JOSE: An Open Testbed for Field Trials of Large-scale IoT Services

Yuuichi TERANISHI, Yuki SAITO, Sakae MURONO, and Nozomu NISHINAGA

We have developed a testbed called JOSE that can provide functionalities required to build Internet of Things (IoT) services as virtualized network services. JOSE is an open testbed and at the same time, a service platform which can accommodate multiple large-scale field trials that consist of sensor networks, storage resources and computation resources. In this paper, the base technologies of JOSE and operation status are described.

1 Introduction

New generation network services that are so-called 'Internet of Things (IoT)' services, which make use of network-connected things (such as sensors, mobile terminals, robots and cars) to provide labor-saving, safe and efficient ICT services at anytime, anywhere, are attracting great attention. The network-connected things continuously generate and provide data such as observation data from sensors (sensor data) as network services. The continuous data generation occurs periodically or intermittently. The number and volume of the accumulated data become huge: this is the so-called "big data," even if the size of each sensor data is small, when a massive number of things are connected to the network. By obtaining, transmitting, accumulating and analyzing continuously generated data, network services are expected to have new abilities to enable fast disaster countermeasures, stimulation of local industries, and so on.

The IoT services need to be constructed as unified and secure systems, which consist of functional elements such as sensor networks, storage resources, computation resources, and networks that connect functional elements. These functional elements are necessary to construct systems that handle the processes described above (obtain, transmit, accumulate and analyze). We developed a platform to provide such functional elements as services on a virtualized network, which we call "JOSE: Japan-wide Orchestrated Smart/Sensor Environment[1]." JOSE is a sort of Infrastructure as a Service (IaaS) specialized for IoT services, which can provide required functional elements of IoT services reducing the investments required for computer servers and network facilities.

JOSE parallelizes and automates the configuration of storage resources and computation resources using virtualization technology (virtual machine technology and virtual network technology). JOSE also utilizes P2P network technology to construct a sensor data sharing network as an autonomous and distributed network. These features enable JOSE to reduce the operation steps and time that are needed to manage large-scale resources.

This paper describes the principles of JOSE, and its operation status as a testbed.

2 IoT services platform provided by JOSE

2.1 Elements comprising IoT services

Figure 1 shows functional elements of IoT services handled by JOSE. For each IoT service accommodated by JOSE, the functional elements are provided as virtualized services on physically shared hardware. Each functional element is described below.

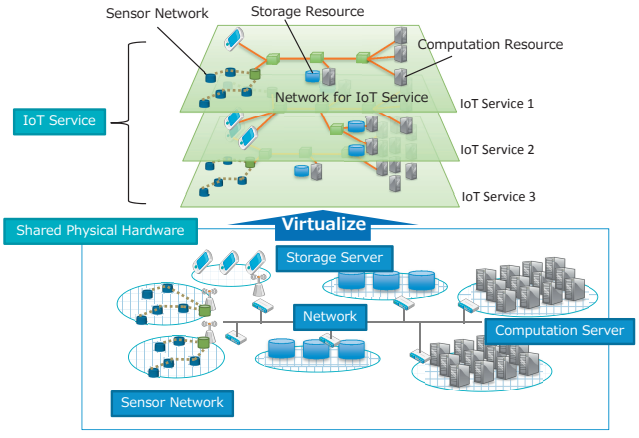


Fig. 1 Elements comprising IoT services

1) Sensor network

Sensor networks (or sensor actuator networks; hereafter, "sensor networks") are unique functional elements needed for IoT services. In IoT services, sensor networks are deployed in areas targeted for observations. In the advanced IoT services, actuators (terminals, PCs, devices, etc.), which notify some events or change behavior, are connected as sensor networks' functional elements. There are two types of devices for a sensor network: one is a high-power device, which has high-level communication ability such as Internet protocols to connect to servers etc. The other is a low-power device, which only has low-level communication ability such as short-distance, simple network protocols, and constructs an ad-hoc sensor network which requires a gateway device to connect to servers. The gateway relays the short-distance, simple network protocols and Internet protocols.

There are the following two types of sensor networks in JOSE:

- Sensor networks on which IoT service is newly deployed
- Sensor networks which are already deployed and sensor data is shared among multiple services

2) Storage resources

Each IoT service has its own dedicated storage resources. Storage resources are component elements that accumulate sensor data obtained from sensors. Storage resources in JOSE are provided by virtual machines on the *storage servers* equipped with large volume hard disk drives.

When continuously generated sensor data is accumulated over a long time, then even if the individual data size is small, huge capacity will be required. At the same time, to make sensor data usable for analysis or machine learning, the sensor data must be searchable. To enable high-response searches, the sensor data must be stored in an indexed database.

In addition, when there is cooperation between IoT services, coordination between storage resources is required. In this case, data on the storage is shared: access rights to the stored data are given to other IoT services and/or access rights to the stored data of other IoT services are obtained.

3) Computation resources

Computation resources are component elements to process sensor data. In JOSE, the computation resources are provided by virtual machines on physical *computing servers*. When a complex process such as weather forecasts by simulation is running, the required computational power becomes large. Therefore, to finish processing within a certain time, adequate computational resources must be assigned to the process. In big data analysis, to obtain outputs from a process in a short period of time, a *scale-out* architecture that improves performances by increasing the number of computational resources is required. In the scale-out architecture, the larger the number of computers, the less time is needed for processing. Therefore, the number of computational resources that are needed to be assigned to the application depends on the required process completion time and the scale of data to be processed.

In IoT services, one must also consider cases in which the required computational resources change depending on situations (events) that occur in the real world. For example, in an application that forecasts the status changes of rainfall conditions using simulations, appropriate computational resources are required when a heavy rainfall occurs. However, when there is no rainfall, no simulation process runs, thus no computation resources are required. Therefore, in this application, the required computational power changes depending on the occurrence of rainfall events.

4) Dedicated network for IoT service

As described above, the required structure of functional elements of IoT services can be changed in a short time. Considering that the demands can change frequently, the providers of IoT services should have rights to change the structure of functional elements freely by themselves. On the other hand, in IoT services, there are great needs for maintaining the confidentiality of each service. Sensor data is not always usable as open data; for example, IoT services that utilize personal data such as people's location information must be operated under individual privacy policies with confidentiality. Even for environmental sensor data such as temperature and amount of rainfall, in cases harmful rumors have effects on tourism, the sensor data must be handled carefully. When such sensor data is handled, to avoid threats of attacks such as illegal packet capturing, they may want to adopt a security policy that inhibits sharing the network with other services. Moreover, even without malice, one should consider network malfunctions and network traffic congestion that occur due to failures and misconfigurations. Countermeasures against such situations are important since JOSE is a testbed, and verifications of technologies under development are conducted on daily basis.

Therefore, in JOSE, each IoT service is configured on an independent virtual network. Each virtual network of JOSE implements the following:

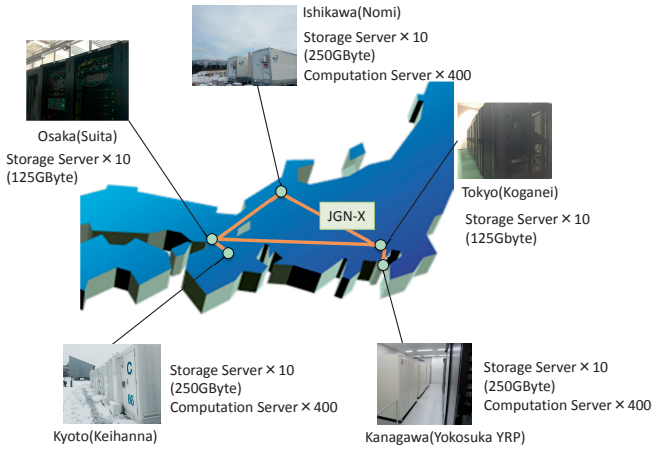


Fig. 2 Hardware environment of JOSE

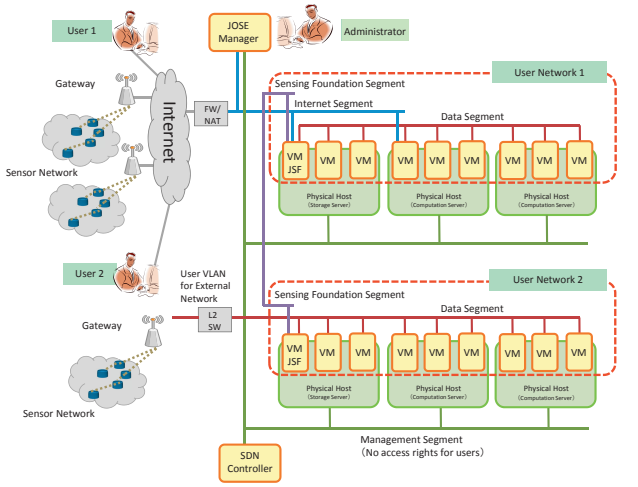


Fig. 3 Example of configuration of IoT services provided by JOSE

- Data traffic is separated for each IoT service
- Structure of IoT services can be changed independently

2.2 Hardware environment of JOSE

JOSE aims to achieve an IoT services platform that accommodates many IoT services simultaneously, by combining the functional elements shown in the previous subsections. As hardware equipment, 1,250 physical servers were deployed in data centers in five distributed locations (Tokyo, Osaka, Kanagawa, Ishikawa and Kyoto). In each physical server, up to 16 virtual machines can run. Therefore, up to 20,000 virtual machines are usable as computational resources. The total storage size in all the data centers is 1 PByte.

The data centers are connected to each other in a wide area layer 2 network provided by JGN-X[2]. The network facilities have Software Defined Networking (SDN) functions. The data centers are linked by a 10 Gbps core network and each physical server has a 1 Gbps dedicated access link. There are 3 divided SDN domains on JOSE since there is a limit to the number of virtual servers that can be controlled by one SDN controller.

Based on this hardware environment, JOSE provides an IoT services platform that virtualizes and provides the functional elements needed in IoT services.

3 JOSE implementation technologies

This section describes the technologies to implement JOSE as the IoT services platform shown in Subsection **2.2**.

3.1 Basic configuration of IoT services provided by JOSE

Figure 3 shows a sample configuration of IoT Services provided by JOSE.

In JOSE, a "user" is a provider of IoT services (service provider: experimenter of IoT services as a testbed). The virtual network of each IoT service (called a user network) is configured on the shared hardware equipment of JOSE.

In JOSE, there are also "administrators." Administrators of JOSE register and delete users, create IoT services, receive users' requests, change configurations and monitor the status of IoT services.

JOSE users have dedicated storage resources and computational resources on the user network, and can freely customize them by installing original programs, etc. Also, JOSE users can change the configuration of IoT services following the requirement changes. In JOSE, according to the levels of customization for the user network, the following 3 types of service classes are provided.

1) Static class
2) Dynamic class
3) RISE class

In the static class, requests of user network configuration changes are submitted to the administrator and the administrator changes the configuration with administrator's access rights. Requests for IoT service configuration changes by users are not immediately reflected in the user environment; the network is configured after approval by the administrator.

In the dynamic class, a user can change the configurations of the user network with user's rights. Requests for IoT service configuration changes by users are immediately reflected in the user environment.

In the RISE class, users can customize the SDN controller itself and can change the configurations of the network with user's rights. Users can control the user network more freely than the dynamic class using their own controller. For details on SDN virtualization methods, refer to the RISE article[3].

In Figure 3, there are two user networks, and each user network is configured by the static class. Each user logs in via the Internet to the JOSE Manager (see Subsection **3.2**), and operates his/her allocated virtual machines such as boot, shutdown, etc.

In user network 1, virtual machines (VM) are generated on storage servers and computing servers, and the sensor network is connected via the Internet. Data exchanges between storage resources and computational resources are executed on *data segments*. Also, in user network 1, virtual machines on storage servers and computing servers connect to the *Internet segment* (NAT segment), and can access the Internet. In a virtual machine on the storage server, the JOSE Sensing Foundation (JSF: see Subsection **3.3**) is installed and the sensor data is shared with other linked IoT services via the *sensing foundation segment*.

In user network 2, the sensor network or terminals for virtual machine management do not access JOSE via Internet, but access via s wide area layer 2 network service such as JGN-X. To access JOSE, a user's network is connected directly to the data segment via a user VLAN. Therefore, the Internet segment does not exist in user network 2.

The *management segment* is a network for administrations to change the configurations. The configurations are changed by *JOSE Manager*, which will be described in Subsection **3.2**. The management segment only connects to a physical host; it does not connect directly to a virtual machine. By this configuration, it maintains the independence and confidentiality of IoT services.

The JOSE Manager operates virtual machines and virtual networks via the management segment. Virtual machine settings and status monitoring from JOSE Manager are executed via the physical host, and virtual network settings and status monitoring are executed via the SDN controller.

Dynamic and RISE classes can also be configured similarly. These classes configure the virtual machine and virtual network on hardware with a physically separated network. In the dynamic class, the administrator does not operate the user network via JOSE Manager; the user operates using the user's rights. In a RISE class, virtual network settings are done via a dedicated SDN controller customized by the users themselves.

3.2 JOSE Manager

JOSE users can specify the following requests for the user network.

1) Virtual machine-related requests
   (a) Number of virtual machines
   (b) Template image (described later)
   (c) Deployment location
   (d) Memory size
   (e) Number of CPUs
   (f) Storage size
2) Requests for virtual network
   (a) Data network settings (L3)
   (b) Internet for external connections (L3)
   (c) User VLAN network for external connections (L2)
   (d) Sensing foundation network (L2/L3)

JOSE has a management mechanism called "JOSE Manager" which handles various management operations, i.e. receives user requests, reflects them on the physical host and SDN, collects and visualizes the usage status, etc. In the above virtual network settings, the network noted as "L3" is a layer 3 network; IP address allocations and routing settings are done via JOSE Manager. The network noted as "L2" is a layer 2 network; users themselves must set up IP addresses, etc.

The JOSE administrator, upon receiving user requests, reflects these requests in JOSE and monitors the usage status via JOSE Manager. JOSE Manager accounts are assigned to the JOSE users. The users can login and operate via JOSE Manager's web interface. After the login, the users can see a list of virtual machines allocated to them. They can also add/delete running virtual machines, via JOSE Manager.

3.2.1 Management of virtual machines and storage

In JOSE Manager, virtual machines with basic functions (Linux OS kernel + user land) are prepared as "template images." A template image can be selected from among multiple candidates, according to the user's needs. Currently, there is a template image based on the Ubuntu Linux distro.

JOSE Manager generates a virtual machine image (user image) from a template image, and runs it on a physical host that is allocated to the user. When the virtual machine first launches, the user image is made from a template image. Thus, a new user can always start using the updated latest OS version.

Virtual machine setting commands based on user requests from JOSE Manager are sent from physical hosts as a parallel batch program, which reduces the labor of administrator to make settings in virtual machines one by one.

Each virtual machine generated from a user image has the following functions by default which are needed to run in JOSE.

(1) Network control
The function to synchronize the network settings with SDN settings.

(2) Storage control
The function to allocate/mount file systems on the virtual machine.

(3) Status notification
The function to notify the status of the virtual machine to JOSE Manager. It is launched as a background process when the virtual machine is started. JOSE Manager is notified of the virtual machine's running status and usage status.

By the function (1) above, network settings in virtual machines follow the request from JOSE Manager. In principle, in JOSE, each user cannot directly change network settings on the virtual machine.

By the function (2), storage space according to the user request is provided for each virtual machine. According to the storage size specified in JOSE Manager, the "storage disk image" is generated on the physical host, and it is mounted from the virtual machine. A large size storage disk can be allocated if the physical host of the virtual machine is the JOSE storage server.

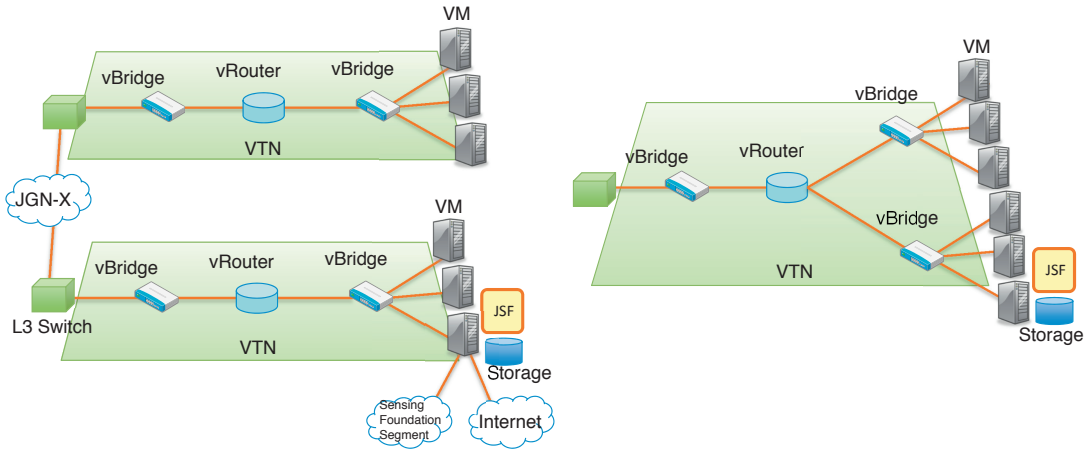
By the function (3), running status and usage status information of the physical hosts and virtual machines can be obtained in JOSE Manager. This process notifies the machines' statuses to JOSE Manager at certain time intervals. In JOSE, there are tens of thousands of virtual machines. Therefore, if each machine sends status notifications directly to the JOSE Manager, the number of data received for each time period from different physical hosts becomes huge, causing a large communication load by accepting connections. To limit the communication load on JOSE Manager, some physical hosts are selected as relay servers for each certain number of virtual machines and constructs a tree structure topology for data collection. The status information is aggregated on each relay server, then it is forwarded to the next above in the tree structure, and finally to the JOSE Manager which is the tree structure's root. This function reduces the number of connections that JOSE Manager must accept.

3.2.2 Virtual network administration

JOSE Manager has the following network functions, to configure a network that handles tens of thousands of virtual machines spread across multiple data centers.

1) Multi-domain SDN control

JOSE has three SDN domains, so from JOSE Manager, each SDN controller's API is invoked in parallel. In JOSE, for static classes and dynamic classes, an interface called a Virtual Tenant Network (VTN)[4] is used to configure each user network's virtual network. The VTN API can be used to configure a virtual network, using virtual routers, virtual bridges, etc. If a request from a user for deployment locations of computational resources and storage resources is spread across multiple SDN domains in JOSE, then the VTN configuration command is sent to multiple SDN



(A) User network over multiple SDN domains (B) Splitting virtual layer 2 network

Fig. 4 Example of user network configuration

controllers simultaneously.

2) Virtual network division

Figure 4 shows examples of data networks that connect between resources. Each data network contains virtual bridges (vBridges) that interconnect resources as the virtual layer 2 network, and virtual routers (vRouters) that interconnect resources as the virtual layer 3 network. In a user network that consists of multiple SDNs, the physical layer 3 switch on the upper link does routing between VTNs (Fig. 4 (A)).

Usually, the maximum number of end hosts that can be accommodated on one L2 network is around several hundred. Therefore, if the number of virtual machines that should be accommodated on one user network exceeds a certain number, then JOSE Manager splits the virtual layer 2 network into multiple networks, and layer 3 routing between these virtual layer 2 networks is configured (Fig. 4 (B)). IP address settings for the virtual machines and vRouters are automatically assigned by JOSE Manager.

The network for external connections (L3), user VLAN network for external connections (L2), and sensing foundation network (L3) correspond to one VTN. In a virtual machine, a network interface is allocated for each connected network. In a layer 3 network, IP addresses are automatically assigned to the network interfaces.

3.3 JOSE Sensing Foundation (JSF)

In JOSE, as already described, the following two types of sensor network exist.

- Sensor networks on which IoT service is newly deployed
- Sensor networks which are already deployed and sensor data is shared among multiple services

For sensor data access in these types of sensor network, we developed a new sensor data management mechanism: JOSE Sensing Foundation (JSF). JSF is deployed along with storage resources in each IoT service, and becomes a functional element for access to the sensor network from computational resources.

In the sensor network gateways and in the JSF, IEEE1888[5] was adopted as the common access protocol between storages in JOSE. In IEEE1888, the names of observation data can be freely decided, but in JOSE, unique IEEE1888 rules (syntax) are defined, to enable data sharing between JOSE users. Even if a sensor network is built by a user, it can be connected as a sensor network of JOSE as long as it complies with JOSE's IEEE1888.

Computational resources of IoT services use IEEE1888 and access JSF, so they can obtain observation data in the past and the latest observation data. JSF has an API that can obtain event-driven data in real time by using a Web Socket when observation data is sent from the sensor network. Moreover, among sensors in a sensor network of another IoT service that is already deployed, one can search for and get sensor data from sensors to which access rights are granted. The JSFs between each IoT service are connected by sensing foundation segments and form a wide area network for cooperation (Fig. 5).

A sensor network is owned by a user, and is operated within the scope of the user's responsibility. To achieve stable JSF operations between inter-connected sensor networks, cooperation between owners is necessary. However, we must consider cases where a sensor network with insufficient administration exists. Therefore, JOSE provides a function of JSF that interconnects between different sensor networks autonomously by a Peer-to-Peer (P2P) network; when a new JSF is added or an existing JSF is deleted, instead of the user doing registration tasks etc., the JSFs automatically connect to each other and keep the search operation between JSFs ready. This JSF function was implemented using PIAX[6] middleware. In the JOSE's IEEE1888 rules, a sensor's location information is a mandatory field,

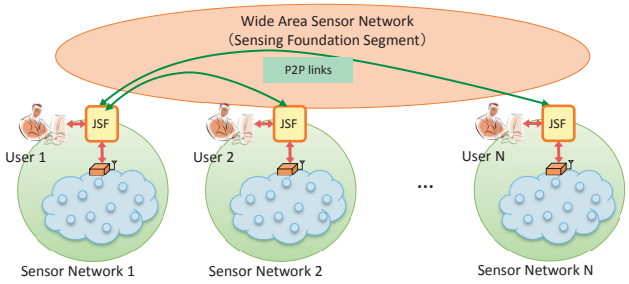


Fig. 5 P2P connections between JSFs

Request URL:

http://host:port/sensors/Location in rect(130,30,10,10)/values/Temperature

Response:

```
[{"id":"http://jose.jp/1888/00/0x0001",
"values":[{"id":"http://jose.jp/1888/00/0x0001/Temperature",
"value":[{"value":27.5,"time":1401334500000}],
"profile":"weather"}]},
{"id":"http://jose.jp/1888/00/0x0002",
"values":[{"id":"http://jose.jp/1888/00/0x0002/Temperature",
"value":[{"value":29.5,"time":1401334500000}],
"profile":"weather"}]},
{"id":"http://jose.jp/1888/00/0x0003",
"values":[{"id":"http://jose.jp/1888/00/0x0003/Temperature",
"value":[{"value":29.0,"time":1401334500000}],
"profile":"weather"}]}]
```

Fig. 6 Example of sensor search in JSF

and sensors can be searched by specifying location coordinates as a key. From an application, if one specifies the area of the observation data needed and does search and access in JSF, then the latest observation data can be accessed without bothering other users who administrate sensor networks that observe the corresponding area.

Figure 6 is an example of a sensor search in JSF. In this example, temperature observation data is obtained from sensors located in the rectangular area within 10 degrees of 130 degrees east longitude, and within 10 degrees of 30 degrees north latitude. Search results are provided in JSON or XML format (this figure is an example of JSON format).

4 JOSE testbed and its operation

Currently, as a testbed, JOSE accommodates field-trial experiments that relate to 27 IoT experimental services. In this section, JOSE's operating situation as a testbed is described.

4.1 Sensor devices example

We developed several types of JOSE sensor equipment in compliance with IEEE1888, and lent them to testbed users.

Figure 7 (A) shows a smartphone sensor we developed. Its form is like a normal smartphone, but in addition to sensing functions for location and acceleration, which are equipped on normal smartphones, it contains sensing functions that observe geomagnetism, illumination, temperature, humidity and barometric pressure. It also incorporates a software function that transmits observed data to JSF by IEEE1888, via wireless LAN, 3G or LTE.

Also, Figure 7 (B) shows a wireless environment sensor that runs autonomously with a self-charged battery by solar panels and a gateway device for the wireless environment sensor. The environment sensor can have equipped a solar radiation meter, infrared radiometer, sound level meter, and sensors that observe vibration, snow, CO_2, wind speed and direction, temperature, humidity, barometric pressure, precipitation intensity, etc. Users can customize the equipped sensing function on the sensor. Each wireless environment sensor is wirelessly connected to a gateway device (Fig. 7 (B) right). If multiple wireless environment sensors are installed, the wireless environment sensors form an ad-hoc network between themselves, and they send observation data by multi-hop communications. The gateway device transmits observation data to JSF by IEEE1888, using communications via wired or wireless LAN, 3G or LTE.

(A) Smartphone sensor

(B) Wireless environment sensor and gateway (right)

Fig. 7 Sensor device examples

4.2 Sensor network example

In JOSE, some sensor networks are already installed and operating in several places. Below are some examples of running sensor networks. Several field trials utilize these sensor networks.

Koganei City: Environmental sensor network (Environmental sensors)

Fukushima City: Building and bridge sensor network (Vibration sensors)

Chikuma City: River sensor network (Water level and rain volume sensors)

Hiroshima City: Environmental sensor network (Environmental sensors)

Tokyo Prefecture: Bridge sensor network (Vibration sensors)

JOSE users can access sensor networks via JSF (each sensor network has its own access right settings).

4.3 JOSE's performance

4.3.1 Operation time

Figure 8 shows the operation time in JOSE (from setup time to provision time) for an IoT service configured with 200 virtual machines deployed in different locations. In the figure, "No Manager" is the operation time when the user network was built without using JOSE Manager. "JOSE Manager" is the operation time when using JOSE Manager. When the user network is built without using JOSE Manager, for each of the 200 virtual machines, the following operations must be done by an operator: account setup (user setup), launch setup of the virtual machine itself (VM setup), storage setup, and address settings for the network

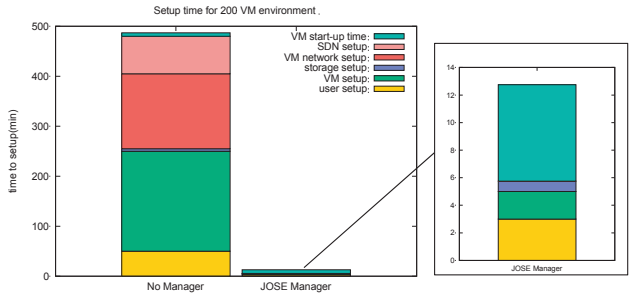


Fig. 8 Setup time for user network that has 200 virtual machines

interface (VM network setup). Also, commands for SDN need to be entered (SDN setup: In this case, the GUI interface of an SDN product is used). Figure 8 also includes the time taken when VM images of virtual machines are transmitted to all the physical hosts, and the OS boot time (VM start-up time). Some of the setup process was automated by a script but 487 minutes were needed until setup was complete for 200 virtual machines.

On the other hand, if JOSE Manager is used, SDN command inputs and address setup for virtual machines are done automatically, so they take almost 0 working time. Also, account setup and virtual machine setup are done in batches in parallel, so setup can be completed in a very short time. For VM start-up time, the wait time is similar to when JOSE Manager is not used, and this takes over half of the total operation time. In total, the setup took about 13 minutes until all the virtual machines were in running status, so the work time was reduced by about 97.4%, from 487 minutes when JOSE Manager was not used.

4.3.2 JSF search response time

Figure 9 shows the measurement results of the search completion time in JSF. The horizontal axis shows the number of sensors and sensor networks that have location information. We measured the average delay in a situation with up to 1,000 devices and gateways (nodes) that have location information. These sensors were evenly distributed between three SDN domains in JOSE. In the evaluation, one search condition is used and the number of matching sensors was 10 or 100. This shows the average response time of ten trials.

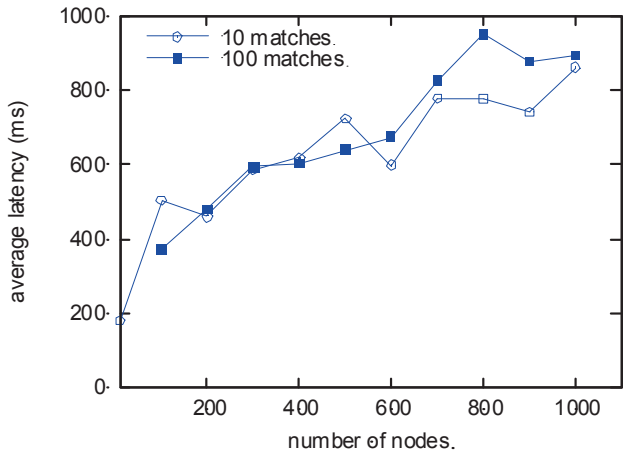


Fig. 9 Search response time in JSF

From the figure, first, we can confirm scalability for the number of search matches. Comparing 10 matches vs. 100 matches, there is almost no difference in time until search completion, as both searches were done within 1 second for 1,000 nodes.

We also confirmed scalability when the number of nodes increased. Even if the number of nodes is increased by 10 times, we see that the delay time only increased by about 2 times. JOSE uses the PIAX overlay network, which is implemented based on Skip Graph[6]. With Skip Graph, for N number of nodes, it is known that an average response time O(logN) can be achieved, and this verification also confirmed these properties.

5 Conclusion

This paper described the JOSE IoT services platform we developed, and its operating situation as a testbed.

Many base technologies of IoT services are still ongoing and not in a practical phase. Also in Japan, various R&D is being done in each field, but to commercialize the technologies, one needs field trials to verify usefulness using real sensors, computers, storage and networks. A large-scale testbed environment that can support the design and evaluation of such applications has not been provided so far, and JOSE is the first testbed for such IoT services technologies in Japan or overseas. Field trials and verifications have been running in JOSE by various R&D projects, and observed data has been shared and utilized. Through such activities, we aim at more efficient R&D on IoT and big data, and to eventually stimulate this research field in Japan and overseas.

References

1 "Large-scale open test-bed JOSE," http://www.nict.go.jp/en/nrh/nwgn/jose.html, June 2015.
2 "Network Testbed JGN-X," http://jgn.nict.go.jp/, June 2015.
3 Y. Kanaumi, S. Saito, E. Kawai, S. Ishii, K. Kobayashi, and S. Shimojo, "RISE: A wide-area hybrid OpenFlow network testbed," IEICE transactions on communications, Vol.96, No.1, pp.108–118, Jan. 2013.
4 "OpenDaylight Virtual Tenant Network (VTN)," https://wiki.opendaylight.org/

view/OpenDaylight_Virtual_Tenant_Network (VTN): Main, June 2015.
5 IEEE1888-2011: standard for ubiquitous green community control network, 2011.
6 Yuuichi Teranishi, "PIAX: Toward a Framework for Sensor Overlay Network," In Proc. of CCNC 2009, pp.1–5, Jan. 2009.
7 J. Aspnes and G. Shah, "Skip Graphs," ACM Transactions on Algorithms, Vol.3, No.4, Article No.37, Nov. 2007.

Yuuichi TERANISHI, Ph.D.

Research Manager, New Generation Network Laboratory, Network Research Headquarters
Ubiquitous Computing, Overlay Network, Multimedia, Database, Mobile

Yuki SAITO

Technical Expert, New Generation Network Laboratory, Network Research Headquarters
Network

Sakae MURONO

Research Expert, New Generation Network Laboratory, Network Research Headquarters
Network

Nozomu NISHINAGA, Ph.D.

Director of New Generation Network Laboratory, Network Research Headquarters
New-Generation Network