International Journal of Grid and Distributed Computing
Vol. 9, No. 5 (2016), pp. 157-174
http://dx.doi.org/10.14257/ijgdc.2016.9.5.15

The "Container-Cloud" Architecture and Scene Perception in IoE Era

Bo Yu[1, 2], Zhongliang Guan[1], Yan Jiang[3, 4], Cuiyu Qi[1], Shifeng Liu[1]

[1]School of Economics and Management, Beijing Jiaotong University, Beijing, China
[2]Mash5 Technologies (Suzhou), Ltd. , Suzhou, China
[3]College of Computer Science and Technology, Changchun Normal University, Changchun, China
[4]College of Computer Science and Technology, Jilin University, Changchun, China
Corresponding author: swallowjiang@21cn.com, 14113121@bjtu.edu.cn

Abstract

This paper introduces a new innovative mobile application architecture called "Container-Cloud", which could be probably an inevitable trend in complex cloud, connected devices and network ecosystem in IoE(Internet of Everything) era. The significance of C-C is comparably to Java for web, and android for mobile.

There are some innovations in platform operations, in service operations, in developing and in use. The "Container-Cloud" architecture can solve those application silo, enable applications developed, debugged and deployed transparently in cloud, allows mobile applications to be generated on demand, and be migrated and scheduled automatically among cloud, and creates an opportunity for implementation of "context-aware" application in cloud. The paper also gives some "context-aware" application examples based on C-C support in IoE era.

Keywords: *Container-Cloud; Cloud compiler; Software-Defined Application; Application Container; IoE (Internet of Everything); Context-aware*

1. Introduction

This paper puts forward for the first time, an innovative "Container-Cloud" application architecture for building enterprise mobile applications eco-system led by the author, and an implementation of scene perception in IoE (Internet of Everything) [1] [2] era based on the research and development works.

The basic principle of Container-cloud is to install only one application container as an application entry which is an application virtual machine on mobile devices. This app executes the applications as native apps to implement the business functions defined in the application functional units. All these applications are programmed in JavaScript and dynamically loaded to mobile devices from the cloud. In other words, the boundary between the mobile application container and cloud platform is blurred in the "container" architecture. The mobile terminal becomes a part of the cloud, and becomes the smart personal mobile cloud.

The applications are controlled by attributive values in different scenes and finally pushed to the Container-App at the mobile devices, they are also deployed, operated, and managed in the PaaS(Platform as a Service) cloud [3].The key content of its architecture schematic is shown in Figure 1.

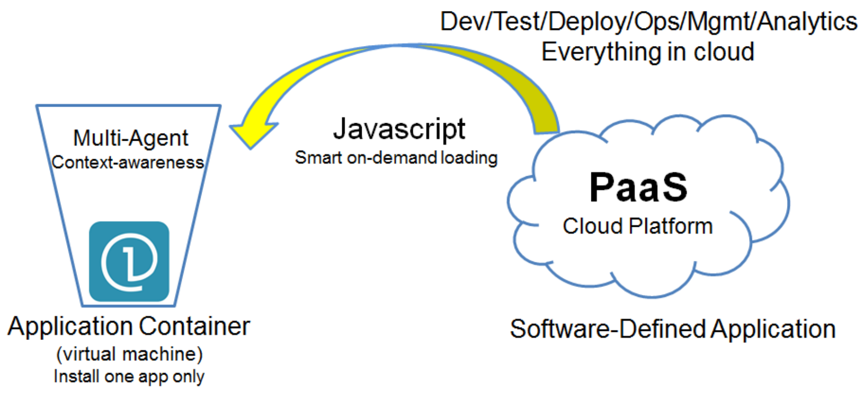


Figure 1. Application Logic Figure

The backbone of implementation uses cloud compiler to reason, fuse and generate mobile applications event-driven and on-demand. By then, cloud side provides all sorts of application layer resource management, provisioning, scheduling, collaboration, and the interface to external systems.

From the compiler's point of view, software can be defined flexibly according to the scenes, events, and application's business logics. And then, cloud compiler, a guarantee mechanism running in background, and is the basis for the application generated automatically, which focuses on how to create in the cloud and run applications. From this perspective, service-oriented design makes even the entire Internet a big compiler.

Meanwhile the "C-C" architecture can provide smart services through the cloud in IoE environment. Mobile application can be deduced from the nature in the different circumstances of the diverse solutions, fused and generated by SDA on demand, and then pushed to the mobile terminal device to meet specific business needs.

Figure 2. Mobile Application in IoE era based on Container-Cloud

The "Containers-Cloud" architecture can be widely used in business, government and other fields of information technology. It provides good protection for building situational, enterprise-class, on-demand intelligent mobile applications, and especially for developing mobile intelligent applications, deploying, managing, and building a business eco-system.

2. Literature Review

From the latest four years of strategic technology trends analysis reports by Gartner, the trend of mobile applications including cloud Personal, app Mobile, anything Software-defined, everywhere Computing and computing Cloud/client and other aspects of [2].

Table 1. Strategic Technology Trends

2012	2013	2014	2015
Big Data	Strategic big data	Smart machines	Smart machines
Extreme low-energy servers	Intergrated ecosystems	Web-scale IT	Web-scale IT
Next generation analytics	Actionable analytics	3D printing	3D printing
App stores and markeplaces	Enterprise app stores	Software-defined anything	Software-defined applications/infrastructure
IoT	IoT	IoT	IoT
In-memory computing	In-memory computing	Cloud/client architecture	Cloud/client compting
Mobile-centric applications/interface	Mobile applications/HTML5	Mobile apps and applications	Risk-based security/self protection
Cloud computing	Hybrid IT/cloud computing	Hybrid cloud & IT as a service broker	Advanced pervasive/invisible analytics
Media tablets and beyond	Mobile device batties	Mobile device diversity/management	Computing everywhere
Contextual/social user experience	Personal cloud	Era of the personal cloud	Context-rich systems

2.1. Mobile Container

There are HTML5, fog computing and virtual machine forms in the mobile application container.

The rise of open source "Docker" containers over the past year has created a de-facto standard for how applications can extend from one platform to another running as micro-services in Linux server and Open Shift PaaS environments such as Cloud Foundry.

Nearly every major enterprise infrastructure software and cloud provider has jumped on the "Docker" bandwagon including Google Inc., IBM Corp., Red Hat Inc., Microsoft, Rackspace and VMware Inc., among others [4].

Google announces cloud container Engine using Kubernetes, Google Container Engine (GKE) automates both the process of provisioning, running and stopping VMs and the process of deploying a containerized application on a variable number of Docker containers based on demand [5].

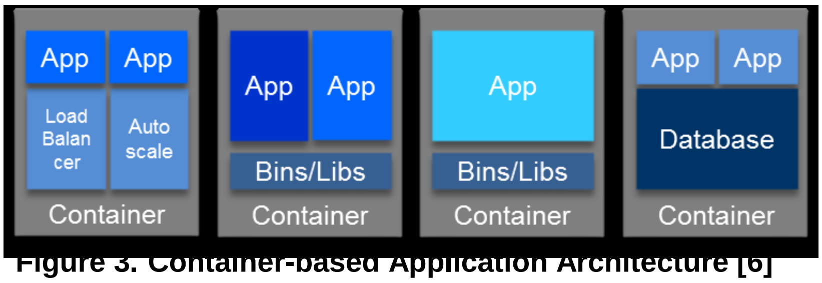


Figure 3. Container-based Application Architecture [6]

But most of the containers which reside in the cloud server only, in the operating system level virtualization, don't forward to the client sides [7].

2.2. Software-Defined Application

Google launched in April 2008 Engine App by virtue of GFS, BigTable and MapReduce technology, and then form its own PaaS, that is dependent on the Google application of multi-tenant PaaS system. Some pioneer systems fall into this category such as Sun's Zembly [8] in 2007's, and Salesforce's Force.com in 2009[9].

The container in this context is in fact a virtualization scheme for applications. Now the question is how to "pack" the applications at the cloud for the container? How to develop the applications ready to pack? How to easily manage and operate container-capable applications? We have to answer these questions from cloud perspective.

Software can define anything in compiler thinking. When Sun coined the term "cloud compiler" in industry, the idea was to make cloud work as a giant compiler, to build and run applications on cloud without glitch. This concept, the first stage is the SOA, the cornerstone of its implementation, is supported by a dynamic evolution, long history of the programming system, can be referred to as mobile cloud compiler [10] [11].

They view SDN in networking space shares the similar idea, considering it's a compiler effort when a networking solution is treated as an application.

2.3. Mobile Cloud

Mobile cloud is a collection of Internet based data, applications and related services through smart phones, laptops, tablet PCs, and other portable devices access, application and related services [12].

At present, most mobile applications (especially HTML5 based mobile applications) are cut from the ability, rather than the platform or framework, which only solve the point of a single problem, but not from a platform or ecological perspective to provide sufficient capacity. The root of these challenges is to delve into, the application mode of mobile enterprise is not appropriate, must subvert!

When Fitbit and other smart objects bloom, mobile phone becomes the second place as an operation interface, but it always to be a management platform, responsible for the complete configuration, monitoring, collecting data, collaboration, cloud synchronization function. It becomes a mobile platform for intelligent objects, and app features become very complex, and take into account the scene, in order to facilitate the management of different intelligent objects, data acquisition and display [13].

From the literature above, we can find that many apps installed on the mobile terminal, seemingly powerful, conveniently, but they affect the user experience. Especially for enterprise applications, the application of each other isolated, the information needed for the application of the information cannot be shared.

So we should use an new programming paradigm, "Containers-Cloud" mixes the mobile container, cloud and Software –Defined Application, it can change and find the necessary content easily from the mass of information tools on demand.

Most important, the philosophy part as we experienced is in procedural, object-oriented, declarative, functional programming, *etc*. For cloud, the paradigm is service-oriented programming because cloud transforms everything in IT as a service. When we use software to define applications, applications and its building blocks must be presented as services at cloud.

3. Technical Conception

In accordance with the current mobile application mode, ten thousands different connected devices will need ten thousands different apps, which will seriously affect the development, operation, maintenance, and user experience of mobile applications. In addition, the issue of information silo can emerge since the apps are isolated from each other. Furthermore, there are also challenges in application deployments, updates, offline operations, security, and so on. All of these serious challenges will become more serious in the IoE era and should be addressed.

The studies on C-C (we use C-C to refer the Container-Cloud in this paper) in the enterprise and customer are very limited. It takes consideration of a variety of variables, which will be discussed below. They can be divided into IoE, application container, Software-Defined Application, cloud complier.

The C-C is an application virtualization system architecture, which is composed of terminal containers and cloud complier. Cloud complier means the cloud-side "Software-Defined Application" supporting system, and terminal containers are smart mobile application container.

The C-C architecture ensures application to be developed, debugged, deployed transparently, while allow applications to be generated on demand, and automatically migrated and scheduled among clouds.

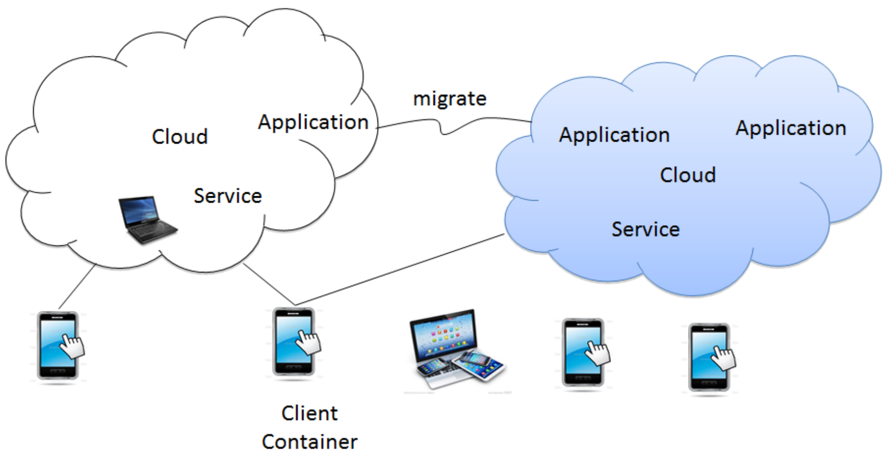


Figure 4. Application Migrate under C-C Architecture

3.1. IoE and IoE Paradox

The IoE can be simply understood as the integration of mobile Internet and IoT (Internet of things). IoE aggregates people, events, places and things and make equipment, network and information be connected more closely and more meaningfully [1]. It can transforms information into actions, and then create newer and richer experiences.

When IoE explodes, everything needs an app! How can we fit millions of applications for everything into our smart phones? How can users easily use these millions of applications? How can businesses manage and operate so many applications cost-effectively? We realize that today's app is not smart at all, and the existing app model no longer works well for IoE.

IoE technology provides a more effective platform for application software, logistics, services and operations. But one of the serious challenges in IoE era is that users have to face too many apps to choose.

Facing so many challenges, especially the issues of too many apps to promote and use, we need to create intelligent technologies to lead people to a more locomotive and integrated world. But how can we do that?

3.2. Application Container

Sun micro system promotes Java is the first experiment of container from 1990s, and JVM's "write once, run anywhere" is the slogan for container. The container in this context is in fact a virtualization scheme for applications resides in server side. In this sense, Java virtual machine is a container for byte code for Java. It is a compiler job to create and to pack byte code from Java source code, and Java VM is part of Java compiler [14].

And there are many containers or virtual machine like VMware, Hyper-v, Xen, They now can be used as an alternative to OS-level virtualization to run multiple isolated systems on a single host. Containers within a single operating system are much more efficient, and because of this efficiency, they will underpin the future of the cloud infrastructure industry in place of VM architecture.

Compare with heavyweight virtualization, from an infrastructure manager's point of view, Docker is lightweight virtualization. But all of them are virtualization machine, support applications in IaaS level.

In the old web era, the container actually was the browser. It has been further extended in the IoE era. Conceptually, these mobile application containers can be created by different virtual machines, and they can also build new development tool framework and infrastructures in new ways.

In a hybrid environment of cloud and intelligent terminal, the biggest change of this "C-C" architecture is to distribute native codes to generate, load and extend dynamic applications by intelligent agent and local cache. In the meantime the components and applications also can be converted into data, which can migrate through cloud, and then be restored in the new environment. Is it magic?

Further, following the same standard, these mobile containers can be combined or reduced to a single container-that is a unified mobile application entry similar to the web portal, a unified internal application. However, different from a web portal, the users can get their own applications dynamically through situational push by intelligence agents and cloud compiler support, they work in aPaaS level.

3.3. Software-Defined Application

Software-Defined Application (SDA) is the higher evolution form of PaaS, which automatically configures to define freely the applications in the cloud, so those applications have app-creation ability.

So cloud services are software-configurable through API calls, and the applications have rich APIs access their function and content programmatically too. To deal with the rapidly changing demands of digital business and scale systems up - or down - rapidly, the SDA (Software Defined Applications) computing has to be transformed from static model to dynamic model. This way, rules, models and codes of application can be dynamically assembled and configured.

To realize SDA, the application generation system needs to follow the principle of unified refactoring. The "C-C" is just a very novel, practical architecture to push the system.

3.4. Cloud Complier

This invention of cloud-side concentrates on the issues of cloud complier. It focuses on the application life cycle of complex system, and studies how to make it transparent, how to generate and schedule application on demand, and how to migrate them between cloud and container automatically. In this invention, the proposed cloud compiler contains all sorts of application layer resource management, provision, scheduling, collaboration, and the interface to external systems.

Thus the logic of Cloud-side can be much more complex than that of client-side. It can accomplish more various and more comprehensive operations. Service can aggregate other contents including the social contribution of the application, the inter-cloud collaboration across the cloud border, service aggregation technology, *etc*. Service can be declared, registered and managed within the scope of cloud, associated and scheduled through the intelligent agent strategy.

By cloud compiler, we can program in the browser to realize the runtime and the cloud storage, to provide a programming model, application development, and to template component, multi-tenants management, and other functions. We can not only retain the flexibility associated with the definition, but also reduce the cost of application call through predefined configuration. Moreover, each cloud may undertake different various kinds of industry or business, thus, we can integrate or generate the service across the cloud or through dynamic combination. Users can assemble the cloud applications on their own requirements in such virtual software factory.

4. Implementation Framework

In this paper, we present "Container-Cloud" architecture for mobile applications to support most IoE smart devices. "C-C" is comparable to the inventions of Java for web [10], and Android for mobile, and it is a virtual system of application as Figure 4, with the aid of cloud complier and mobile application container, it completes applications develop, test, deploy, migrate and other service processes.

Cloud compiler provides the ultimate goal of cloud computing because it maximizes the freedom of software and further disrupts the existing model of software development and deployment. In cloud complier, software developers develop application modules, and the complier provides various module components. Users can leverage these module resources and construct flexible and individualized cloud applications, even can produce one-time-use applications.

4.1. Framework Benefits

The basic idea is straightforward: mobile users install a container app on either iOS or Android device. It is empty and contains no application initially. The container app automatically loads the applications real-time on demand, and then executes at the virtual machine inside the container. In this view, we say this new mobile application model completely subverts the existing mobile mode.

The container concept, such as "docker" is a hot topic today, it is gaining attention at Silicon Valley in cloud computing community. However, the type container resides within cloud environment or at server side [15], while our container resides at client device side, serving different purpose. The container-cloud solution delivers the following benefits:

- Reduce the cost of application development and operation, and speed up time-to-market
- Auto updates and application can change dynamically
- Cross-platform support similar to Java VM’s "write once, run anywhere"
- Avoiding app fragmentation by consolidating many apps into a single app
- Avoiding app segregation as applications can communicate to each other
- And more derivative use cases from this new paradigm

4.2. Application Container

The basic app of container is an empty native app installed on mobile device, which loads application codes from the NoSQL database and provides dynamic loading, upgrading, maintenance and other management functions. Communication between the containers and cloud is guaranteed by container agents and cloud complier.

The container provides following three function categories:

4.2.1. Mobile SDK: SDK is the abstract of existing applications, which can be encapsulated or mapped in the system with restful APIs.

The SDK library was encapsulated in a container which can provide local services, hardware interfaces, and other third-party native API. SDK load code automatically from the background in the cloud based on the update frequency or the appropriate scheduling mechanism set in advance. At the same time, SDK's head needs to mash-up services according to the rules from the cloud agent engine.

4.2.2. Intelligent Agent: An independent app is only a shell app whose functions like the computer BIOS startup. All local code libraries are packed into this app package, including the intelligent agent.

It supports users to assemble their personalized applications according to their situational needs. The applications could be even one-time use ones. For users to deal with complex business logic, it provides the tools like service mash-up, event-driven and complex business rule engines, and so on.

4.2.3. Local Data Cache: Local data cache mechanisms support offline caching services which is easy to load the locally stored codes execution, or uninstall the codes to release space. And the local timing mechanism guarantees distribution and monitoring of applications component services.

4.3. Cloud complier

Compiler – the core implementation of the programming scheme. At front-end, the compiler processes source code at lexicon and syntax level, and limited static semantics. At backend, the compiler implements all semantic actions of the programming construct. In cloud compiler, the services of application components are assembled according to the abstract model and semantic construction. For example, conventional object-oriented compiler needs to implement class inheritance, while service-oriented cloud compiler needs to implement widget view as a service when it is reused, adapted, mashed up in model-view-controller design pattern.

4.3.1. Program Model: The philosophy part as we experienced in procedural, object-oriented, declarative, functional programming *etc*. For cloud, the paradigm is service-oriented programming because cloud transforms everything in IT as a service. When we use software to define applications, applications and its building blocks must be presented as services at cloud. This is new to any previous programming paradigm.

For it uses MVC design pattern to reuse, adapt and mash-up components or services, cloud compiler exposes widgets/views as service for external call, and its runtime exposed as services in API form.

4.3.2 Cloud Store: Cloud load data, dynamic and program fragments from the unstructured database on cloud platform, store system uses object-orient libraries instead of file system, all of them are considered data management uniformly. With C-C and scheduling management Intelligent Agent, the content is loaded from the cloud to the native application container. The size of data, program fragments or service components can be split very small so they can be compiled fast enough to generate applications dynamically.

4.3.3. Runtime: On cloud, the runtime is the collection of all possible cloud services. Moreover, it’s easy to accumulate through "social programming" (another term coined at Sun) contributed by open source developers, and also through adapted APIs to various third-party cloud services such as Google Map, Facebook, Flickr, and Zillow, *etc*.
For cloud compiler, the environment for life-cycle support extends to distribution (app store), operation (DevOps), management (*e.g.* entitlement and security), maintenance (build-fix), and analysis (*e.g.* Google Analytics) on cloud. Note that "operating system

camp" of cloud implementation has no knowledge on application internal, and therefore is not suitable to application operation at fine-grain level. The cloud compiler must support application level of cloud operation through the whole life cycle.

4.3.4. Inter-Cloud [16]: In order to interconnect with other clouds, to call some service from other cloud and mush-up them to a new service, a cloud should be extended by a cloud stub. A cloud stub contains transport layer, data layer, semantics layer, behavior layer total four abstracted layers.

Aside from the four layers in the cloud stub, a global resource registry is needed to maintain cloud node information in the inter-cloud environment. The registry is a directory service that lists the capability data of each layer for a specific cloud. Virtualization will rely on the look up operation on the directory. Inter-cloud registry can be physically centralized service, or logical centralized but actually

4.4. Scenario Application

Strictly speaking, scenario application isn't the kernel of cloud compiler, but the further expansion based on intelligent Agent and situational matrix.

Meanwhile, Cloud compiler binds scenario matrix with smart agent, it can load, unload or destroy scenario applications dynamically generated from the cloud and pushed to the container.

And cloud complier constructs the distributed intelligence system by multi-agent with socialization, real-time event stream processing, event associated engine model, to derive the personalized scenarios and manage the scenario-based flow strategies and the complex events.

We can define context info and business services separately, and combine them with the event life cycle to definite scenarios matrix, thus can evolve various applications. All of those contexts are driven by the motive events from the angle of time, people, places, objects, and so on. Of course, there are other scenarios mechanisms of discovery which can be designed and used.

4.5. Relationship

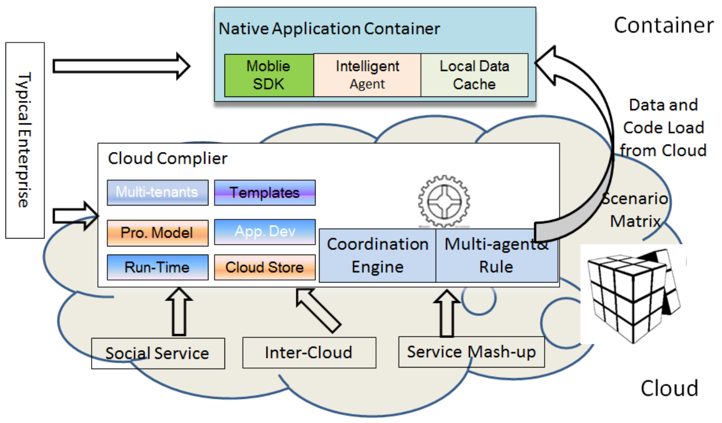


Figure 5. Container-Cloud Framework

As mentioned at the beginning, the top-level system is composed of the container and the cloud. The container is installed at client mobile devices as a native app. The virtual machine (not shown) inside the container is implemented with a truly native tools to device via a "language bridge", and easy to use hardware resource.

The local stored code can be fetched to execute, or unload to release memory. The container needs to connect to cloud such that it has a networking support layer with certain security mechanism. The SDK provides a large JavaScript library to program on this container, including hardware interface, and some 3rd party native APIs like payment interface. On top of the SDK, there is an intelligent agent engine to process rules loaded from cloud. The topmost part is application modules, also loaded from cloud, serving as the application building blocks. Examples including user profile, avatar management, and business rule engine, *etc*.

The C-C concentrates main effort on the cloud portion of the platform. The core runtime of the it provides all foundation support for PaaS and deals with Tomcat and MongoDB like service initialization by configuration and DB connection maintenance. The cloud store is implemented with MongoDB for its advantage of meta-data friendly and supporting for unstructured data, also its JavaScript native and easy to build API services.

In cloud, C-C treats JavaScript code and user data equally, the data cache is in the memory-cache fashion and mixed implemented with database and memory. As discussed in earlier section, the main objective is to implement the application components as a service, and to assemble them into applications under many constraints like role-based access control, agent rules, perceptions bundle for dynamic assembly. Along with the compiler there is a runtime library exposed as APIs to operate various services.

5. Findings

There are many shortcoming of mobile cloud architecture by now, for example:

- Entrance through portal. When there are too many fragmented apps, users will be lost in finding the entry for a particular app when needed, and they are available on-demand.
- Operation for variety and change. The vast amount on-demand applications dramatically increase the complexity of management and operation, involving application life-cycle for development, testing, deployment, update, operation, analytics, security, *etc*. Application change is a norm.
- Integration for connectivity and continuity. Those applications need to connect various IoT hardware devices, need to integrate with legacy enterprise systems, and need to interoperate with existing applications.

The writer's team researches and promotes the "C-C" architecture and cloud programming paradigm in early study. The study considers that building eco-system problems of enterprise mobile applications in cloud from the view of virtual machine container.

5.1. The New Understanding of Cloud Computing

In general, the design criteria of cloud computing is to think from the perspective of how to use the resources effectively. The computing resources, the application resources, and so on were unified as a fixed element to allocate and scheduled. Thus, we can call them "operating system camp" of cloud computing, from which the IaaS, PaaS and SaaS are derived.

Another angle is the compiler concept, which we call them "compiler camp" of cloud computing. At this point, the elements in the cloud such as computing resources, especially the application of resources are no longer fixed. The application resources in the cloud will be reassembled with the change of time, location, users or processes. The change gives the resources a certain degree of operational capacity, makes the resources continue to change and unfix the boundaries of applications. The cloud of cloud compiler concept supports application-level cloud operations in the whole life-cycle. In the contrast, the cloud implementation of "operating system camp" doesn't consider the fusion among the applications, and therefore isn't suitable for operation in the fine-grained level.

The cloud compiler can combine module components intelligently and assemble these resources to form a new unique application based on the smart rules, even the one-time application. Among them, these resources or services can be registered, split, mashed-up from the inside of a system or a cloud area, or even across one or more cloud service systems.

5.2. The New Understanding of Container

In fact, JVM has been explored to solve the applications problem by virtual machine since 20 years ago. The goal of JVM was designed to provide a computer model with an abstract specification which could not only solve the problem of cross-platform compatibility, but also provide a lot of flexibility for developers. In this case, JVM was designed from the perspective to across hardware platforms. It didn't care the Java source files on the top.

In this paper, cloud side virtual machine is cloud complier, and the client side virtual machine is container.

Unlike docker container only resides in the cloud service side only, the C-C model also uses client container. App container is a native app on mobile device, it provides dynamic loading, upgrade, maintenance and other management functions.

The cloud compiler on the cloud side and the virtualization container on mobile end together constitute a complete cloud compiler system in the form of the C-C model.

5.3. Scene Perception in IoE

Because IoE aggregates people, events, places and things, connects the equipment, network and more relevant information, and turns information into action, which creates new capabilities, richer experience. Gartner claims that 2014 was the first year of IoE, and 2020 there will be 37 billion interconnected smart devices globally.

This requires entry application must be intelligent, even on demand, with the change of scene [17] [18]. In particular enterprise-class mobile applications can be generated changing with the scenario on demand through the cloud compiler, so the applications form changed to a whole new form.

We can combine the context-aware [19] [20] with the mobile application container, which is an effective solution in IoE era for enterprise to response agilely and to build an information eco-system.

To handle complex businesses, the C-C architecture combines the Programming for Service Mash-up, event-driven and complex business rules engine, to achieve a smart innovative software workshop with user's participation.

To help developers build an application on cloud, C-C supplies a component library and application templates to bootstrap the usage. Developers can contribute open source JavaScript components as well as the finished applications for social programming. We are eager to establish an ecosystem by following Github model. Its advantage over Github is that users can run the application right away on container without download and compile on local IDE, and later install on device. Also they can mash up components from cloud to build a new application in seconds or minutes, and run immediately.

The user management gives the cloud operator full control on users. A user can access multiple tenants (*e.g.* use case of one student belonging to several clubs). Finally, the application development tool is the browser-based IDE such that the developer can write JavaScript code in a browser window without installing any IDE on a local computer. All she needs is a browser. The developer uses the tool to manage the whole development process on web or even on mobile.

6. Scene Perception Implementation

In related researches, the authors found that the "C-C" programming paradigm is also suitable for generating mobile applications and constructing ecosystem in IoE environment.

In accordance with the current mobile application develop mode, ten thousand different connected devices need ten thousand different apps, this seemingly be powerful and convenient, but mobile terminal need to install a lot of apps, each place need an app, each object need an app, each service requires an app, every time activities need an App. So many long tail low frequency applications but too many entrances make users edgy instead. For enterprise applications, they isolated from each other, the information and services can’t be shared and interactive which seriously affect the development, operation, maintenance of the effectiveness and accuracy.

Figure 6. Outrageous Number of Apps

With the emergence of smart devices such as fitbit, smart phones tend to become the second interface in mobile world, but it will always likely play the role of a management device that are responsible for configuration, monitoring, collecting data, collaboration, cloud synchronization, and other functions of all kinds of smart devices. It becomes a moving background service platform of smart devices, so the function of the apps on smart phones cloud becomes rather complicated.

It is an effective means of technology that C-C architecture combine with context-aware, which can realize agile and build enterprise information eco-system in the IoE era.

Based on multi-agent strategy [21], the situational application can push different application functionalities to user silently, according to the different situations, which will increases the application depth. Programmed and compiled once, mobile applications can deploy on a variety of mobile operating system platform, guarantees the cross-platform.

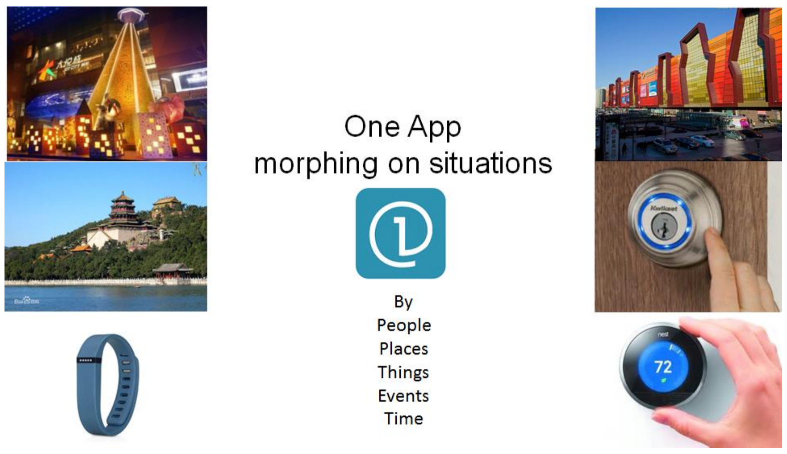


Figure 7. One App for Situations

And how can it implement?

Firstly, the basic situational app is a container app, actual application codes store in cloud. Upon different situations, the cloud pushes corresponding codes to the container.

The native Container-App can load applications dynamically, so business people do not need to install many apps on their smart phone or other devices. There is no need to create a different app for a different device in future since the application or application logic can be changed according to the user scenarios or situations as the description in blow.

Secondly, we can define context and business services separately, and combine them with the event life cycle to definite scenarios matrix, thus can evolve various App business applications. All of those contexts are carding by motive events from the angle of time, people, places, objects, and so on.

Thirdly, the application assembles codes according to the situation which needs to abstract many business scenarios from B2B, B2C, B2B2C G2B, even G2C *etc*. in whole ecosystem. The agent of the situational application can be expressed with a group of five elements [21] [22].

Situational app makes our life easy, but the underneath technology is really complex and there is big technology barrier built up over years.

Situational app can be used in many vertical markets. While we building the technology, we attracted some top-tier customers as early adopters, mostly vertical system integrators in transportation, field services, health care, government, event management, for both B2B and B2C customers. We created white label container apps but also keep a general purpose 1at app, then the user can access across many different services with just one app.

We want to sharp focus on one vertical market – transportation, because we have many unique advantages and high business entry barriers to block competitors

The following screenshots are some situational applications. A passenger takes a travel, the scenario including contract talking, office work, information sharing, ticket check-in, waiting, order gift for his girlfriend, dining, leisure time, on boarding, Itinerary reminding, extract the rent vehicle, the brake capture to expend, hotel check-in and so on. All the applications he used are supporting by several clouds, according to the container of contextualized examples.

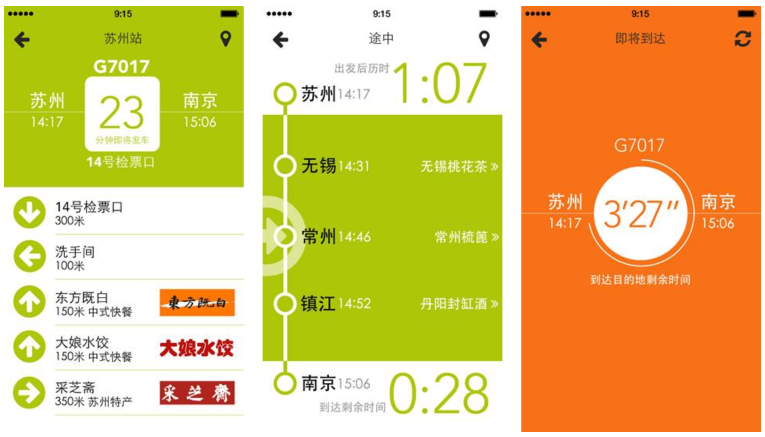


Figure 8. Screenshots of Situational Applications

And further, we can achieve the real IoE by deploying the application container virtual machine into an intelligent hardware chip through mobile phone or a tablet, and interact directly with united equipment. Now we are prototyping a tiny virtual machine for low-end MCU with only 256KB memory such as arduino processor. Hopefully the developer without firmware development experience can write JavaScript code and use a phone to load the compiled "byte code" into the tiny chip, completely "cloudifying" the smart devices. We want to enable the IoE world where all devices are connected to clouds at application level.

7. Discussion

The new "Container-Cloud" architecture had proven by several large complex enterprise projects such as High-speed Rail field services, highway maintenance, state-wide long-distance bus services, commodity trading for agriculture products, and environment monitoring, *etc*.

In the C-C architecture, we must give up the Web side thinking and fully consider the characteristics of the mobile device, use the app container as a unified entrance, generate applications and push services automatically in backstage based on scenario. We must combine online and offline mechanism to provide mobile applications and services wholly from a business perspective. As can be seen from Figure 6, the mobile application evolution roadmap, mobile applications from the Web to mobile Web, and then customize the App, it is now an urgent need to redefine app stage and even eliminate the use of the mobile application containers.

The technology will potentially enable businesses adapting and transforming into IoE era, as Java did for web, Android did for mobile.

The mobile “Container-Cloud" technology discussed in this paper has the following features to achieve contextualized application:

- The cloud programming system contains reusable components, configurations, and service aggregation functions. It can push application components according to the business rules and policies to mobile container and automatically aggregate services on demand.
- The specific applications on the mobile container can manage the functions such as dynamic loading, upgrade and maintenance.
- It provides agile response and high adaptive ability through consultations and cooperation by multi-agent business response of flexible, and a higher level of system function.
- Build business model and business rules socially, crossing enterprise boundaries.

- Nevertheless, no one idea is perfect. There are two inadequateness of the "Containers-Cloud" as below:
- It's not suitable for an application which is stable, or the business function is very simple.
- It's not suitable for a pure offline network operation.

8. Conclusion

In this paper, we introduced an innovative mobile application architecture called "Container-Cloud", some examples of the "context-aware" applications and the C-C architecture that supports managing the whole life cycle of application software on cloud. It produces all kinds of applications and manages their whole life cycle, meeting various business needs at any time quickly.

Unlike traditional container applications, the "context-aware" applications can be automatically loaded and executed according to users' related location (such as the station, shops, workflow, *etc*.), related smart objects (*e.g*., air conditioning, sensors, coffee machine *etc*.) or others. Such "context-aware" applications can be long tail applications or light-weighted applications or even one-time-use applications. This leads to a new application paradigm that may influence all apps and finally eliminate many apps, reaching the vision of application automatic perception.

It can easily match a variety of applications, be ready for deployment at any time, and suitable for applications to be quickly generated.

Meanwhile, the architecture brings about many innovations in programming that are helpful for building IT eco-system：

- Innovation of platform operations: Realization of user-centric, connecting everything.
- Innovation of service operations: Operational scenario, tenants, and other brand apps very easily for the customer service.
- Innovation of development: Users with programming skills can develop in the code layer, or mix their own applications through simple tools such as a building-block type places.
- Innovation of use: Apps can simply create real-time connectivity required by the use cases and scenarios.

Finally, the native container of "C-C" platform has the advantages of accessing all device hardware and system level resources, providing native user experience, providing better performance, enabling less messy code base, needing less efforts on support and enabling debugging on various browsers.

Acknowledgements

This paper is supported by 1at PaaS platform of Mash5. The authors will thank Dr. Chun Ray Xia, Dr. Robin Li, and Ms. Xin Yao.

References

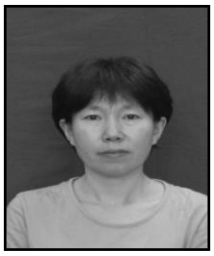
[1] http://www.techopedia.com/definition/30121/internet-of-everything-ioe
[2] Mike J. Walker & David W. Cearley, Research Guide: The Top 10 Strategic Technology Trends for 2015, Gartner G00273142, January 20**(2015)**
[3] http://www.mash5.com https://m.mash5.cn/
[4] Are Containers the Beginning of the End of Virtual Machines? , https://virtualizationreview.com/articles/2014/10/29/containers-virtual-machines-and-docker.aspx
[5] Abel Avram, Google Announces Cloud Container Engine Using Kubernetes, http://www.infoq.com/news/2014/11/google-cloud-container-engine, Nov 05**(2014)**
[6] Claus Pahl, Brian Lee, Containers and Clusters for Edge Cloud Architectures – a Technology Review
[7] Milos Gajdos,containerops.org/2014/12/19/docker-vs-rocket-gimme-a-break/

[8] Web App and Widget Engine Zembly Launches, http://mashable.com/2008/06/07/zembly/
[9] Force.com multitenant, http://www.salesforce.com
[10] Mahendra Mehra, Kailas.k. Devadkar, Dhananjay Kalbande ,Mobile Cloud based Compiler: A Novel Framework for Academia, International Journal of Advancements in Research &Technology, **(2013)** April, Volume 2,Issue4, pp.445-449
[11] Mash5 Technologies: The cloud compiling platform - completely cloud develop process, http://www.cyzone.cn/a/20140903/262451.html
[12] Mobile cloud，http://searchcloudapplications.techtarget.com/definition/mobile-cloud
[13] Intel, Implementing a Cross-Platform Enterprise Mobile Application Framework, http://www.intel.com/content/dam/www/public/us/en/documents/best-practices/implementing-a-cross-platform-enterprise-mobile-application-framework-paper.pdf
[14] https://www.oracle.com/java/index.html
[15] App Container Specification, https://github.com/appc/spec/blob/master/SPEC.md
[16] Chun Xia, Bin Li, and Bo Yu, Inter-Cloud Application Virtualization. U.S Patent US 2012/0239825 A1 **(2012)**
[17] Gerhard Weiss, Multi-agent Systems: A Modern Approach to Distributed Artificial Intelligence, New Ed edition, The MIT Press **(2000)**, pp.165-200.
[18] Jianyuan Yan, Juijung Liao, Chinghui Shi, Multi-Agent hybrid mechanism for financial risk management, Journal of industrial engineering and management, **(2015)** Vol. 8, No 2 , pp.435-452
[19] DEY AK, ABOWD G D. Towards a better understanding of context and context-awareness, October 2 **(2008)**
[20] Jinggang Wang, Xiaobing Zhang, Xiaojian Hu, and Ju Zhao, Survey on logistics service mode based on cloud computing, Journal of Logistics, Informatics and Service Science, **(2014)** Vol.1, No. 2, pp.63-74
[21] Cong Du, Suozhu Wang, Research on mobile web cache prefetching technology based on the combination of context and user interest degree, Journal of Logistics, Informatics and Service Science, **(2014)** Vol.1, No.1, pp.69-78
[22] Bo Yu, Zhongliang Guan, Cuiyu Qi, and Shifeng Liu, Scene Perception in IoE Era, Proceedings of the 2015 IEEE International Conference on Logistics, Informatics and Service Sciences (LISS' 2015), July 27-29 **(2015)**; Barcelona, Spain & Beijing, China

Authors

Bo Yu, received his Master in the Business School from Jilin University (2004), and received his B.S. degree from Northern Jiao Tong University (1991), respectively. He is currently pursuing the Ph.D. degree in School of Economics and Management, Beijing Jiaotong University. He is co-founder, vice president and professorate senior engineer of Mash5 Technologies, Inc. His current research interests are focused on mobile cloud, data mining, marketing analysis and management information system.

Zhongliang Guan, received his PhD degree from Beijing Jiaotong University. He is currently professor with School of Economics and Management at Beijing Jiaotong University. His current research interests are focused on management theory and method, information and service, Information theory and practice, digital logistics.



Yan Jiang, received her Master in the College of Computer Science and Technology, JiLin University (2008), and received her B.S. degree from Xidian University(1993), respectively. She is currently pursuing the PhD degree in College of Computer Science and Technology, JiLin University. She is a vice professor of College of Computer Science and Technology, Changchun Normal University. Her current research interests are focused on cloud, graphics, bioinformatics.

Shifeng Liu, received his PhD degree from Beijing Jiaotong University in 1998. He is currently professor with School of Economics and Management at Beijing Jiaotong University. His current research interests are focused on management theory and method, information of transportation, quantitative analysis of economic problems.

.

Cuiyu Qi, she received her Master in the College of Information Science & Engineering from Ocean University of China (2011), and received her B.S. degree in the School of Management from Xi'an Jiaotong University (1994), respectively. She is currently pursuing the PhD degree in the School of Economics and Management at Beijing Jiaotong University. She is currently a scientific research manager in the China Center for Industrial Security Research, Beijing Jiaotong University. Her current research interests are focused on big data, enterprise transformation and upgrading, and enterprise strategic response.