Cloud Terminals for Ticketing Systems

João Ferreira[1,2], Porfírio Filipe[1]
[1]Instituto Superior de Engenharia de Lisboa
Lisbon, Portugal and [2]Centro Algoritmi, Univ. Minho
{jferreira, pfilipe}@deetc.isel.pt

Gonçalo Cunha, João Silva
Link Consulting, SA
Lisbon, Portugal
{goncalo.cunha, joao.r.silva}@link.pt

***Abstract*— In this research work, we introduce the concept of a thin device implemented on a cloud platform for terminal devices on the front end of ticketing systems. Therefore, we propose 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 transport operators that pay-per-use. Ticketing terminal devices (e.g., gates, validators, vending machines) are integrated in the cloud environment creating the concept for a 'thin' device. 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-tenancy 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 operators.**

Keywords-Cloud Computing; Software as a Service (SaaS); Ticketing System; Terminal Device;

I. INTRODUCTION

In this work, we propose the definition of a thin device terminal in an Android platform that allows building a common transportation ticketing services to which the terminals can connect through a simple Plug-and-Play model that reduces human interventions.

Therefore, it will 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 global consolidation of all business logic and move terminal specific logic to the cloud; therefore reducing the overall system complexity.

This change of paradigm 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.

In this paradigm, the consolidation of business logic facilitates the adoption of open and secure protocols, making the terminal simple benefiting from being online with the global 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 [1]. 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 [2, 3].

There are several advantages in having lightweight terminals connected to cloud business logic, such as: (1) consolidated logic with easier maintenance and lower IT costs; (2) improved physical security (avoid secure elements distribution and logistics); (3) enable functionality by subscription for devices; (4) support offline and online operation models over the same infrastructure; and (5) reduced complexity for supporting new terminals, by using open interoperable protocols.

This paper is organized in seven sections: Section I defines the work context; Section II describes the electronic ticketing survey; Section III describes the proposed cloud ticketing architecture; Section IV describes the proposed thin device concept; Section V describes the proposed multitenant approach; Section VI describes the device provisioning associated with thin device implementation; and finally, Section VII presents the conclusion.

II. ELECTRONIC TICKETING SURVEY

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 a set of trips using transportation service. A survey of electronic ticketing systems can be found at [4, 5].

The customer obtains an electronic ticket medium (smart card, mobile device ticket) which is the storage location for electronic tickets. When an operator sells a ticket, the sale is registered in the ticket storage medium and will be validated before use [6]. In association with the sale process of electronic 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 and private network.

A ticketing system operation is based on a 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 transport network. This means that every user who is in possession of a token that has been issued by a genuine issuer is considered an authorized user. For instance, a user U wants to travel from a place X to some location Y. Before U is allowed to enter the transport system at X, he must first prove to a verifier Vin, at the entrance of the transport network that he is authorized to access it. When Vin verifies successfully the user's token, U is allowed to enter. Otherwise, access will be denied. On the other hand, during his trip, U may encounter arbitrary inspections where he must prove that he is authorized to use the transport 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 transport network and may be punished for using it without authorization. After arriving at location 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 a token is typically bound to some limitations. For instance, it may be bound to some geographical or time 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.

A. A Review of Proprietary Solutions

In Europe, there exist several implementations of the e-ticketing paradigm, mainly on the national level (limited to a single country). The information concerning system specification is for the most part publicly unavailable, which is a hurdle when a review of privacy solutions in the area is considered. However, certain pieces of information are openly accessible. Main European platforms are: (1) In the UK, ITSO (Integrated Transport Smartcard Organization) has developed a specification for interoperable smart ticketing [7], which is similar to the guidelines of the respective standards; (2) Another popular proprietary e-ticketing standard developed in Europe is called "Calypso" [8]; and (3) The e-ticketing systems based on MIFARE cards, such as Dutch OV-chipkaart, London's.

B. Main Ticketing Project

One of first initiatives in electronic ticketing was the Cubic Transportation Systems [9]. From this project, several world projects emerged. Some of them are already operational in countries like United Kingdom, Germany, France, Australia, Netherlands, and South Korea.

One of the first projects that used the contactless smart card based ticketing was Octopus [10]. It started in 1994, and became operational in the year 1997 in Hong Kong. The system was built by AES Prodata [11], which is a member of ERG Group [12] using the Sony Felica card [13] for contactless payments. The company ERG Group owners automated fare collection systems cooperating also with transport system projects in Manchester and Hertfordshire (Great Britain). The name of the project is Herts Smart Scheme using the Philips [14] platform for ticketing MIFARE. It is based on the EPT policy, is having different cards to handle special fares and is operative since 1997.

Another ERG Group project is Metrebus Card [15] operative at the moment in Roma (Italy). Metrebus Card uses a combi-card, which stores tickets like the project SIROCCO in Spain but it does not have an anonymous option.

In San Francisco (USA) ERG group has implemented the project TransLink [16] that will start its first phase in 2003. In this project, ERG Group works with Cubic Transportation Systems to develop an EPT solution which uses in the beginning a personalized card.

Another project that started around 1995 was ICARE [17]. It concerns Lisbon (Portugal), Constance (Germany), Venice (Italy) and Paris (France). This project evolved, with the entrance of Brussels into the consortium, creating a telematics platform, which defines a card-terminal ticketing standard called Calypso [18]. The protocol used in ICARE was a proprietary protocol developed by Innovatron and further on implemented by ASK [19] in France.

III. CLOUD TICKETING ARCHITECTURE

The vision of the present proposal is illustrated in 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.

The proposed architecture is composed of the following layers of services (see Figure 1):

- 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) read 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 is 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.

In order to implement a full ticketing system multiple use cases must be considered. However, we highlight only the relevant use cases, 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.

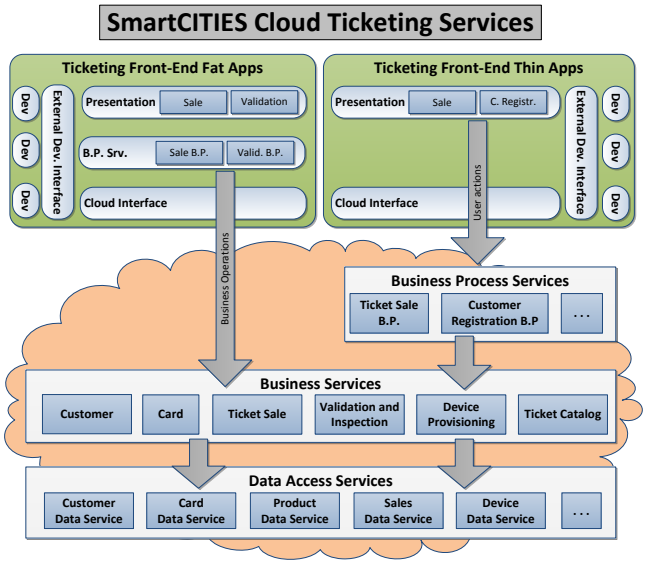

Figure 1. Cloud Ticketing Architecture.

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

The interoperability goal implies the existence of common security and privacy measures (e.g., an agreement on mutually recognized and accepted security and privacy suits).

The need for security is widely acknowledged by transport companies, since insecure solutions may result in substantial revenue losses.

Privacy, namely customer privacy, to the contrary, is not in direct interest of service providers. The reason for this is that possible risks associated with privacy violation have far less serious implications for company business compared to security. The interoperability goal poses a further challenge to privacy since sharing of privacy-critical data, which is needed for a proper delivery of transport services by cooperating companies, should be performed in a privacy-preserving way.

A. Cloud Ticketing vs Traditional Ticketing

In the cloud, ticketing system architecture is based on consuming services organized in a layer available in a cloud platform. The services have published interfaces. These interfaces support the development of personalized ticketing systems. The main effort is the definition and development around the implementation of services. Cloud platform is important to reduce/manage hardware costs and to publish the transport operator's services. Therefore, the cloud ticketing has the following benefits:

- Reduce system development cost: the creation of a robust service layer available in a cloud platform has the benefit of a better return on the investment made in the creation of the software. Services map to distinct business domains, opening the possibility of personalized ticketing systems for small transportation companies, with small budgets.
- Code mobility: since location transparency is a property of a service-oriented architecture, code mobility becomes a reality. The lookup and dynamic binding to a service means that the client does not care where the service is located. Therefore, an organization has the flexibility to move services to different machines, or to move a service to an external provider.
- Focused developer roles: cloud ticketing approach will force an application to have multiple layers. Each layer has a set of specific roles for developers. For instance, the service layer needs developers that have experience in data models, business logic, persistence mechanisms, transaction control, etc.
- Better testing/fewer defects: services have published interfaces [20] that can be tested easily writing unit tests.
- Support for multiple client types (multitenant, see Section V): as a benefit of a service-oriented architecture on a cloud platform, personalized ticketing systems can be easily developed.
- Service assembly: the services will evolve into a catalog of reusable services. Everyone will benefit from new applications being developed more quickly as a result of this catalog of reusable services. The big issue on this topic is the standardization.
- Better maintainability: software archeology [21] is the task of locating and correcting defects in code. By focusing on the service layer as the location for your

business logic, maintainability increases because developers can more easily locate and correct defects.

- More reuse: code reuse has been the most talked about form of reuse over the last four decades of software development.
- Better parallelism in development: the benefit of multiple layers means that multiple developers can work on those layers independently. Developers should create interface contracts at the start of a project and be able to create their parts independently of one another.
- Better scalability: one of the requirements of a service-oriented architecture is location transparency. Typically, to achieve location transparency, applications lookup services (in a directory) and bind to them dynamically at runtime. This feature promotes scalability since a load balancer may forward requests to multiple service instances without the knowledge of the service client.
- Higher availability: because of location transparency, multiple servers may have multiple instances of a running service. When a network segment or a machine goes down, a dispatcher can redirect requests to another service without the client's aware.
- Main disadvantages: the security and the privacy of data and latency issues.

IV. THIN DEVICE CONCEPT

An e-ticketing system manages terminal devices to sell and validate (check passengers authorization) tickets before, during and after the travel. This system provides an alternative to conventional ways for proving the existence and validity of the travel rights (e.g., paper tickets) through transferring the needed information to an electronic storage medium (e.g., an RFID card). For more details about radio frequency identification (RFID) see [22].

An e-ticketing system can be coarsely analyzed into two main parts: front-end devices and back-end systems. A front-end device is a terminal (described in Figure 2) with a RFID card reader that interfaces with the back-end systems. Typically, these terminals include business logic to control their functionalities, mainly for selling or validating tickets). These devices communicate to an IT infrastructure to request information and report performed transactions. This kind of terminal device, in current electronic ticketing systems, we designated by fat device.

Our main idea of the current work is illustrated in Figure 3, where we propose: a common interface to card readers (and other peripherals); and a common interface to the devices where the business logic is located in the cloud. The adoption of the thin device (client) concept opens several issues, such as dependency of communications, high latency issues (usually, a validation process at a gate should take less than 300ms) and additional security and privacy issues.

The main advantage of the thin device adoption is the easier implementation of new devices and the reuse of business logic across multiples types of devices, facilitating software updates. Figure 4 shows this concept, where different terminals devices can share common modules and a dedicated company terminal device can be created by the change business process, presentation layer and perhaps the card reader process (only in the case of the usage of a different smart card).

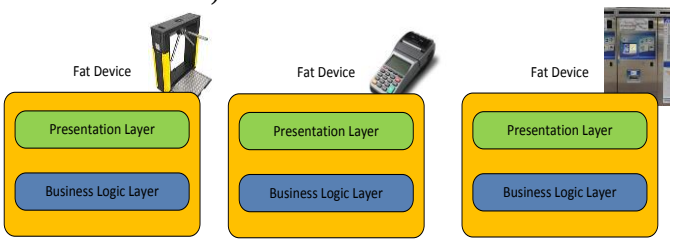

Figure 2. Fat Device architecture.

As in the case of a thin device, the term is often used to refer to software, but it is also used to describe the networked computer itself. When the applications require multimedia features or intensive bandwidth it is important to consider going with thin devices/clients. One of the biggest advantages of fat clients rests in the nature of some operating systems and software being unable to run on thin clients. Fat clients can handle these as they have their own resources.

The proposed architecture for cloud ticketing systems is designed to support two 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 online but have higher processing capacity. The first set of devices has what we called “thin apps”, the second set are the “fat apps”.

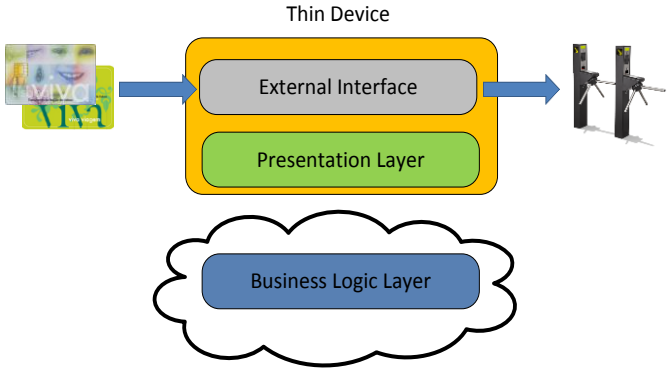

Figure 3. Thin Device architecture.

Thin apps know few about business logic and have presentation logic built-in. Typically, they receive the screens to be displayed and send back requests. Global process coordination and business logic are located in the cloud. The operation depends on network connection to access cloud services on the Business Process Services Layer. In Figure 5, we show a generic workflow of a thin app performing an action in a cloud service, where a few points are highlighted, namely:

• The thin app interacts with one business process service, which coordinates multiple business services;

• The thin app receives presentation information and sends back commands;

• Every app interaction communicates with the cloud;

• When the operation 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.

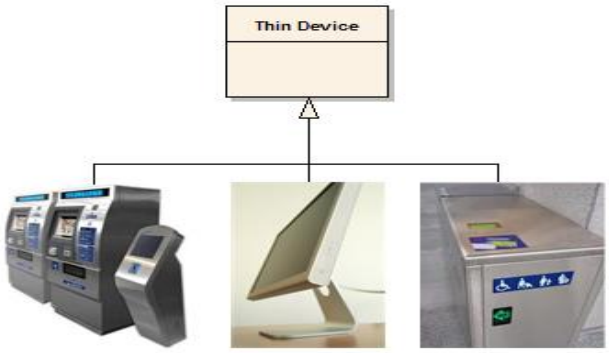

Figure 4. Usage of thin device approach.

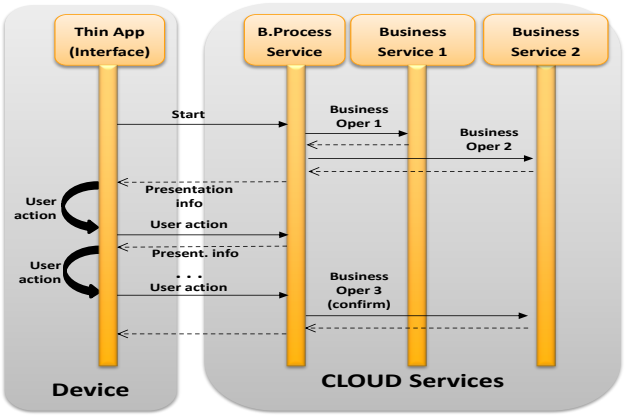

Figure 5. Thin client workflow.

On the other hand, fat apps are, for instance, running in PCs, tablets or even POS and typically have the process coordination 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. An example is a ticket validation device aboard a bus. In the bus scenario, there are zones of the route without network coverage and the timing requirement from the moment the user puts the cards on the validator till the moment of the approval should be bellow 300ms [23].

Erro! A origem da referência não foi encontrada. shows a generic workflow of fat apps, the general case is to have the process coordination installed in the device, with local interactions (e.g., ticket validation), and at the end of the operation the cloud service is informed of the operation result. A generic workflow of the interactions follows the highlights:

• The fat app performs every interaction locally (possibly using cached reference data);

• 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

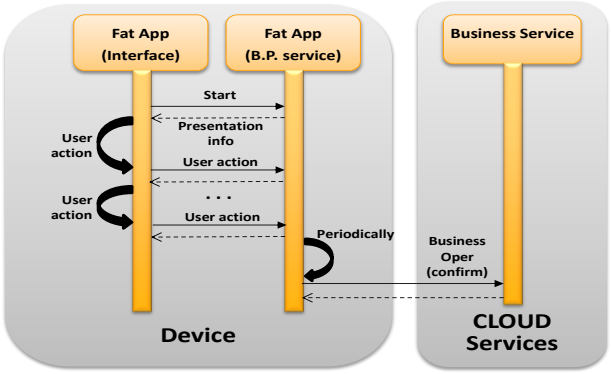

Figure 6. Fat client workflow.

A. *Thin Device Development*

Our development is based on an Android device with the following characteristics: (1) usb host; size bigger than 4,1’’; communication 3G or 4G; android version 4.0 or upper; NFC (near field communication), two USB with dedicated power. Our first development step was the communication interface between the reader (terminal) of RFID chip with the standard ISO14443 [24], which consists of: (1) physical characteristics; (2) radio frequency interface power and signal interface; (3) initialization and anti-collision; and (4) transmission protocol.

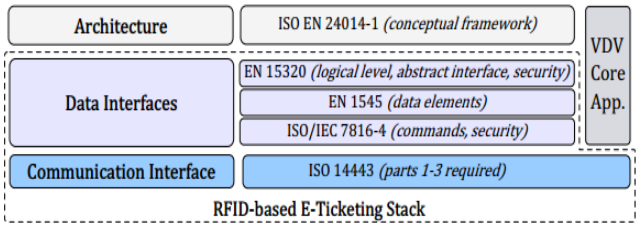

Figure 7. Standards to implement thin device concept.

We transfer most of adopted standards available in Calypso e-ticketing system [25] adopted in several countries such as, Belgium, Canada, China, France, Israel, Italy, Portugal, etc.). Figure 7 illustrates our standards for this interoperation necessary for thin device concept implementation.

The standard, ISO EN 24014-1 [26] introduces a conceptual framework for developing an interoperable architecture for transport fare management systems. It describes the structure of an interoperable platform, its main actors, and general flows of information exchange. Privacy is considered at a conceptual level by requiring the definition of a security scheme that should provide privacy.

The data interface layer, EN 15320, defines the logic data structure in the card and defines the communication interface between the card and the terminal. The card data interface and data group interface handle security based in the specification of the security subsystem (SSS), illustrated in

Figure 8. Security related operations are defined in the card profile and data group profiles (this to handle privacy issues).

The ISO/IEC 7816-4 defines the commands exchange as well as the retrieval of data structures and data objects in the cards. Security is taken into account specifying the methods for secure messaging and security architecture defines access rights to files and data in the card. A list of algorithms is available in [27].

Communication interface layer is based on ISO 14443 [28], which handles the connection of terminal to the card reader. The communication between terminal and the cloud is based on our work described by [2].

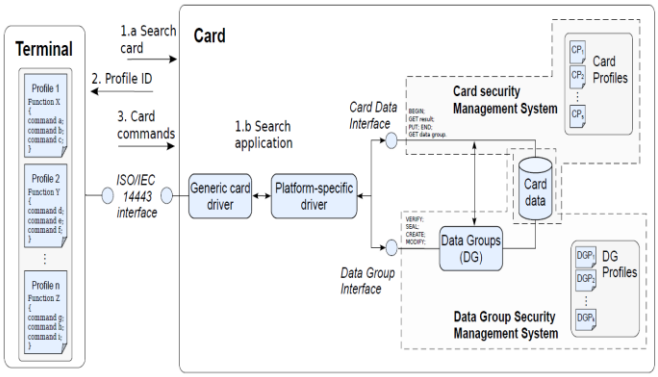

Figure 8. Interaction process between card and terminal.

V. MULTI-TENANCY

Multi-tenancy, which lets multiple tenants (users) share a single application instance securely, is a key enabler for building such a middleware. The main idea is to apply this concept to terminals devices adopting the thin device concept to consolidate globally the available operations in ticketing back offices systems. Multi-tenancy has been proposed as a way to achieve better resource sharing and to provide almost zero cost when unused.

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 [2]. 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 target scenario, we propose a hierarchy of tenants with multiple roots (see Figure 9). 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:

• Lower level tenants (operators) can view information about their private customers, as well as business information common to the metropolitan area.

• Upper level tenants can read and consolidate common business information to the lower level tenants (e.g., customers, cards, sales and validations).

• Upper level tenant may not see information about private customers, and sales/validations of private tickets.

Here, we discuss the option of having a shared database or separate database/schema implementation of multi-tenancy.

The main concerns with this decision were privacy, security and extensibility. It is necessary to avoid risks of having one operator accessing information belonging to 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?

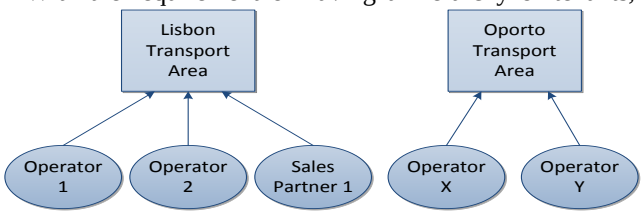

Figure 9. Sample hierarchical structure of tenants.

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.

VI. DEVICE PROVISIONING

To enable device installation automation, we propose a device provisioning model, where devices can detect and download the relevant software to support the functionalities assigned to them. This procedure depends on cloud platform and device operating system. This is a working project, where we are starting our approach with a Windows Azure platform and an Android operating system.

In this model, we assume that devices have some generic bootstrap software pre-installed, that asks the cloud service to provide it with the device specific software it needs. Either this bootstrap could be factory installed or user installed, however it does not bring any device specific configurations. When the bootstrap starts (Figure 10), it sends a bootstrap command to the cloud, requesting information on what it needs to install, and receives back information about the download location and metadata. Next, it downloads the software and configurations and starts the newly installed software.

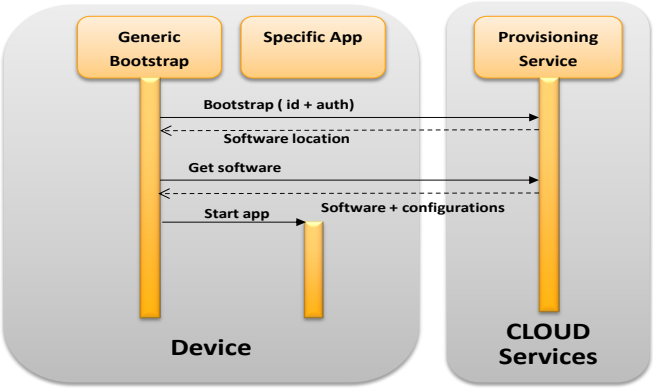

Figure 10. Provisioning workflow.

VII. CONCLUSION

The proposed system uses a novel approach based on SaaS model for the development of a personalized ticketing software. This project started 12 months ago in a synergy between the technology company Link Consulting [29], with 10 years of experience in ticketing business and researchers in computer science at ISEL [30]. 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. This topic not covered in this paper handles the problems of losing connectivity and the security issues; (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.

The architecture described in this paper illustrates the adoption of the thin device concept within our cloud ticketing approach. In current work stage we have an Android platform, where we 'migrate' current process of a selling ticketing operation and validation process at a gate using calypso platform [2]. Part of this research effort belongs to Link company in the scope of the "SmartCITIES Cloud Ticketing" project [28], which is focused on designing an interoperable, cost-efficient, multi-supplier, cloud based ticketing solution, where transport operators may opt in and out when they need/want to.

This project brings together two complementary sets of experiences: the engineering experience applied to ticketing solutions of Link Consulting [28] and the computer science and research experience of ISEL – Instituto Superior de Engenharia de Lisboa, which lays out the path to a solid foundation of a cloud ticketing solution.

REFERENCES

[1] "Impact of changes in the airline ticket distribution industry", www.gao.gov/assets/240/239237.pdf, [retrieved: April, 2013].

[2] J. C. Ferreira, P. Filipe, C. Gomes, G. Cunha, and J. Silva, "Taas – ticketing as a service," in proc. of CLOSER 2013 - 3rd Int. Conf. on Cloud Computing, 8-10 May, Aachen-Germany.

[3] A. Benlian, T. Hess, and P. Buxmann, "Drivers of SaaS-adoption–an empirical study of different application types," in Business & Information Systems Engineering 1.5, 2009, pp. 357-369.

[4] M. Mut-Puigserver, M. M. Payeras-Capellà, and J. L. Ferrer-Gomila, A. Vives-Guasch, and J. Castellà-Roca, "A survey of electronic ticketing applied to transport," in Computers & Security, Volume 31, Issue 8, November 2012, pp 925-939.

[5] E.A. V. Vilanova, R. Endsuleit , J. Calmet, and I. Bericht, "State of the art in electronic ticketing," Universität Karlsruhe, Fakultät für Informatik.

[6] TCRP Report 115, "Smartcard interoperability issues for the transit industry," in Transit Cooperative Research Program, www.nap.edu/openbook.php?record_id=14012, [retrieved: April, 2013].

[7] ITSO Technical Specification 1000, "Interoperable public transport ticketing using contactless smart customer media. Version V2.1.4," http://www.itso.org.uk//Home/Itso-Specification, 2010, [retrieved: April, 2013].

[8] Frederic Levy, "Calypso functional presentation. SAM and key management," www.calypsostandard.net/ index.php?option=com, [retrieved: April, 2013].

[9] "CubicCoorporation", www.cubic.com, [retrieved: April, 2013].

[10] Octopus Cards Ltd, "The Octopus Project, 1994," www.octopuscards.com/e index.html, [retrieved: April, 2013].

[11] "An overview over ticketing projects," http://www.tick-et-portal.de/, [retrieved: April, 2013].

[12] ERG Group, www.erggroup.com, [retrieved: April, 2013].

[13] "Sony felica card," http://www.sony.co.jp/en/Products/felica/ contents02.html, [retrieved: April, 2013].

[14] "Philips semiconductors." www.semiconductors.philips.com, [retrieved: April, 2013].

[15] "Metrebus card," http://europeforvisitors.com/rome/ transportation/rome-metrebus-tickets-and-fares.htm, [retrieved: April, 2013].

[16] "TransLink," www.translink.co.uk/, [retrieved: April, 2013].

[17] "Régie autonome des transport parisiens," France, ICARE, 1998. Philippe Vappereau, Project Coordinator. http://www.cordis.lu/telematics/taptransport/research/ projects/icare.html, [retrieved: April, 2013].

[18] "Calypso standard for card terminal ticketing," http://www.calypso.tm.fr, [retrieved: April, 2013].

[19] "ASK," France. http://www.ask.fr, [retrieved: April, 2013].

[20] M. Fowler, "Public versus published interfaces," in IEEE Software, Vol. 19, No. 2, March/April 2002, pp. 18-19.

[21] A. Hunt and D. Thomas, "Software archaeology," in IEEE Software, Vol. 19, No. 2, March/April 2002, pp. 20-22.

[22] Smart Card Alliance, "Transit and contactless financial payments: new opportunities for collaboration and convergence," http://www.smartcardalliance.org/resources/pdf/Transit_Financial_Linkages_WP_102006.pdf, [retrieved: April, 2013].

[23] "RFID," http://en.wikipedia.org/wiki/Radio-frequency_identification, [retrieved: April, 2013].

[24] ISO 14443 Standards family, "Identication cards - contactless integrated circuit cards - Proximity cards," www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39693, [retrieved: April, 2013].

[25] ISO/IEC 7816-4:2005, "Identication cards - integrated circuit cards - part 4: organization, security and commands for interchange," - http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=36134, [retrieved: April, 2013].

[26] Calypso Networks Association, "Calypso handbook," http://www.calypsonet-asso.org/downloads/100324-CalypsoHandbook-11.pdf, [retrieved: April, 2013].

[27] GlobalPlatform's Value Proposition for the Public Transportation Industry, "Seamless, secure travel throughout multiple transportation networks," http://www.globalplatform.org/documents/whitepapers/GP_Value_Proposition_for_Public_Transportation_whitepaper.pdf, [retrieved: April, 2013].

[28] "ISO14443," www.openpcd.org/ISO14443, [retrieved: April, 2013].

[29] "SMARTCITIES projet", http://www.link.pt/smartcities, [retrieved: April, 2013].

[30] "Polytechnic Institute in Lisbon", ISEL – www.isel.pt, [retrieved: April, 2013].