BLUEPRINTING

the

CLOUD

MIKE P. PAPAZOGLOU
EXECUTIVE DIRECTOR - ERISS
TILBURG UNIVERSITY.
THE NETHERLANDS
EMAIL: MIKEP@UVT.NL
HTTP://WWW.ERISS.ORG
The ERISS vision focuses on how emerging technologies will impact organizations & society and provide critical insights to influence policy makers & businesses.

**Expertise**
- Dedicated highly-qualified R&D team working at the intersection of business and technology
- Builds on a proven track record of envisioning the future, inventing and delivering the next wave of cutting-edge research solutions

**Vision**
- The ERISS vision focuses on how emerging technologies will impact organizations & society and provide critical insights to influence policy makers & businesses

**Collaborative Research**
- Joint research conducted with top research institutes in Europe, N. America, Australia & East Asia (China & Japan)

**Scientific Conferences, Programs & Summer Schools**
- Founders of the flagship International Conference on Service Oriented Computing (ICSOC)
- Founders of the International Master in Service Engineering (IMSE)
- Joint founders of the highly successful Service and Software Architectures, Infrastructures and Engineering (SSAI&E) Summer School – supported by the EU FP-7 IST SSAI&E unit
ERISS - Areas of Expertise

• Service Lifecycle & Governance:
  - Service-based application development & governance
  - Design of SOA composite business services and multi-party apps

• Service Evolution & Adaptation:
  - Theory of changes for service interfaces, behaviour, policies

• Service Networks & Open Innovation:
  - Coordination & conformance of multi-party protocols
  - Business-aware Transactions

• Business Processes Mgt & Regulatory Compliance:
  - Auditable controls - SOX, Basel II
  - Declarative languages for specifying these requirements

• End-to-end SLAs, QoS & event/activity monitoring:
  - Managed QoS, SLA compositions & complex event processing

• Advanced Cloud Computing Delivery Models – Blueprints:
  - Service design patterns for dynamic and reconfigurable cloud computing architectures spanning organisations
✧ Overview, Vision & Aim
✧ Brief Introduction to Cloud Computing
✧ The Cloud Delivery Model Landscape
✧ Managing the Cloud
✧ Blueprint Example: Interactive Telco Services
✧ Final Remarks
Overview, Vision & Aim
The Services Tsunami
The Cloud Computing Revolution
The world needs to get smarter – more instrumented, interconnected & lead to better decision making. Smarter Service & Cloud technologies are central to this vision.

(Process-intensive & event-driven applications)

Michael P. Papazoglou ©
Barcelona, Spain
May 22, 2012
Describe an approach that supports the effective deployment of global-reach service-based apps into a variety of different implementation platforms - in particular federated cloud computing formations.
Brief Introduction to Cloud Computing
Cloud: Consumption & Delivery Models Optimized by Workload

“Cloud” is:

- A new consumption and delivery model inspired by consumer Internet services.

Cloud enables:

- Infrastructure configuring
- Highly virtualized infrastructure
- Sourcing options
- Economies-of-scale

“Cloud” represents:

- The Industrialization of Delivery for IT supported Services

Multiple Types of Clouds will co-exist:

- Private, Public and Hybrid
- Workload and/or Programming Model Specific
Cloud Computing Delivery & Deployment Models

Cloud service delivery model (provided by Cloud Providers)

**Cloud Service Applications**

- **Software as a Service (SaaS)**
  - Applications (ERP, SCM, CRM)
  - Processes
  - Information

- **Platform as a Service (PaaS)**
  - Middleware – application servers
  - Process automation middleware
  - Database servers
  - Provisioning, etc.

- **Infrastructure as a Service (IaaS)**
  - Virtualized servers
  - Memory, CPUs, Disk space
  - Networking

**Cloud Deployment Models**

- Public
- Private
- Hybrid
SOA in the Cloud

End-to-end Logistics Processes

Order Management

Inventory

Billing & Collections Management

Service Provider

Service-based Application

Service interface

Cloud environment
(platform & infrastructure providers)

Service Provider

Service-based Application

Service interface

Service Provider

Service-based Application

Service interface

Service Provider

Service-based Application

Service interface

Service Provider

Service-based Application

Service interface

Michael P. Papazoglou ©
Barcelona, Spain
May 22, 2012

14
Cloud Services

Cloud Service

- Modular, Shared, Standard & Easily Integratable:
  Commodity-like, *built for a market (public) not for a single consumer*

- Solution-packaged:
  A *turnkey offering that integrates requires services & resources*

- Elastically Scalable:
  *Dynamic & fine-grained*

- Metered:
  *Reflects app & infrastructure QoS characteristics*

- Published service interface/API:
  *Internet accessible via web services or other common Internet APIs*

Cloud-delivered services (or simply cloud services) are fundamentally an emerging delivery/consumption model that is applied to all cloud offerings, e.g., BPaaS, SaaS, PaaS and IaaS.

Cloud services can be constructed of many service-enabled component parts: operating systems, applications, middleware, databases, monitoring tools, and so on - to create a truly federated cloud environment.
The Cloud Delivery Model Landscape
Inflexible Monolithic Cloud Delivery Models

- A monolithic one-size-fits-all SaaS/PaaS/IaaS stack architecture and vendor lock-in prevails.

- PaaS offerings are constrained by providers’ capabilities. They don’t allow easy extensibility, mashup, or customization options at the consumer or developer levels.

- Rigid service orchestration practices tied to a specific resource/infrastructure configuration at the application level.

- SaaS is predominantly tethered to proprietary application platforms in which the cloud provider runs all elements of the service and presents a complete application to the client. They’re hard to extend or customize.
Difficult to compose SaaS solutions in end-to-end processes

Difficult to re-configure/customize SaaS solutions
Stairway to the Clouds: Breaking the Cloud Delivery Monolith

Michael P. Papazoglou © Barcelona, Spain May 22, 2012
Benefits

- Increased interoperability
- Protection against vendor lock-in in the cloud
- Increased quality, callability, performance, (low latency, bandwidth)
- Control, reliability, simplicity, faster deployment, enabling new applications
- New business models, innovation, cost reduction
Managing the Cloud
Meta-data Templates

- **Meta-data templates**: Templates describe how a cloud offering is presented & consumed. The offering is abstracted from the specific type of cloud resources offered. The provider uses service templates to describe in a general form what a cloud service can offer.

  - The Open Virtualization Format (OVF) is an open, portable, & flexible format for the packaging & distribution of virtual appliances (pre-built solutions comprised of one or more VMs).
  - By packaging virtual appliances in OVF, vendors can create a single, pre-packaged appliance that can run on customers’ virtualization platforms of choice.
Model-driven Approaches

• IaaS model-driven approaches automate the deployment of complex IaaS services on cloud infrastructure.

  ➢ A virtual appliance model treats virtual images as building blocks for IaaS composite solutions.
  ➢ Virtual appliances are composed into a virtual solution model which helps developers determine deployment-time requs in a cloud-independent manner using a parameterized deployment plan.
  ➢ Composite appliances automate the deployment of complex app services on a cloud infrastructure.
The Blueprint Cloud Delivery Model
The Blueprint Model

• The term "blueprint" refers to any detailed architectural plan, e.g., a technical drawing documenting an architecture or an engineering design.

• The blueprint model builds & documents a holistic cloud architectural solution as assembled & operated, by engaging available cloud stack modules at all layers.

• It configures a unique optimized cloud environment to meet specific a broad range of application requirements & policies, which specify **what is desired** (e.g., consistency, security and privacy requirements) & **not how it is to be accomplished**.
Features of the Blueprint Model

• It lets developers syndicate, configure, partition & deploy virtual service app payloads on VM & resource pools in the cloud by clearly separating service processing concerns.

• It alleviates vendor lock-in & promotes interoperability.
  – Any service can interoperate horizontally with another service at the same level of the cloud stack provided elsewhere.
  – Upstream or downstream (vertical) interoperation of cloud services is possible.

• It maps declarative configuration points for abstract cloud service specs to available resources, & composes them into complete solution models (using simple aggregation and cross-configuration of virtual services).
Overview of the Cloud Blueprinting Approach

- Blueprint Request Language & Patterns
- Cloud service specifications
- Blueprint Definition Language API calls
- Cloud Blueprinting Framework
- Mapping to Cloud Operations & Resources
- Deployment
- Custom Cloud
- Mapping to Cloud Management API calls
- Cloud Deployment & Monitoring

Generic Request Patterns

Request Language for the Cloud

Blueprint Images

Barcelona, Spain
May 22, 2012
## Elements of the Blueprint Framework

<table>
<thead>
<tr>
<th>Declarative Blueprint Request Language</th>
<th>Developer/user centric language</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Blueprint Definition Language</strong></td>
<td>Operational service description, performance-oriented service capabilities, resource utilization</td>
</tr>
<tr>
<td><strong>Blueprint Constraint Language</strong></td>
<td>Any aspect of an SLA about a service: includes security, privacy, &amp; compliance requirements</td>
</tr>
<tr>
<td><strong>Blueprint Manipulation Language</strong></td>
<td>Based on model mgt algebraic ops, e.g., match, merge, compose, extract, delete, etc</td>
</tr>
<tr>
<td><strong>Blueprint Query Language &amp; Reasoning Mechs</strong></td>
<td>Query BDL &amp; BCL templates &amp; reason about correspondences, mismatches, etc.</td>
</tr>
</tbody>
</table>
The Blueprint Cloud Lifecycle Model

- **Blueprint Definition Language**
- **Source Blueprint model**
- **Customized source blueprints**
- **Blueprint Query Engine**
- **Blueprint Manipulation Language**
- **Interim Target Blueprint model**
- **Optimized Target Blueprint model**
- **Deployment Plans & Configuration Options**
- **Testing & Monitoring**
- **Cloud resources**

- **Application Developer (AD)**
- **User**
- **End user**
- **Service provider (SP)**
BPDL- Specifying Blueprints

- Functional characteristics, including:
  - Service type
  - Messages
  - Interfaces
  - Operations

- Defines the KPIs associated to the services, e.g.,:
  - Ranges of service availability
  - Latency
  - Bandwidth

- Describes the physical infrastructure & resources that are required to run the service described in the blueprint. Prototypical items include: average and peak workload requirements

- Policies:
  - Prescribe, constrain and specify any aspect of a business agreement needed to use a service, including items such as security, privacy and compliance requirements.

- Inter-connected abstract stating functional inter-dependencies & deployment dependencies & options
Example of a Virtual Resource Network

- Horizontal Link: a functional dependency
- Vertical Link: a deployment dependency
- AbstractResourcekLink: connection to an abstract resource

Directed Graph
- Vertices
  - Service Offering
  - Implementation Artefact
  - Resource Requirement
- Edges
  - Abstract Resource Link
  - Horizontal Link
  - Vertical Link
Blueprint Delegation

- Blueprint delegation is an attractive solution when a cloud provider does not need to bind statically to a specific service delivery option but keeps some degree of freedom to pursue a service provision option dynamically, while respecting the original request.

Provide a PaaS authentication service & place VMs in the same location and racks

Delegation (choice at run-time)
## BML - Unary Blueprint Operators

<table>
<thead>
<tr>
<th>Operator Name</th>
<th>Input</th>
<th>Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateBlueprint()</td>
<td>vector&lt;Property&gt; initProps</td>
<td>Blueprint bp</td>
<td>- Create a new Blueprint using a set of initial properties</td>
</tr>
<tr>
<td>GetBlueprintByID</td>
<td>String blueprintID</td>
<td>Blueprint bp</td>
<td>- Retrieve a blueprint using the provided blueprint ID</td>
</tr>
<tr>
<td>GetBlueprintByProperties</td>
<td>vector&lt;Property&gt; inputProps</td>
<td>vector&lt;Blueprint&gt; resultBP</td>
<td>- Retrieve a set of blueprints using the provided property values</td>
</tr>
<tr>
<td>ModifyBlueprintProperties</td>
<td>Blueprint bp, vector&lt;Property&gt; updatedProps</td>
<td></td>
<td>- Modify the value of a blueprint property</td>
</tr>
<tr>
<td>GetProperty</td>
<td>Blueprint bp, String propName</td>
<td>Property p</td>
<td>- Get the property</td>
</tr>
<tr>
<td>DeleteBlueprint</td>
<td>String blueprintID</td>
<td></td>
<td>- Delete a blueprint (from a repository)</td>
</tr>
</tbody>
</table>
### BML - Binary Operators

<table>
<thead>
<tr>
<th>Operator Name</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>CompareProperty</td>
<td>Property p, Property p'</td>
<td>{'restricts'</td>
</tr>
<tr>
<td>CompareBlueprints</td>
<td>Blueprint bp, Blueprint bp'</td>
<td>vector&lt;Property, Property&gt; eq, vector&lt;Property, Property&gt; res, vector&lt;Property, Property&gt; ext, vector&lt;Property, Property&gt; ueq, vector&lt;Property, Property&gt; ada,</td>
</tr>
<tr>
<td>MergeBlueprints</td>
<td>Blueprint bp, Blueprint bp', vector&lt;String&gt; conds</td>
<td>Blueprint bp”</td>
</tr>
<tr>
<td>SplitBlueprint</td>
<td>Blueprint bp, vector&lt;String&gt; serviceIdList1, vector&lt;String&gt; serviceIdList2, vector&lt;String&gt; conds</td>
<td>Blueprint bp’, Blueprint bp”</td>
</tr>
<tr>
<td>Operator Name</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>HorizontalLink</td>
<td>- Check the Conditions for linking:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- create a horizontal link between so and so'</td>
<td></td>
</tr>
<tr>
<td>VerticalLink</td>
<td>- Check the Conditions for linking:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- create a vertical link between so and so'</td>
<td></td>
</tr>
<tr>
<td>AbstractResourceLink</td>
<td>- create an abstract resource link between $S_i$ and $S_{i+1}$</td>
<td></td>
</tr>
<tr>
<td>ComposeBlueprint</td>
<td>- Check whether the offerings of bp’ can satisfy any requirements of bp.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>- If yes, create a horizontal or vertical link between bp and bp’</td>
<td></td>
</tr>
</tbody>
</table>
Blueprint Constraint Language

Is composed of

- Blueprint Predicates (BP)

### Resource Utilization Constr.
- Example:
  - FP1: ((Servlet 2.5 Container Exists) Runs On (Intel Dual Core 2Ghz Exists))
  - FP2: Composition Engine Exists
  - FP4: Network Link 2Gbit Exists
  - FP5-Link 2:N etwork Link 2Gbit Exists
  - FP6-Link 3 2 Network Links 3Gbit Exists

### SLA/QoS constraints
- Example:
  - NFP1: Throughput >= 100 req/s
  - NFP2: Availability >= 98% on 24/7

### Security/Compliance const.
- Example:
  - RP1: use WS-Security & XML-Digital Signature

### Deployment/Data residency constraints
- Example:
  - RP1: Only used in the Netherlands

BCL is grounded on:
- Linear Temporal and Monoidal t-norm based Logic (LTL/MTL)
- Compliance patterns.
The CRL in BCL is designed for:

- the formal specification of compliance requirements.
- enabling automatic design-time compliance verification.
- Grounded on:
  - Temporal logic; Linear Temporal Logic (LTL/MTL)
  - Compliance patterns.

- Supports **non-monotonic requirements** to relax rules and handle exceptional situations.

- CRL is an open extensible language.
Blueprint Example:
Interactive Telco Services
Cloud Computing and Rendering

• High quality of video & services
  – Bandwidth availability -higher priority to game traffic during network bottlenecks
  – Video encoding is computationally demanding

• Low latency for interactive applications
  – Real time new view rendering at the browser client end
  – Adaptive Stream management to handle user requests and network loads
Interactive Mobile Video Gaming

Out of resources!

Server

CG application

Full-frame Rendering

Video Encoding

Internet

Bandwidth: 2-6Mbit per client

Clients

Video Decoding

Video Decoding

Video Decoding

CG app controls the shape, appearance, and motion of objects drawn using programmable graphics hardware.

Michael P. Papazoglou © Barcelona, Spain May 22, 2012
Gaming app Integrating Iaas/Paas/Saas Components
Blueprints for Interactive Video Application

**Authentication service BP**
- Platform Description
  - API
  - Endpoint
  - Authentication conf.
- Infrastructure Description
- Policies Description

**Database service BP**
- Platform Description
  - API
  - Endpoint
  - Database conf.
- Infrastructure Description
- Policies Description

**Platform Configuration**
- Authentication Configuration
- DB Configuration
  - Medium VM
  - Large storage capacity
- IaaS Configuration
  - Medium size VM
  - Large storage capacity

**Application Configuration**
- Application Description
  - Metrics: delay, availability
  - Location: URI
  - Version: ...
  - Package: caching service
  - Environment: servlet

**Deployment**
- Check dependencies
- Deploy
- Deploy with rendering service
- Use Amazon’s autoscale
- 99.999% availability

**Caching service BP**
- Application Description
  - Metrics: ...

**Rendering service BP**
- Application Description
  - Metrics: delay, availability
  - Location: URI
  - Version: ...
  - Package: caching service
  - Environment: servlet

**Public interfaces**
- Authentication service
- Rendering service
- Caching service
- Database service

**Components**
- DML composition point
- External configuration points
- Exposure of horizontal parameters

**Check dependencies**
- Estimate avg delay – if delay goes below 100 ms migrate to other IaaS

By Michael P. Papazoglou © Barcelona, Spain May 22, 2012
Closing Remarks
• Contemporary cloud technologies are fraught with problems. New technologies are required to support the mgt of clouds and allow for the dynamic deployment and management of services.

• Blueprinting allows cloud creating cloud formations dynamically to comprise an arbitrary assembly of virtual cloud services (business processes, virtual platforms, virtual machines, & virtual storage volumes) connected into whatever design of IT service-based application & associated infrastructure a customer desires.
  – It provisions cloud services, effectively manages workload segmentation and portability, and provides cloud architectures that automatically manage the lifecycle of cloud services, partition work & optimize workload distribution.