In short

- We recognize the value of the cloud computing model, and would like to capture its benefits
In short

- We recognize the value of the cloud computing model, and would like to capture its benefits
- Valid concerns prevent widespread and expedited adoption
In short

- We recognize the value of the cloud computing model, and would like to capture its benefits.
- Valid concerns prevent widespread and expedited adoption.
- It is possible to expedite adoption by internally adopting the cloud computing philosophy; we propose a seven-step roadmap.
A definition of cloud computing (NIST)

A model for enabling ubiquitous, convenient, on-demand access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.
A definition of cloud computing (NIST)

A model for enabling ubiquitous, convenient, on-demand access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction

- It is a model for using existing technology
- It favors the what over the how; concepts over concrete devices (e.g., storage over hard drive, or computing power over workstation)
A vision to reduce complexity

Sources of Complexity

- Proliferation of systems
- Proliferation of formats
- Proliferation of protocols
- Siloing of datasets
A vision to reduce complexity

<table>
<thead>
<tr>
<th>Sources of Complexity</th>
<th>Architectural Evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proliferation of systems</td>
<td>Associative Elasticity</td>
</tr>
<tr>
<td>Proliferation of formats</td>
<td>Semantical Hyperdata</td>
</tr>
<tr>
<td>Proliferation of protocols</td>
<td>Living Workflows</td>
</tr>
<tr>
<td>Siloing of datasets</td>
<td></td>
</tr>
</tbody>
</table>
A vision to reduce complexity

Sources of Complexity
- Proliferation of systems
- Proliferation of formats
- Proliferation of protocols
- Siloing of datasets

Architectural Evolution
- Associative Elasticity
- Semantical Hyperdata
- Living Workflows
- Abstraction
A vision to reduce complexity

<table>
<thead>
<tr>
<th>Sources of Complexity</th>
<th>Architectural Evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proliferation of systems</td>
<td>Associative Elasticity</td>
</tr>
<tr>
<td>Proliferation of formats</td>
<td>Semantical Hyperdata</td>
</tr>
<tr>
<td>Proliferation of protocols</td>
<td>Living Workflows</td>
</tr>
<tr>
<td>Siloing of datasets</td>
<td>Abstraction</td>
</tr>
</tbody>
</table>

**Summary of Architecture**

Expose data and algorithms as resource-oriented Web services, coordinated via messaging and running on virtual machines
A vision to reduce complexity

**Sources of Complexity**
- Proliferation of systems
- Proliferation of formats
- Proliferation of protocols
- Siloing of datasets

**Architectural Evolution**
- Associative Elasticity
- Semantical Hyperdata
- Living Workflows
- Abstraction

**Summary of Architecture**
Exposé data and algorithms as resource-oriented Web services, coordinated via messaging and running on virtual machines

- Virtual Machine
- HTTP Server
- Database
- Message Broker
- Serialization Format
- Web Application
- Web Client
Hypervisor: key to elasticity
Message broker: key to abstraction

Work decoupling

Producer → Broker → Consumer
Message broker: key to abstraction

Work decoupling

Producer → Broker → Consumer

Work streamlining and decoupling

Producer → Broker → Consumer/Producer

Consumer ← Broker
Message broker: key to abstraction

Work distribution

Producer → Broker

Consumer

Consumer
Message broker: key to abstraction

**Work distribution**

Producer -> Broker

Producer

Consumer

Consumer

**Work aggregation, distribution, and decoupling**

Producer -> Broker

Producer

Producer

Consumer

Consumer

Consumer

Consumer
Message broker: key to abstraction (example)
Data Evolution

(1) adopt a common data-interchange format → make data usable

**typical legacy output**

- **EPOCH:** 18-Dec-2012 00:12:34.567891 UTC
- **FRAME:** EME2000
- **POSITION:** 1.234567891D+06 -7.6543210D+06 3.4567891D+02
Data Evolution

(1) *adopt a common data-interchange format* → make data usable

**typical legacy output**

- EPOCH: 18-Dec-2012 00:12:34.567891 UTC
- FRAME: EME2000
- POSITION: 1.234567891D+06 -7.6543210D+06 3.4567891D+02

**simple JSON translation**

```json
{"epoch":{
    "day":18, "month":12, "year":2012,
    "hour":0, "minute":12, "second":34.567891,
    "iso":"2012-12-18T00:12:34.567891",
    "clock":"utc"
},
"frame":"eme2000",
"position":[1.234567891e6, -7.6543210e6, 3.4567891e2]}
```
Data Evolution

(1) adopt a common data-exchange format → make data usable

typical legacy output

EPOCH: 18-Dec-2012 00:12:34.567891 UTC
FRAME: EME2000
POSITION: 1.234567891D+06 -7.6543210D+06 3.4567891D+02

simple JSON translation

{"epoch":{
  "day":18, "month":12, "year":2012,
  "hour":0, "minute":12, "second":34.567891,
  "iso":"2012-12-18T00:12:34.567891",
  "clock":"utc"
},
"frame":"eme2000",
"position": [1.234567891e6, -7.6543210e6, 3.4567891e2]}

(2) evolve raw data to semantical hyperdata → increase information content

augmented JSON translation

{"_meta": {"_id":"c461e383ba7f549319a3e63f36649b05e",
  "product":"abcd", "mission":"mymission",
  "user":"jarrieta", "tstamp":"2012-12-18T00:11:22.334455", "sw":"myapp", "1.2.3",
  "sources": [{"id":"63f6bc22ef35f62e8c8832f64f6f159"},
    {"gin":"a53db90452346e80055e7c430cb9f0c"}],
  "xcheck": [{"nav":true, "rep":"jdoe", "tstamp":"2012-12-18T00:11:23.334455"},
    {"sys":false}, {"prp":false}, {"acs":false}],
  "epoch": ...}
(3) decompose workflows into simple actions → granular, reusable units of work

**Workflow evolution**

**Produce Maneuver Design**

- **Retrieve OD**
- **Run optimizer**
- **Generate data**
- **Integrate deliverables**
- **Deliver design**
Workflow evolution

(3) decompose workflows into simple actions → granular, reusable units of work

Produce Maneuver Design

Retrieve OD → Run optimizer → Generate data → Integrate deliverables → Deliver design

(4) categorize data and algorithms as resources → addressable, independent entities

| Retrieve OD | GET /od/solutions/latest/ |
| Run optimizer | POST /optimizer/queue/?priority=1 |
| Generate data | POST /man/designs/latest/ |
| Integrate deliverables | POST /man/designs/latest/approval/ |
| Deliver design | PUT /man/designs/latest/ |
Protocol evolution

(5) Provide a common interface to resources via HTTP → access standardization

<table>
<thead>
<tr>
<th>Verb</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEAD</td>
<td>read the metadata provided in a resource’s headers</td>
</tr>
<tr>
<td>GET</td>
<td>read a resource in a specified representation</td>
</tr>
<tr>
<td>POST</td>
<td>create a resource by providing a specific representation</td>
</tr>
<tr>
<td>PUT</td>
<td>update a resource in whole or in part</td>
</tr>
<tr>
<td>DELETE</td>
<td>delete a resource</td>
</tr>
</tbody>
</table>
(5) Provide a common interface to resources via HTTP → access standardization

<table>
<thead>
<tr>
<th>Verb</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEAD</td>
<td>read the metadata provided in a resource’s headers</td>
</tr>
<tr>
<td>GET</td>
<td>read a resource in a specified representation</td>
</tr>
<tr>
<td>POST</td>
<td>create a resource by providing a specific representation</td>
</tr>
<tr>
<td>PUT</td>
<td>update a resource in whole or in part</td>
</tr>
<tr>
<td>DELETE</td>
<td>delete a resource</td>
</tr>
</tbody>
</table>

(6) Coordinate the system interaction via messaging → living, adaptable workflows
Protocol evolution (continued)

(7) Deploy worker and data nodes in virtual machines → abstract, elastic, configurable system

```
<table>
<thead>
<tr>
<th>Database</th>
<th>Legacy App</th>
<th>Optimizer</th>
<th>Optimizer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Hypervisor</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Hardware</td>
</tr>
</tbody>
</table>
```
Conclusion

- It is possible to immediately capture some benefits of the cloud computing model.
- The architecture may help reduce some common sources of complexity.
- The implementation may enable teams and agencies to evaluate the cloud computing model in their specific context, with minimal infrastructure changes, and before committing to a given provider.