Cloud Computing: Infrastructure and Application Architecture Considerations

Uli Hitzel, Cloud Architect (ASEAN) – August 1st, 2012

Acknowledgements: this presentation leverages material from IBM cloud computing presentation decks, 360 degree architect session (Sugandh Mehta, Bernard Glasner, Harirajan Padmanabhan) and “Architecting applications for the cloud @ IBM developer networks” by Peter Bell (Systemsforge)

This handout is a modified version of the original presentation deck.

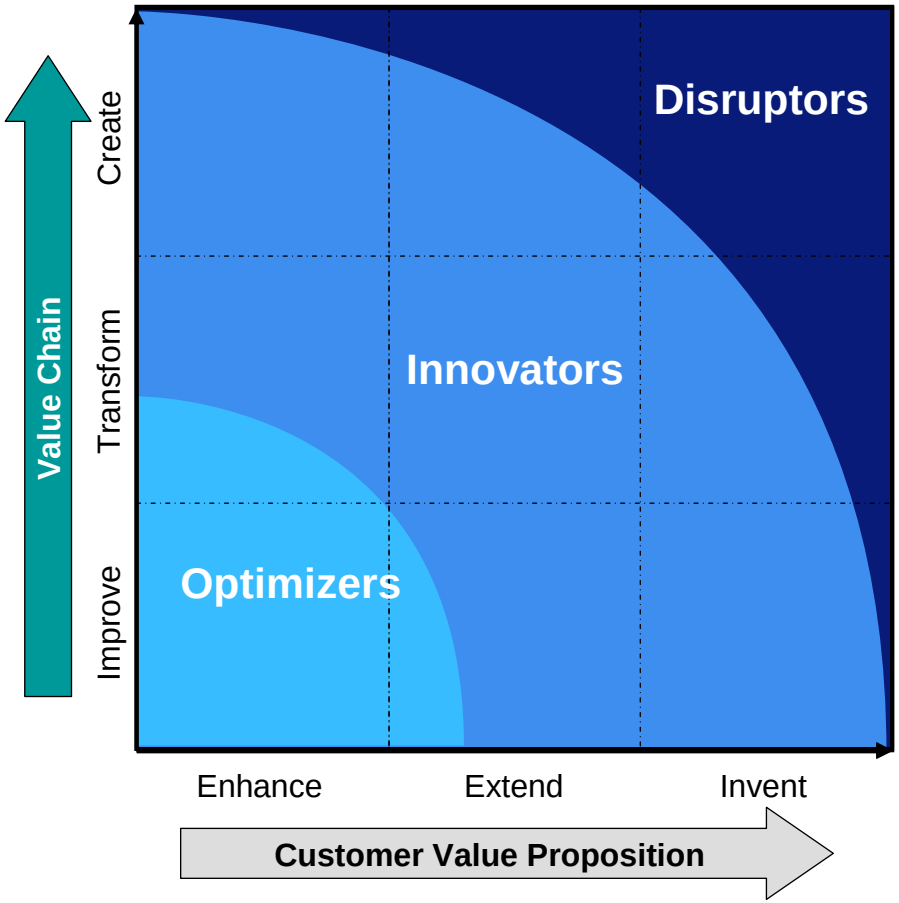
About Uli

- Architect (ASEAN)
- projects at banks, telcos, service providers & in public sector
- Technical Expertise
  - **virtualization**
  - **automation**
  - **business process integration**
  - **web technologies**

Session Agenda

- **Introduction**
- **Infrastructure and Platforms**
- **Cloud Applications**

Cloud helps clients to…



Save costs - profit from flexibility in IT spends (CAPEX/OPEX)

Increase agility and organizational efficiency – faster time to market

Innovate and create new business models – additional revenue streams

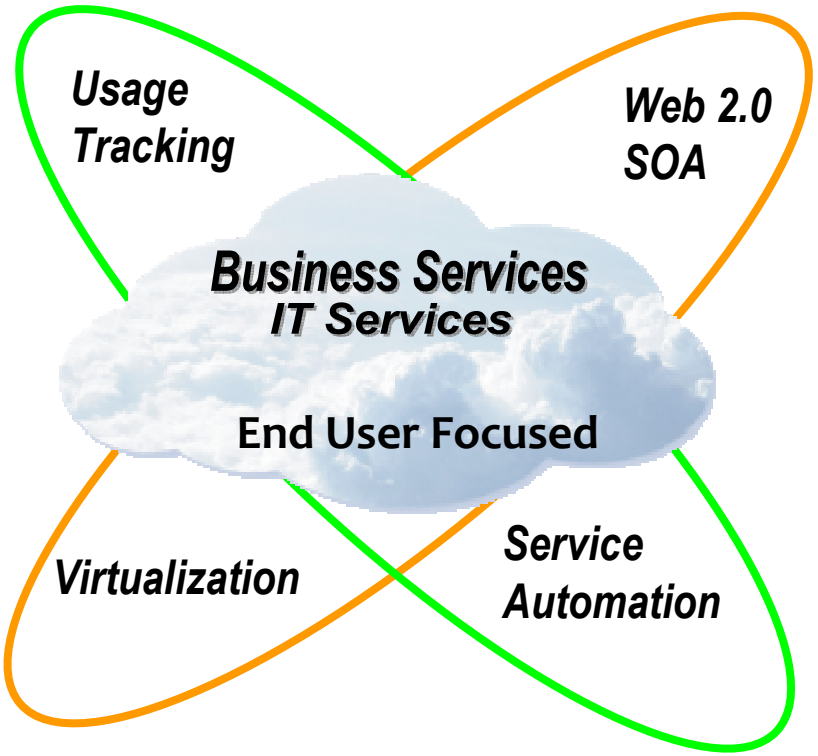
Disrupt the market – coming up with services that only work on the cloud model

Cloud Computing Definition

Cloud computing is a new consumption and delivery model inspired by consumer internet services.

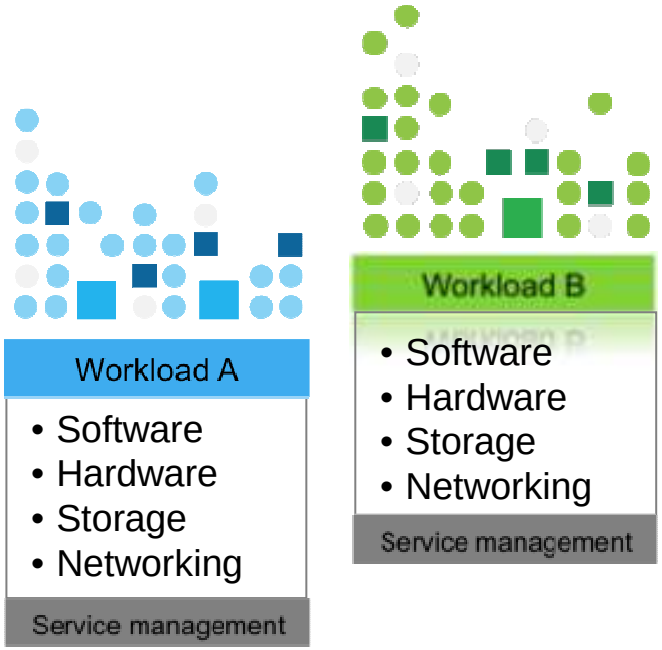
Essential characteristics:

1. **On-demand self-service**
2. **Broad network access**
3. **Location independent resource pooling**
4. **Rapid elasticity**
5. **Measured Service**

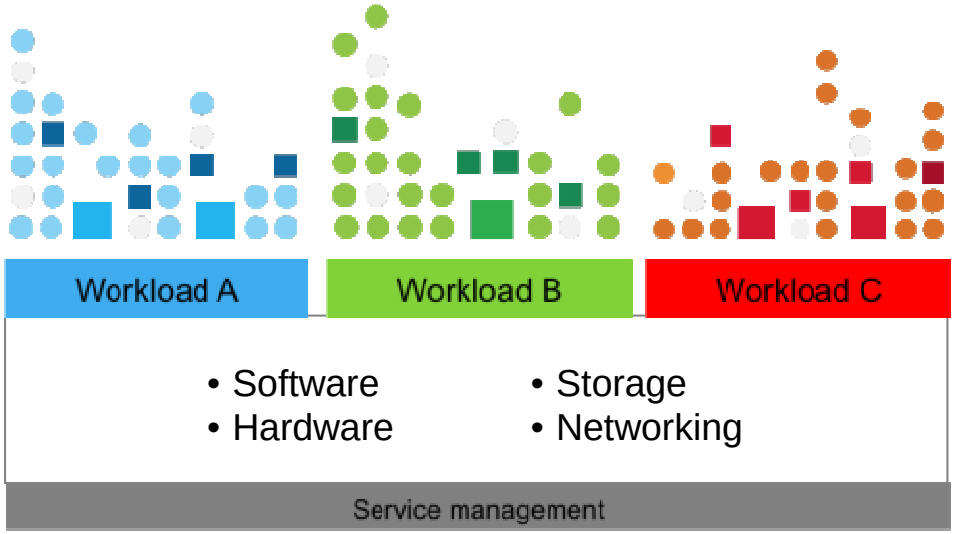


What's different with cloud computing

Without cloud computing



With cloud computing



- Virtualized resources
- Automated service management
- Standardized services
- Location independent
- Salability & Elasticity
- Self-service

Shopping Mall – B2B IaaS Cloud

- Service provider owns the system
- Business rents units to offer services to their clients → consumers
- Shared resources (building, elevators, parking, washrooms)
- Shared Services (security, cleaners)
- Country-wide, world-wide network of shopping malls

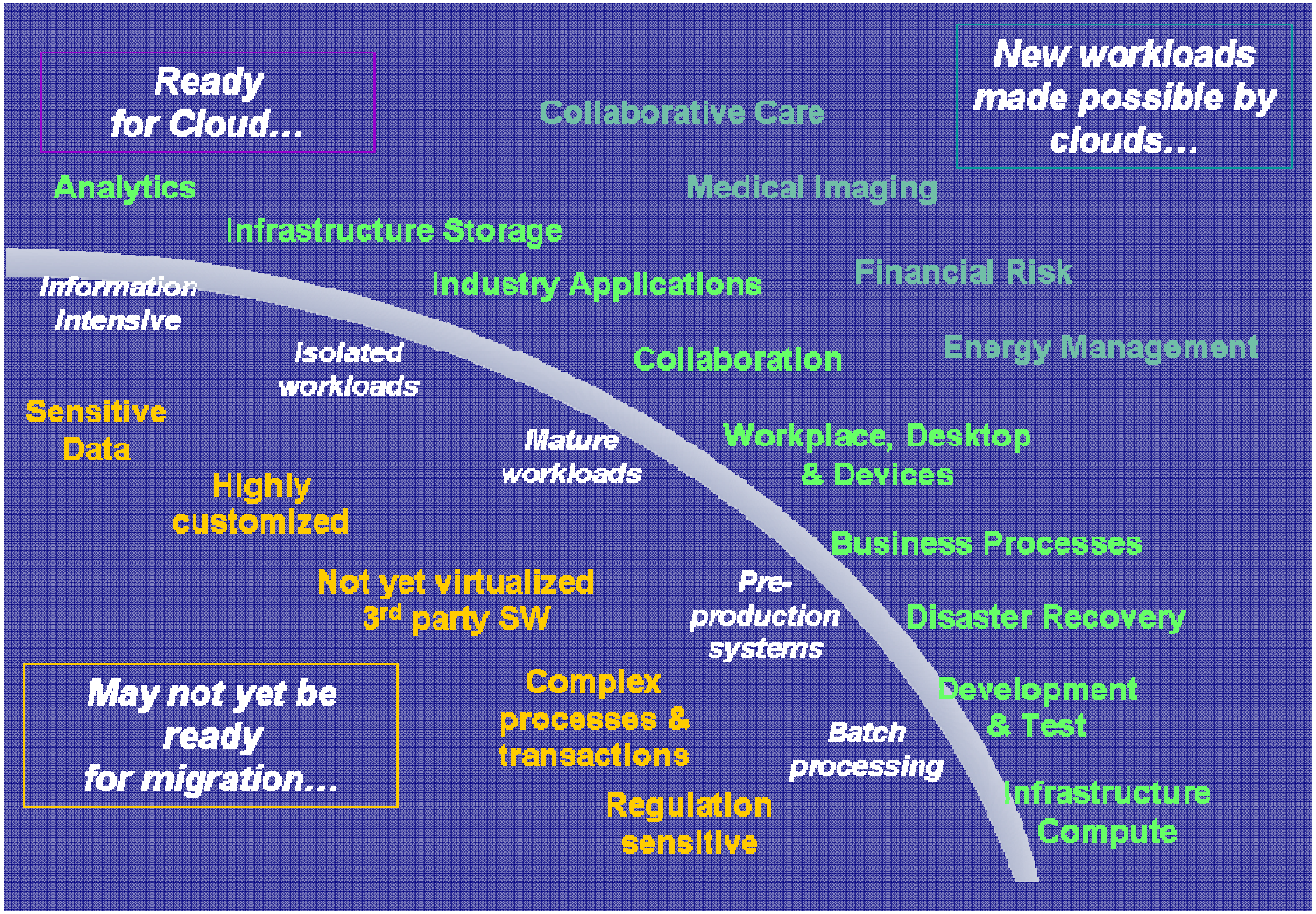
→ better ROI, larger audience, more flexibility & less risk for the shop owners

“One Size Does not Fit All”

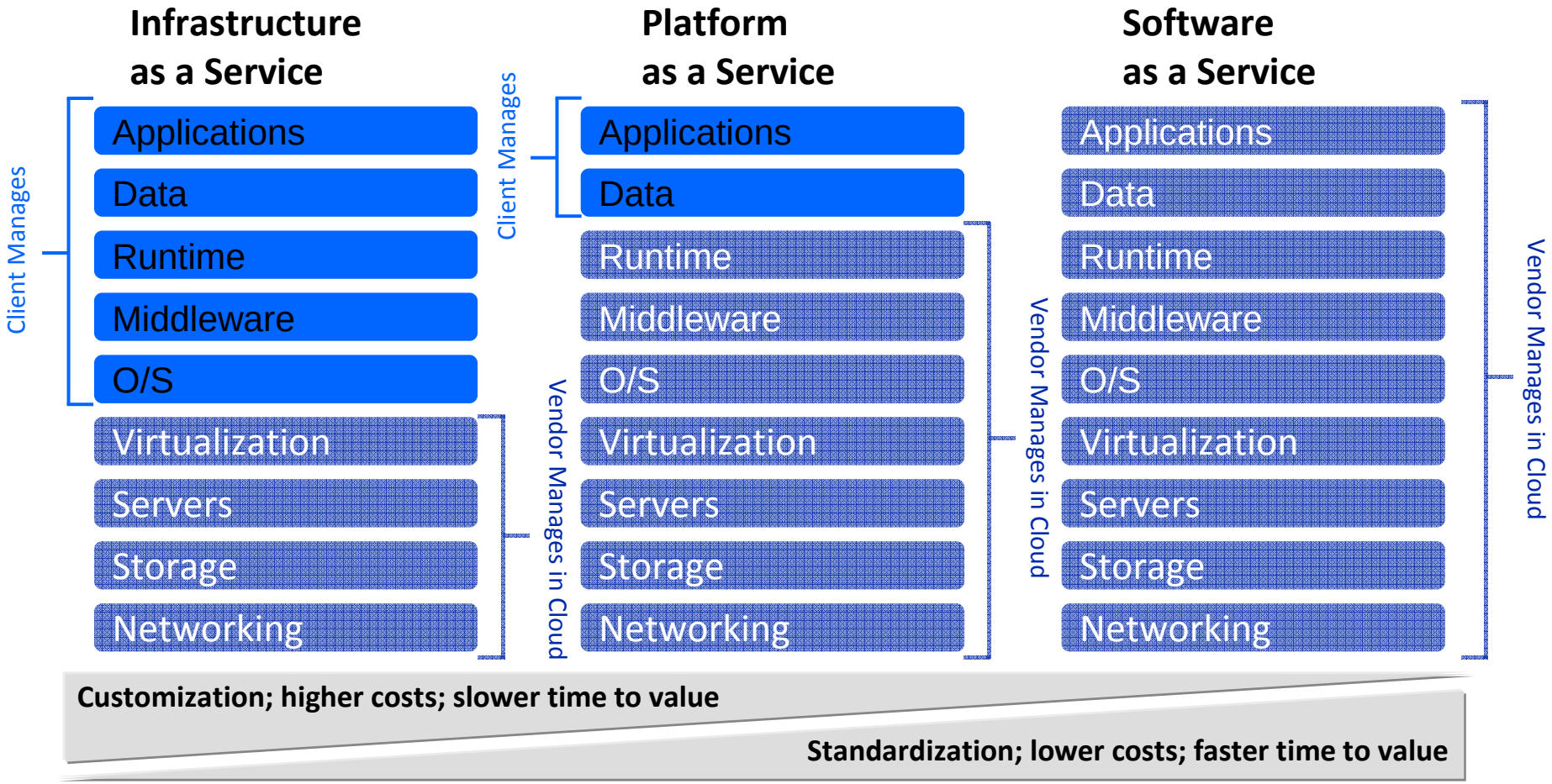
What exactly do we want to do?

- Workloads → which ones? → required functionality and performance?
- Service models
- Deployment models
- Organizational Motivations & Constraints

Workloads



Definition – Cloud Service Models



Service Layers

Software as a Service (SaaS)

Collaboration
CRM / ERP / HR
Industry Applications
Financials

- **CRM , ERP Applications**
- **Human Resources**
- **Analytics**
- **Financial Services**
- **Mail, Web Conferencing**

Platform as a Service (PaaS)

Web 2.0 Application Runtime
Development Tooling
Middleware
Database
Java Runtime

- Applications
- Middleware
- Development tools
- Java and Web 2.0 runtimes

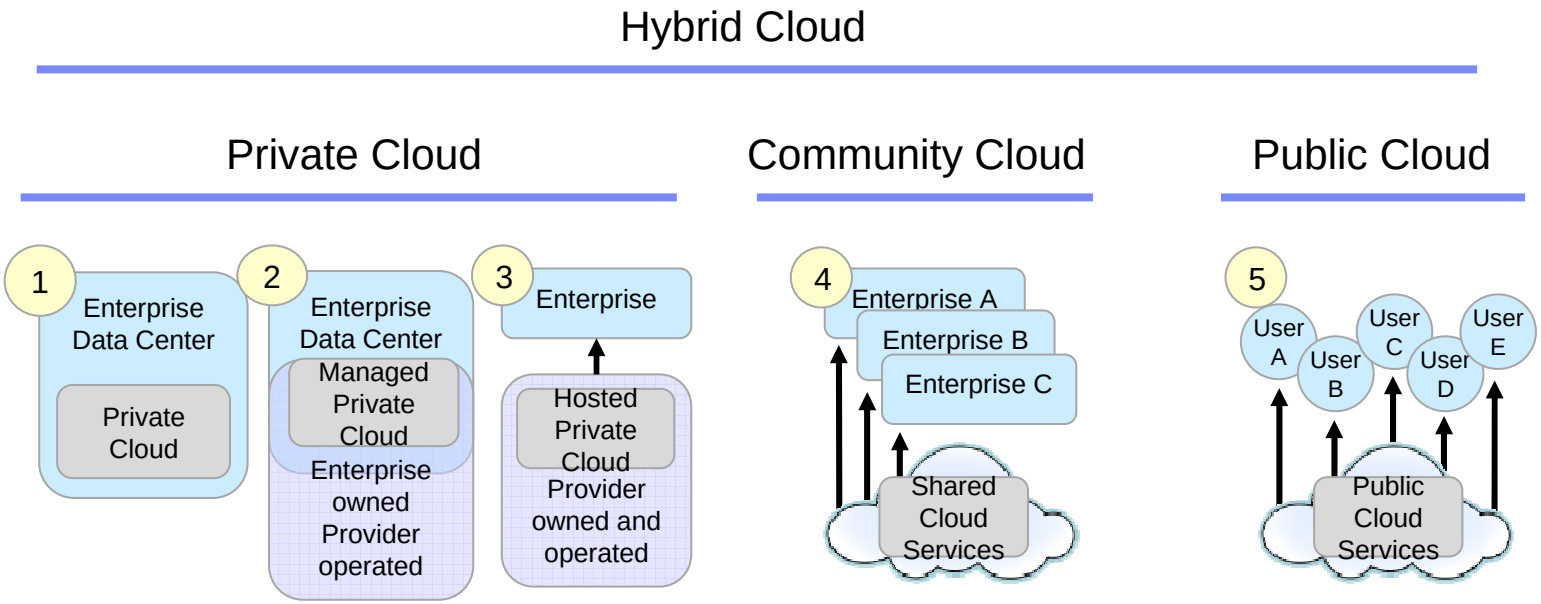
Infrastructure as a Service (IaaS)

Storage
Shared Pool
Virtual Machines
Images
Virtual Network

Shared virtualized dynamic provisioning for:

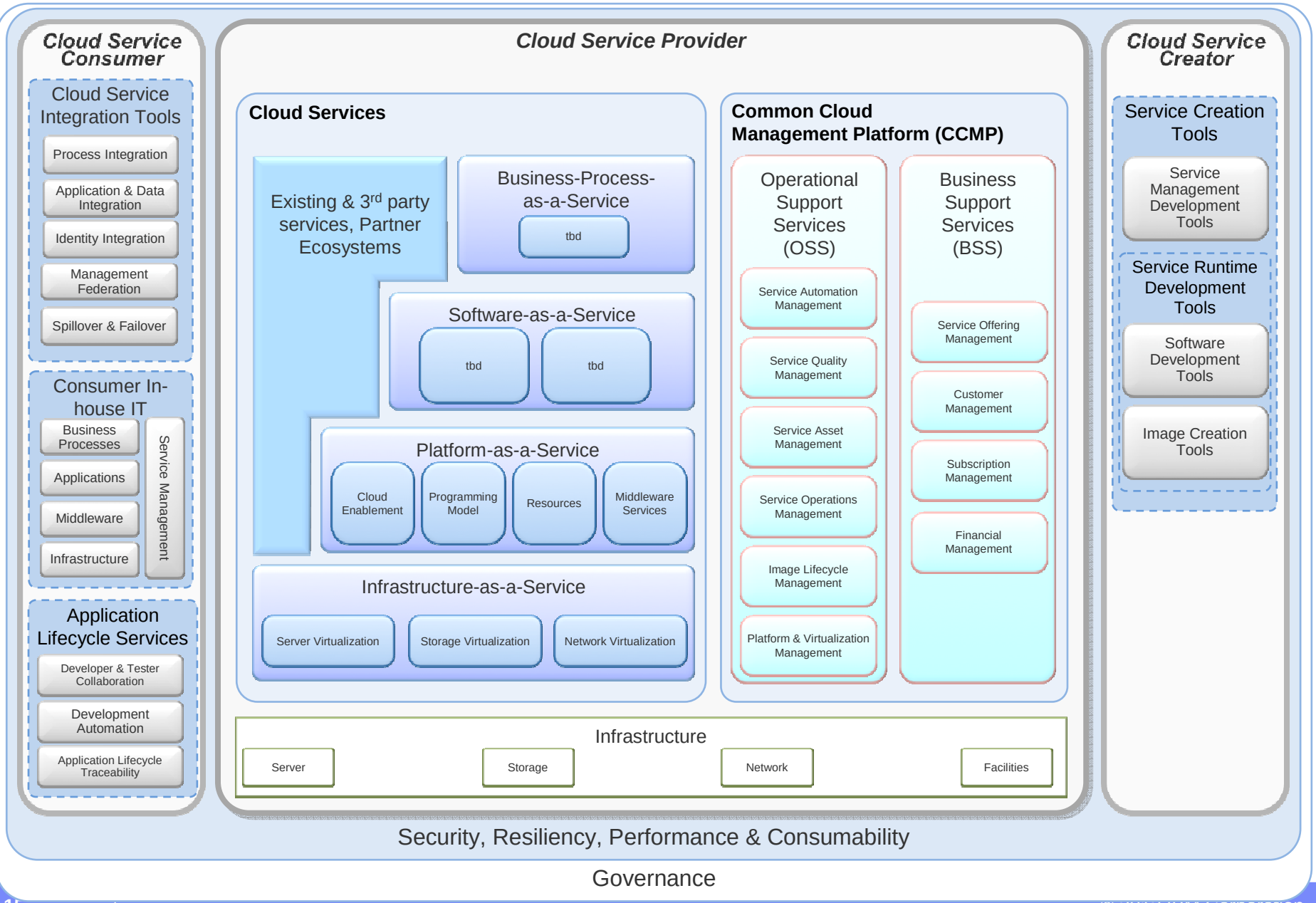
- Server functionality
- Networking functionality
- Data center functionality
- Storage functionality

Cloud deployment models



1. Customer managed private Cloud
2. Customer premise, provider operated private Cloud
3. Provider premise, provider managed, public Cloud
4. Provider premise, provider managed, provider applications, public Cloud
5. Provider premise, provider managed, provider applications, public Cloud

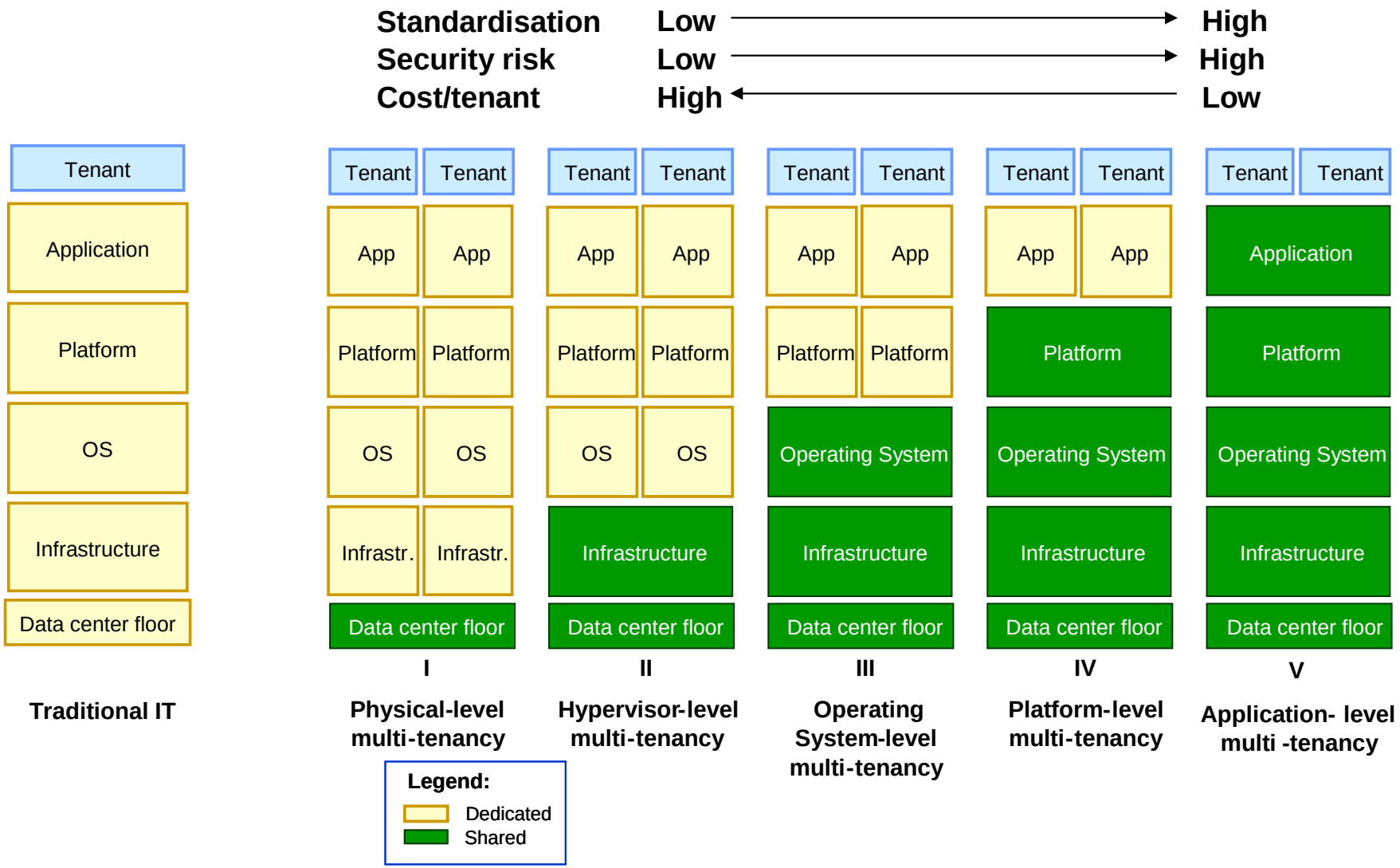
IBM Cloud Computing Reference Architecture



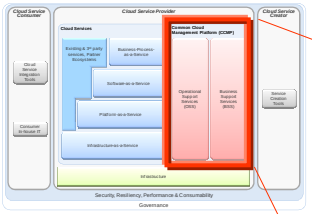
Session Agenda

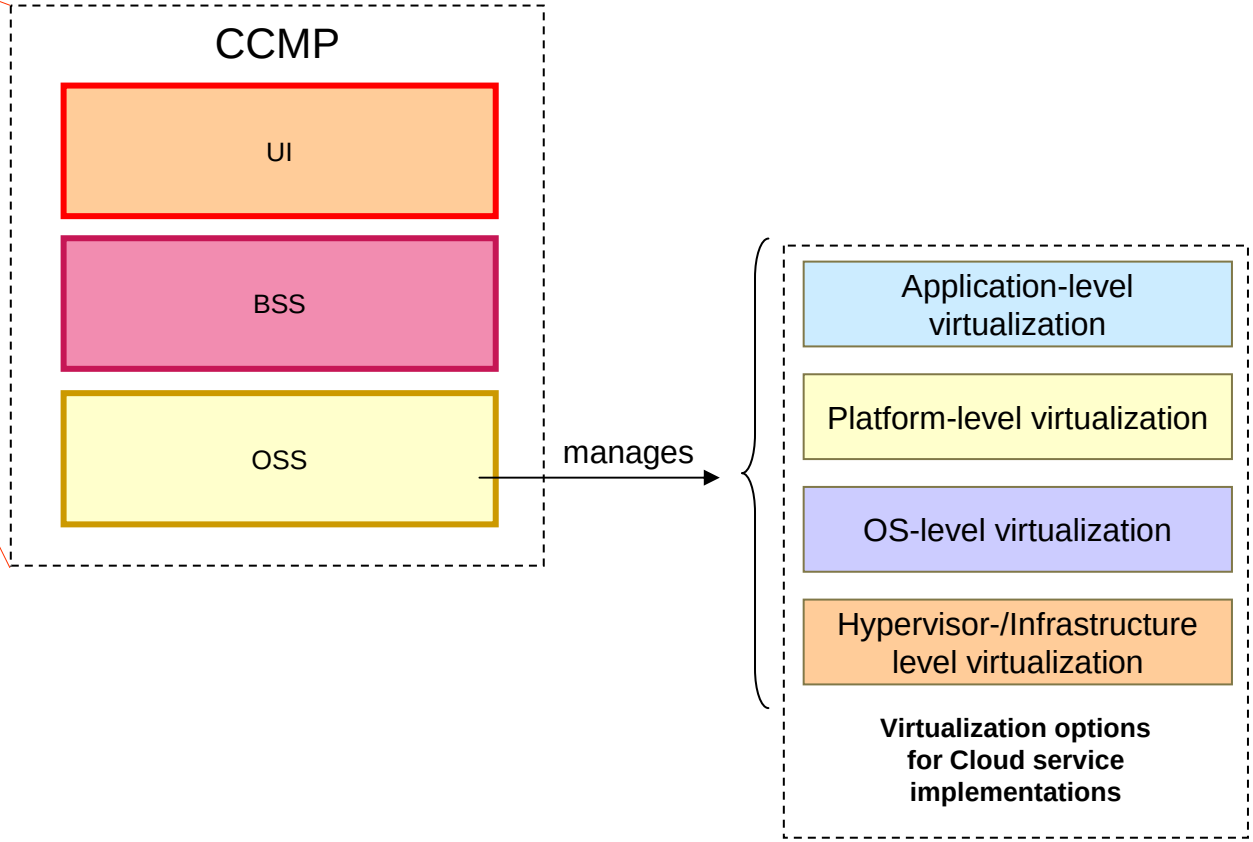
- Introduction
- **Infrastructure and Platforms**
- Cloud Applications
- Designing and delivering SaaS applications
- Client case studies

Resource sharing considerations



Virtualization can be on any level





Virtualization is a great start – but it adds complexity!

What is the utilization of your environment? How do you ensure adequate capacity?

How do you rapidly provision services?

How do you manage image sprawl?

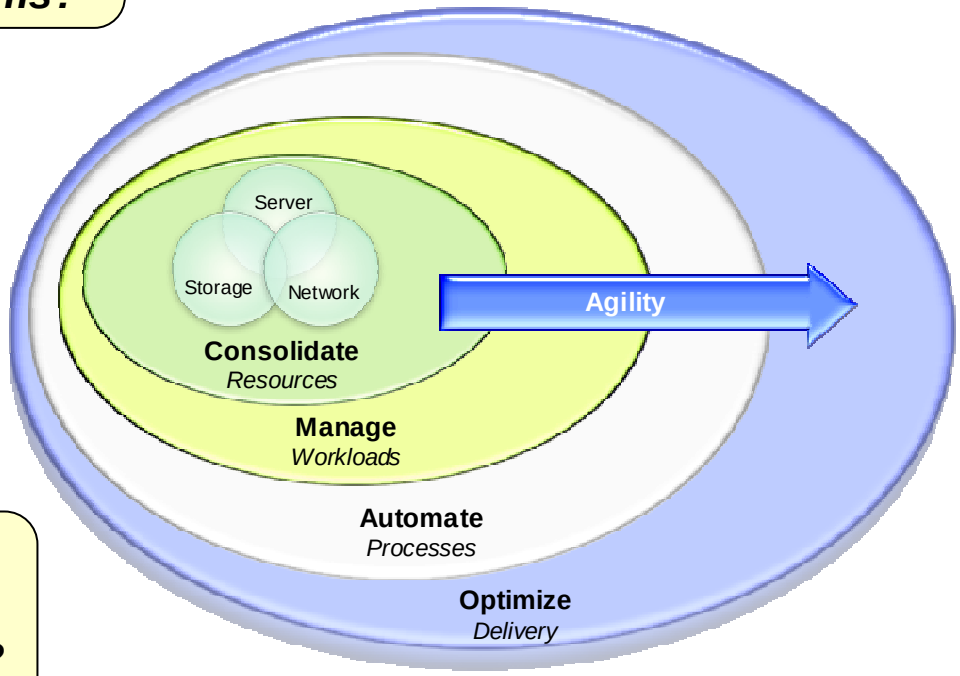
How do you secure your infrastructure & protect data?

How do you charge for shared resources?

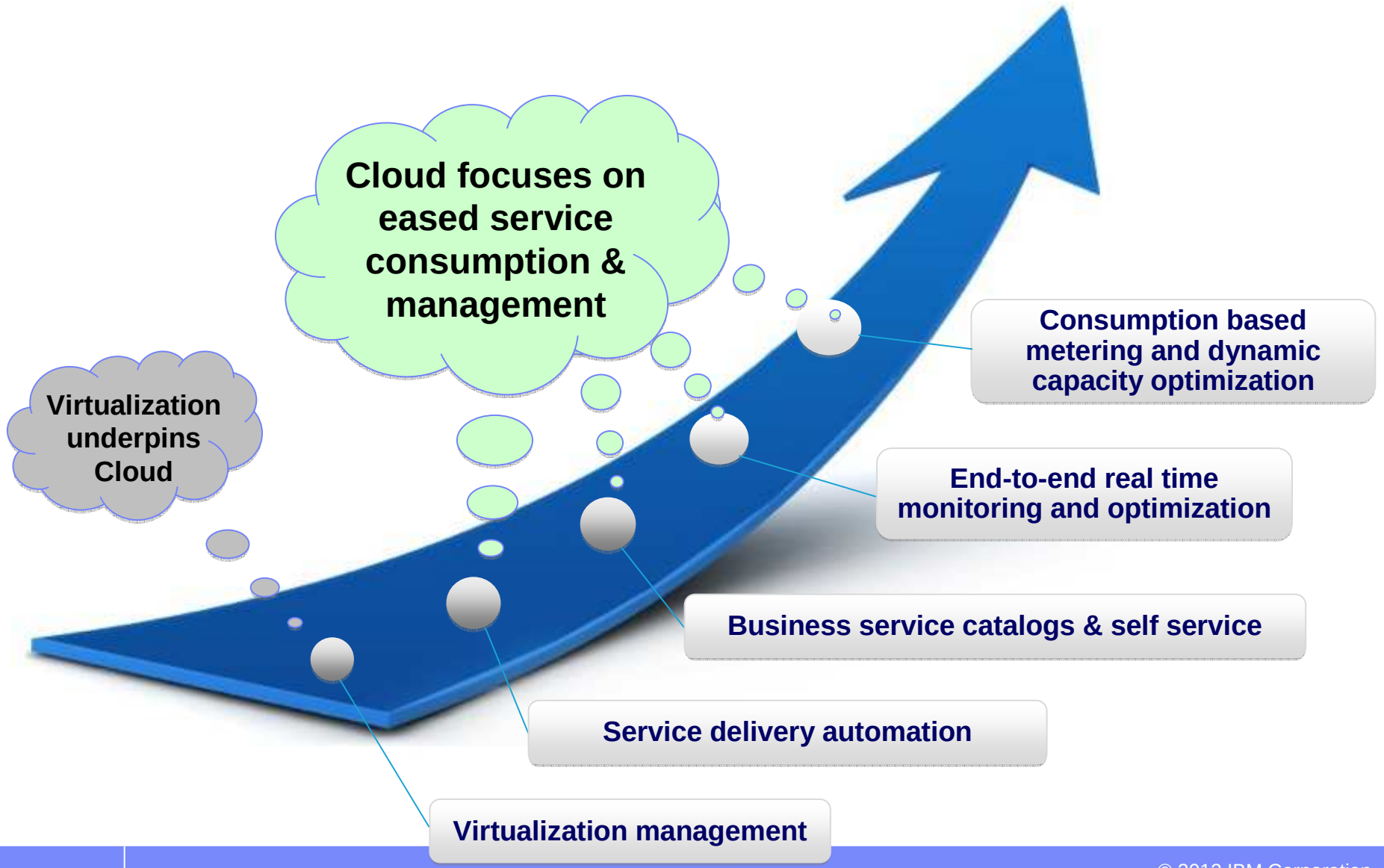
How do you reduce cost of service delivery?

Where are your services running?

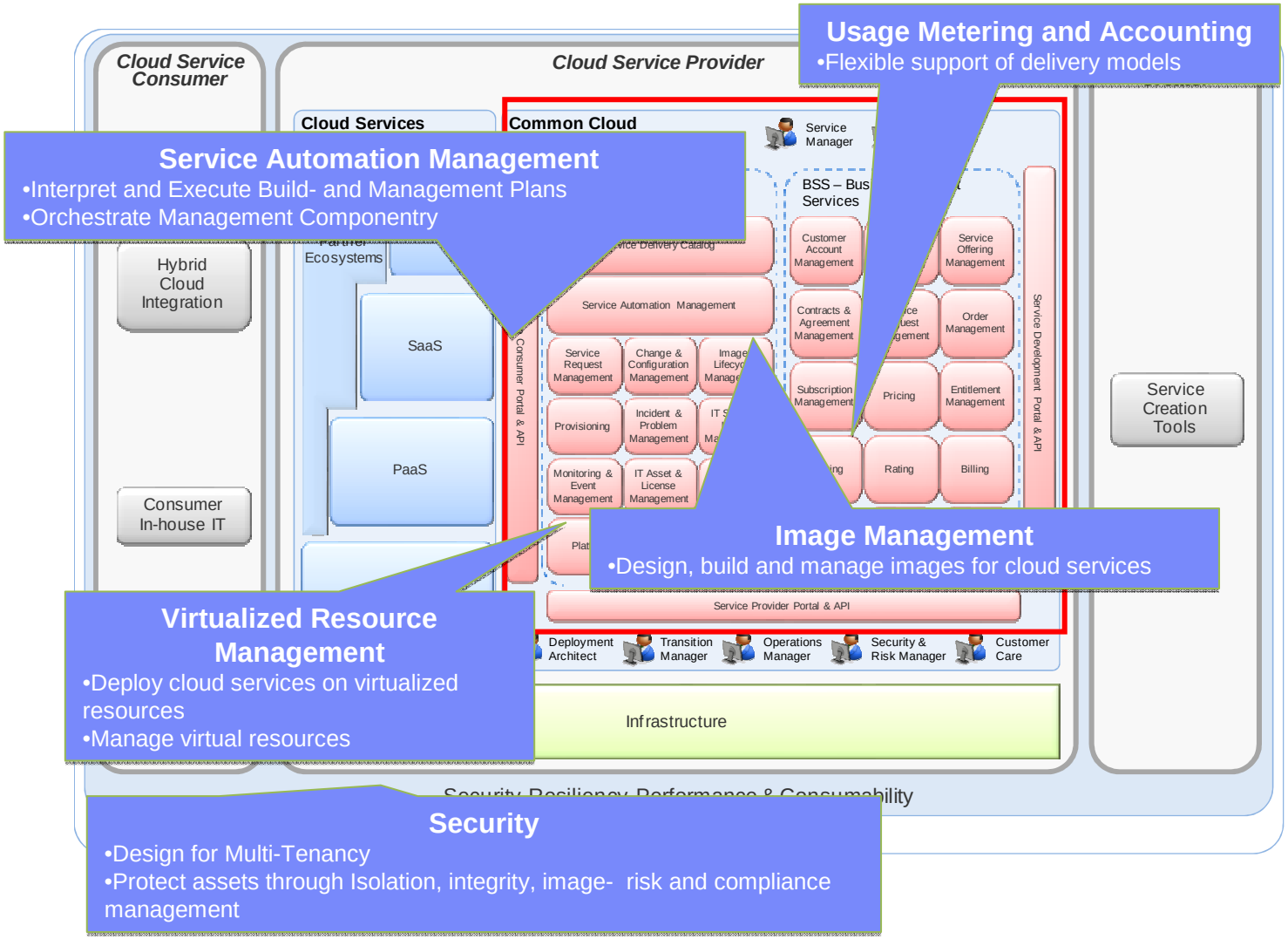
How are your services performing? How do you isolate problems?



Journey to Cloud Computing



The Cloud Computing Reference Architecture has best practises for these aspects



Session Agenda

- Introduction
- Infrastructure and Platforms
- **Cloud Applications**

Traditional vs cloud applications

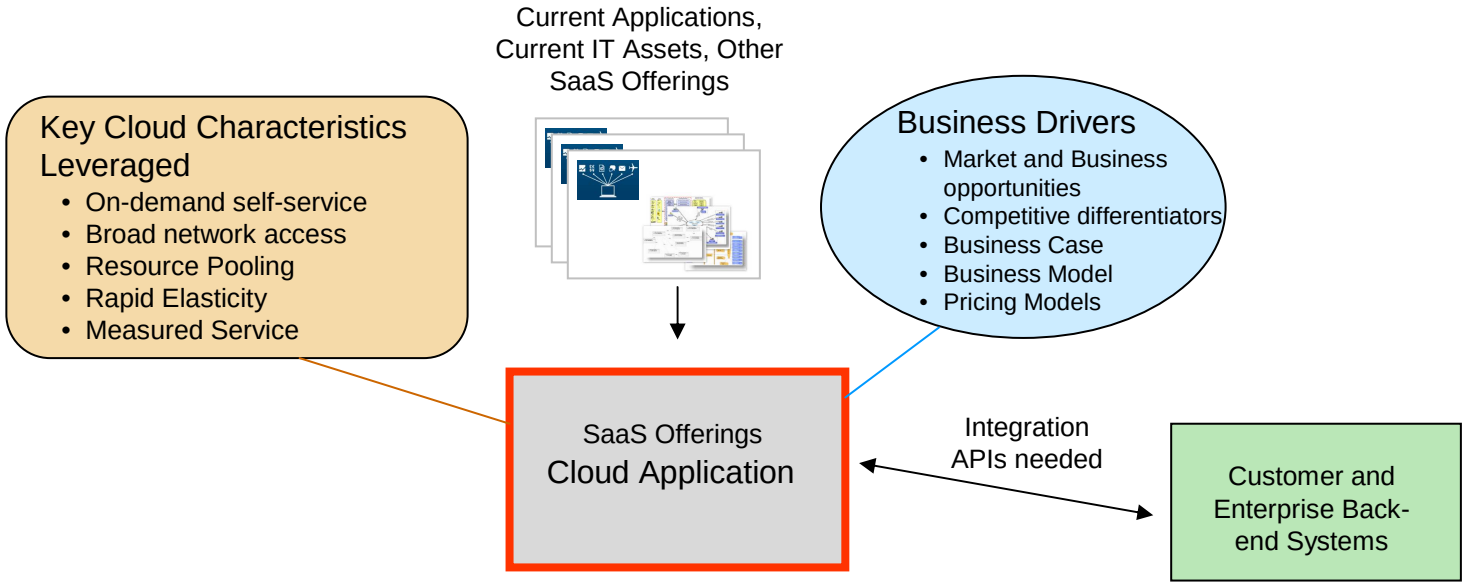
Traditional applications:

- ✓ Extend the life of these applications
- ✓ Add new functionality more cheaply and quickly
- ✓ Connect these applications to the cloud
- ✓ Maintain the security features built into on premise systems

Cloud applications:

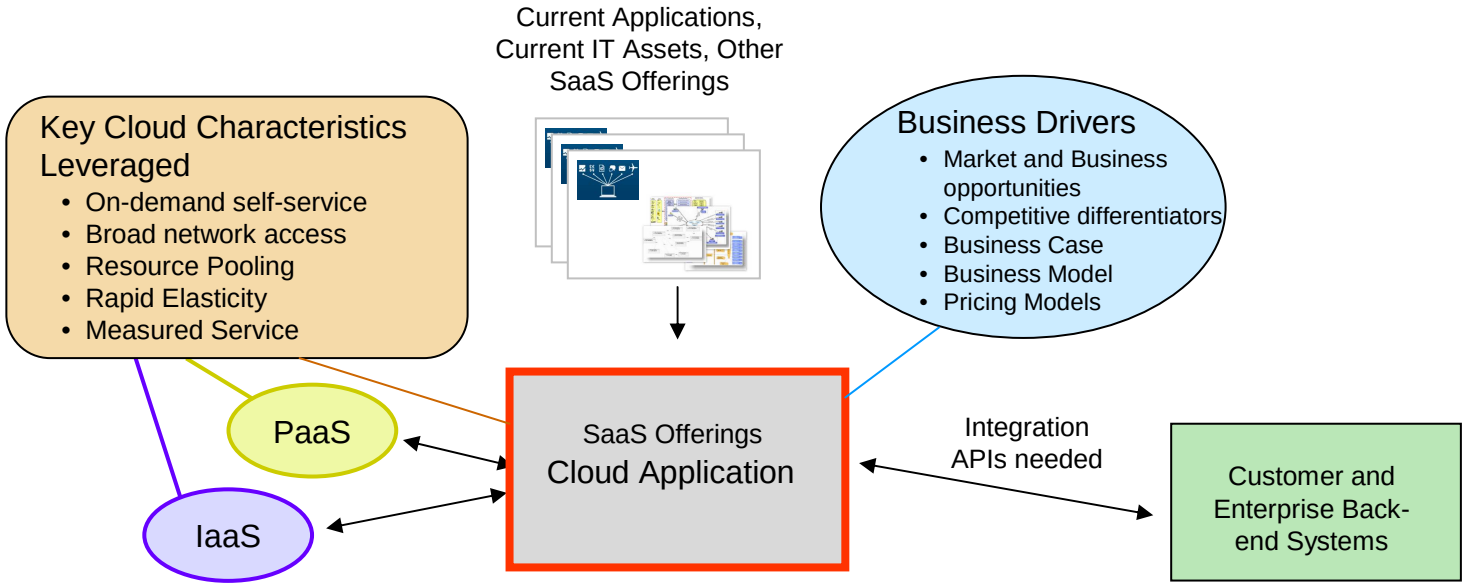
- ✓ Connect to legacy applications
- ✓ Connect to other cloud applications
- ✓ Enhance capabilities at the lowest possible cost

Implementation Considerations (1 of 4)



- Cloud applications supporting multiple SaaS offerings
- SaaS offerings composed of several existing applications
- Business requirements for each offering could be different
- Integration with Backend systems

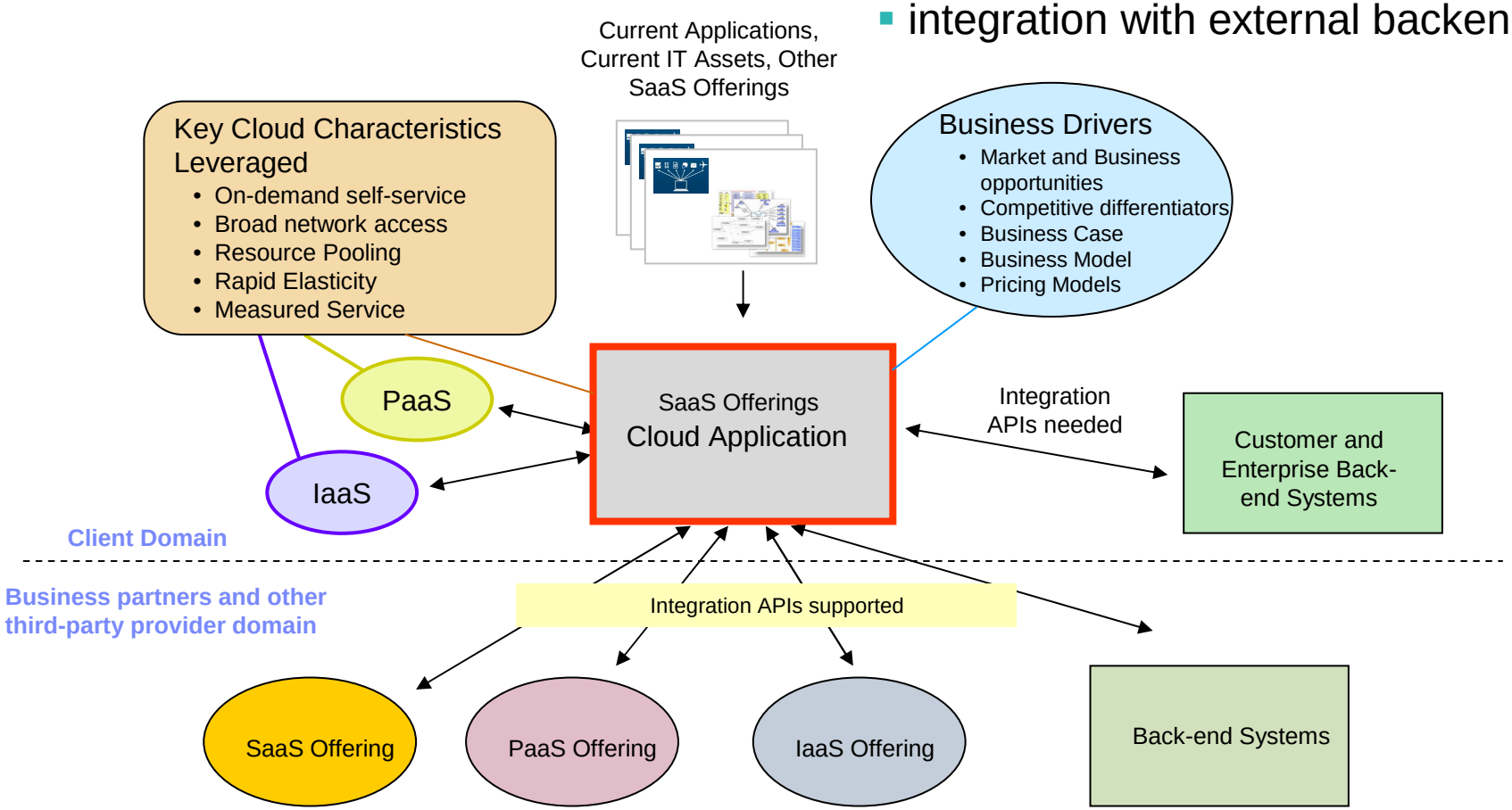
Implementation Considerations (2 of 4)



- choosing the appropriate underlying service layer: IaaS or PaaS
- each layer has own set of business drivers and cloud characteristics

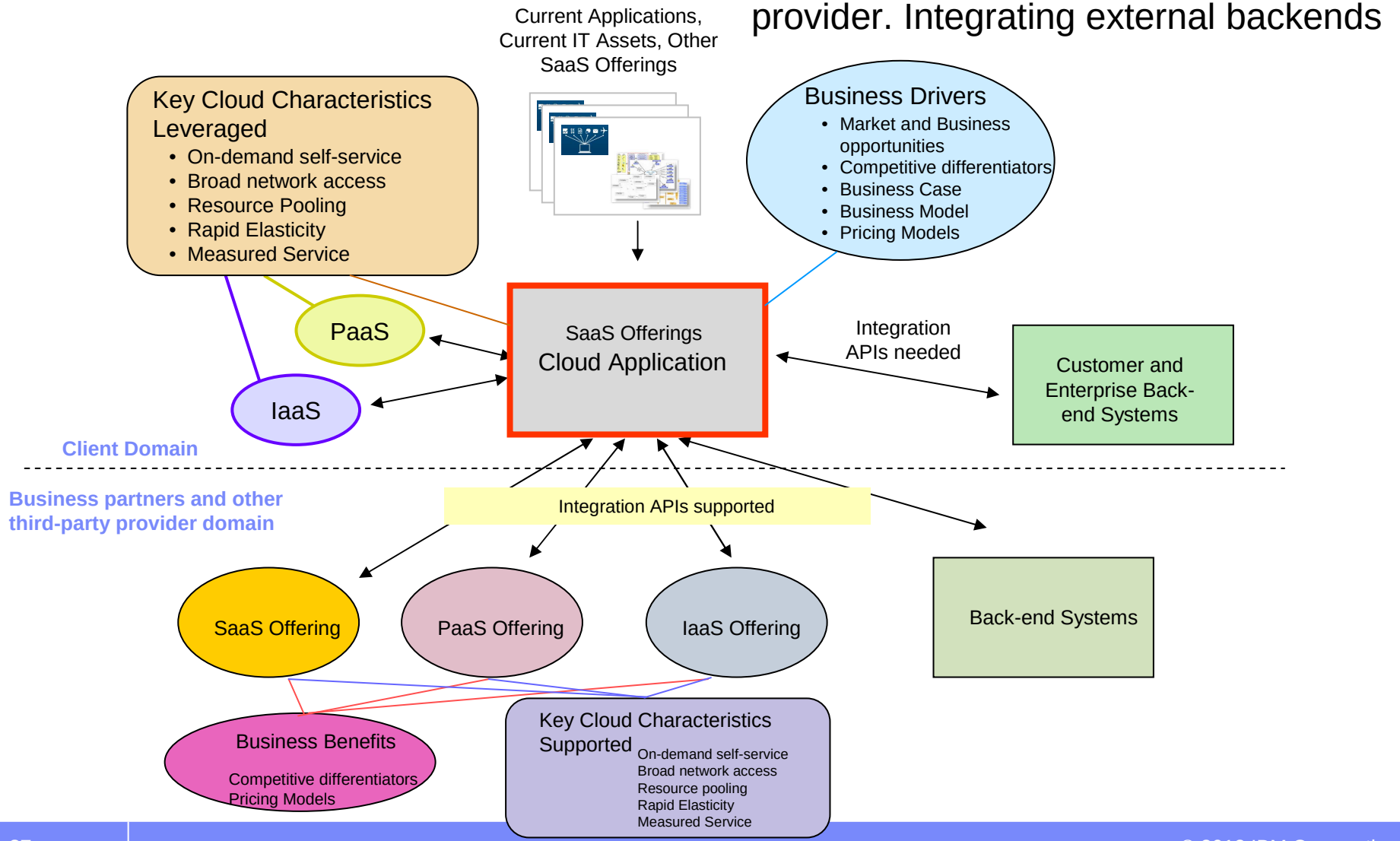
Implementation Considerations (3 of 4)

- consume external XaaS cloud services
- integration with external backends

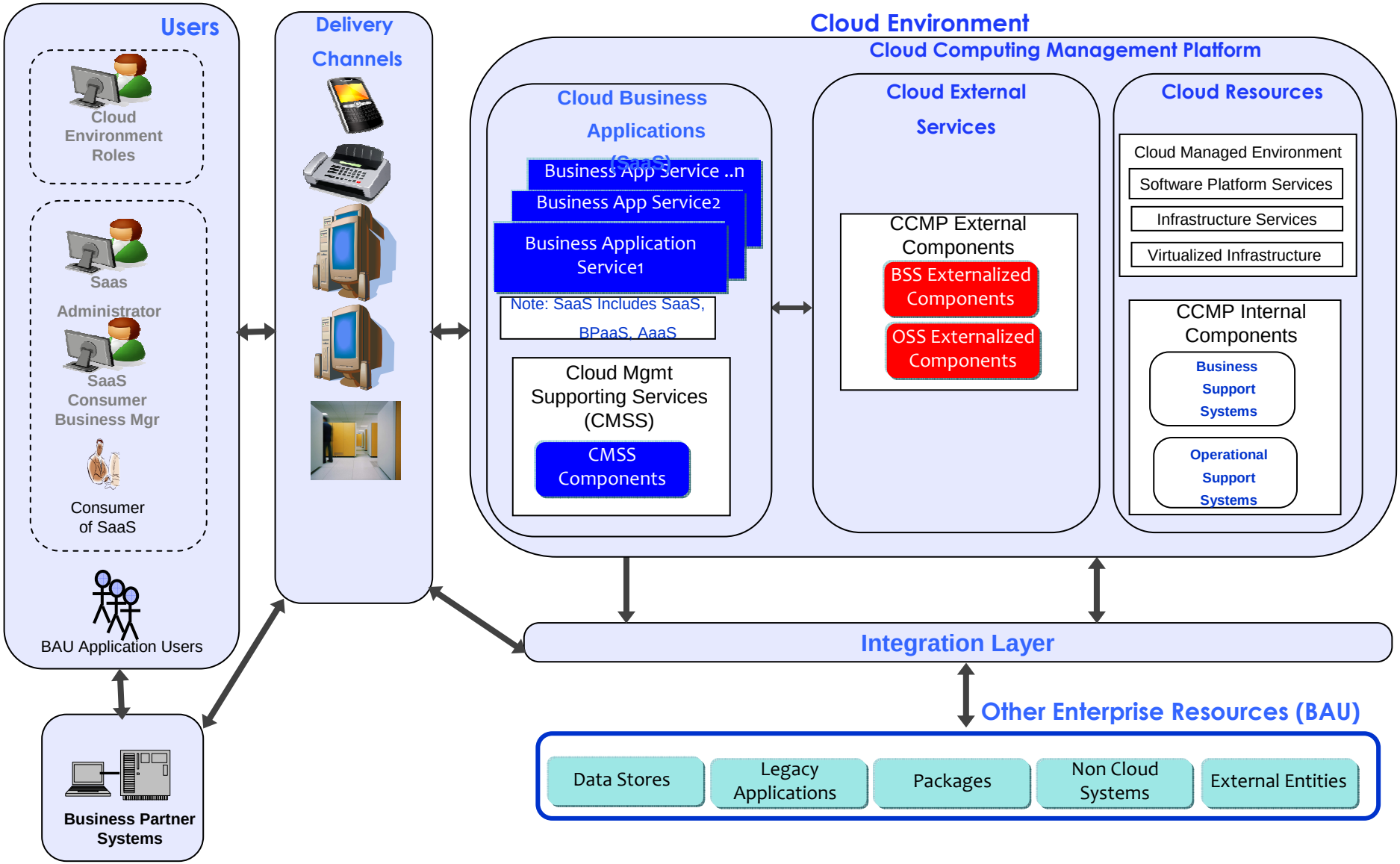


Implementation Considerations (4 of 4)

- ROI, risks, cloud characteristics & business benefits are different for each provider. Integrating external backends

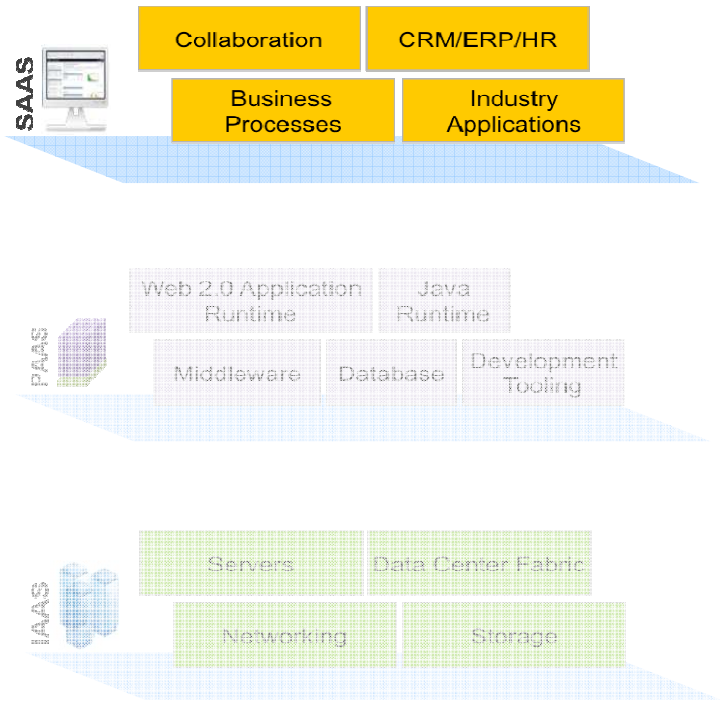


Use Guidance from the Cloud Computing Reference Architecture



Build Cloud Applications, not applications in the cloud !

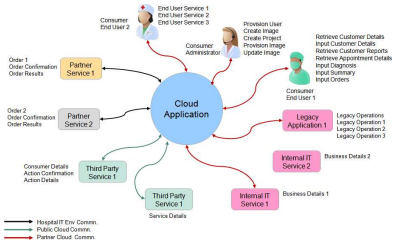
Key Principles

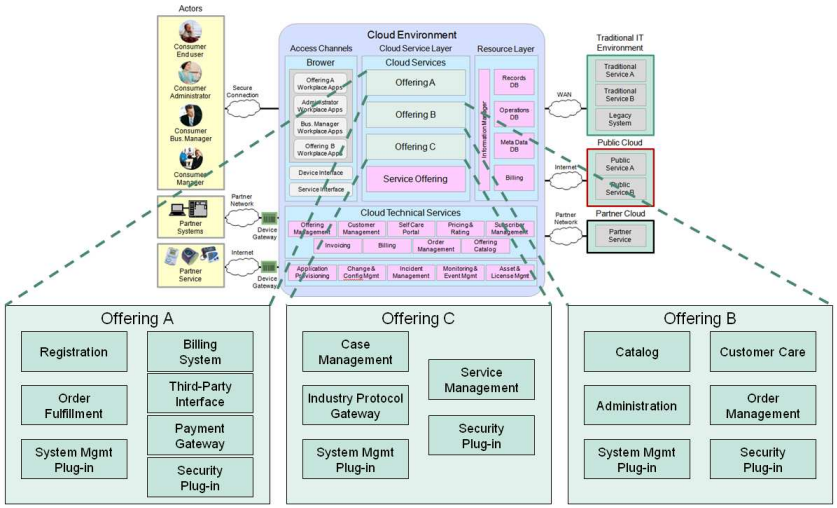


- **Virtualize the Application Stack**
- **Componentize. Decouple & design all components as a 'Black Box'**
- **Design for scalability**

NIST Definition of Cloud Computing

Design for scaling **OUT** !





Traditional way

- *add more RAM*
- *use faster servers*
- *expensive 'micro-optimization'*
- *complex caching*
- *faster hard disks*

Design for scalability

- *minimize mutable state*
- *create asynchronous services*
- *alternative data stores*
- *automate deployment*
- *design for failure*

Design for Scalability: Design for failure

"Everything fails, all the time“

Werner Vogels, CTO Amazon.com

- find single point of failures
- evaluate scenarios. What levels of risk is acceptable?
- failure tolerance
- don’t overinvest

Design for Scalability: Create components & asynchronous services

- Offload work from main application servers – Web 2.0
- Break tasks into separate services, run by different components
- Scale independently
- Use message queues for guaranteed delivery

Design for Scalability: Minimize Shared Mutable State

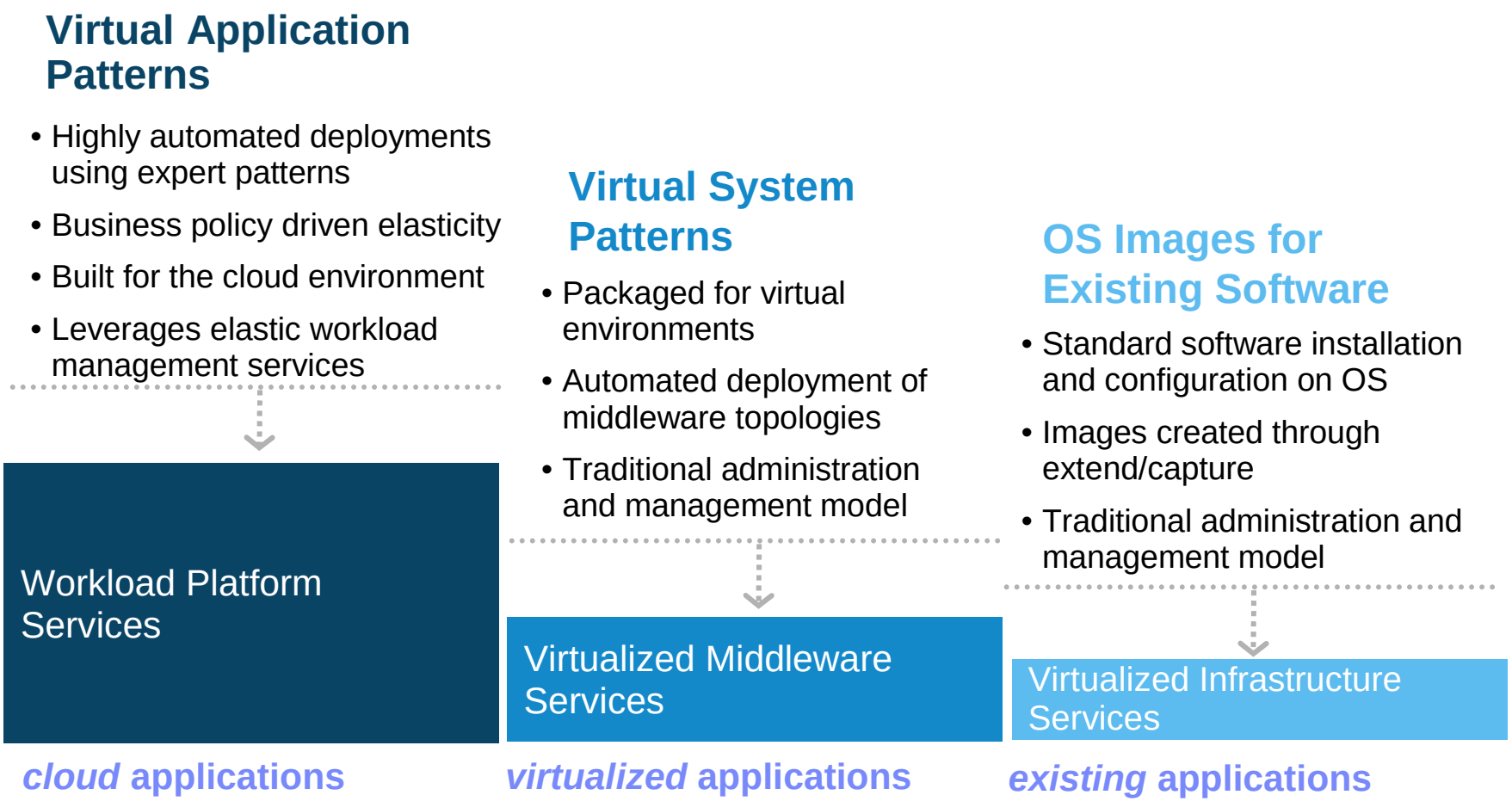
- Variables shared across application
- Multiple servers and processes trying to update the same variables at the same time result in deadlocks, time-outs, and failed transactions

→ minimize or eliminate those in webservers, application and the database

→ specific considerations for filesystems, applications and datastores

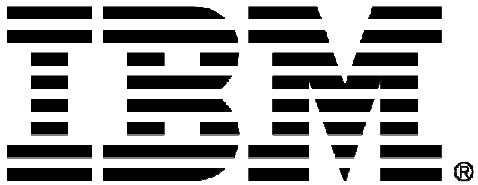
→ look at cluster filesystems, object stores, NoSQL / CouchDB, MongoDB – asynchronous ‘fire & forget’ updates

Design for Scalability: Automate Deployment



3 take aways

- **Cloud Computing is more than just technology. It's a business transformation that can help saving costs, increasing agility**
- **The IBM Cloud Computing Reference Architecture can help you design and implement cloud environments based on best practises and industry standards**
- **Design applications specifically for the cloud, design for scalability & failure**



Trademarks and Disclaimer