Taas – Ticketing as a Service

João C. Ferreira[1,3], Porfírio Filipe[1], Carina Gomes[1], Gonçalo Cunha[2], João Silva[2]
[1]*ADDETC-ISEL,Lisbon, Portugal,* [2]*Link-SA,Lisbon, Portugal,* [3]*Centro Algoritmi, Univerdidade Minho*
{jferreira,pfilipe,cgomes}@deetc.isel.ipl.pt, {goncalo.cunha,joao.r.silva}@link.pt

Keywords: CLOUD COMPUTING, SOFTWARE AS A SERVICE, TICKETING, AUTOMATIC FARE COLLECTION.

Abstract: The goal of this research work is to introduce the concept of a lower cost flexible system for ticketing purposes implemented on a cloud platform. We propose therefore the evolution of the traditional architecture of ticketing for a cloud based architecture in which the core processes of ticketing are offered through a Software-as-a-Service (SaaS) business model, which can be subscribed by operators that pay-per-use. Ticketing terminal equipment (e.g. gates, validators, vending machines) are integrated in the cloud environment. This approach is achieved by moving business logic from terminals to the cloud. Each terminal is registered to be managed by each own operator, configuring a multi-tenant implementation which is vendor hardware independent, allowing to address elasticity and interoperability issues. The elasticity of the cloud will support the expansion/implosion of small (transport) operators business around electronic ticketing. In the near future, this ticketing solution will promote collaboration between transport operators.

1. INTRODUCTION

In this work we propose the definition of an architecture that allows building a common transportation ticketing services to which the terminals can connect to a simple Plug-and-Play model, so the cloud automatically recognize and configure any ticketing equipment at installation time by eliminating manual configuration.

It will therefore be necessary to define the architecture of the cloud services, as well as the characteristics of terminals to consume those services. The goal is to achieve centralization of all business logic and move terminal specific logic to the cloud, therefore reducing the overall system complexity.

This change of paradigm to benefits from the fact that cloud ticketing services can be accessed through the Internet and they can be elastically grown or shrunk, providing easier scalability and high availability.

Thus the entire application logic can be consolidated and centrally implemented on open and secure protocols, making the equipment simple frontends benefiting from being online with the central ticketing system to offer value-added features on lower capacity terminals.

In the aviation industry there are already systems for seat reservations and ticketing to be offered "as a service" for several airlines, often at a cost of only pennies per ticket [GAO, 2003]. In fact, very few low cost operators manage and maintain its own ticketing system because SaaS options available in the market do it more efficiently and at a lower cost.

Having lightweight devices connecting to the business logic on the cloud are also have the following advantages:

- Consolidated logic with easier maintenance and lower IT costs;
- Improved physical security (avoid secure elements distribution and logistics);
- Enable functionality by subscription for devices;
- Support offline and online operation models over the same infrastructure;
- Reduced complexity for supporting new terminals, by using open interoperable protocols.

This paper was written within the scope of the "SmartCITIES Cloud Ticketing" project, which is

focused on designing an interoperable, cost-efficient, multi-supplier, cloud based ticketing solution, where transit agencies may opt in and out when they need to. This project brings together two complementary sets of experiences: the engineering experience applied to ticketing solutions of Link Consulting [http://www.link.pt/smartcities] and the computer science and research experience of ISEL – Instituto Superior de Engenharia de Lisboa [http://www.isel.pt], which lays out the path to a solid foundation of a cloud ticketing solution.

2. ELECTRONIC TICKETING

An electronic ticketing system is designed to enable fare collection in a simple, secure and efficient way. In the public transport operators market, electronic ticketing is associated with a trip or set of trips of a transportation service. A survey of electronic ticketing system can be found at [Mut-Puigserver, 2012] and [Vilanova, 2002].

The customer obtains an electronic ticket medium (smart card, mobile device ticket) which is the storage location for electronic tickets. When a ticket is sold by one operator, the sale is registered in the ticket medium and validated before use [TCRP, 2006]. In connection with the sale and use of tickets, electronic information is stored and processed for the purpose of producing: (1) Billing data are used in the sharing of ticket revenues among the various operators involved in the ticketing system; (2) Revenue data; and (3) Statistics (about the sale and use of tickets).

An electronic ticketing system may also incorporate a number of other functions: (1) Ticket inspection function; (2) Internet services (for example online sales of tickets); (3) ticket vending machine; and (4) entrance/exit ticket validation machines. This operation is performed at front end system (entrance and exit ports) with dedicated equipment solution using a private dedicated network.

A Ticketing system operation is based on one token issuing entity (issuer), a set of users, tokens, and verifiers who verify whether tokens are valid, performed in a dedicated solution using a private network to deal with security and privacy issues. Typically, a user U must buy a token from token issuer I. Therefore, U selects his desired ticket and pays it. Issuer I then checks whether U is eligible to obtain a token (e.g., whether U paid for the ticket), and, if applicable, issues a token T and passes it to U. From now on, U is able to use token T to prove that he is authorized to use the transit network. This means that every user who is in possession of a token that has been issued by a genuine issuer is considered to be an authorized user. If a user U wants to travel from a place X to some location Y . Before U is allowed to enter the transit system at X, he must first prove to a verifier Vin at the entrance of the transit network that he is authorized to access it. If Vin can successfully verify the user's token, U is allowed to enter. Otherwise access will be denied. During his trip, U may encounter arbitrary inspections where he must prove that he is authorized to use the transit network. Thus, a verifier V may check the user's token T. If verification of T is successful, U is allowed to continue his trip. Otherwise, U must leave the transit network and may be punished for using it without authorization. After arriving at Y , the user's token T can be checked for a last time. Again, if T cannot be verified successfully, U may be punished.

Note that the token is typically bound to some limitations. For instance, it may be bound to some geographical or tim usage restrictions. Additionally, a token may be bound to the identity of its owner (i.e., the entity that bought the ticket).

Most of these ticketing systems are based on proprietary solution with terminal at transportation stop and a central system to handle all related operations.

3. CLOUD TICKETING

The vision of present proposal is illustrated on Figure 1, where a set of dedicated services are available in an SaaS approach and front end devices (e.g. validators, vending machines, gates and others) 'migrate' from an integrated fat device to a flexible and modular thin device with all or part of business process logic executed remote in a SaaS approach. The idea is to interact with several equipment interfaces and integrate related business process in a SaaS approach. The proposed idea for a ticketing system on the cloud can be simply described as a set of standards based services on the cloud to perform a specific business function (e.g., card issuing, ticket sale). These services are available through a communication protocol that is common to all registered devices. When front office devices (PCs, POS, Smartphones, tablets, web browsers, etc) first announce themselves to the cloud services, they identify themselves, as well as the tenant they belong to and automatically downloading the relevant software and configurations for the functions assigned to them. After the registration occurs, the device is able to interact with the cloud services, for instance a tablet computer connects to the cloud provisioning service, authenticates itself, and

automatically downloads the ticket sale software. Afterwards is allowed to start selling tickets to customers.

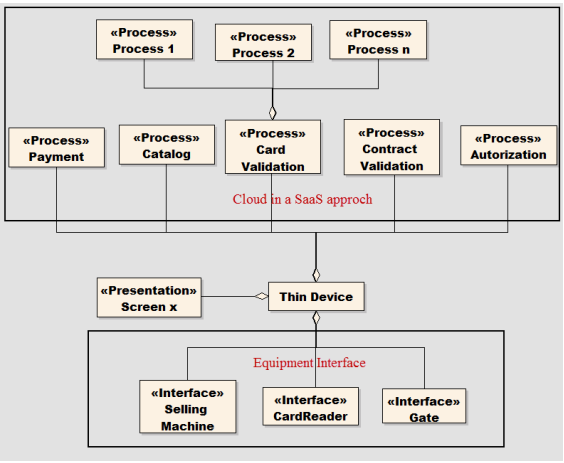

Figure 1 – Vision of current proposal for electronic ticketing system development

The proposed architecture is composed of the following layers of services (see Figure 2):

- Data Access Services – internal services to access business data (customers, cards, sales, validations, etc);
- Business Services – cloud exposed services to implement business operations like registering a new customer, authorizing a ticket sale for a specific customer, or consulting a catalog of tickets available to the specific card;
- Business Process Services – services that coordinate among multiple business services to implement specific use cases, e.g., ticket sale use case, which generally involves: (1) reading the card; (2) browse the ticket catalog for available products; (3) choose the ticket to buy; (4) pay; (5) load the card; and (6) confirm and register the sale. The output of this service the information to present to the user on the screen, as well as available operations. The inputs of the service are the actions performed by the user.

The Data Access Services Layer is a lower level internal layer, used to abstract the access to the data provider. The Business Services Layer should implement the service business logic of the overall system, including data validations, user authorization, accounting algorithms and data correlation. Here we highlight the proposed cloud services on the Business Services Layer: (1) Customer Service; (2) Card Service; (3) Ticket Sale Service; (4) Validation and Inspection Service; (5) Device Provisioning Service; and (6) Ticket Catalog Service.

To implement a full ticketing system multiple use cases have to be considered, here we highlight only a few of them, which are included on the Process Coordination Services Layer: (1) Ticket Sale Business Process Service; (2) Customer Registration Business Process Service; (3) Card Renewal Business Process Service; and (4) Card Cancellation Business Process Service.

To complement the exposed cloud services, there is also a range of backoffice applications, to manage the system as a whole (e.g., CRM, Product Catalog Management, Reporting, Device Management, etc).

The proposed architecture is designed to support two different sets of front-end devices on the customer side: The ones with lower processing capacity but are always online; the other that at some point in time need to work offline but have higher processing capacity. The first set of devices has what we call “thin apps” the second set has the “fat apps”.

Thin apps know nothing about business logic and only have presentation logic built-in. They receive the screens to be displayed and send back commands. All process coordination logic, as well as business logic is located on the cloud. Every operation must be made with an online connection to the cloud services on the Business Process Services Layer.

A generic workflow of a thin app performing and action with a cloud service, where we highlight a few points: (1) The thin app interacts with one business process service, which coordinates multiple business services; (2) The thin app receives presentation information and sends back commands; (3) Every app interaction communicates with the cloud; and (4) When a use case ends, the app sends an action which generates a confirm operation (e.g., ticket sale confirmation). The confirm operation commits the information to persistent storage.

On the other hand, fat apps are for example PCs, tablet computers or even POS and typically have the process coordination logic installed locally and some offline data to enable offline usage. In ticketing applications they are still required for some use cases, where short timing requirements exist and offline capability is a must, for instance a validation device on a bus. In this example, there may parts of the bus route where it does not have network coverage and the timing requirement from the moment the user puts the cards on the validator to the moment his access is approved should be typically bellow 300ms [Smart, 2006].

A generic workflow of fat apps, the general case is to have the process coordination logic installed on the device, with interactions local to the device (e.g., ticket validation), and at the end of the operation the cloud service is informed of the operation result. Below is a generic workflow of the interactions, with the following highlights: (1) The fat app performs every interaction locally (possibly using cached reference data); and (2) The fat app periodically sends the confirm operations to the cloud service (e.g., ticket validations). These confirm operations commit the information to persistent storage.

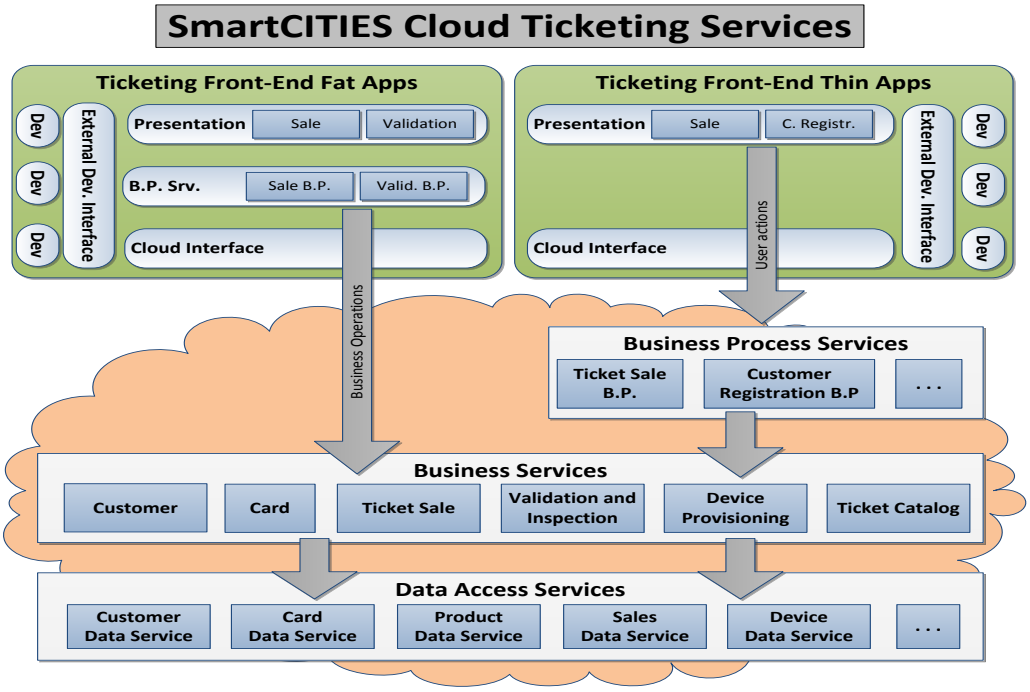

Figure 2 - Cloud Ticketing Architecture

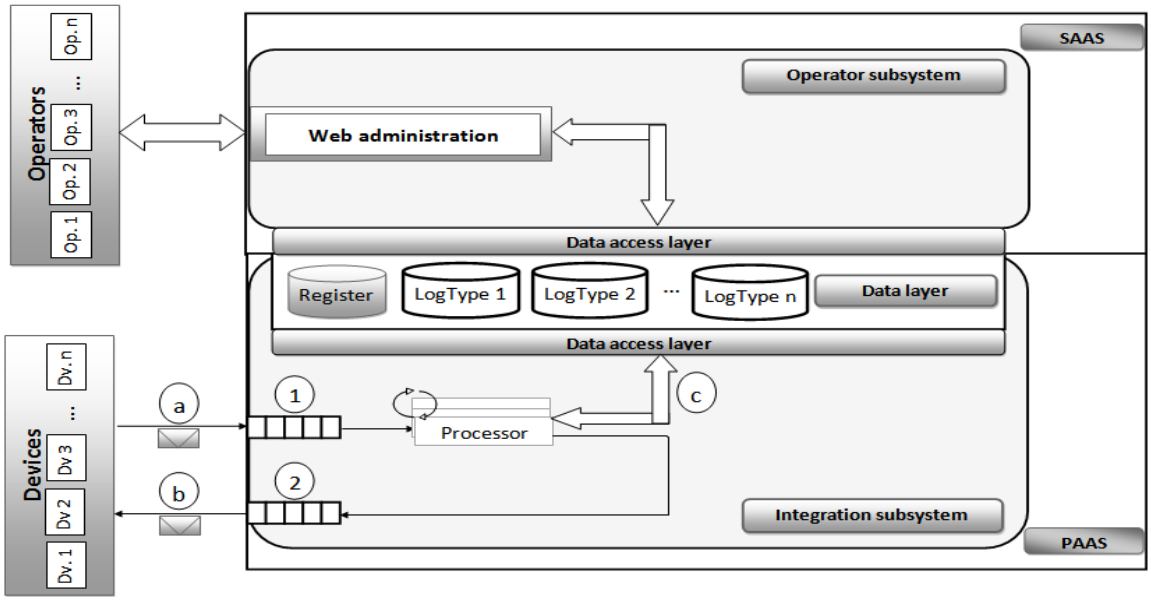

Figure 3 – Architecture for device integration

4. CLOUD INTEGRATION

The proposed architecture for integrating terminal devices in a cloud computing platform is depicted in Figure 3. The integration protocol admits that part of the logic of each device runs in the cloud platform. Figure 3 shows in the periphery, two sets of entities: operators and devices.

The operator manipulates the historical data of the device and the respective configuration accessed via the data layer by an administration cloud application. The operation of the subsystem comprises a multi-tenant feature, which interacts with the data layer. The device activation can be made by its owner/operator, using the application, where is selected previously defined device's type and returned the device's unique identifier. The application's main responsibilities are to setup, access to the register operation and activation / deactivation of device.

A request / response protocol allows equipment (e.g. ticket validator or sale operation) to call remote operations hosted in the cloud. The data manipulated by the devices flows across queues "1" (input for request) and "2" (output for response). Within the integration subsystem, instances of processors handle messages from queue "1", to dynamically instantiate the type of device and perform the requested operation indicated in the message.

The session context associated with the maintenance status of the device is dependent on their type. Objects, that store configuration and historical data of the interactions, are managed in a data layer. To ensure load balancing and isolation of data, the history of each type of equipment is stored in isolated data object, called a "LogType n" where "n" represents the type of device. The configurations are stored in a data object, referred as "Register". The quantity of entries in this object coincides with the quantity of integrated devices.

Is interesting to note, for sending the requests made by the device there is only one message. Instances of processors read the messages and validate its format. Validation consists in confronting the XML document, which represents the message, with appropriate XML Schema Definition (XSD). If one assumes that the message is malformed, relevant information is added to the history and the processing of the current request is finished. Subsequently, the processor accesses data register (interaction "c"), so that, in the context of the session handling information specific to a particular type of device. With the information received, it is checked if the device has been previously registered and is in active state. If any of the validation is unsuccessful, the history is updated and the processing of the current request is finished. In the next phase, the processor checks whether the request corresponds to the first message from the equipment designated by check message. If so, the processor creates the queue "2" for response messages and sends through this queue a message notifying that the cloud component of the device is ready to be used. Given the information present in the device's configuration, the message parameters lead the processor to dynamically instantiate the processing of its contents in accordance with the identifier of the requested operation. After the instantiation and execution of the operation, in the context of the current session, the log history is updated. Finally, the processor sends a message to the queue "2", with the result of the operation, the device that was blocked (interaction "b") receives the message. For details of the common message model for messages exchanged in interaction "a" and "b" and for the definitions of the validation rules for messages and data objects are described and a Windows Azure Demonstrator, see [Gomes, 2012].

5. TICKETING AS A SERVICE

Taking into account a small transport operator, that wants to implement an electronic ticketing system. Our approach for the system development involves three main actors, the cloud provider, the transport operator and the technological partner. The Cloud Provider gives three levels of cloud services: SaaS, PaaS e IaaS. In our concept demonstrator, we use Microsoft Windows Azure (WA) platform.

The Technological Partner (TP) is responsible for developing and maintaining the ticketing system, and creating a set of services to enable a personalized business process for the transport operator. From this modular approach TP can use previous developed services to implement the required system able to support the business and the front end device can also be provisioning and using past developed projects. The current proposal is oriented for a SaaS approach, but with more work development can be implemented in a PaaS or even used in dedicated solution. A complete description is available on [Gomes, 2012], where two terminals were integrated in the cloud using resources of cloud provider (e.g. processors, queue messages). The operator buys the front-end devices (e.g., gates and validators) that are

register by TP on the cloud system taking into account the device provision procedure. At this point we are exploring a novel approach of a terminal on an Android device.

In order to support multi-tenancy on the cloud services it is important to consider that multiple operators may be organized in a common metropolitan area, having some (not all) common customers, smartcards and multi-modal tickets [Globalplatform, 2009]. In these cases, it is important to have a consolidated view of common business information (customer, cards, sales, validations, etc) by all operators to enable revenue distribution. With this scenario in mind, we propose a hierarchy of tenants with multiple roots (see Figure 4). Each root is a transport area with multiple operators where some parts of the business information (customers, cards, etc) are common to several operators. The hierarchy of tenants has the following rules: (1) Lower level tenants (operators) can view information about their private customers, as well as business information common to the metropolitan area; (2) Upper level tenants can read and consolidate common business information to the lower level tenants (e.g., customers, cards, sales and validations); and (3) Upper level tenant may not see information about private customers, and sales/validations of private tickets.

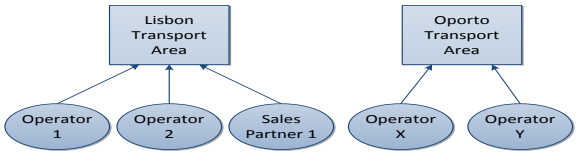

Figure 4 – Sample hierarchical structure of tenants

Here we discuss the option of having a shared database or separate database/schema implementation of multi-tenancy. The main concerns on this decision were privacy, security and extensibility. It is necessary to avoid risks of having one operator see information about other operators (they may be competitors). On the other hand it is very common for an operator to require customizations specific to its business. Therefore we have chosen to have a separate database approach.

With the requirement of having a hierarchy of tenants, using a separate database approach, has an additional challenge – how to consolidate common business information (e.g., sales of multi-modal tickets) on the upper levels, which is generated at the lower levels? The answer is to have the lower levels ship the common business information to the upper levels, where it is consolidated and becomes its master repository. Private information on the lower levels is never shipped

6. CONCLUSIONS

The proposed system uses a novel approach based on SaaS approach for the development of personalized ticketing software. This project started 12 months ago in a synergy between the technology company Link Consulting (www.link.pt), with 10 years of experience in ticketing business and researchers in computer science at ISEL (www.isel.pt) a polytechnic institute in Lisbon. Starting from the initial kickoff meeting several Masters thesis are running: (1) network security, where we study the privacy and security problems of moving business logic from terminals to the cloud; (2) Cloud Computing projects, where we are developing the concept for different cloud platforms as well the implementation of a set of services regarding the complete ticketing process; and (3) the thin devices are being developed in an Android platform, where ticket selling and validation processes are also under development.

The architecture described in this paper shows how devices may communicate with cloud services to enable a business process and a general overview of our vision for the development of electronic ticketing system in a modular service approach available in the Cloud.

REFERENCES

Carina Gomes (2012). Estudo do Paradigma Computação em Nuvem. Master projet at ISEL, (in Portuguese).

GAO (2003). AIRLINE TICKETING - Impact of Changes in the Airline Ticket Distribution Industry, available at http://www.gao.gov/assets/240/239237.pdf.

GlobalPlatform's Value Proposition for the Public Transportation Industry (2009). Seamless, Secure Travel Throughout Multiple Transportation Networks. http://www.globalplatform.org/documents/whitepapers/GP_Value_Proposition_for_Public_Transportation_whitepaper.pdf.

Macià Mut-Puigserver, M. Magdalena Payeras-Capellà, Josep-Lluís Ferrer-Gomila, Arnau Vives-Guasch, Jordi Castellà-Roca (2012), A survey of electronic ticketing applied to transport, Computers & Security, Volume 31, Issue 8, November 2012, Pages 925-939.

Smart Card Alliance (2006). Transit and Contactless Financial Payments: New Opportunities for Collaboration and Convergence.

TCRP Report 115 (2006). Smartcard Interoperability Issues for the Transit Industry, Transit Cooperative Research Program.

Eugenia V. Vilanova, R. Endsuleit , J. Calmet, I. Bericht (2002). State of the Art in Electronic Ticketing. Universität Karlsruhe, Fakultät für Informatik.