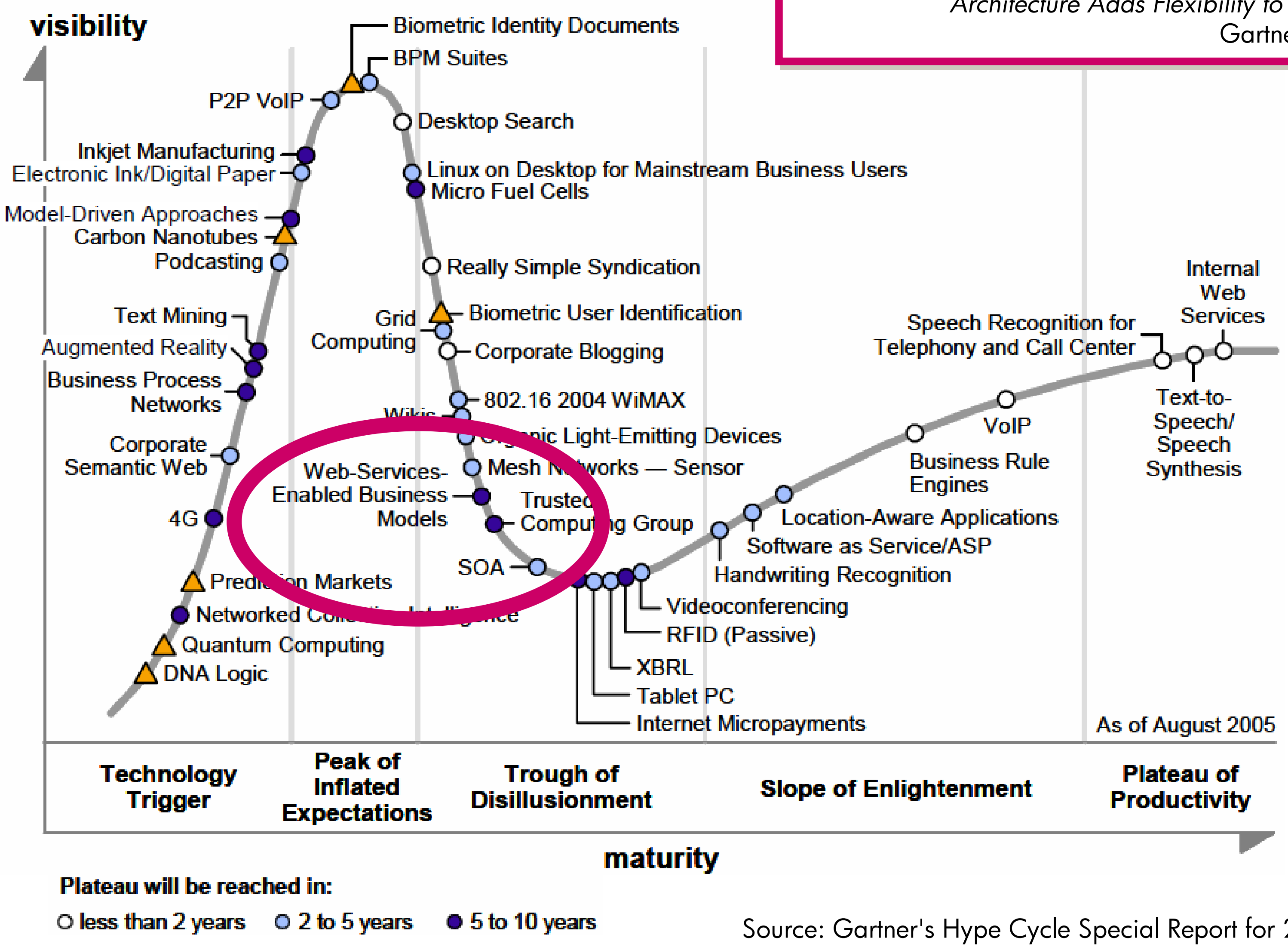
SOA is becoming real

What’s changed? SOA is becoming real

By 2008, “SOA will provide the basis for 80 percent of new development projects”

— S. Hayward, *Positions 2005: Service-Oriented Architecture Adds Flexibility to Business Processes*
Gartner, Inc. Feb. 2005.

Figure 2. Hype Cycle for Emerging Technologies, 2005

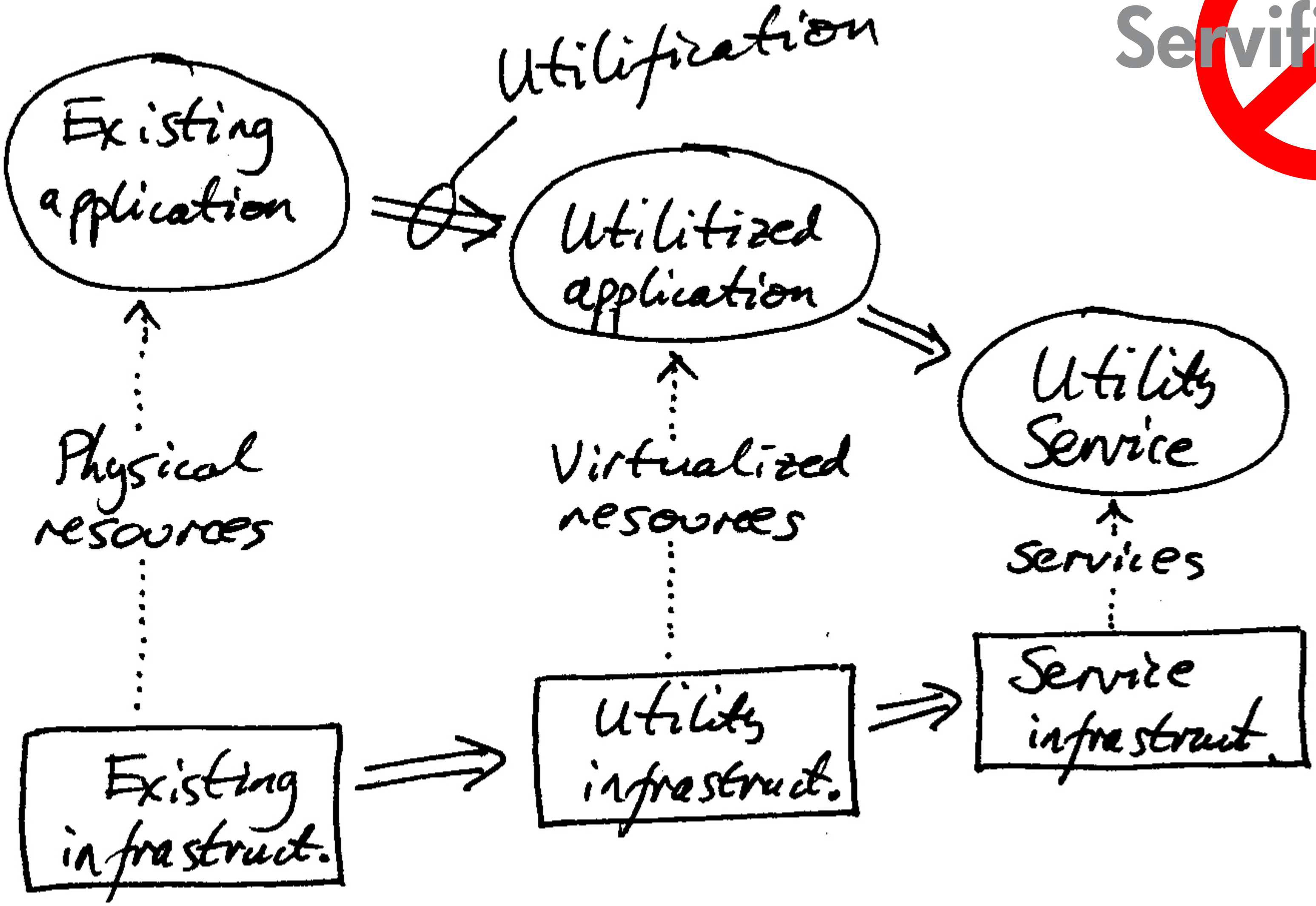


Source: Gartner's Hype Cycle Special Report for 2005, Aug 2005, ID Number: G00130115

Utilification: the process

➔ don't stop at the "utility" stage

Servification



Utilification: the process

→ turn applications into services

- Utilification allows resource sharing
  - flexibility in scale and placement
  - decisions about <u>resource</u> priorities and allocations
- Service equivalent
  - focuses on service/client relationships, not resources

Utilification: the process

→ turn applications into services

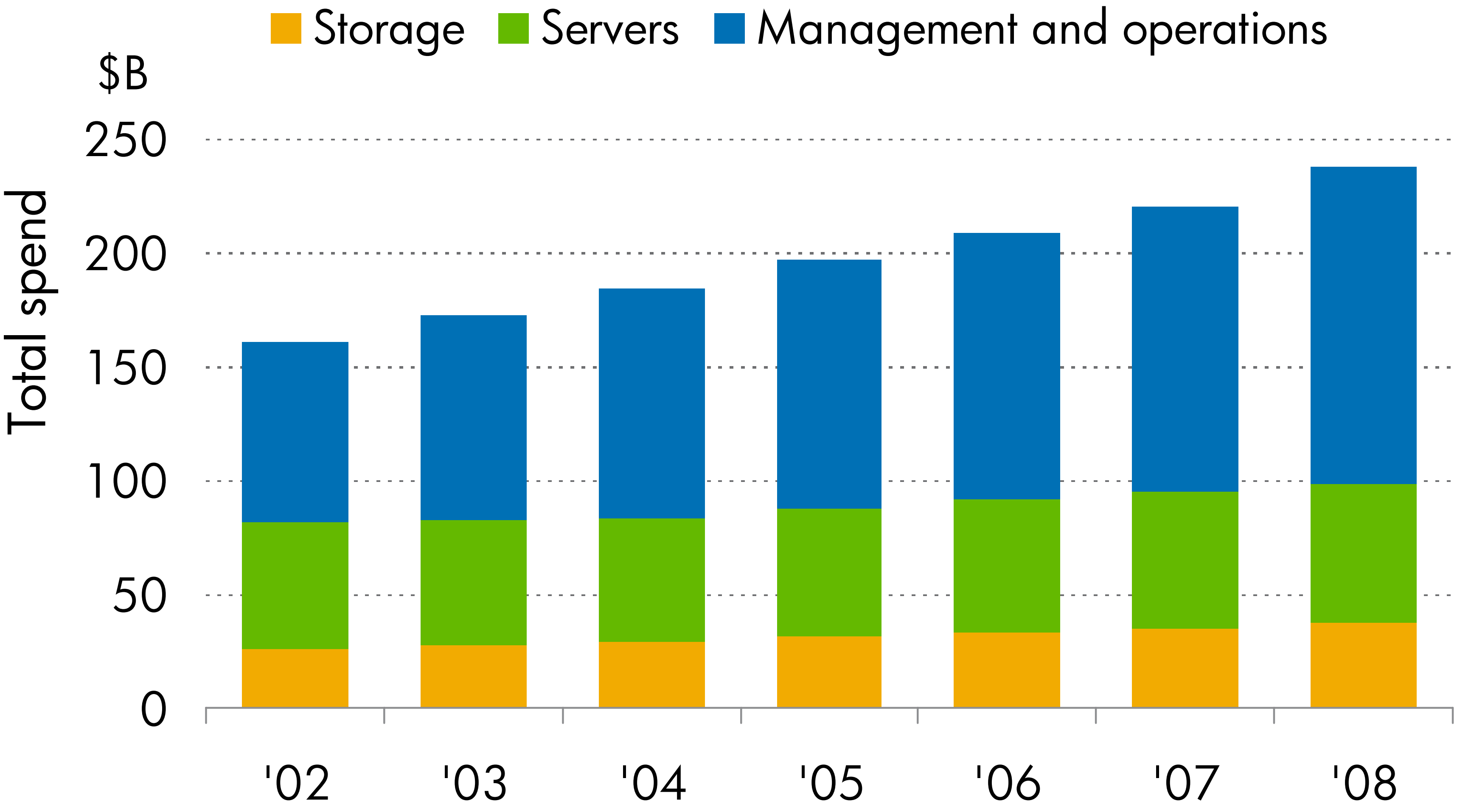
- Software as a service (SaaS):
  - reuse, multiple customers, asynchronous development, dynamic invocation, … ☺
- Running a service as a business (service provider)
  - forces decisions about service/client priorities and allocations

→ **economy-based approaches**

What's changed?

IT economics are becoming more important

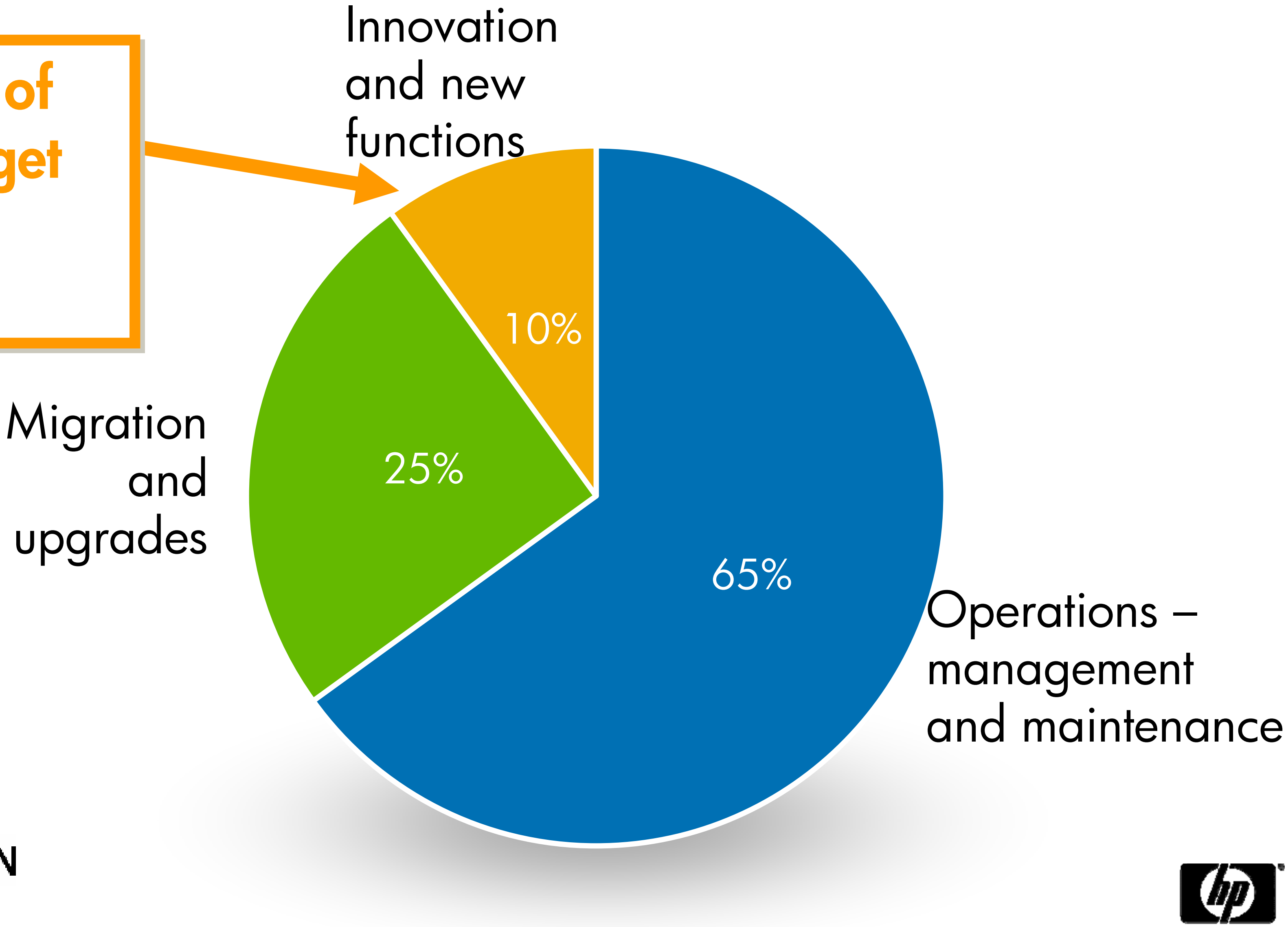
IT spending: OPEX growing 3x faster than compute capacity spend



Sources: IDC, "On-Demand Enterprises and Utility Computing: A Current Market Assessment and Outlook," July 2004; HP analysis.
Storage spend includes storage management SW.

IT spending: where does all the money go?

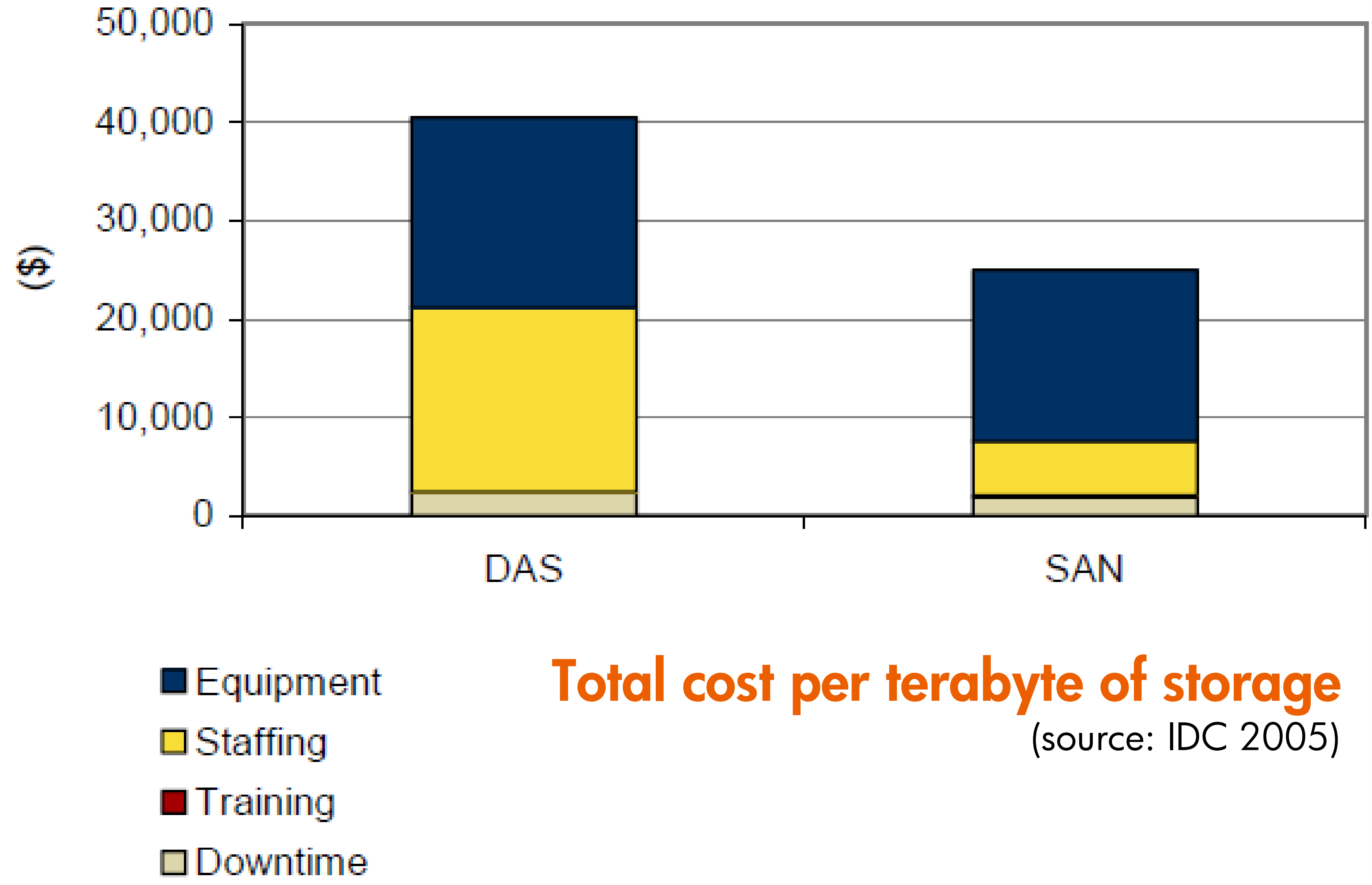
Only 10% of the IT budget is left for innovation



IDC Analyze the Future

ALINEAN

IT spending:
hardware vs administrator costs

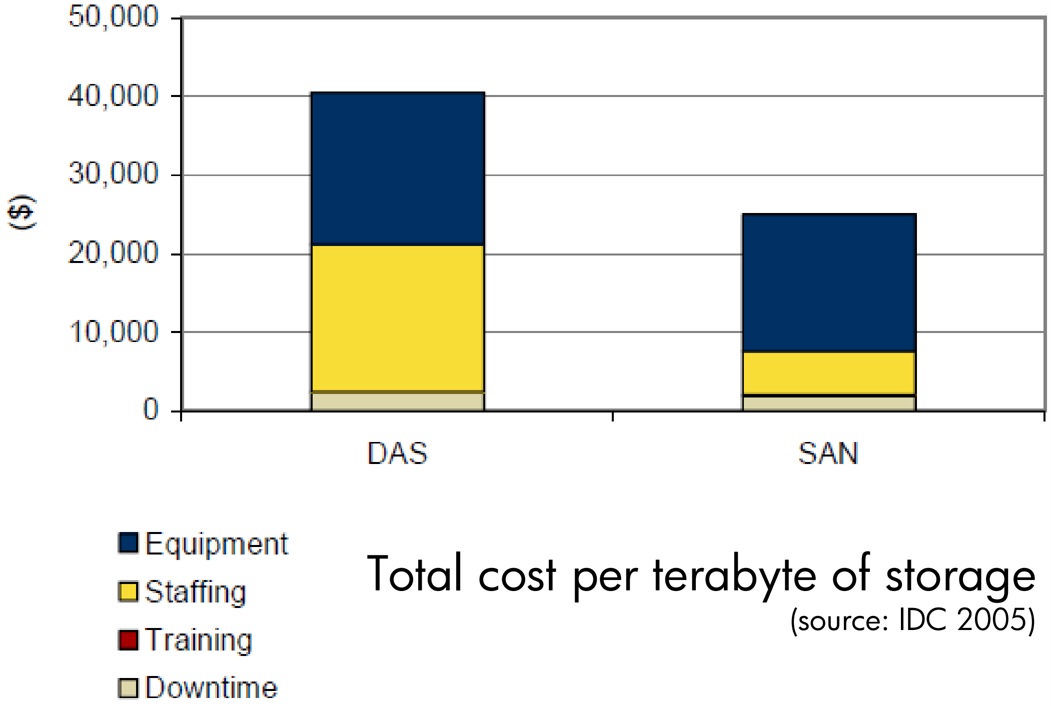


Total cost per terabyte of storage

(source: IDC 2005)

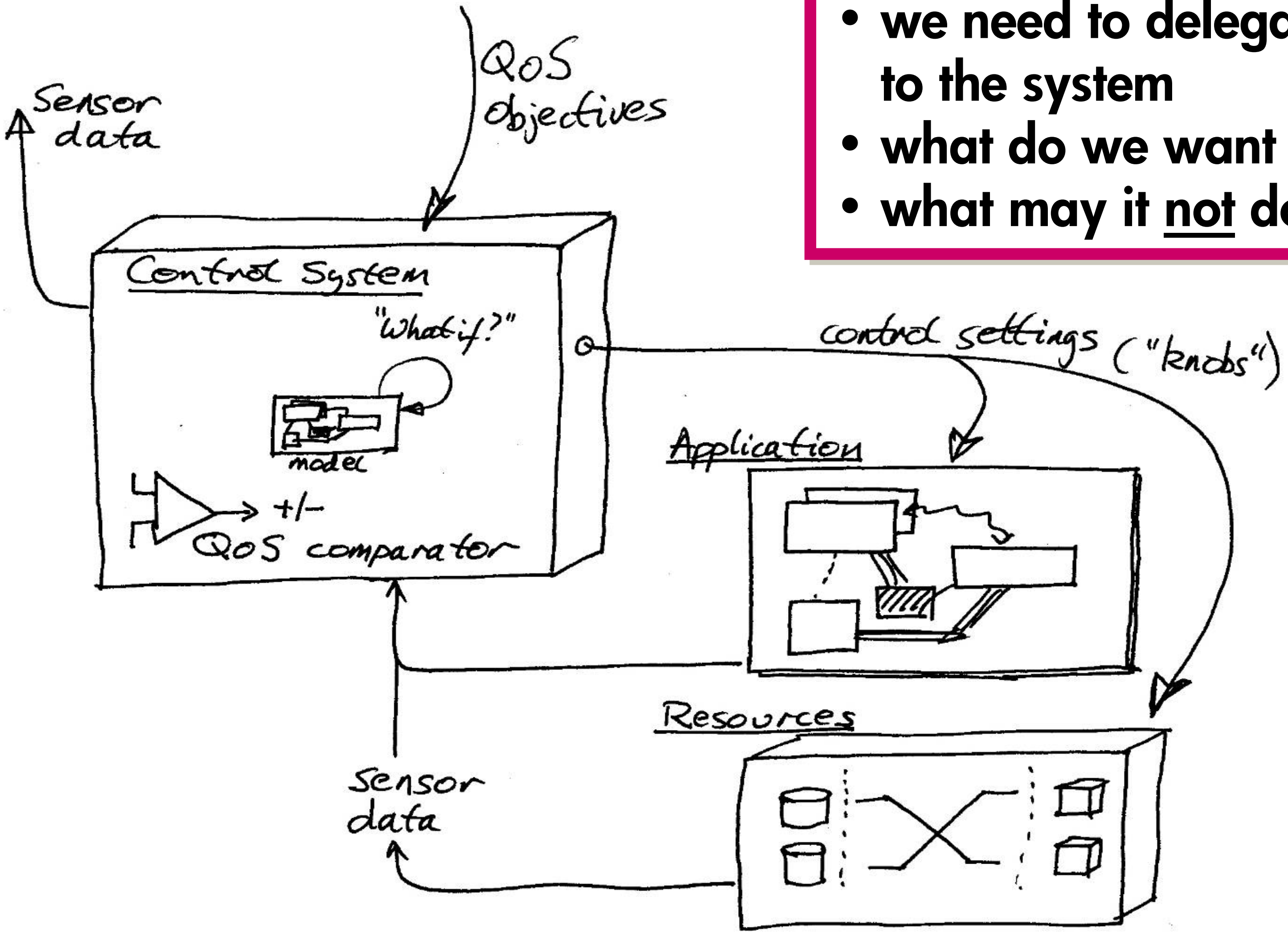
IT spending:
hardware vs administrator costs

- Storage costs are dropping
  - 1995: ~$5000/GB raw
  - 2005: $0.5/GB raw
- People costs are not:
  - 2004–5 admin salary: US$68k
  - growing ~0–6%/year [SAGE-USA survey]



Total cost per terabyte of storage
(source: IDC 2005)

IT spending: Solution: automation



For this to work:
- we need to delegate authority to the system
- what do we want it <u>to</u> do?
- what may it <u>not</u> do?

What's changed?

running IT like a business

We didn't spend $4.5 billion just to improve our business. We spent it to improve yours.

Today, HP is turning the world of I.T. on its head with our acquisition of Mercury and their powerful Business Technology Optimization software. With BTO, you make sure that good I.T. outcomes equal good business outcomes. Forward-thinking CIOs around the world are already using BTO to do just that. Join their ranks at OptimizeTheOutcome.com

Business results. The next big thing in I.T.

Good I.T. outcomes don't always equal good business outcomes. That's why HP is helping CIOs turn I.T. on its head with Business Technology Optimization software to run I.T. like a business. Learn how at OptimizeTheOutcome.com

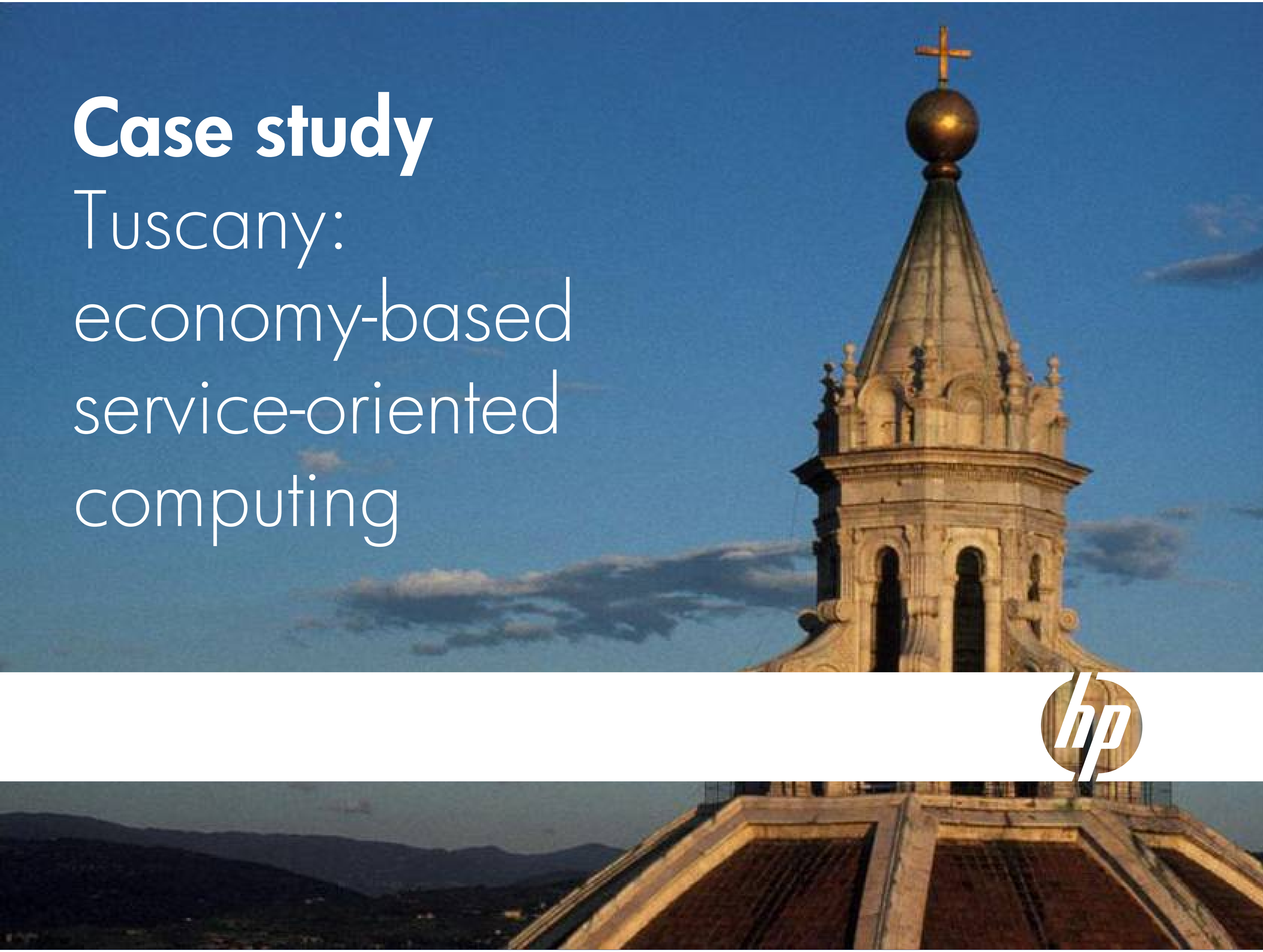
THE WALL STREET JOURNAL.

Running IT like a business

Optimizing the outcome of SOA

SOA governance

	Initiate	Build	Operate
Providers	Will others see value in my services?	Can other groups develop against my services?	How do I manage changes without disrupting my consumers?
Consumers	Can I find and trust services?	How do I develop/test against services out of my control?	Do I get the expected quality of service?
CTO office	Does the enterprise work on the right services?	Can I establish a repeatable implementation process?	Can I control services in production?



Case study

Tuscany: economy-based service-oriented computing

Tuscany key ideas

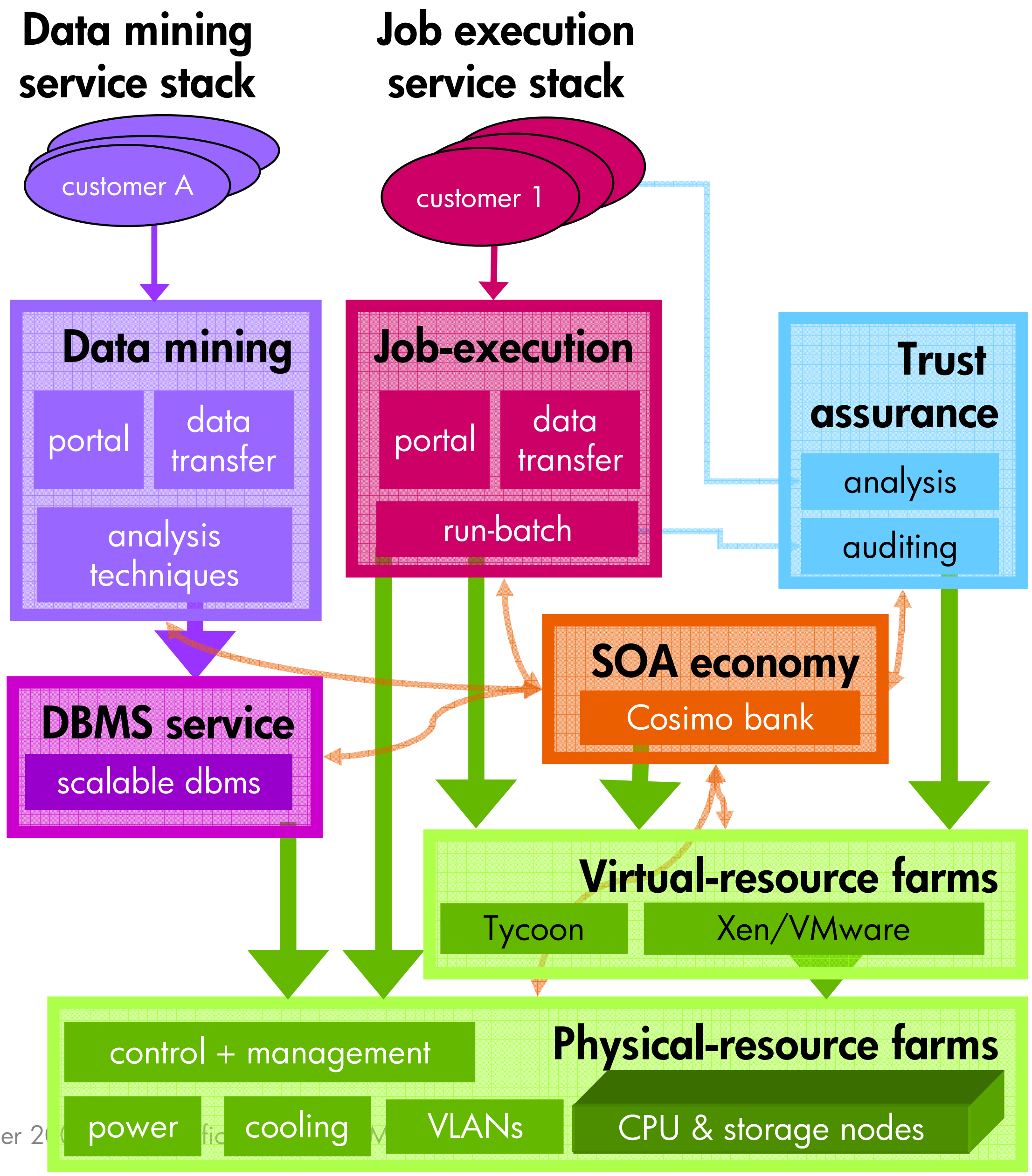
- Self-interested service providers

  ➔ SOA + economic rewards to steer behavior

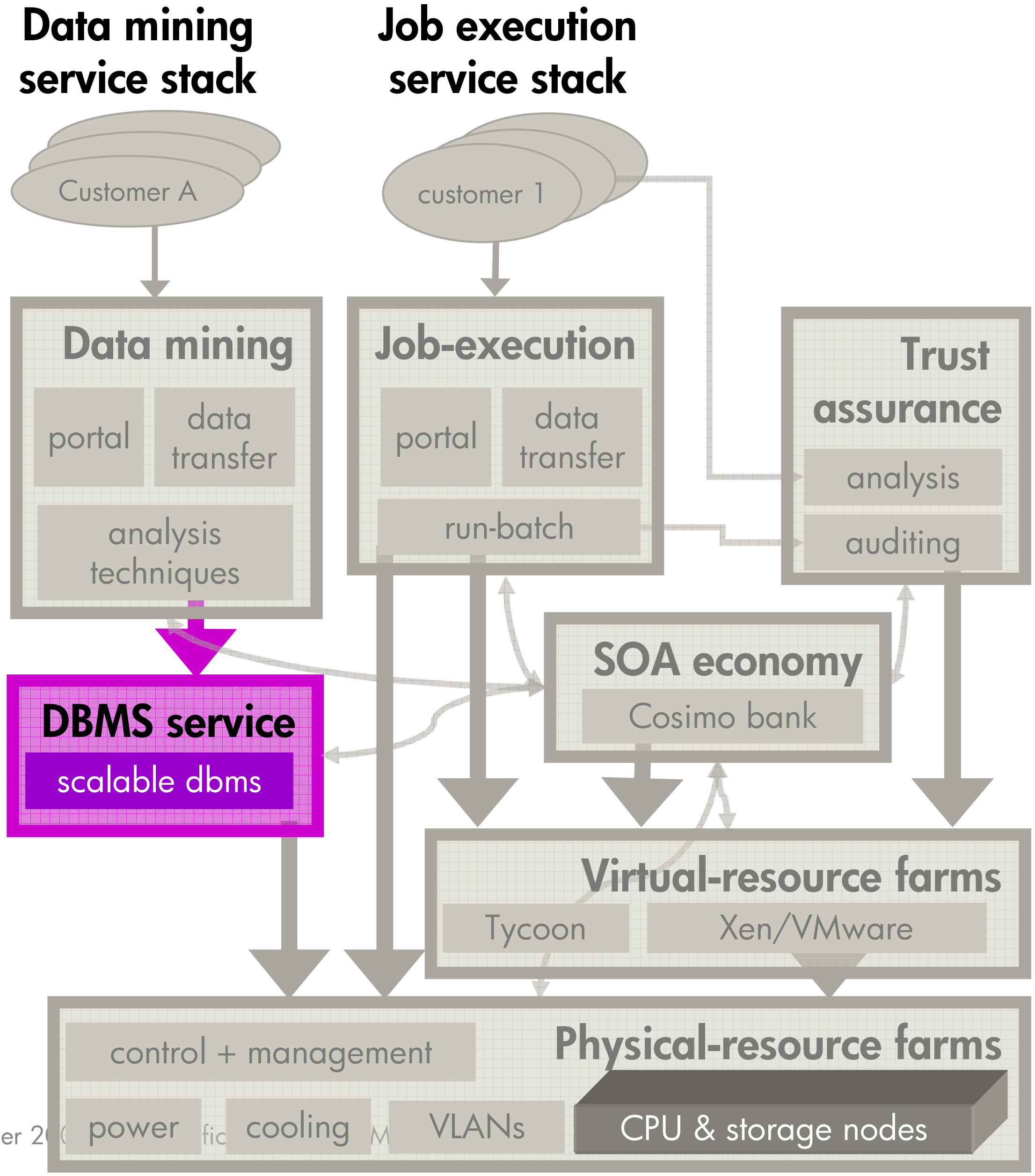
- Automated self-management

  ➔ cost-effective, lights-out, agile operation

Tuscany ecosystem



Tuscany ecosystem
Prato: dbms-on-demand service



Tuscany ecosystem
Prato: dbms-on-demand service

- a self-managing service provider

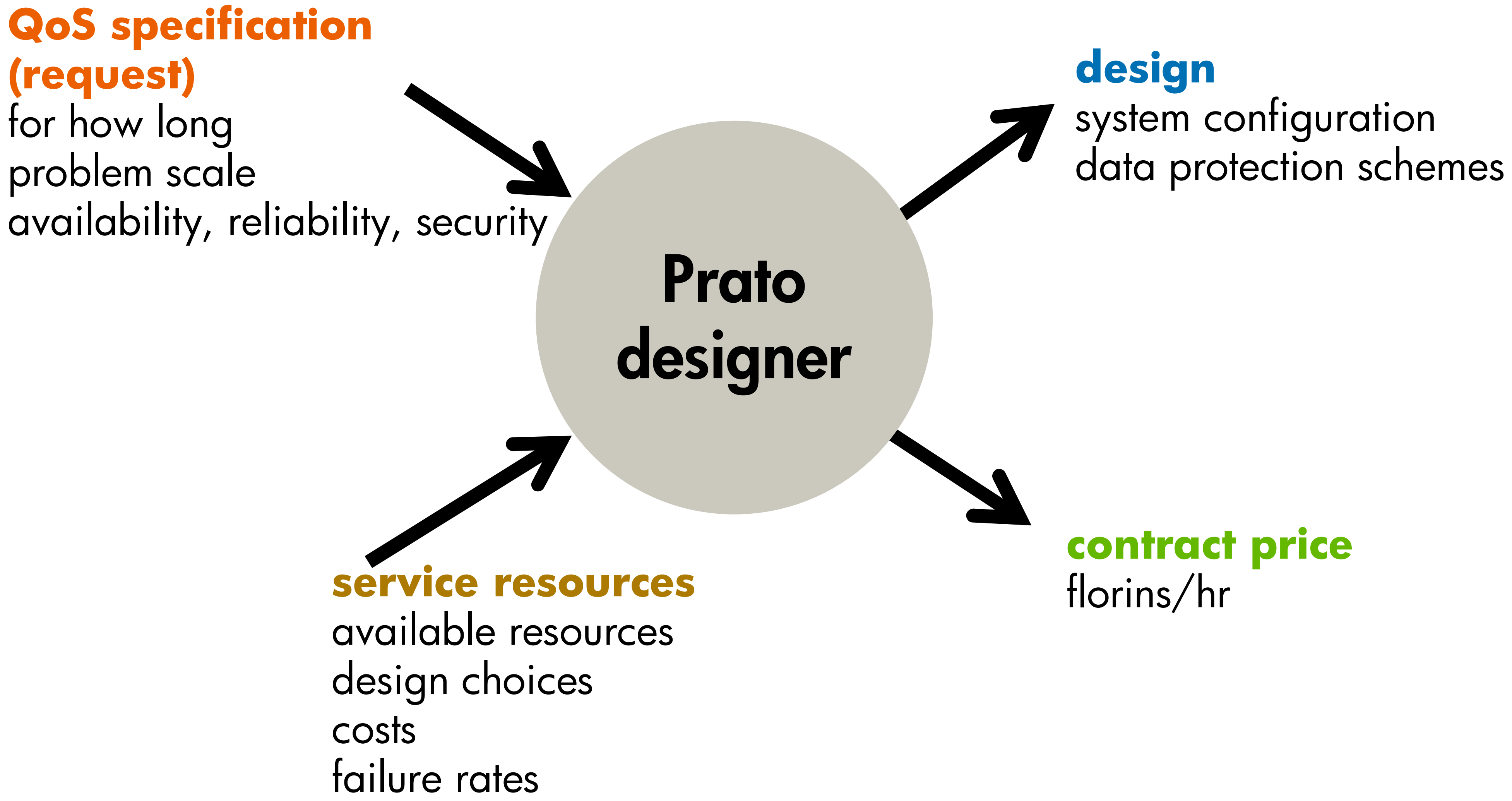
- that offers a dbms-on-demand service
  - 2x capacity for 3 days!

- by providing each client with their own virtual dbms appliance
  - hiding the complexity of:
    - setting up the service
    - managing the service (e.g., if it breaks)

Prato
research focus

1. representing customer needs
   - expressing what they need without dictating the solution
2. translating needs into implementation choices
   - automatically selecting between different designs
3. automating service provider management
   - lights-out self-management is the end goal
4. composing service providers
   - Prato is just one service provider: how does it integrate with others?

Prato research focus
automatic QoS spec ➔ service design



QoS specification (request)
for how long
problem scale
availability, reliability, security

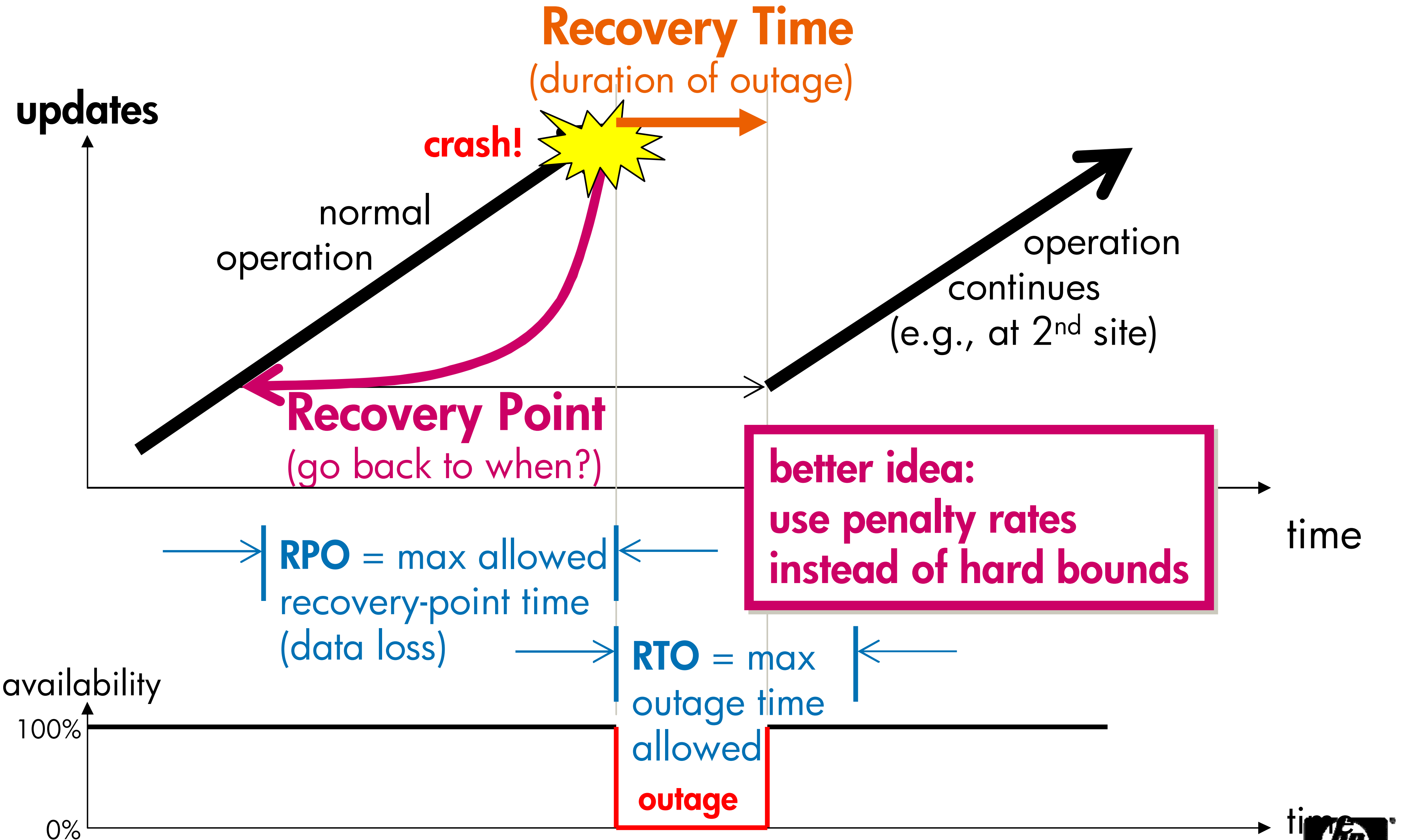
service resources
available resources
design choices
costs
failure rates

Prato designer

design
system configuration
data protection schemes

contract price
florins/hr

Designing data protection
Anatomy of a failure



Recovery Time
(duration of outage)

operation continues (e.g., at 2nd site)

Recovery Point
(go back to when?)

better idea:
use penalty rates instead of hard bounds

RPO = max allowed recovery-point time (data loss)

RTO = max outage time allowed

Prato research focus
automatic QoS spec ➔ service design

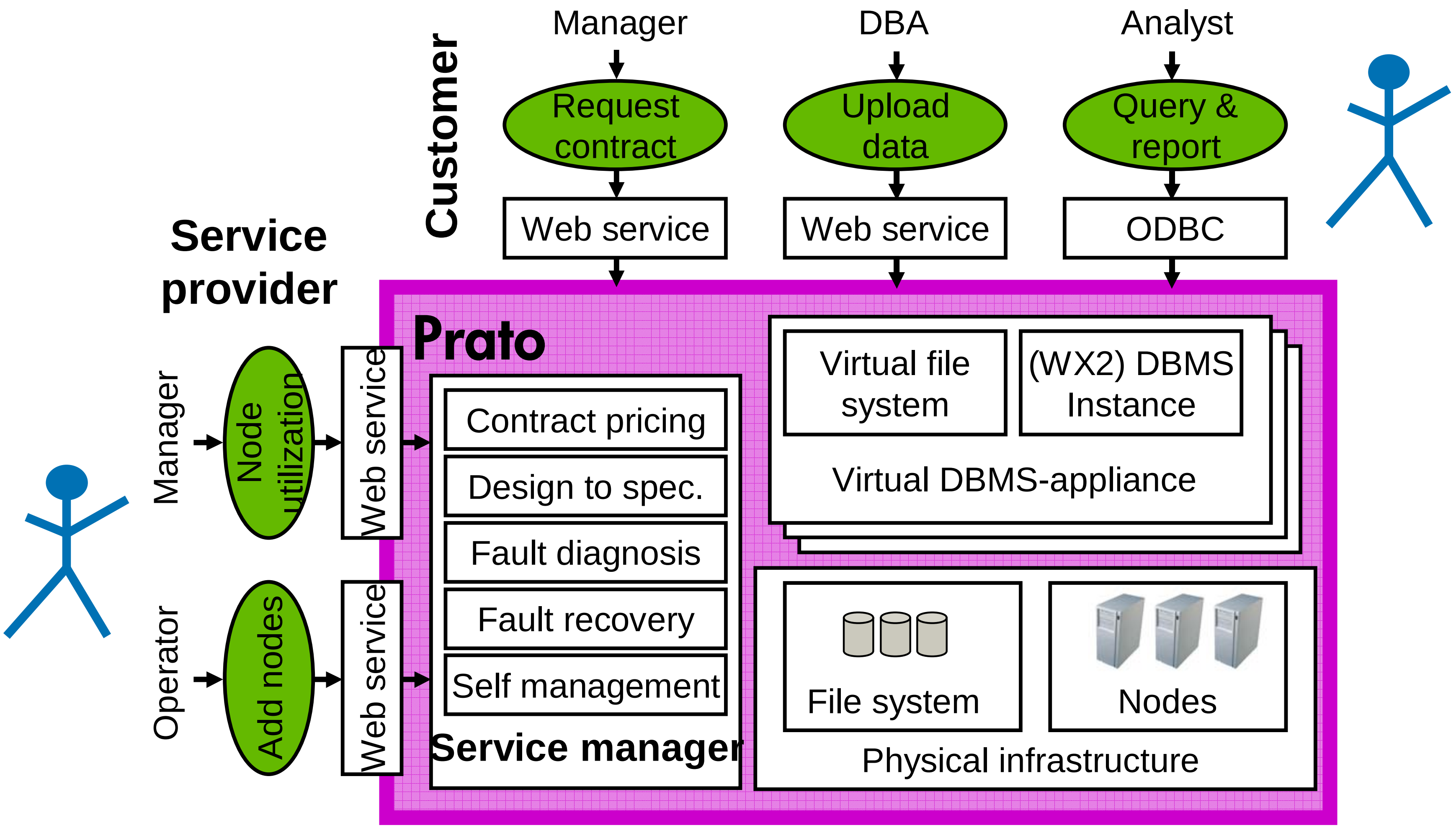
Client specifies:

- dbms size
  - RAM (GB), disk (GB)
  - "in-memory" ➔ high-performance
- outage and data-loss penalty rates
  - florins/hour
- contract start-date + duration
  - date, hours
- data-isolation breach penalty
  - florins/occurrence

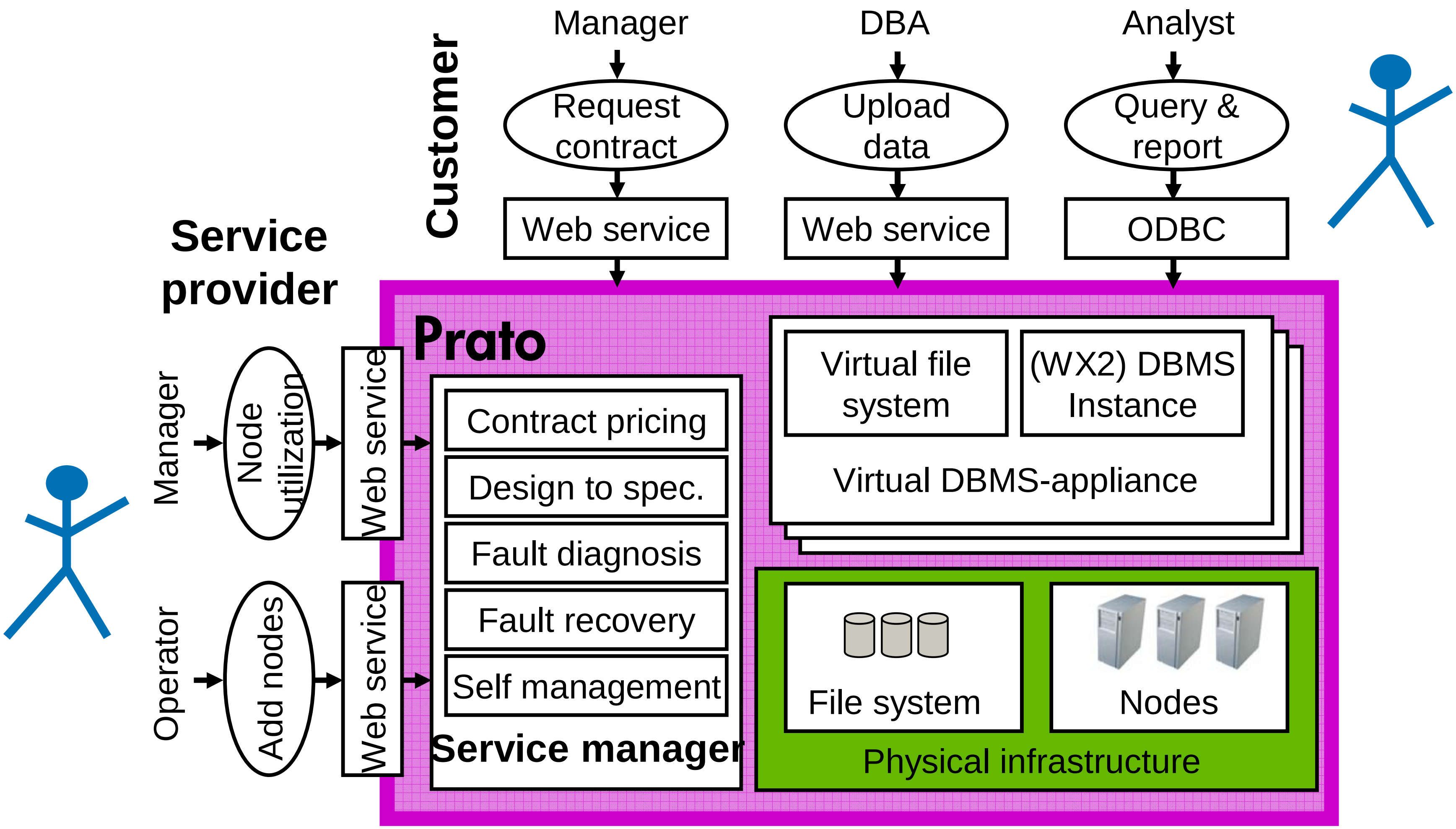
Prato chooses:

- capacity + speed
  - number of nodes, amount of disk, amount of RAM
- data-protection approach
  - mirrored disk/dbms RAID-5
  - reload from remote/local copy
  - snapshot frequency
  - dedicated/hot spare nodes
  - cold/hot standby dbms
- contract price
- security-isolation approach
  - air-gap; VLANs
  - virtual machines; dbms-protection

Prato service
design driven by use cases

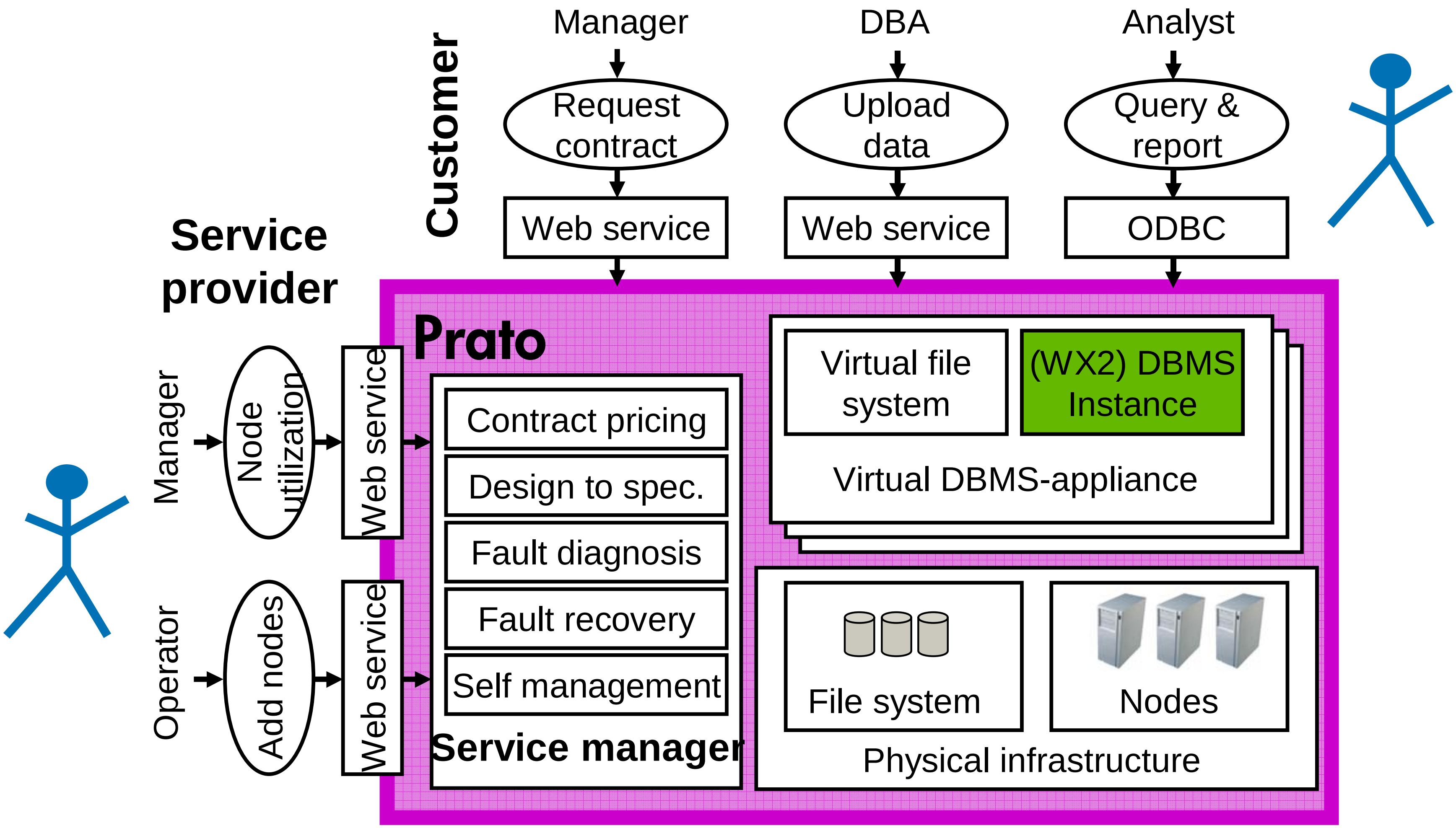


Prato service
hardware and software

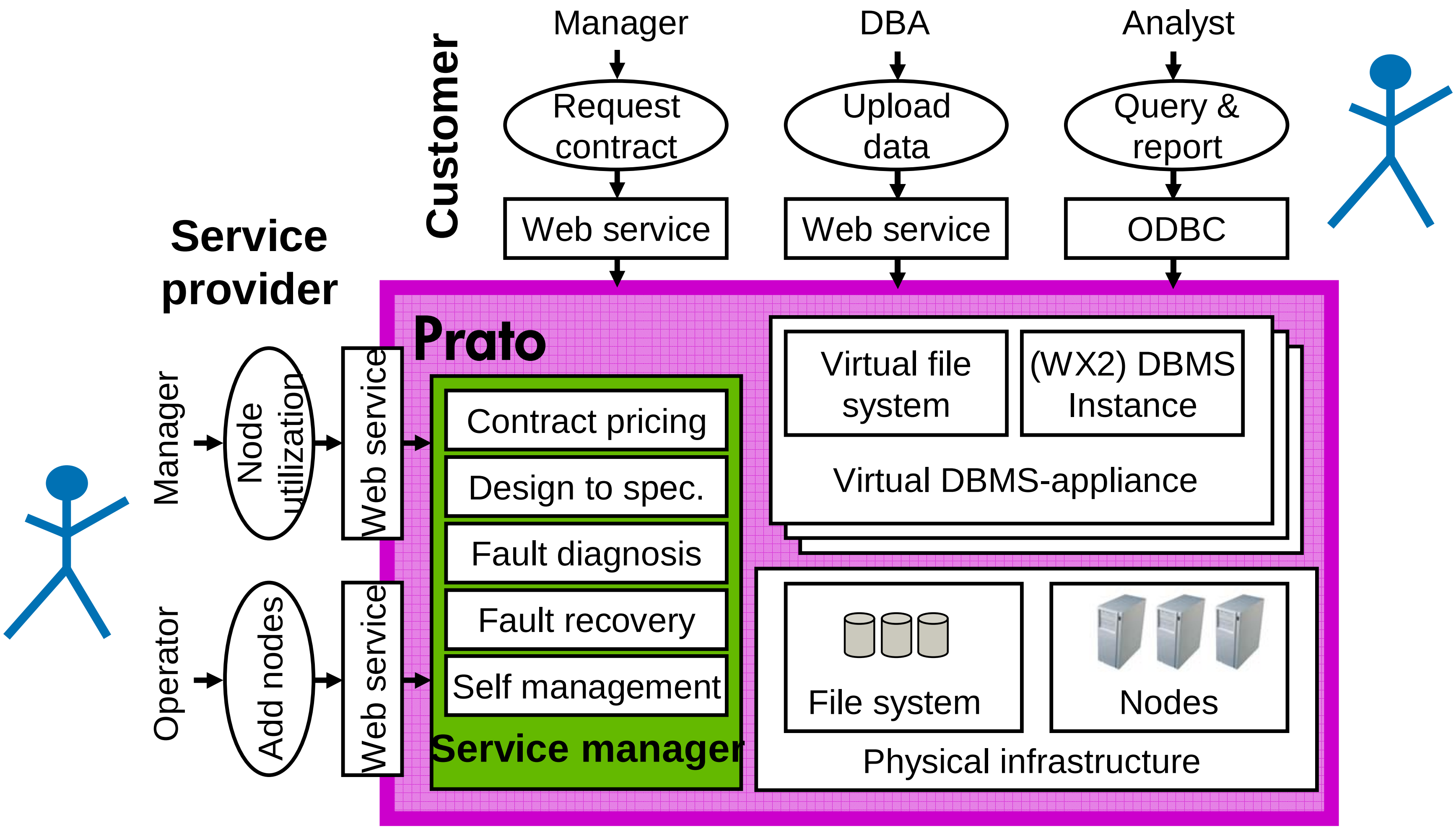


Prato service

DBMS = Kognitio WX2



Prato service
Enigmatec EMS service manager

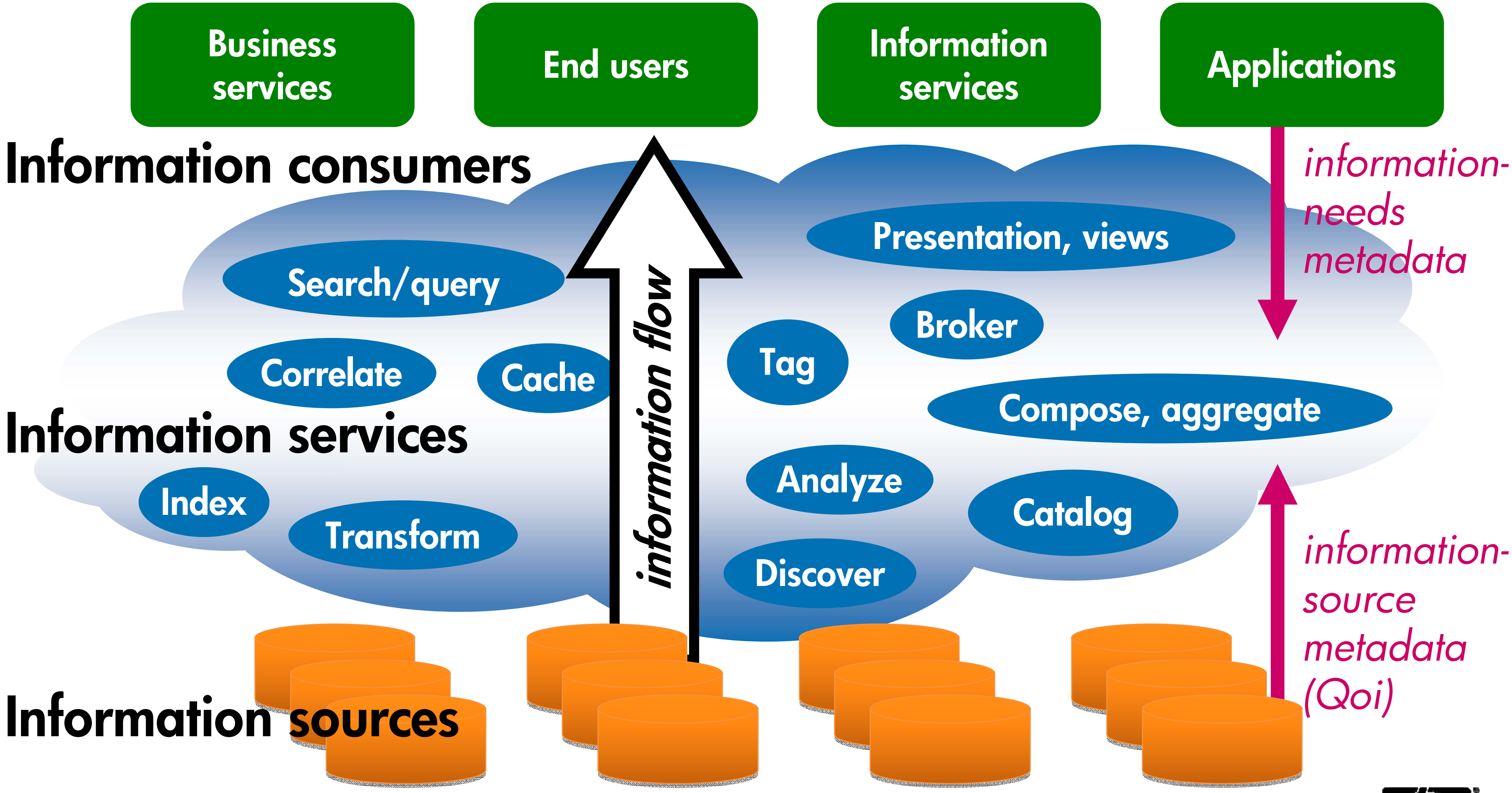


What’s changed?

need Quality of information (Qoi), not just Quality of Service (QoS)

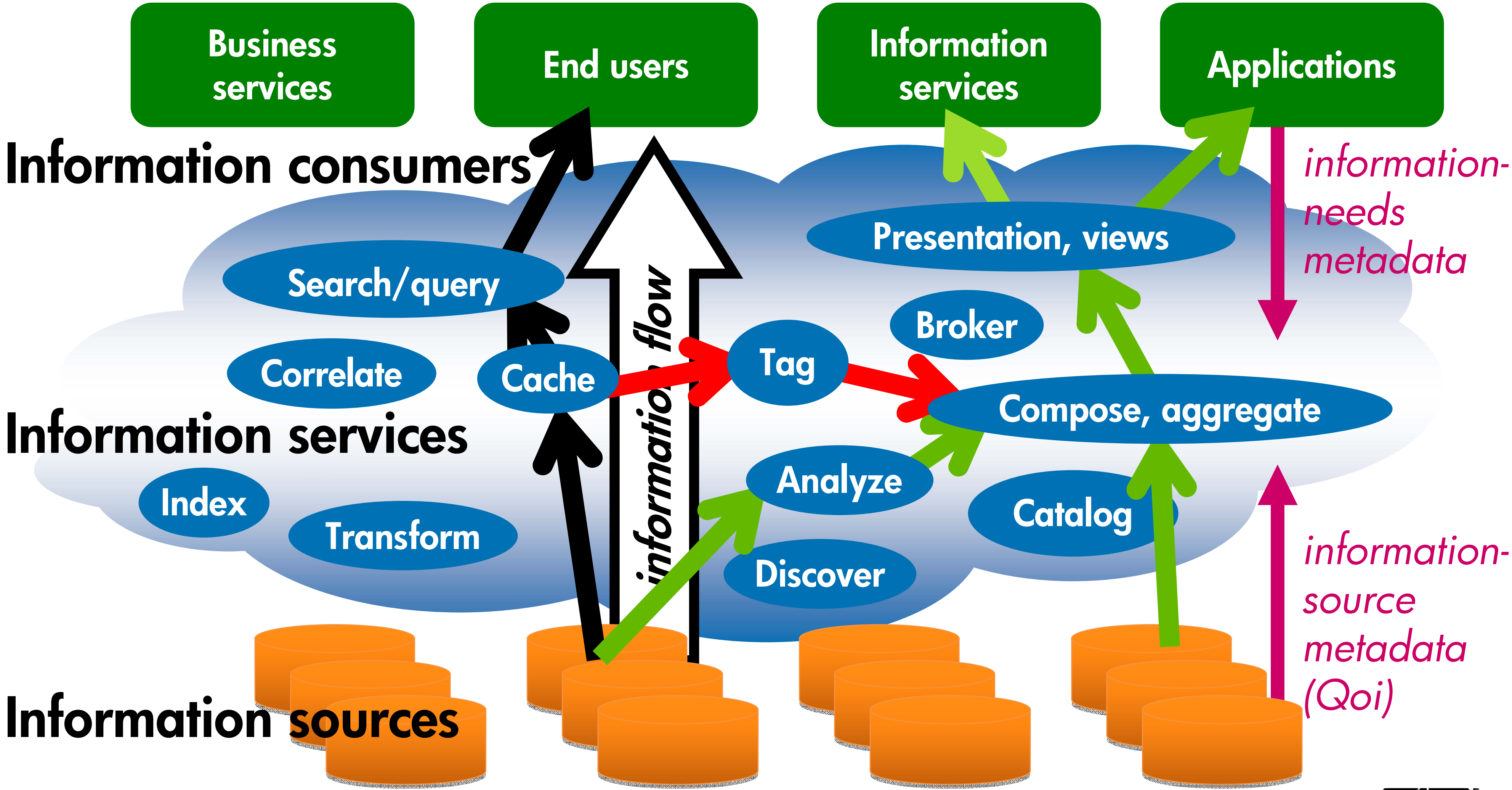
Quality of information

In-the-middle services



Quality of information

In-the-middle services



Quality of information

Key observation

- **QoS** is great – but only addresses half the problem
  - is the service available? fast? cost-effective?
- **Qoi = quality of information**
  - is the information fresh? complete? accurate? clean?
  - what was its provenance? is it original?
  - is it believable? why?

➔ key idea: ask for the Qoi you need …
and get the information system to deliver it
… **automatically**

Quality of information
A few research opportunities

- how to build the processing DAG?
- how to express Qoi?
  - what metrics to use? how to measure them?
  - suppose you had the metrics – what would you do?
- how do processing steps affect Qoi?
  - can we predict their effects?
  - can we design processing DAGs to meet Qoi goals?
- how much Qoi is needed?
  - who decides? how?

Summary

Summary

- Utilification + SOA: still a good idea!
  - but: many “opportunities” remain

- Trust, trust, trust
  - automation requires delegation
  - vital to understand what QoS is wanted
  - predictability, reassurance

- Going beyond QoS → Qoi

→ **This is all middleware’s turf!**