3rd International Symposium on High Confidence Software
December 21-22, 2013, Beijing, China

InternetWare

SM@RT Cloud

Runtime Software Architecture Based Cloud Management

From paper to product

Gang Huang

高可信软件技术教育部重点实验室
Key Lab of High Confidence Software Technologies (Peking University), Ministry of Education

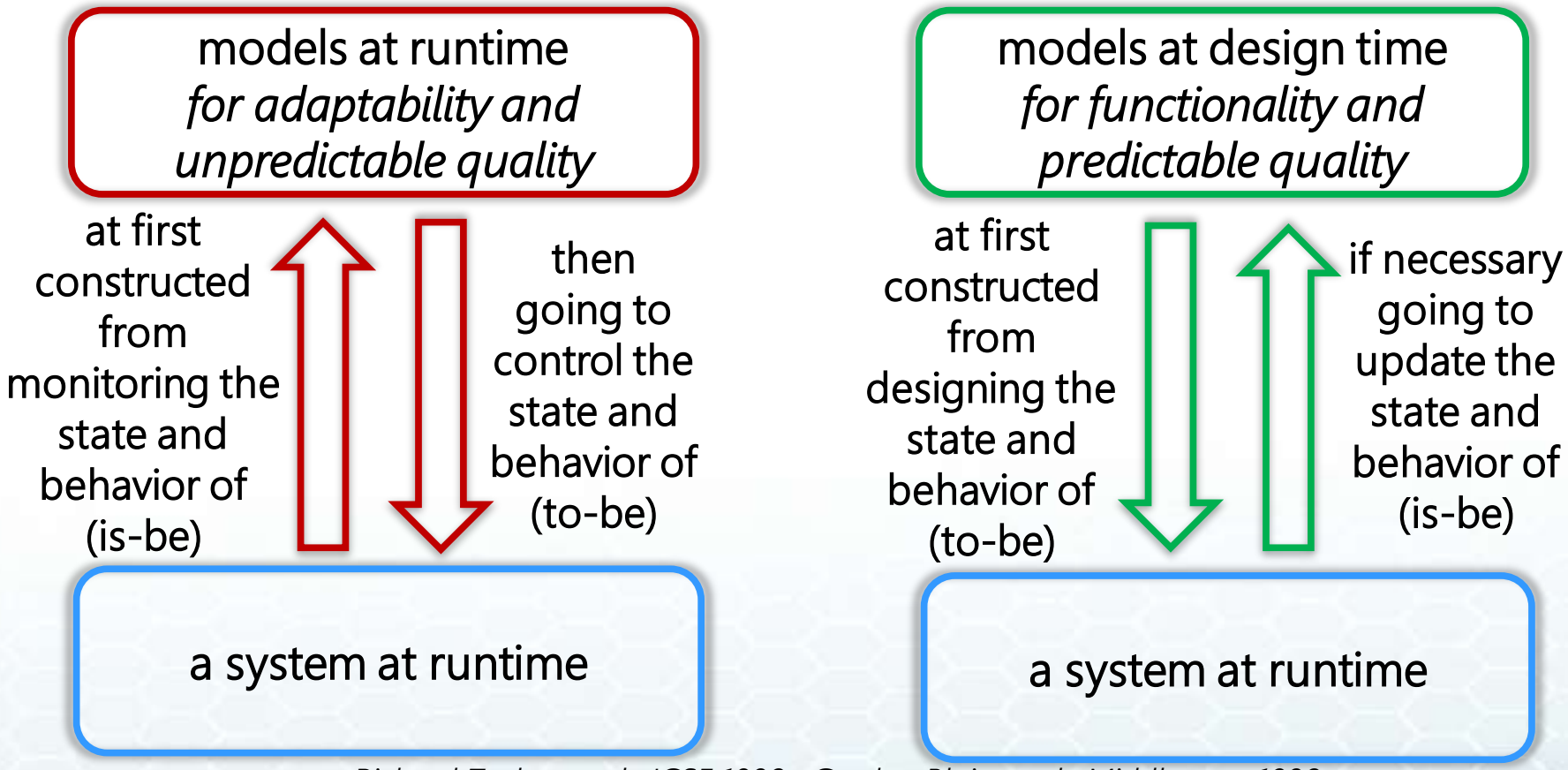
Agenda

- Background of this work
- Challenges to Cloud Management
- Runtime Software Architecture Approach
- Conclusion and Future Work

Models@Runtime

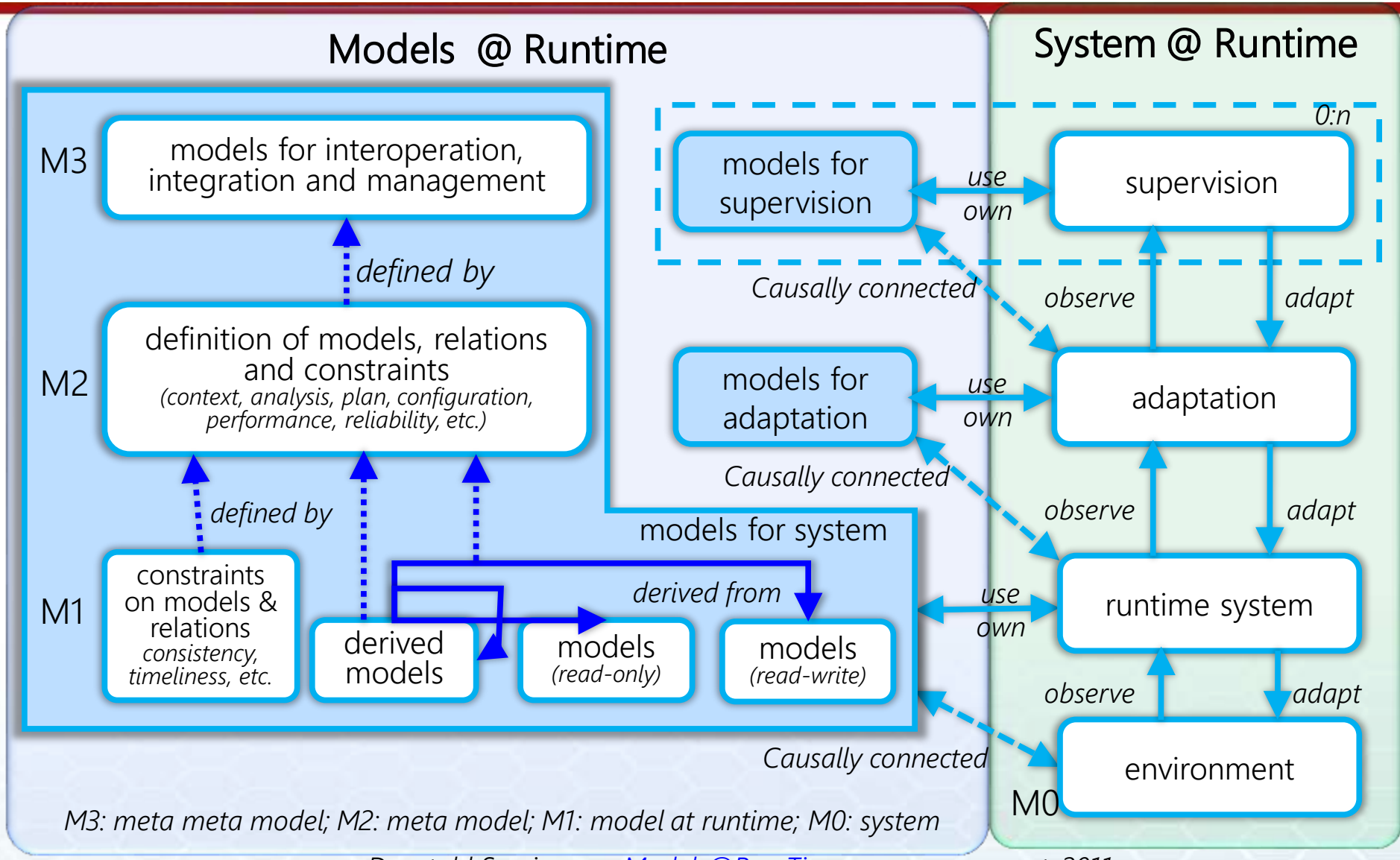
- Models@Runtime are models *causally connected** to the state and behavior of a runtime system

**means any change in the model will be immediately propagated to the system, and vice versa*



Richard Taylor, et al., ICSE 1998; Gordon Blair, et al., Middleware 1998

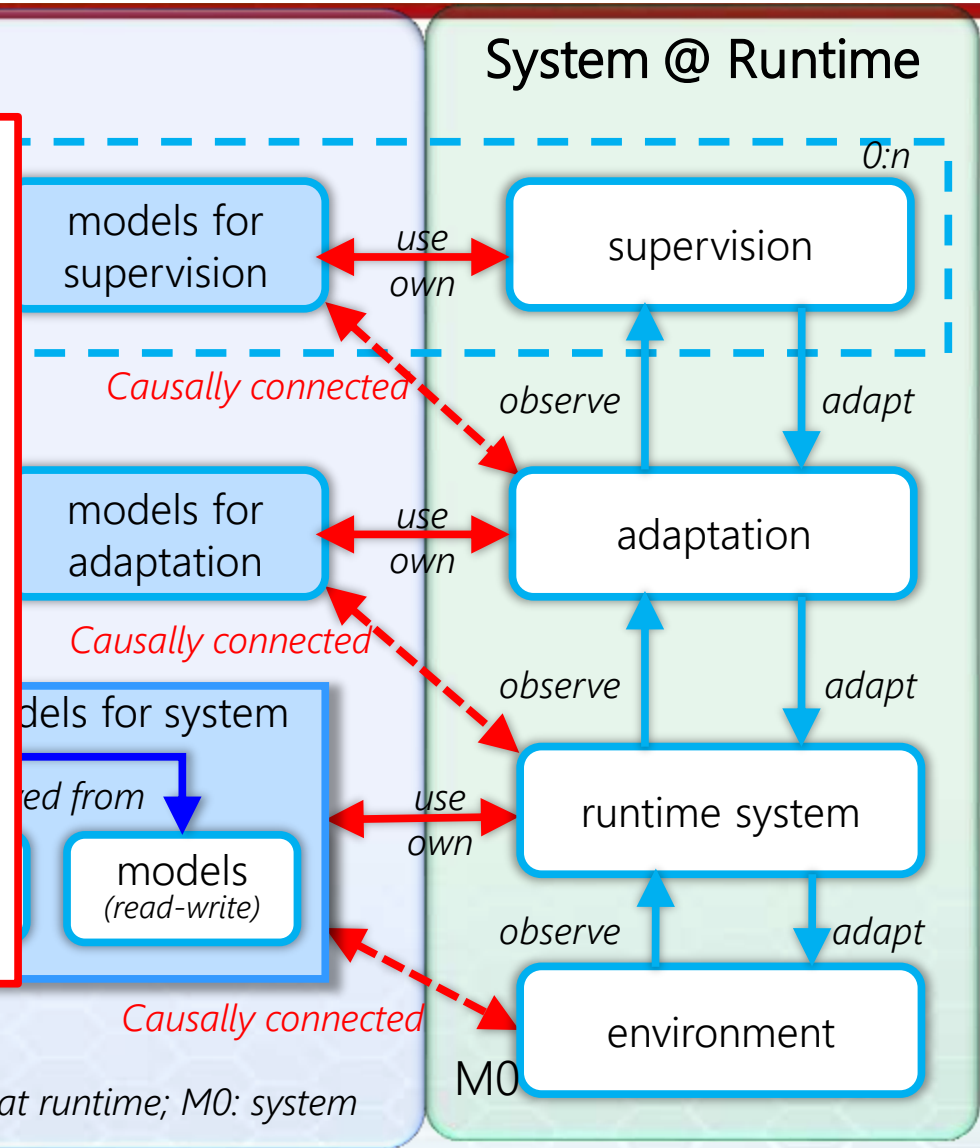
Conceptual Architecture of Models@Runtime



Dagstuhl Seminar on Models@Run.Time, summary report, 2011

Engineering Challenges to Models@Runtime

- How to implement and maintain the causal connections?
  - Systems use different programming languages and frameworks, provide different monitoring and controlling mechanisms
  - Users prefer different models
  - Legacy systems are the biggest obstacle in practice

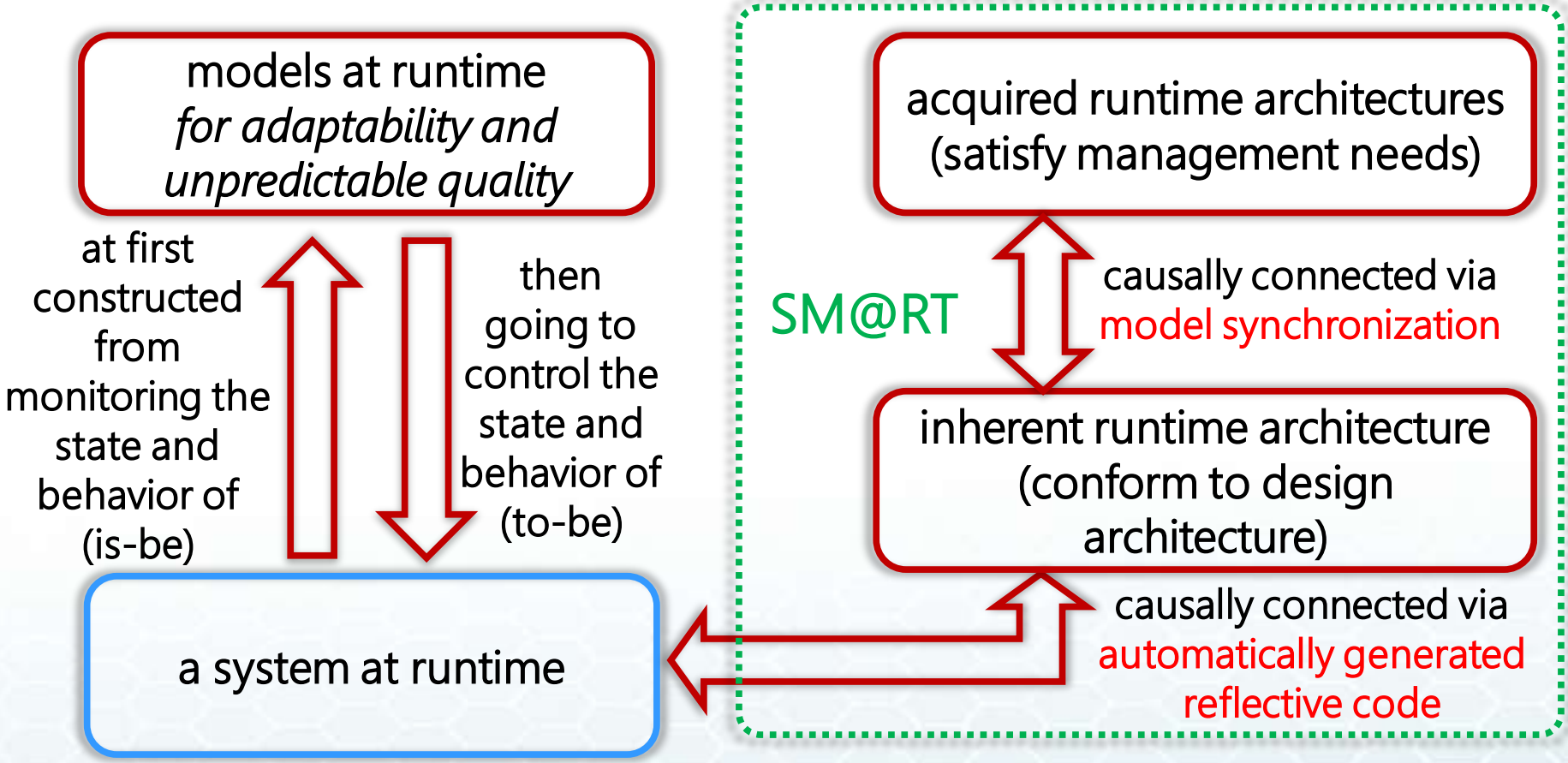


M3: meta meta model; M2: meta model; M1: model at runtime; M0: system

Dagstuhl Seminar on Models@Run.Time, summary report, 2011

SM@RT: Supporting Models at Runtime

- SM@RT: A model-driven framework for constructing the causal connection between the architectural models and runtime systems in an automated manner (using MOF/QVT standards)



Model-Driven Generation of Models@Runtime

InternetWare

Package javax.management

Provides the core JMX classes.

See:

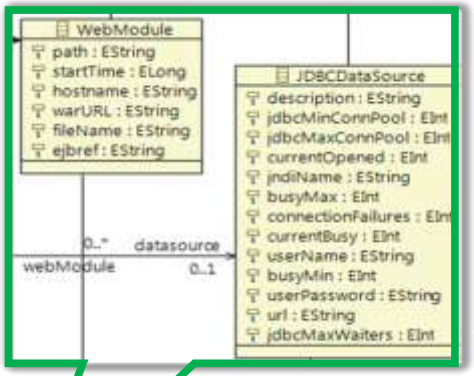
Description

Interface Summary

Descriptor	This interf
DescriptorAccess	This interf class which
DynamicMBean	Defines the that expose
MBeanRegistration	Can be impl and after b
MBeanServer	This is the

3) define QVT-based mapping between the meta model and system manageability

```
mapping SysElement::xAdd(prop:Property,v:Object):
   result:Object
from(self,type=MBeanServer; prop=ejb)
 var fileName=v.fileName;
 var server:=self.parent.auxiliary.server;
 \[ String[] signature=\"java.lang.String";
    String[] params=new String[$fileName];
    String deployedName=(String)mgmt.invoke($server,
      "deployJar", params, signature);
```



2) select the manageability of

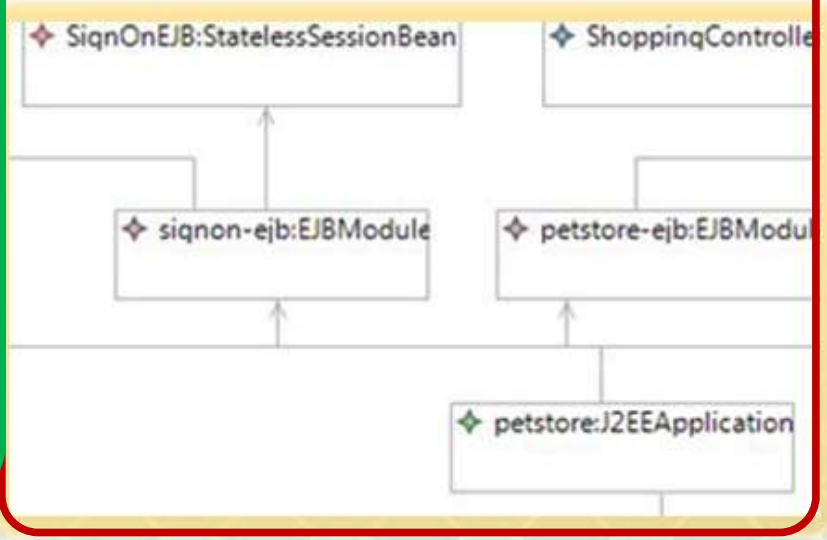
4) generate reflective code

1) define MOF-based meta model of

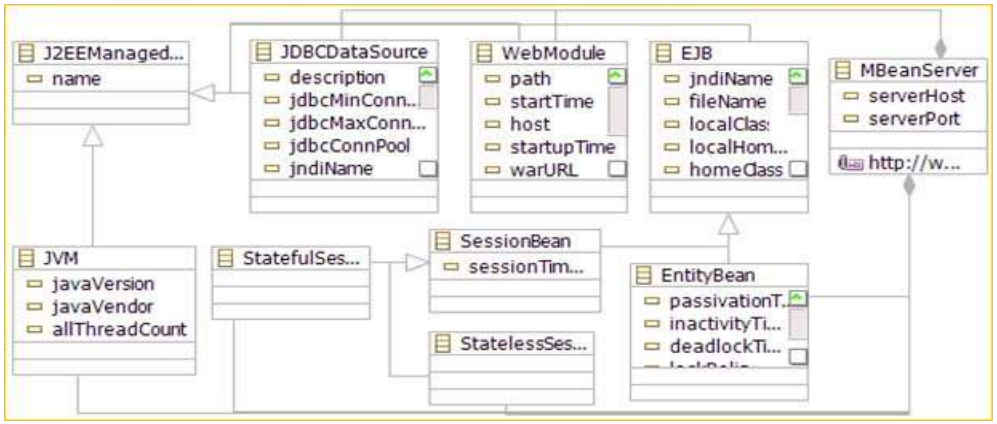
a system at runtime


```
1 public String getJndiName() {
2   try{
3     MBJB mainEntry=(MBJB)
4       PkuamanagementPackageImpl
5         .getMainEntry();
6     ObjectName core=getCore();
7     Object res=mainEntry
8       .getAttribute(
9         getCore(),"jndiName");
10    return (String)res;
11  }
12  catch(Exception e){
13    return null;
14  }
15 }
16
17 public Integer getMaxInstancePool(){
18   try{
19     MBJB mainEntry=(MBJB)
20       PkuamanagementPackageImpl
21         .getMainEntry();
22     ObjectName core=getCore();
23     Object res=mainEntry
24       .getAttribute(
25         getCore(),"maxInstancePool");
26     return (Integer)res;
27 ...
```

inherent runtime architecture

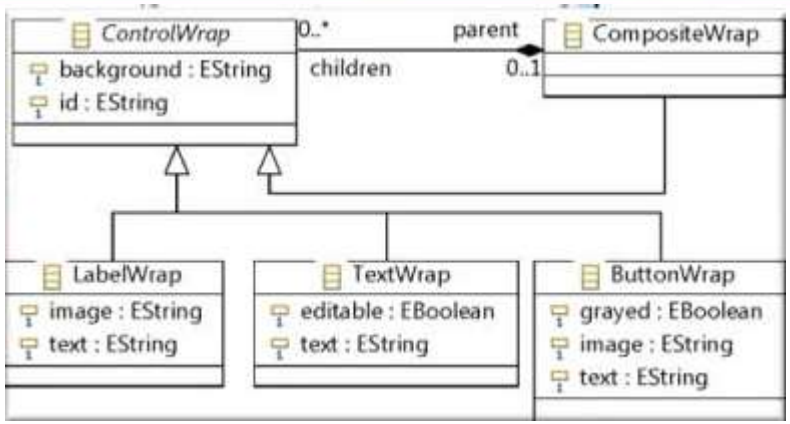


Case Studies



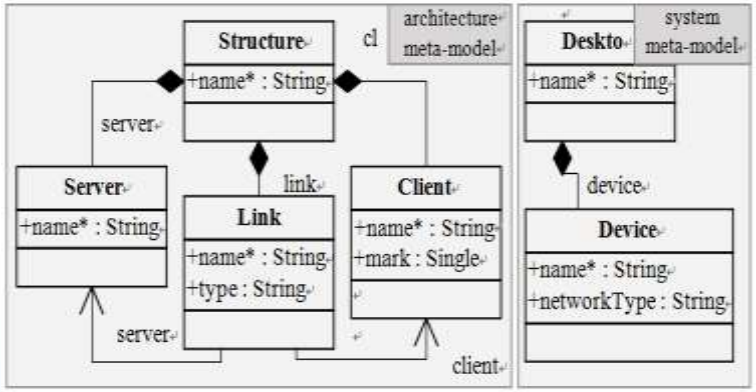
JEE (JonAS/PKUAS, Apusic) Inherent RSA MM

305ele+28map+310loc=22151loc



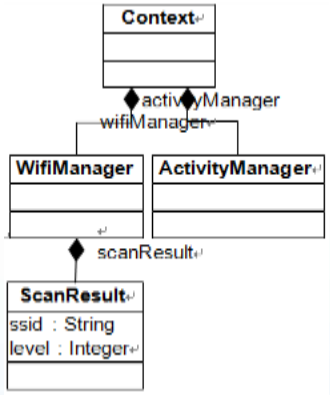
Eclipse SWT

19ele+23map+178loc=11209loc



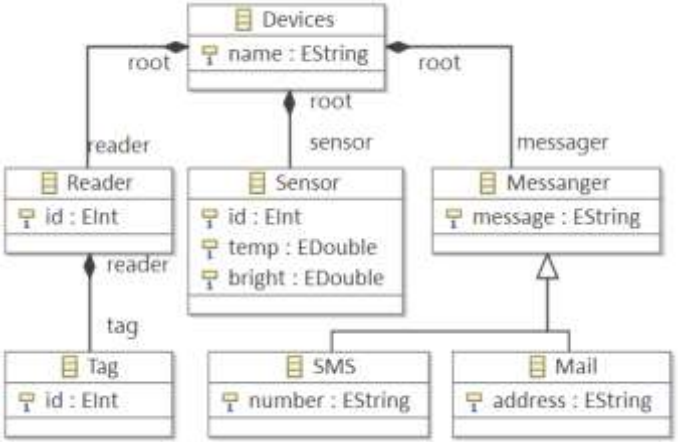
PLASTIC

6ele+13map+547loc=9126loc



Android

87ele+95map+431loc=21732loc



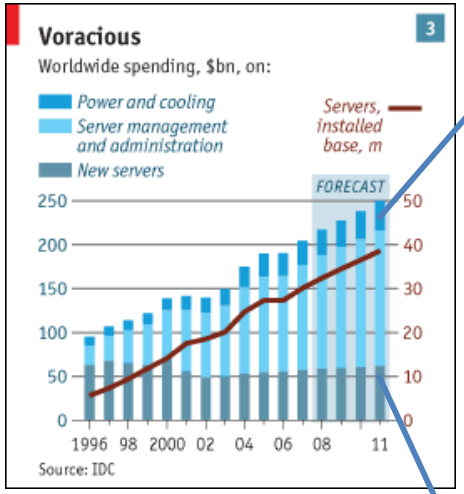
IoT Devices

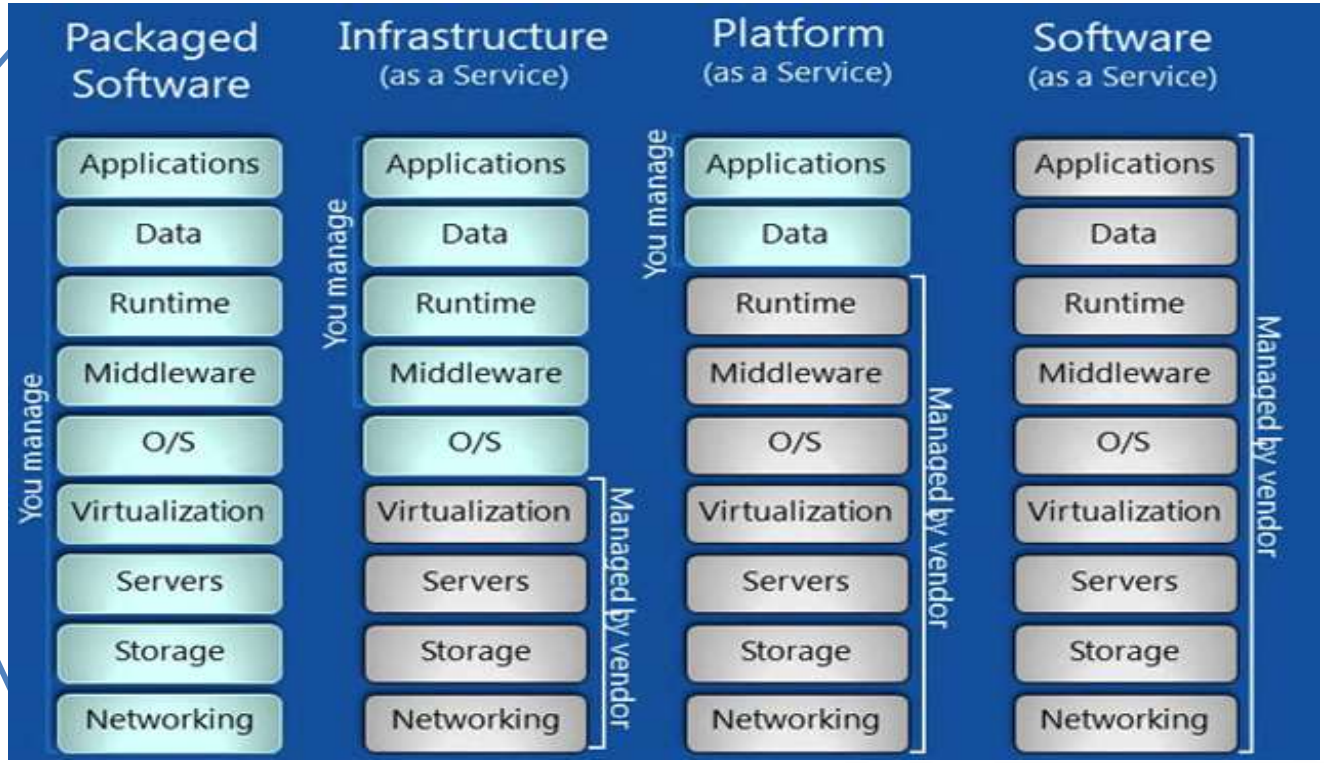
29ele+15map+267loc=8732loc

Cloud Computing

Visual Model of NIST Working Definition of Cloud Computing, 2011

Cloud Resource Management





IT management becomes a business
besides the traditional hardware and software
because they are too complex to be managed manually
which had been predicted by IBM in 2001*

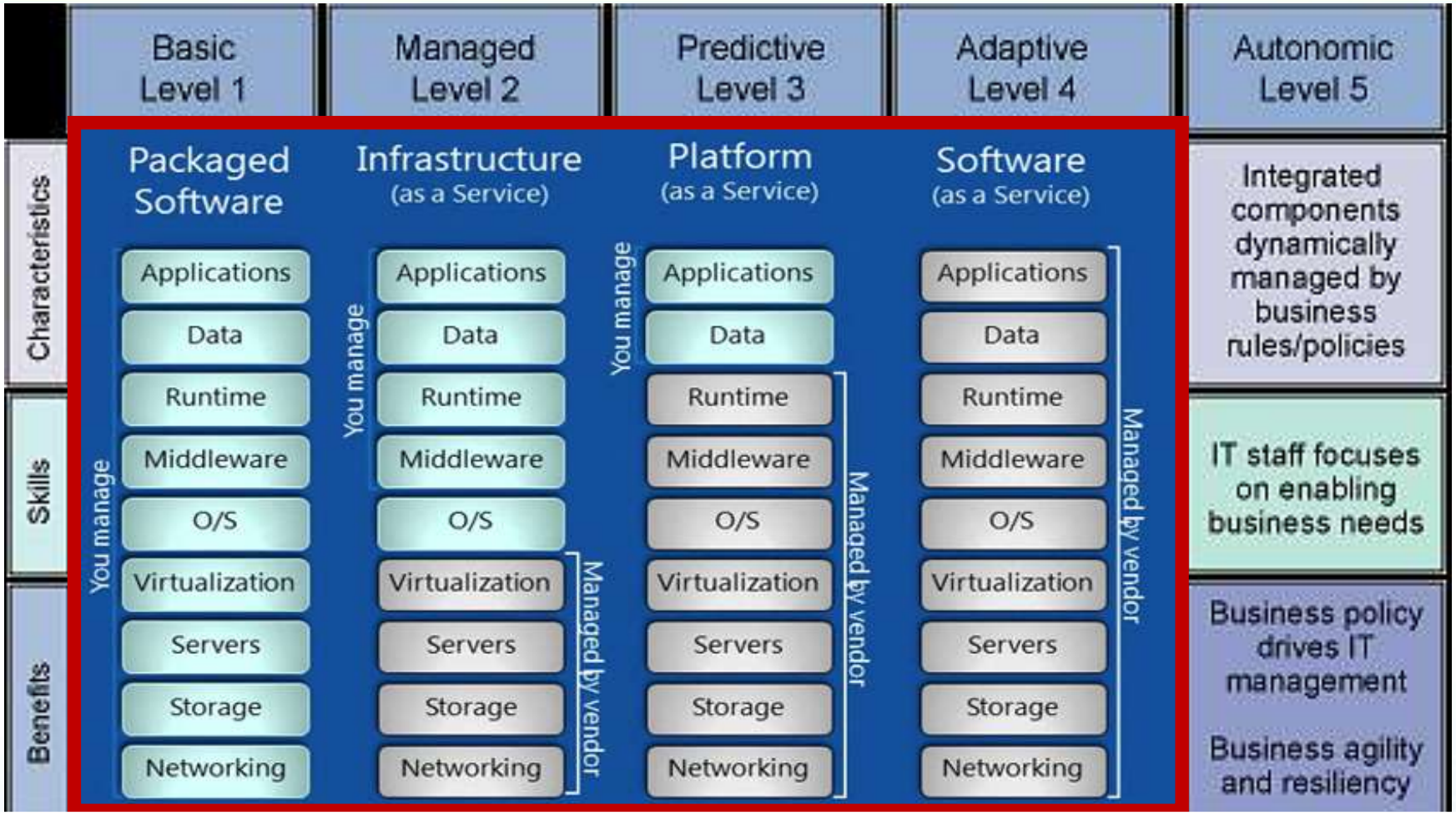
**Autonomic Computing: IBM's Perspective on the State of Information Technology, 2001*

Autonomic Management

InternetWare

	Basic Level 1	Managed Level 2	Predictive Level 3	Adaptive Level 4	Autonomic Level 5
Characteristics	Multiple sources of system generated data	Consolidation of data and actions through management tools	System monitors, correlates and recommends actions	System monitors, correlates and takes action	Integrated components dynamically managed by business rules/policies
Skills	Requires extensive, highly skilled IT staff	IT staff analyzes and takes action	IT staff approves and initiates actions	IT staff manages performance against SLAs	IT staff focuses on enabling business needs
Benefits		Greater system awareness Improved productivity	Reduced dependency on deep skills Faster/better decision making	Balanced human/system interaction IT agility and resiliency	Business policy drives IT management Business agility and resiliency
	Manual				Autonomic

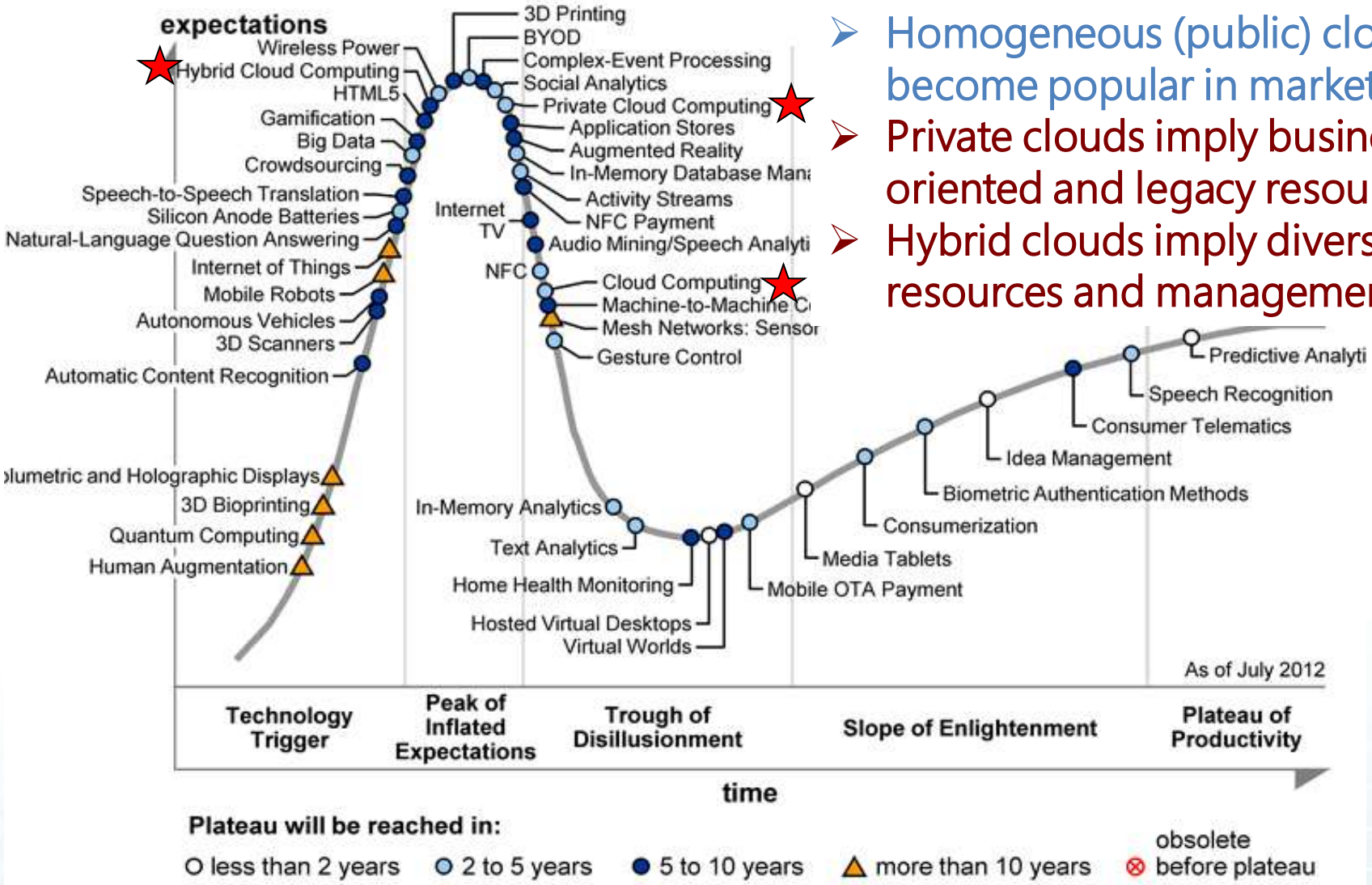
Autonomic Maturity of Cloud



More specific to application, better maturity but more private

None cloud can be proved/regarded as fully autonomic (Level 5)

Proliferation of Clouds



- Homogeneous (public) clouds become popular in market
- Private clouds imply business oriented and legacy resources
- Hybrid clouds imply diverse resources and managements

Gartner Hyper Cycle 2012.7

Challenges to Cloud Management

InternetWare

- How to build up **your** clouds with **your** requirements and resources ?

ABC Management Requirement

BC Management Requirement

big gap and complexity

cloudstack

openstack

Windows Azure

IBMSmartCloud

vmware

salesforce

Resource A

Resource B

Resource C

while existing cloud management systems have their predefined requirements and resources?

Models@Runtime for Autonomic Management

InternetWare

- How to build up **your** clouds with **your** requirements and resources ?

ABC Management Requirement

BC Management Requirement

Models at Runtime

3) define QVT-based mapping between the meta model and system manageability

2) select the manageability of

4) generate reflective code

1) define MOF-based meta model of

a system at runtime

inherent runtime architecture

Resource A

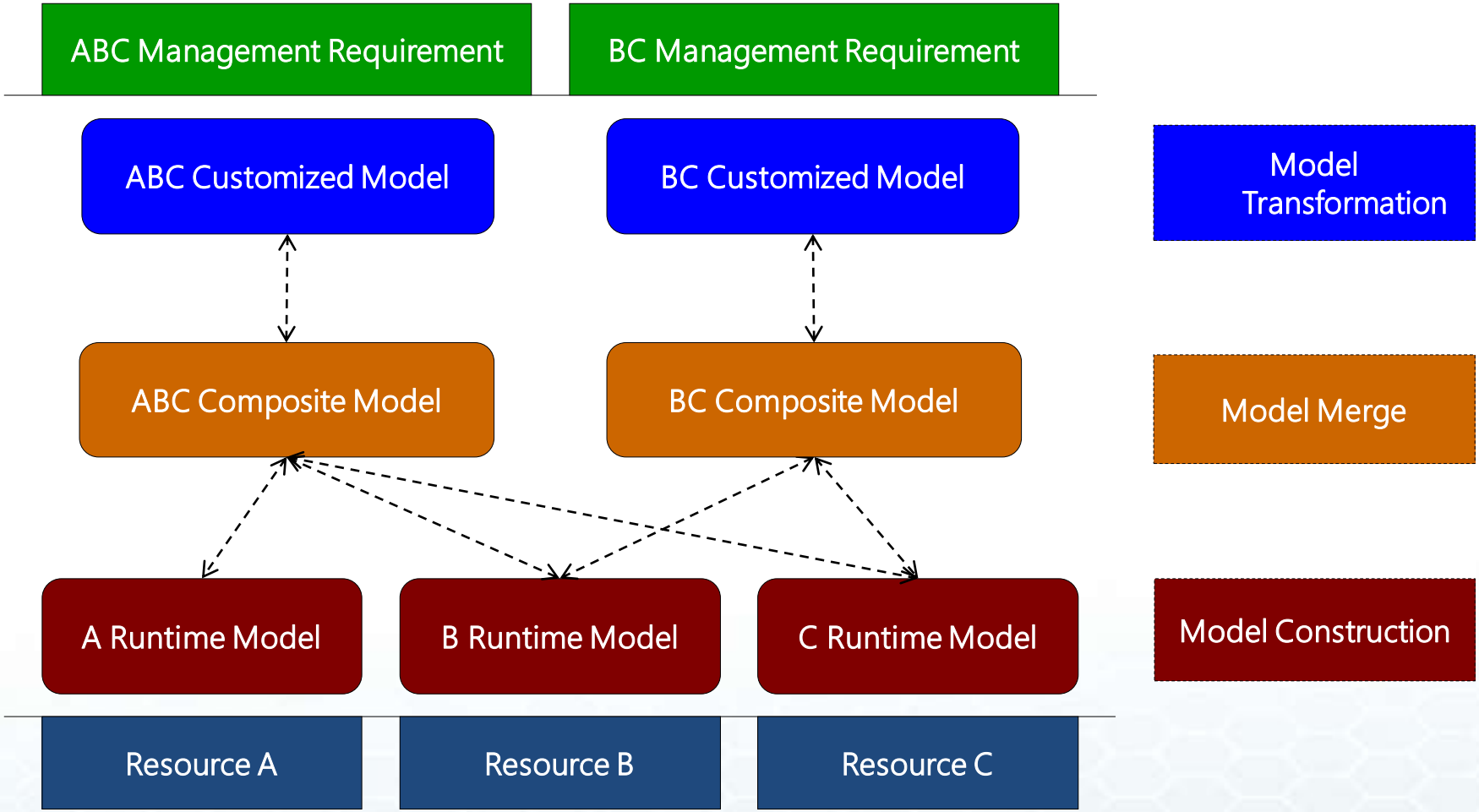
Resource B

Resource C

define your own management via models

manage any resources via models

SM@RT Cloud Management



Real Requirements on Cloud Management

I focus on the management of infrastructure

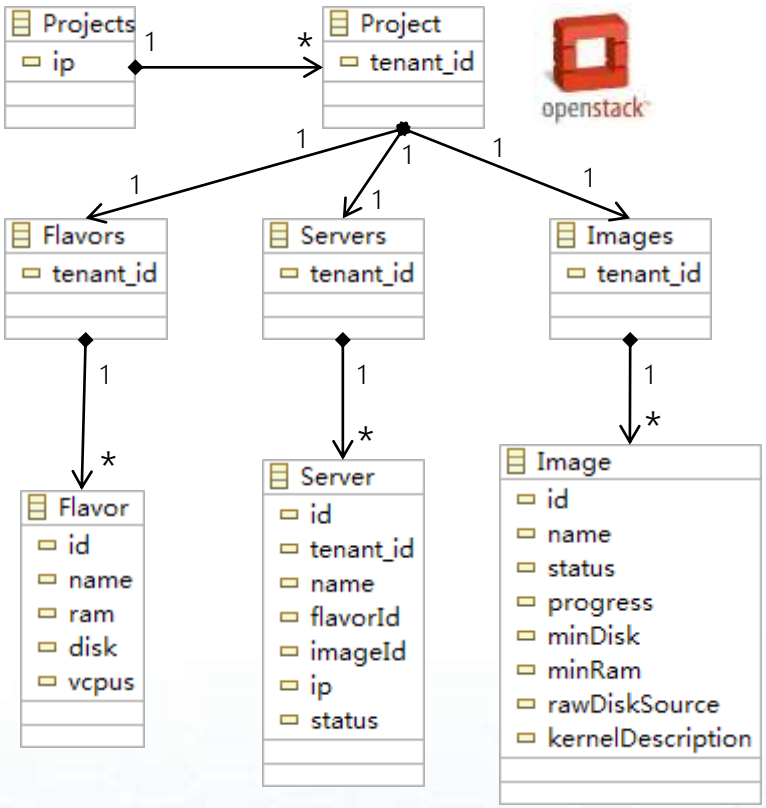
I focus on the monitoring and management of software

In our cooperation with the Information Center of Guangdong Power Grid Corporation, they propose the strong demand for managing both of the infrastructure and software resources in a unified manner.

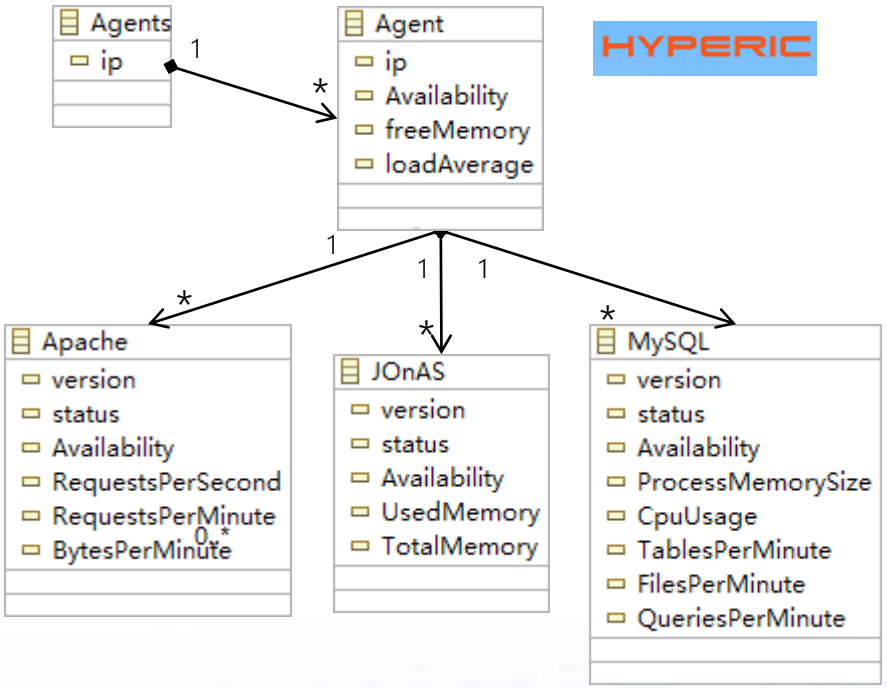
Construction of Cloud Resource Runtime Model

InternetWare

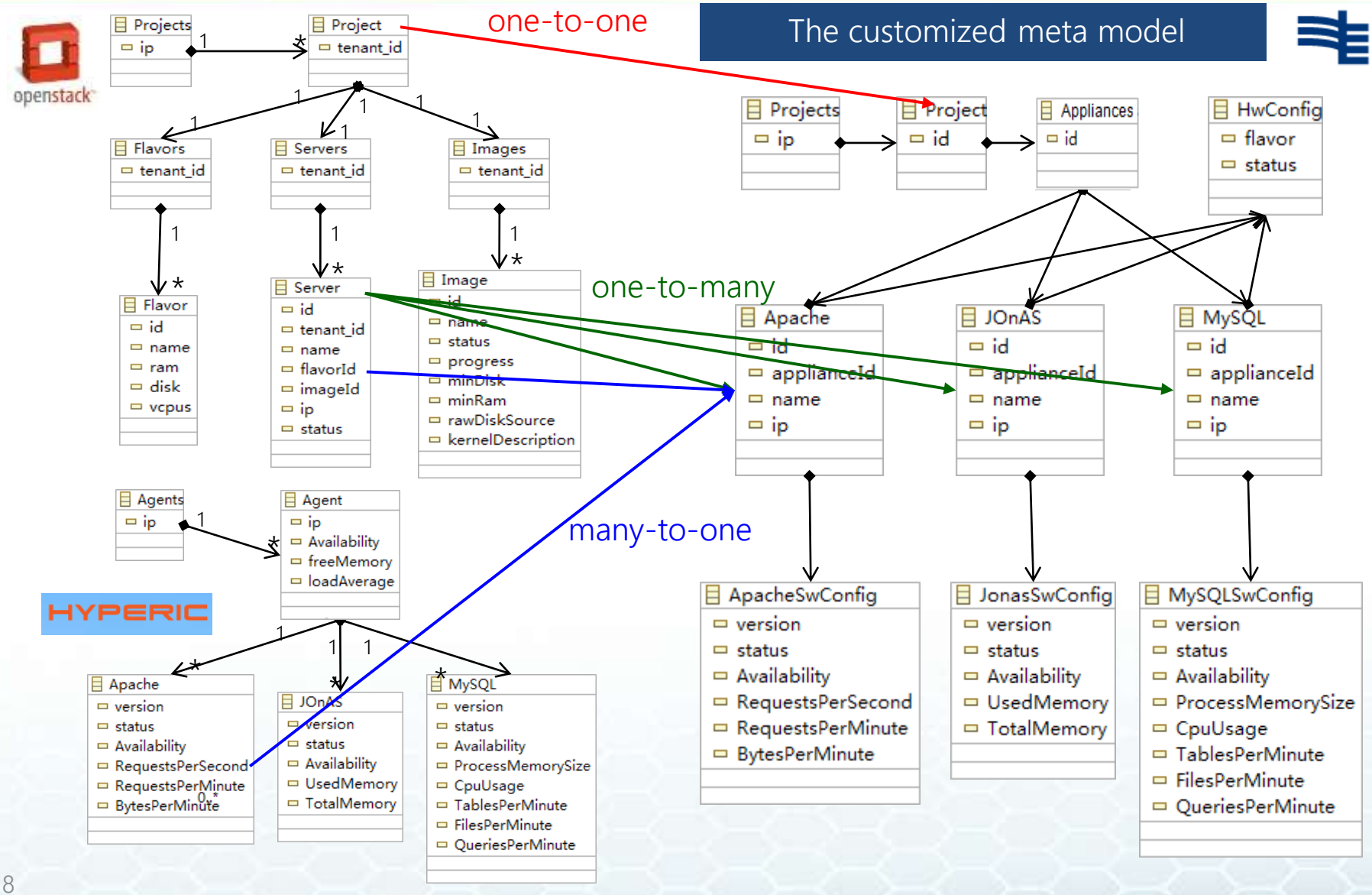
The meta model of OpenStack



The meta model of Hyperic



Model Transformation



QVT based Analyzer and Effectuator

//Fragment of Java Program

//Object: To check if there are nodes whose memory utilization is below 40 percents.

//1st Function - getAMemFreeNode(): It contains the main logic of the management task.

//2nd Function – getMemUtilizationOfNode(): It deals with the detailed implements of the runtime system.

```java
/*
 * JAVA: To get a node whose memory utilization is
 * less than 40%.
 */
public String getAMemFreeNode()
{
    String[] nodes = getAllNodes().split(";");
    int len = nodes.length;
    // To check every node.
    for(int i = 0; i < len; i ++)
    {
        String nodeId = nodes[i];
        String nodeIp = getNodeIp(nodeId);
        NodeClient nc = new NodeClient(nodeIp);
    /*
     * To get the nodes' memory utilization.
     */
        double memUltilization = Double.parseDouble(
                nc.getMemUtilizationOfNode());
        if(memUltilization < 0.4)
            return nodeId;
    }
    return null;
}
```

```java
/*
 * JAVA: To get a node's memory utilization
 * by invoking the management script.
 */
public String getMemUtilizationOfNode()
{
    try{
        String[] args = new String[2];
        args[0] = "/bin/sh";
        args[1] = "/opt/xen/getUsedMem.sh";
        ProcessBuilder builder =
            new ProcessBuilder(args);
        Process process = builder.start();
        BufferedReader br = new BufferedReader(
                new InputStreamReader(process.
                        getInputStream()));
        String line;
        int usedMem = 0;
        while((line = br.readLine()) != null)
        {
            //To get the value by parsing the string.
            String[] tokens1 = line.split(" ");
            if(!tokens[1].equals("ID"))
                usedMem += Integer.parseInt(tokens[2]);
        }
        process.waitFor();
        double memUtilization = usedMem / (double)mem;
        return String.valueOf(memUtilization);
    }
    catch(Exception e){
        return null;
    }
}
```

//Fragment of QVT Program

//Operations in QVT Language: select() – To return a list of the objects which are in a certain condition.

//To get the list of nodes in a free condition

```
var freeNodes = cloudModel.objectsOfType(Node)->select(VM.Memory->sum() < Memory * 0.4);
```

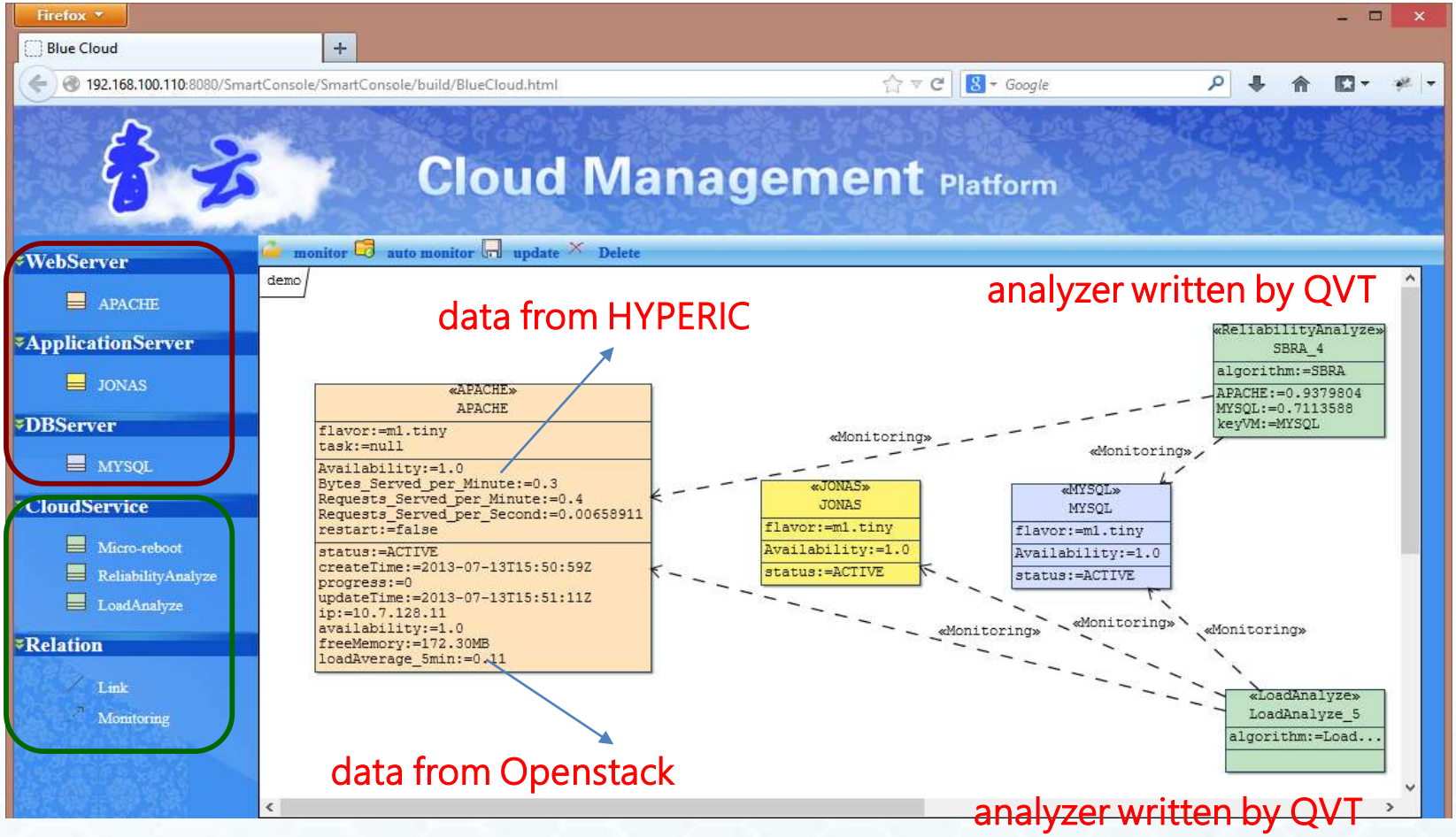
//To get the list of nodes in a busy condition

```
var busyNodes = cloudModel.objectsOfType(Node)->select(VM.Memory->sum() > Memory * 0.8);
```

SM@RT Cloud Prototype

Model based Resource Management

Model based Advanced Management



Evaluation on "Write" Operation

Management Tasks	Number of Appliances	Using Management Interfaces		Using Runtime Model	
		Execution Time (second)	Data Delay (second)	Execution Time (second)	Data Delay (second)
Create new appliances	1	22.9	-	24.1	-
	5	43.4	-	45.5	-
	10	58.7	-	62.1	-
Delete appliances	1	11.2	-	13.8	-
	5	29.7	-	30.1	-
	10	41.4	-	45.6	-
Set "name" attribute	5	2.1	-	4.1	-
	20	8.3	-	11.7	-
	100	39.7	-	44.2	-
Restart Apache	5	9.1	-	12.3	-
	20	37.6	-	41.2	-
	100	192.4	-	207.3	-

For the "write" operations, the execution time of Java programs is less than the QVT ones, because the two sets of programs are based on the same management APIs and there are some extra operations in runtime model based approach, which ensure the synchronization between the runtime model and real system.

Evaluation on "Read" Operation

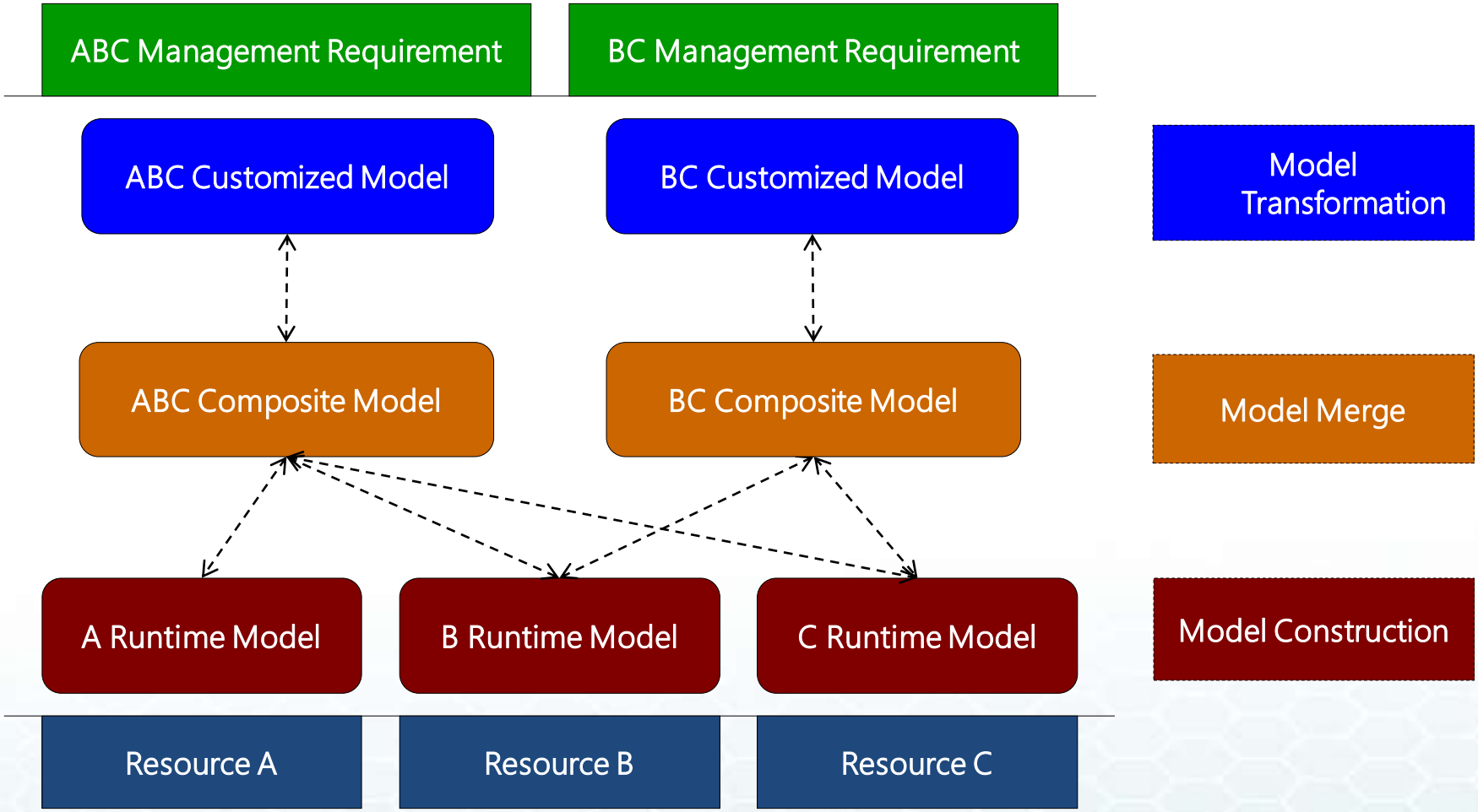
Management Tasks	Number of Appliances	Using Management Interfaces		Using Runtime Model	
		Execution Time (second)	Data Delay (second)	Execution Time (second)	Data Delay (second)
Get "usedMemory" attribute	5	1.2	0.6	0.6	30
	20	4.2	2.2	0.8	30
	100	18.6	11.4	1.6	30

For the "Read" operations, the execution time of Java programs is longer than the QVT ones, but the data delay of QVT programs is longer than the Java ones. On the one hand, the Java programs query the attributes of appliances through directly invoking the management interfaces, so the execution time increases linearly with the number of the appliances and the data delay is very little. On the other hand, the runtime model is equivalent to the snapshot of system metrics and getting the attributes of appliances just needs a reading operation, so the execution time of the QVT programs is shorter.

The Eclpise EMF framework is not efficient and scalable for runtime !

Conclusion

SM@RT Cloud: Runtime Software Architecture Based Cloud Management



Future Work

Commercialization of SM@RT Cloud

-manually write the reflective code of models@runtime

-develop new MOF/QVT engine for models@runtime

Thanks

SM@RT Cloud: Runtime Software Architecture Based Cloud Management

Gang Huang, Peking University, hg@pku.edu.cn

SM@RT free download: http://code.google.com/p/smatrt