1. Hybird cloud would satisfy their needs. They have their own infrastructure so that they must have their own private cloud and intranet with their data, storge and compute services. In this private they have total control with everything. But they would need extra resources occasionally, cause whey would need expanding its capability. And that would need public cloud to provide some resources.
2. ActiveMQ will provide interoperability. The language that the producers and consumers used are not revelant.

As a consumer you will have to create a connection. You have to set up a session and know who you are communicating with. The producer will do the same thing. Then sending and receiving messages.

This is interoperability issue. And not relevant with programming language.

The consumer can set handler, which is very much a sort of proxy. Once the message is copied into the queue, the consumer get automatic notification. Or alternatively, the consumer can simply pu the message in the queue. Simply check the queue if there are any incoming message.

1. With server consolidation, you need to limit or reused the number of servers currently in used in your cloud data infrastructure. You do this through what we called a virtualized based service consolidation.

And servers are running VM. So if you move all the existing running VM on the other server. You can then switch off the idle servers. Therefore reducing the actual cost inherently in managing your cloud data infrastructure. So the improvement is less server running, you have less cost or infrastructure to manage.

1. Automatic configuration of libraries in terms of software needed. For example, the networking like IP address, Mac address, etc. All this is supposed to be done transparently and automatically. It’s also hidden to end user of the VM.

With recontextualisation, we take the sernario where the VM has been set up and running in the cloud environment. Then the VM is going to run in a completed different environmnet, like a different service provider. The recontextualisation will take the information avaliable regarding the contextualization phase. And update that information so that your VM is able to run in different cloud environment.

You have a number of VM, and it’s need to configuations. Then contextualisation can do it automatically, like operating system, networking IP address, Mac adress, etc.

Then you VM is currently running on a cloud infrastructure. Then you decide to move VM to another infrastructure. Like you have a VM on Amazon cloud, then you want to migrate to your own cloud. And you also don’t want to stop it. Recontextualization can help you to take the contextualization information, and update that information, so that your VM will keep running in the new cloud infrastructure.

1. For capital expenditure. 基础建设支出You have to think about the building (real estate), need to buy machine, these are your servers in thousands. And the shelf or racks to put on these machine and to boot them. You need to set up the network and cooling system, etc. You need to invest your money in relation to building your data center.

For operating expenditure. The costs that are inherent in running in your cloud datacenter. For example, the electricity cost, you’d have to keep running your cloud data center and keep consuming electricity. If you are providing services, you are certainly running software and software have a licence associated with it. Itself is seen as an operating expenditure. Server containers are very much software that runs on a physical server. So containers are seen as running software costs.

1. The number of container that I will need to run applications should be known in advanced. Clarify you need this number of containers to run. Every time the system scales, I will need to re-estimate the number of containers for my application.

When it comes to serverless computing, it means that a number of containers is not known in advanced. But the number of containers depends on the demand. The code run automatically, create the container when needed.

Cotainers can scale automatically. The thing is the flexibiility from the programmer persepective, then it comes to deploying application. You need to bulid a executed file when you actually need it

If you need to invoke a container, the serverless computing is that the container is already up running. Then we invoke the function on that container. If it’s not running , we create one.

About cost, if you have to manage the end user on your own containers. The whole manage is on the container.

But the effort you put in setting up your containers for your application is hidden with the serverless computing architecture. Because there is no back end to manage. The cost is very much inherent to that, the serverless code is always executed when needed.

Then you will need to count the actual time for your containers to be set up. For example, you need to configure the system setting in the library. The driver is still extra work to do on you deployment docker. The function will need millisechonds to be deployed. It’s not your problem, the whole back end is managed by the service provider itself. Again, the time for deployment is the certainly Containers more than serverless.

They are certainly serverless better when it comes to managing contains. But containers are still very important and are the core of serverless computing. More importantly, any serverless computing architecture is there would rely on containers. The manage behind serverless computing seems to be more on the cloud provider side rather than the cloud customer side. That’s why a lot of hassle, in terms of setting up containers, managing these containers is somehow hidden in a serverless architecture

1. Uner-provisioning means, as a cloud provider, we tend to make less resources avaliable for your customers. Wheras the over-provisioning make more resources avaliable that there is a waste.

If you have SLA with the customer in an under-provisioning of resources scenario. You may not have many resources avaliable for everyone. It’s likely to lead to a SLA breach. It’s a breach of contract. Because you have signed a contract with your customers. The customers want to accuss the resources. Unfortunately, they are not getting the resources that they have asked for.

For over-provisioning, you have more resources that you need as a cloud provider. You customer will access the resources they need. They are all happy and satisfied. However, you may end up with wasting those resource. And there is a waste of resources in terms of CPU, memories, electricity and the light. There is always a good chance that your SLA will be fulfilled.

As a cloud provider, you want to cut on your cost. But with the risk that may end up in an SLA breach. If an SLA is not fulfilled into an official contract, you may end up paying penalty to your customer.

1. Cloud heterogeneity.

Because the cloud infrastructure is not changeless, it constantly evolved. And many service providers have increased their offerings and update their hardware, such as CPU, GPU, FPGA, to meet customer demands and improve performance. And different provider may use different hardware as well as the operating system. Heterogeneity always exists, and people are trying to use cloud computing to solve this problem. With heterogeneity, it can use co-processor and achieve data parallelism, low latency processing and low power consumption capabilities.

Also in terms of the network, the IP address and so on.

Cloud automation. You could potentially collect data about their cloud infrastructure analyse that data and plan for your infrastructure to be managed differently. Therefore, need to monitor the plan. You can expect the management of your infrastructure to be automatic.

And it’s important when it comes to QoS for customers and providers.

Cloud evolution. Users can access elastic virtual resources. And cloud solution is transparent to the users.

1. (i) R

(ii) K over R

(iii) M \* R

1. It looks like the energy efficiency in each layer of the software stack.

The number of transaction inherent to the storage to reading, writing into files storge. Access to storage devices can be seen as consume energy. So I can design and implement software, taking into consideration these issues. Ties application into thread is an excellent way to potentially address the issue with the energy.

With middle. Thinking of a service or an application to be energy aware. Then deploy the service in cluster X, not in Y. Because the cluster X is more energy efficient than cluster Y. So there is a decision that when it comes to scheduling allocation of thoughts into cloud infrastructure.

Resource are very important. You could see virtual resource and the physcial resource. Allocate VM to an energy efficient cloud server or physical server. For example, the PDF is the dynamic all page frequency system is an interesting mechanism. If I reduce the both page on my CPU. Im going to consume less power. But I will accept it will take me longer. DBFS 集群文件系统 (DVFS??) is another mechanism, that is certainly used in modern server and also your mobile phone. If you deploy services are geographically closer, you wouldn’t take account into the issue with bandwith, latency …

From the network perspective, introduce data with input and output and data is transferred over the network. There is efficient network protocol such as routing protocols. It’s likely that your network will consume less energy. How the network is designed to save energy.

Whatever you do, bear in mind that you will always need to take into cost. Save energy and you have to pay a price for performance.

1. The simpler requirements of this is access control. So I need to control the access to this application. For example, for the threat, think about any hijacking. Session hijacking could be a threat, because if the session associated with the application is hijacked. Then create on security issue with application. …that software interruption be the issue how exploited in order to have access to it was to create a threat on my virtual machine.

If I take a phsical server, as part of the requirement, I have a network protection so the physical server is protected through the network. What if I have an intrusion detection in my network as part of a threat. What if I have a DOS, for example an attack on my network. That’s clear a threat to my physical server. Therefore, the protection on the server is compromised. 可以从reference找到例子之类的

IoT scenario.

Define an architecture, the architecture has to be layered, the sensor layer, network layer. Then you need to explain how data is actually sent, what are the rules? Talk about the network infrastructure.

Once the data is sent, it will get stored… What kind of technology are you going to use? Once the data is there, what are you going to do with it? What the data is processed. Some format of software, like hadoop. And you may potentially use prediction models. Because you pick up the data, you will get prediction models. Those models will tell you: according to the data that I have, the level of pollution is defined, or the level of pollution is approaching a threshold. A threshold where do we need to generate a signal or generate a message to the authority and say look, we have to do something about it.

This can be generated on screen available in the throughout the city. 有几个停车位之类的问题。

This kind of information is going to be generated all the time. Because you will keep send data to the cloud for constant processing.

In summary. Start with the architecture, then for each layer of the architecture, explain what it is, what layer actually does from data centre sesing to data transmission to data storage. Insist the technology that you would like to use. Generate usful information and use to address potential problems.

Build the IoT. First, deploy the sensors, cameras and screens everywhere around the city. Maybe we can also put GPS on each sensor to monitor each locations condition around the city. And this make the sensing layer.

Then connect thounds of sensors into the network through the gateways or routers. Like the mobile telecom network, or the information network. Then transfer the data from each sensor to the cloud computing platform. We may use TCP or UDP to do that. This make the network layer.

Up in the cloud, we may use HPC to process these data, or storge the massive data into datacenter, and use MapReduce to deal with these massive data.

Finally, we can build some application to interact with these data, which was processed in the cloud. Like the environment protection app, intelligent traffic app, etc. This make the application layer.

These three layers make the whole IoT. Gathering those useful information, like the traffic flow or the air condition in the downtown. And we can store them into a datacenter or server racks. And with these data we can build a predict model to optimise the city’s traffic, etc.