# Chapter 1 Introduction

文章内容的意见请使用批注形式，

语言表的的意见请直接修改并加黄底。

**1.1 Introduction to the Ningbo E-government Cloud**

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

The Ningbo E-government cloud computing center is designed to play the role of the support platform for cloud applications of the smart city and electronic government project, both of which have swept the world. Based on centralized management aiming at optimization of public foundation applications configurations, it implements a nice loose coupling layout between the applications and middleware. Using the idea of the bus, not only the cloud improves the interagency collaboration between business across different departments, but also creates a positive environment of application interaction.

**1.2 Introduction to the PaaS Cloud Architecture**

Platform as a service (PaaS) is a category of cloud computing services that provides a platform allowing customers to develop, run, and manage applications without the complexity of building and maintaining the infrastructure typically associated with developing and launching an app.

According to characteristics of the network in Ningbo E-government cloud computing center, my project team decided to deploy two sets of PaaS cloud architecture in the internal and external network. The applications on the left PaaS cloud will provide a punch of smart city web services for the public and enterprises and at the same time the right PaaS cloud will run governmental applications sternly. It is worth mentioned that applications running on the exclusive one mostly migrate from original government services platform for the sake of rapid development and reusability of business logic. Indistinguishably and expectantly, the PaaS cloud architecture makes a contribution to the convenience and efficiency of construction, deployment and management in software engineering field because of the unification planning and management of the resources, the elastic expansion of resources and the fabulously rapid application deployment in a timely manner.

Each PaaS Cloud Architecture consists of five participants (the structure index in figure 1.2). In addition to smart city third-party applications outside, they are the underlying IaaS layer by adopting the latest OpenStack version called *Mitaka,* the enterprise services bus (ESB) which is in charge of management and monitoring of public or restricted services, the CloudFoundary Web(cfWeb) which is responsible for management and monitoring of various complex software running platforms such as java or python environment and their dependent middleware ActiveMQ as an illustration and the last but crucial important one on the SaaS cloud domestic consumers’ point of view, the Unify Identity Authentication and Access Control System (UIAACS) which reconstructs the three significant services in E-government cloud environment: federated SSO, tenant management and access control system.



**Figure 1.2 Ningbo E-government PaaS Cloud** **Architecture**

* + 1. **OpenStack Mitaka**

OpenStack is a free and open-source software platform for cloud computing, mostly deployed as an infrastructure-as-a-service (IaaS). The software platform consists of interrelated components that control hardware pools of processing, storage, and networking resources throughout a data center. Users either manage it through a web-based dashboard, through command-line tools, or through a RESTful API. Mitaka brings plenty of benefits to the government information management department: cost savings on hardware and infrastructure, capacity management, disaster recovery/business continuity, cost savings on IT staffing and administration and the ability to access new skills and capabilities.

**1.2.2 Unify Identity Authentication and Access Control System (UIAACS)**

The UIAACS in PaaS Cloud plays a symphonic work. All the other applications or service providers could join the cloud by using UIAACS single sign on mechanism and giving away the user management including the following mentioned cfWeb and ESB. Facing some inalienable identity providers (IdP) actively, the UIAACS choose to docks these the existing authentication systems offered by IdPs with little configuration on applications. Anyone who logs in UIAACS could be authorized to log in any application or experience services on the PaaS Cloud magically and trustworthily. Instead, the UIAACS will take the responsibility for tenant management aiming at compatibility with any potential user management model in E-government environment. Excitedly we design and implement a security solution to protect resources access from wanton destruction on purpose or not.

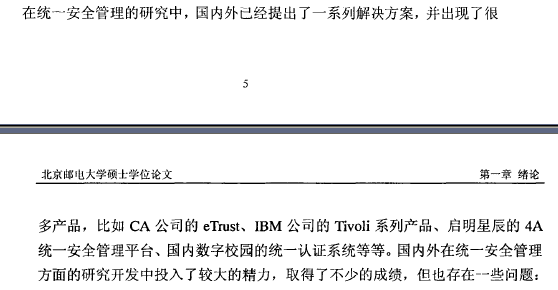
**1.2.3 CloudFoundary Web (cfWeb)**

The cfWeb is one of the nation’s leading CloudFoundary-based independent research and development project supporting some wildly popular language, typical development frameworks and universal back-end services. We develop a friendly responsive web-based interface and message system for administrators as long with enterprise users. Because of some outstanding features that the CloudFoundary project brings out such as collaborative development, continuous integration, rapid deployment, visual supervisory, efficient management and particularly elastic expansion, developers on cfWeb could focus on applications itself and the behind commercial plans, minimizing the time cost.

**1.2.4 Enterprise Service Bus (ESB)**

Enterprise Service Bus is the product of traditional middleware technology and web service technology. The ESB collects all the service in the enterprise internal development, providing a more fine-grained and standard unified control of services including a review and approval process and a supervision service system. ESB logically is divided into three accessible parts which are, in order, the service registry center who receives all the requests from services developers, the service application center who publishes all approved services to all validated applicants and flagrantly the services monitoring center telling different levels and dimensions of resources usage. To carry out the design principle on the centralism unification management, these three parts in ESB hires the UIAACS for authentication and authorization.

**1.3 Introduction to Existing Work**



This section will present the existing related work of how to organize the unify identity authentication and access control system in traditional enterprise.

1.3.1 About single sign on:

The various types of SSO shown in Fig: 1.3.1, fall under different categories, based on where they are deployed (Intranet, Extranet, Internet); how they are deployed (architecture – Simple, Complex); the credentials they use (token, certificate ..) and the protocols they use (Kerberos, SAML, OpenID..).Following picture shows the types of SSO and their classification:



**Figure 1.3.1 Classification of Single Sign-On**

1.3.2 About cross domain user management:

Along with the explosive growth in the number of applications, opening a new account becomes to be a tedious and risky issue. Developers or administrators need to assures consistency for his permissions in applications. It inevitably gets more complicated to modify access permissions as he shifts position or just leaves. Another scary thing is about potential security risks to exchange user information or credentials among applications. Therefore, the following solutions stand out:

* P2P Identity Copy Mode: establish data conversion and transmission mechanism to copy user information from one application to another;
* Identity Mapping Mode: authentication service component keeps a mapping table that records different users according to a unique identifier;
* Agent-based Unified User Management Mode: install a special user agent in applications to transform various structure user data to unify structure

1.3.3 About tenant access control:

After accumulation and sedimentation in a long time since 1960s, the core idea about tenant access control is about the extensible mechanism between user and his role. Bertino comes up the Temporal-RBAC model which imports time parameter into traditional RBAC model. Administrator must consider time constraints when assigning roles to support for temporary permissions’ dependency. Joshi expands Temporal-RBAC model to Generalized Temporal RBAC model which applies time constraints to role abstraction and responsibility. Then more and more impact factors such as environment, location, and even system status join the constraint party. Kumar and McDaniel send out the content-sensitive RBAC model finally. These models exist a common weakness, that these models rely on manual role assignment.

**1.4 Main Work and Content Organization**

The whole Ningbo E-government PaaS cloud is a large system. It consists of four main subsystems: OpenStack, cfWeb, ESB and UIACCS. In this paper, I can’t cover all these four topics because of the limits of the time and knowledge. So I focus mainly on the unify identity authentication and access control system which is also one of my results during the internship. In the rest of this paper, I introduce the UIACCS architecture, data structures and detailed algorithms, devoting myself to break down contradiction in three requirement-driven fields which are the unify identity, tenant management and centralism access control and then constructively provide our on-line practice. Considering the competition restrictions, there is no code leakage.

This is how this article is organized. In Chapter 1, I draw the PaaS architecture outline and UIAACS background including related work. Some unmissable relevant concepts and theories are introduced in Chapter 2. Chapter 3 moves to specific requirements and the preliminary and detailed design shows up in Chapter 4. The implementation based on WSO2 identity server working with some pivotal algorithms completes our UIAACS in Chapter 5. Next Chapter I provide test and related evaluation of the performance. At last Chapter 6 concludes the dissertation and discusses future work.

Reference

