Journal of Information Systems Engineering & Management, 1:2 (2016), 123-140

ISSN: 2468-2071

Banking desmaterialization using cloud computing

Joana Silva[1*], Cláudio Teixeira[1] & Joaquim Sousa Pinto[1]
[1]*Universidade de Aveiro, PORTUGAL*

ABSTRACT

The technological challenges have led the bank sector to actively bet on technological innovation strategy including the provision of applications for mobile devices. However, the development of mobile tools has had a greater expression for customers than for commercial managers of banks. This paper aims to present an architecture that supports the desmaterialization of transactions at the counters of bank branches. The main goal is to foster the relationship between banks and their customers, giving flexibility to managers to conduct banking transactions from any mobile device, anywhere. This architecture is a results of a dissertation carried out under a master's degree in Information Systems.

Keywords

bank system,
desmaterialization,
architecture,
cloud,
hybrid,
mobile

Received: 28 Jan 2016
Revised: 16 Feb 2016
Accepted: 10 Mar 2016

DOI: 10.20897/lectito.201618

INTRODUCTION

Motivation

Nowadays, in financial institutions, the information systems that make up the infrastructure are designated core processes. These processes are responsible for the transactions and financial calculations culminating in deposits, deposits updates and investments. The banks, as we met, are arranged in a network of physical branches spread over different points. It is at the branches of service that customers deal with much of the financial transactions, including cash deposits or checks, open accounts, purchase of credits among or others.

National and international banking bodies have wagered over the past years in electronic solutions for their customers, giving them the freedom to manage and conduct their own movements, namely home banking. Thereby, now the banks offer more than one channel of interaction with customers: home banking, call centres, ATM, portals for partners and sales agents are common channels of communication with the core bank systems.

After the release of these home banking systems, many banking institutions continued to develop services that allow them to be ever closer to its customers. The next step was to make these services available on mobile devices allowing their customers to perform minor operations such as bank transfers, payments and services among others. All are done through their mobile devices, called mobile banking. Juniper Research estimates that in 2019 will be about 1.75 billion users to use this type of applications. Today, the figures point to about 800 million people who resort to these services (Notícias ao Minuto, 2014). These figures result from the massification of smartphones and tablets as well as the use of Internet as a privileged channel to allow remote access to the banks systems information.

Nevertheless, the development of home banking tools has had bigger impact for the end user than for Client Managers. This constraint has to do with the fact that the banking processes are based on a monolithic

*Correspondence to: s.joana@ua.pt

structure that hampers the streamlining of them, as well as the mapping for applications on mobile devices. i.e., for the processes realization account managers need access to core systems and the application of these procedures is made outside of the financial institution network, this raises security issues in network access. Thus, the client manager is limited in relation to local and ways to access data outside the institution.

The mobility indicated by tablets is not revolutionary compared to what already is presented to us by today's laptops; however, they provide us a different mobility, easily transportable, easy to use and intuitive.

It is the relationship between banks and their customers, including through client managers, to be achieved with the Tablet Ecosystem Apps (TEA). The solution aims to provide the banks' commercial management teams a mobile solution, agile and intuitive in with the counter processes are to be dematerialized.

This architecture proposal is the result of a master's dissertation in Information Systems and a paper called "*Support Architecture to Desmaterialization of Banking Operations*" (Silva, Teixeira, & Pinto, 2015).

Main goals

As noted above, there are already many banking applications for end customers, however, the commercial management teams are still limited to banking institutions to carry out operations. Thus, the main goal of this paper was the design of a reference architecture that allows the desmaterialization of the counters of bank operations. Despite being already possible to perform a wide range of transactions through mobile devices, these are only considered small operations. The vast majority of transactions still require a customer's visit to a physical counter, i.e. can only be made in person (King, Branch Today, Gone Tomorrow, 2013) (King, Bank 3.0: Why Banking Is No Longer Somewhere You Go But Something You Do, 2012). For example, opening an account is not yet possible through a mobile application neither is depositing money. It is this type of operations that the TEA plans to dematerialize. Traditional communication channels between the client and the bank are: face contact, call center, ATM and Internet. It's intended that the TEA becomes a new communication channel.

For developed countries, it may not be easy to understand this concept because there are many and various banking institutions with several scattered branches covering most of the population. However, to the African countries of Portuguese Official Language (PALOP) this concept will certainly have a great market penetration.

We have witnessed an increasing number of users, who are also the result of population growth and use the internet from their devices, whether they are computers or mobile devices. In Table 1 we can see how has grown significantly the number of users who use the internet, both in developed countries and in developing countries. (International Telecommunications Unions (ITU), 2013)

Through the analysis of Table 1 we can see that the increased use of the Internet by developing countries was of 23%. This increased access to the Internet reflects the increasing need that the user feels to be connected to the network. With the increasing network access also come the need and the convenience of using digital services. With these services, also comes access to banking through digital means as a necessity.

According to the annual report of the African Economic Outlook of the Organization for Economy, Cooperation and Development, the African Development Bank and the United Nations Development Programme, there is a growth forecast of around 4.5% in the African economy in 2015. (Stevis, 2015) These data reflect an increased customer access to banking infrastructure that at an increasing rate can pose difficulties in the physical counters to meet the demand of customers. Will be needed as physical infrastructure to respond this problem, however, physical construction of counters which is not fast enough to give efficient response. Thus arises the need to accommodate alternative channels to the physical counters which could mean the creation of a new mechanism of communication between the bank and the public. By mechanism, we can understand virtual solutions to access data which can be used outside of the physical infrastructure of the same.

It is expected that this new communication channel to be crucial to the sustainability and growth of banking in Emerging Markets. We have witnessed the banking services reaching all the population; however,

Table 1. Internet users

	2005	2010	2013
World Population	6.5 quadrillion	6.9 quadrillion	7.1 quadrillion
Not use the Internet	84%	70%	61%
Use the Internet	16%	30%	39%
Users in developing countries	8%	21%	31%
Users in developed countries	51%	67%	77%

the growth of physical counters cannot follow up this need, as well as the training of people to support the banking (Ribeirinho & Silva, 2012) (Kshetri & Acharya, 2012). The mobile services, on the other hand, have a high penetration rate in developing countries. For example we can appoint the project of Safaricom M-Pesa in Kenya, where the service / mobile communications infrastructure is used to make payments and transfers (Buku & Meredith, 2013). Most emerging countries have already infrastructure such as 3G or Wi-Fi that allow the use of mobile technology.

Thus, the TEA plans to respond to the identified issues, enhancing personal contact, as it will be the bank to go to the client and not the opposite, changing the concept from bank centered to client centered, as is already happening in several services such as health. Because it's a mobile solution in the hands of account manager, you can also actively contribute to the banking Emerging Countries seeing as it has a faster penetration rate than the construction of physical counters and all associated infrastructure. The aim is to provide processes that currently can only be found in physical branches.

Throughout this paper there will be described some case service studies designed to support the desmaterialization of banking processes. An architecture draft that supports the desmaterialization of banking transactions via mobile devices will also be presented.

Contribution

This paper main contribution is linked with the setting of a reference architecture, in the banking context, for the inclusion of aspects of access to information using mobile devices, virtualization of counters, virtualization of processes and relocation of banking transactions (computing and data storage).

Paper structure

This paper is divided into five chapters, the first being an introduction. In the second chapter there will describe some case studies of services that have already taken the first steps towards desmaterialization operations. The third chapter presents the concept of cloud computing and shows a comparison between some cloud providers, in order to justify the choice made. In the fourth chapter we have the system description and what it is intended to accomplish as well as presented and explained the proposed architecture. Finally, in chapter five, the final considerations will be presented.

STATE OF ART

The defining and implementing software process is sometimes a difficult process to define. This difficulty in defining and the constant technological evolution we are witnessing in information technologies, leads several financial institutions to develop their own internal systems. In most cases the developed systems become isolated, leading to information silos that evolved separately, but that don't always respond to different needs. Needs, which are for example, the sharing of information between the various systems of financial institutions. Consequently, there is a need to maintain and support systems, which after a long time of use can hardly be discontinued. Yet there is a major flaw in the culture of interoperability.

Some solutions intended to respond to this problem have been found, however, there wasn't found any information concerning the bonding, in the same architecture of security aspects, communication, local storage and access systems with an integrated layer, as is the purpose of this paper.

A reference architecture definition intends to respond to the problem in question. The use of SaaS (Software as a Service) in the bank has been run as a way for banks themselves to solve some of their problems such as the reduction of human resources, streamlining processes and also the provision of the aforementioned services home banking. An example of this is the IDEALINVEST offering B-SaaS services with some predefined banking processes and with interface connections to banks' systems (B-SaaS, 2012). This has become an area of great activity and is increasingly attracting the attention of companies who consider themselves advocates of the traditional model, as demonstrated by the TriNovus acquisition by the Temenos company (Camhi, 2013). Moreover the BTR, together with VASCO Data Security, KBC and the financial services section of Oracle launched a specific SaaS platform for retail back office services, covering current and time deposits, payments, etc. (Services, BTR, 2013).

The CPay 360 propose a SaaS solution for community banks (Cpay360, 2013). With this solution their clients can compete (in terms of features presented to customers) with large multinational banking. CSC (CSC, 2013) also presented the EarlyResolution (SaaS) capable of dealing with lending processes. Most of the solutions presented address the SaaS platform as Web interfaces generator / terminal for bank employees (counter) (Cloud Computing for Financial Markets, 2011) to manage the presence of the web bank with

home banking system. In (NCR APTRA, s.d.) it's presented the NCR APTRA Mobile BankingGateway. It's a set of middleware and mobile solution based on mFoundry (mFoundry, s.d.). One of the interesting aspects of APTRA is its presence in three distinct channels: native application, Web application and communication via SMS.

Despite all the above solutions, the lack of architectures that serve as a reference guide to support the needs and regulatory requirements imposed by the financial sector has fostered the growth and evolution of specific and applicable solutions in just about every financial institution: the development of these application systems is characterized by unique requirements in each country/institution according to how each interprets the generic standards published by the Regulatory Authorities. Thus, there is a low level of reuse and of accelerators that promote such development tasks.

The global challenges are forcing financial organizations to seek a new technological basis for business. The combination of Service-Oriented Architecture (SOA) and SaaS allows organizations the greatest possible scalability at a low cost and a quick change to SaaS. The SaaS combined with a repository of SOA components provides the ability to align technology with business. Monolithic on premise applications are no longer viable to provide products and services in highly dynamic environments (ETNA Software, 2011).

Given the examples mentioned above, we can easily conclude that almost all are based on SaaS solutions to get closer to its customers' needs. Moreover, solutions like NCR APTRA have three communication channels with their customers in addition to the middleware and the mobile solution based on the mFoundry. Table 2 presented below, shows a small comparison between the solutions and what they offer.

Table 2. Comparative table between existing solutions

	IDEALINVEST	BTR	CPAY360	CSC	NCR APTRA
Home Banking				X	
SaaS	X	X	X	X	
Legacy Systems	X				
Backoffice		X			
Middleware					X

Through the analysis of Table 2 we can conclude that with the exception of NCR APTRA, all other solutions rely on SaaS. We can also see that the BTR is the only one that offers a backoffice and the NCR APTRA is the only one that provides middleware. The IDEALINVEST is also the only one that has a connection to legacy systems.

Despite all the already shown solutions that present good progress towards using SaaS, none of them can respond to the problem. The solution that comes closest is the IDEALINVEST using a SaaS scenario and it still has connection with legacy systems. Yet it does offer solutions for the materialization operations in mobile environments, communication and local storage, as stated.

There is a clear lack of an architecture that serves as a reference guide to support the needs and regulatory requirements imposed by the financial sector. This gap has fostered the growth and development of specific solutions for each financial institution. Thus, there is a low level of reuse and accelerators to promote such development tasks.

A Service-Oriented Architecture (SOA) and SaaS combination provides the highest scalability possible at a low cost, and allows a more rapid change capability. SaaS combined with a repository of SOA component confers the ability to align technology with business. The SaaS and SOA junction enables banks to redesign as fast as possible business processes (ETNA Software, 2011). As such, and given the potential level required in using the TEA it would be convenient to adopt a SaaS model.

CLOUD COMPUTING

According to the National Institute of Standards and Technology (NIST), the "Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. " (Mell & Grance, 2011).

The cloud model promotes availability and is composed of five essential characteristics: on-demand services, network access, resource pooling, rapid elasticity, and measured service. Key enabling technologies include: large networks, powerful low-cost servers and high-performance virtualization (Mell & Grance, 2011).

Typically, the applications we use are installed on our own devices. In business environments, though, are more often used applications available on servers, which are then used by any device within the network.

However, thanks to changes in technology and telecommunications, the internet is becoming more comprehensive and faster. There are already many countries where you can access the Internet from any location and at very low prices (Alecrim, 2013). It's thanks to this evolution and coverage that the concept of cloud computing is increasingly present and is increasingly popular.

Cloud computing allows the use of applications and data files that are not installed in our devices. Content is available through the internet. The development process, storage and maintenance lies with application vendors, the customer only has the task of using these applications (Alecrim, 2013).

Characteristics - advantages and disadvantages

As mentioned earlier, one of the great advantages of the cloud applications is that they are accessible anywhere via internet; however there are many other advantages. The fact that it can run on, any operating system and is therefore independent of the system or of the hardware. The customer also, doesn't need to worry about the applications structure and inner workings, as these are borne of the service provider. There is the option to scale services in a flexible way according to the ever changing customer needs. Access price depends on what you use. (Teixeira, 2011).

Regarding the disadvantages, the main one is the vendor lock-in problem. For more than a decade that efforts have been done to achieve solutions based on common standards and protocols that can be built, supported and replaced regardless of their supplier (McKendrick J. , 2011). Nevertheless, this concept has regressed when it comes to cloud computing. By using a solution based on cloud we are using protocols, standards and tools from a particular vendor, making a future migration a costly and difficult task. It's usual to mainly care about to the services costs, and not take into account the costs of a possible migration (McKendrick J. , 2011).

In deciding to use cloud solutions, it's necessary to take into account the associated costs. You need to make a careful analysis of what is involved and what are their costs.

Comparison between Cloud Service Providers

Currently there is already a wide market of cloud service providers, and as it will be necessary to choose a cloud solution for the architecture the study of Nasuni was analyzed – *The State of Cloud Storage - A Benchmark Comparisons of Performance*, *Availability and Scalability* (Nasuni, 2013). The choice of this study lies in the fact it has credibility and offers a very organized a set of tests and their respective results. This study used comparative metrics and conducted tests in order to understand the performance of some cloud service providers.

Comparison Metrics

In this study, Nasuni defined three comparison metrics between different cloud service providers:

Functionality: Usually, the vast majority of interactions a company has with the cloud service providers consist of simple API commands like GET, PUT, and DELETE. However, companies should consider a wider range of features when it comes to comparing vendors. Since most companies are global and are all over the world, it's essential that suppliers have servers available in various parts of the world and that support geographic cross-replication. It is also important to consider features like the process of creating and account management, availability of libraries, software for accessing data and billing.

Price: Storage cloud is different from traditional storage, which is reflected in the way in which it is charged. Traditionally, storage is charged TB, however, most suppliers charge on the basis of stored GB per month. However, this score is not as linear as they are often added computing costs and network costs.

Nowadays, providers offer ways to help calculate costs according to the needs of users and predict costs in case of service expansion of.

The services price is a very small part of this comparison and should be used as a last resource to make a decision because the functionality and performance are quite significant indicators.

Performance: It is mainly through performance that Nasuni compares the different suppliers, testing the functioning and stability for long periods of time. Let's meet the minimum performance benchmark in these three areas:

- Read / Write / Elimination: the way how suppliers deal with thousands of reads, writes and deletes was tested. Files with various sizes were tested using different levels of competition. The test ran for 12

hours, using multiple instances of machines and various non serial testing to reduce the likelihood of external network problems that could distort the results.

- Availability: availability tests took place in a period of 30 days and measured each provider response time of a single reading process, writing and eliminating at 60-second intervals. Read and delete files randomly force the suppliers to prove their ability to be sensitive to all the data, all the time, not just for the data that is stored in cache. This test calculated time required to perform all three orders. It is not just about the responsiveness but also the confidence in the supplier and latency.
- Scalability: similar to the availability tests, these measured the ability to perform consistently the number of objects and management increases. This test measured the ability of each supplier to maintain performance levels as the total number of objects stored in a single storage for millions of people.

Methodology

The study of Nasuni includes five cloud service providers: Amazon S3, Microsoft Azure Blob Storage, Google Cloud Storage, HP and Rackspace (Nasuni, 2013).

In this study were used two virtual machines with the features that can be observed in Table 3.

The size distribution of the used files can be seen in Table 4.

Results

a) Benchmark – writing

In the written test, as can be seen in Figure 1, Microsoft leads the writing performance. Furthermore, Microsoft outperformed all other providers in 14 of the 23 individual combinations tested, making it an ideal target for writing data in file format. With writing on files larger than 1MB, as can be seen in Figure 2 Amazon stands out.

b) Benchmark –Reading

In the read test, once more Microsoft leads over other suppliers, both reading files of all sizes and in reading files larger than 1MB, as can be seen in Figure 3 and Figure 4

Table 3. Features of VMs

	Machine 1	Machine 2
RAM	15-16 GB	4 GB
vCPUs	4	2
Operation Systems	Ubuntu 1204, 64 bit – Ubuntu 12.04 LTS	Ubuntu 1204, 64 bit – Ubuntu 12.04 LTS

Table 4. Size of files

1KB	10KB	100KB	1MB	10MB	100MB	1GB
16.8%	24.6%	26.2%	9.7%	22.2%	0.4%	0.1%

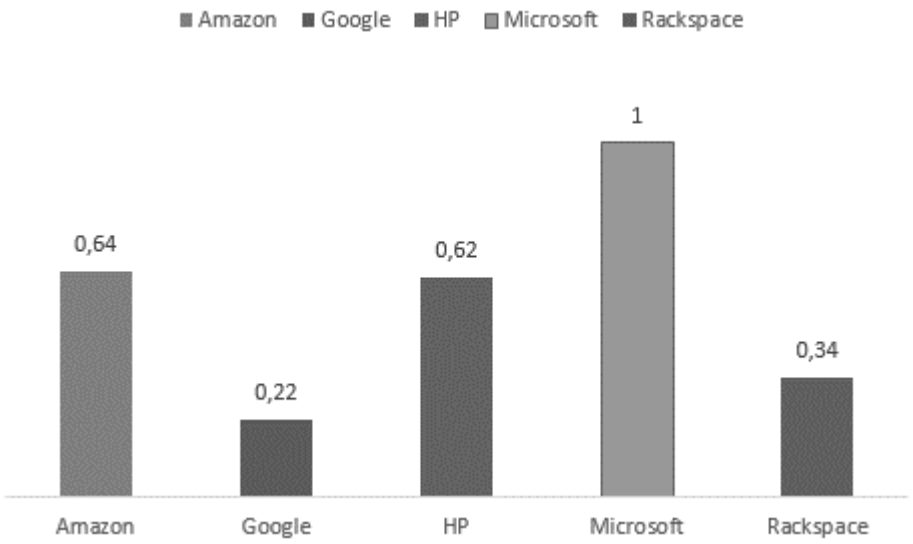

Figure 1. Writing speed storage of all sizes (Nasuni, 2013)

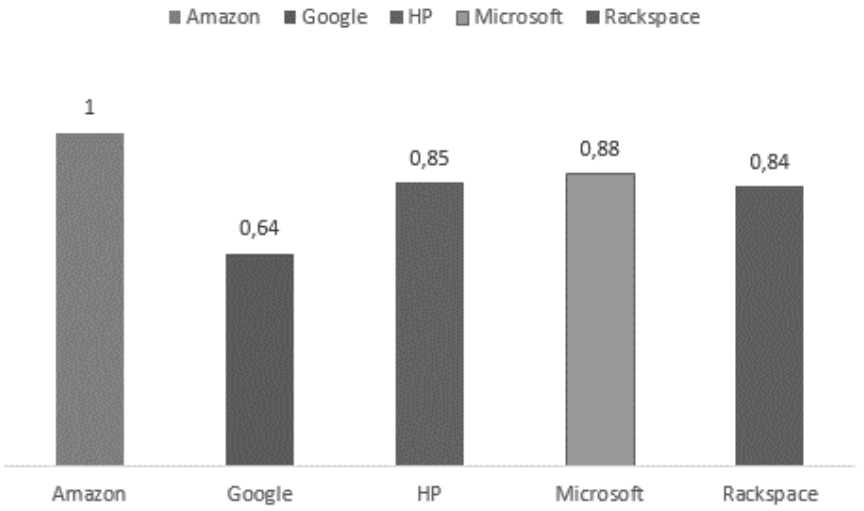

Figure 2. File writing speed storage size > 1MB (Nasuni, 2013)

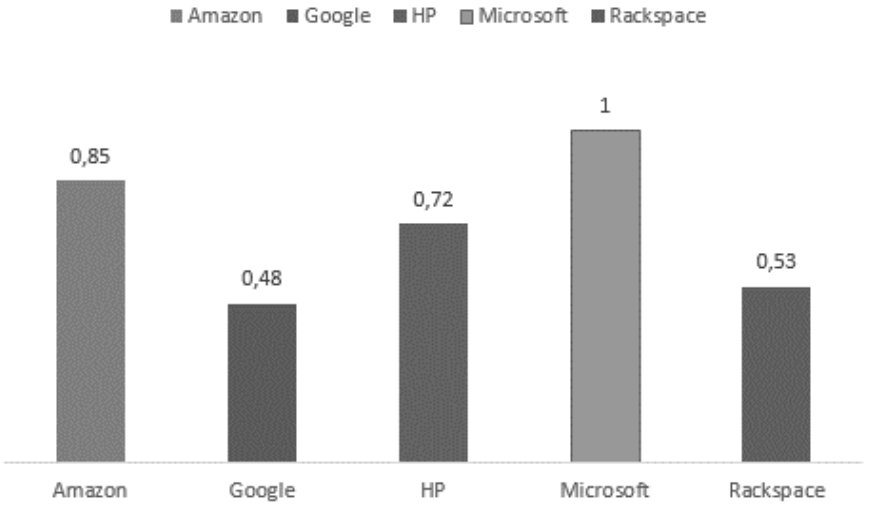

Figure 3. Reading speed storage of all sizes (Nasuni, 2013)

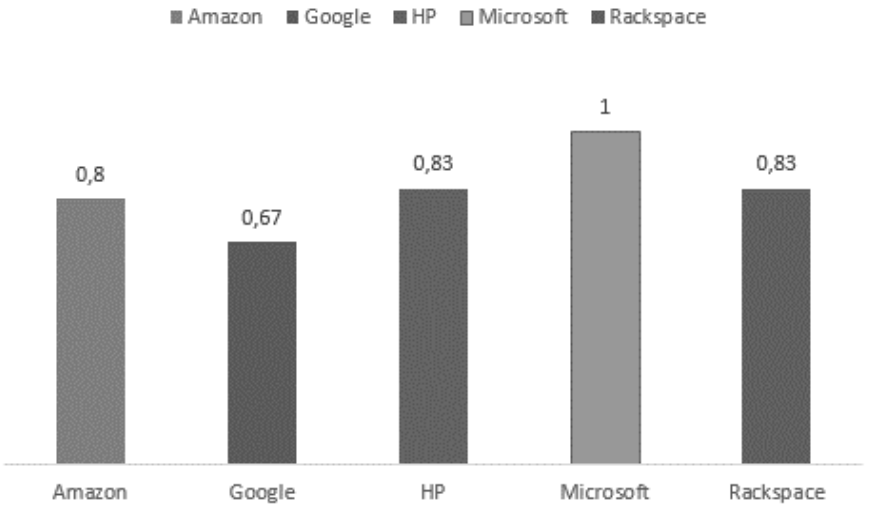

Figure 4. Reading speed storage filesize > 1MB (Nasuni, 2013)

c) Benchmark – Deletion

Confirming the results in the other tests, Microsoft is twice more fast in removing files than any other supplier. Secondly we find Amazon and at a very low performance found Rackspace, as can be seen in Figure 5.

d) Availability

Availability was measured using response time as a metric, which measures the response time of each provider in a single reading, writing or deleting process at 60-second intervals. The response time also includes moments associated with delays and attempts.

Again, Microsoft leads the test with an average of less than 0.5 second response during a period of 30 days. Amazon is close, with a score of 0.65 seconds. Google has the worst response time with almost 2 seconds. These results can be seen in Figure 6.

If we analyze the results in Figure 7 over the month, we can also check the variability of numbers. Microsoft shows some stability, unlike the others that show greater variation in the results.

In addition to the availability of data and the system, the test also measured the uptime or the percentage of time in which the provider is available. All showed strong percentages with Amazon and Google leading, as can be seen in Figure 8.

e) Scalability

When the number of objects increases, the performance of some suppliers degrades or becomes variable. Depending on each provided architecture, some systems are designed to climb through containers, not within

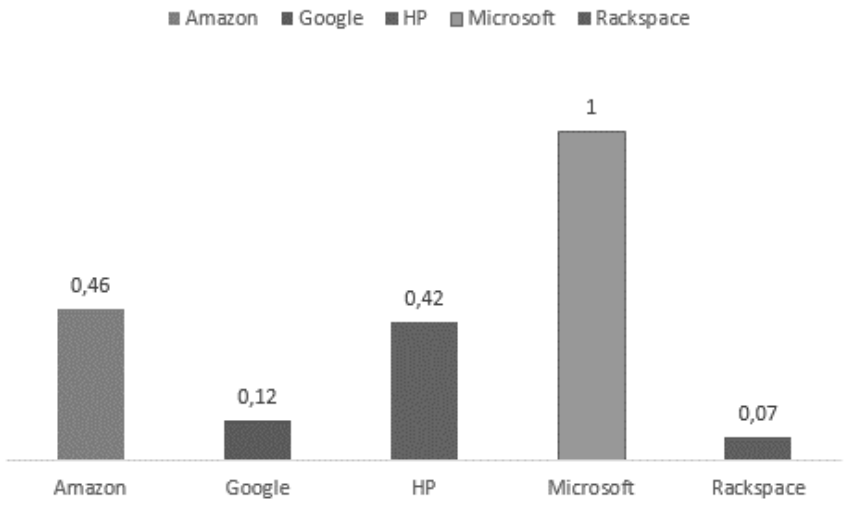

Figure 5. Elimination rate (Nasuni, 2013)

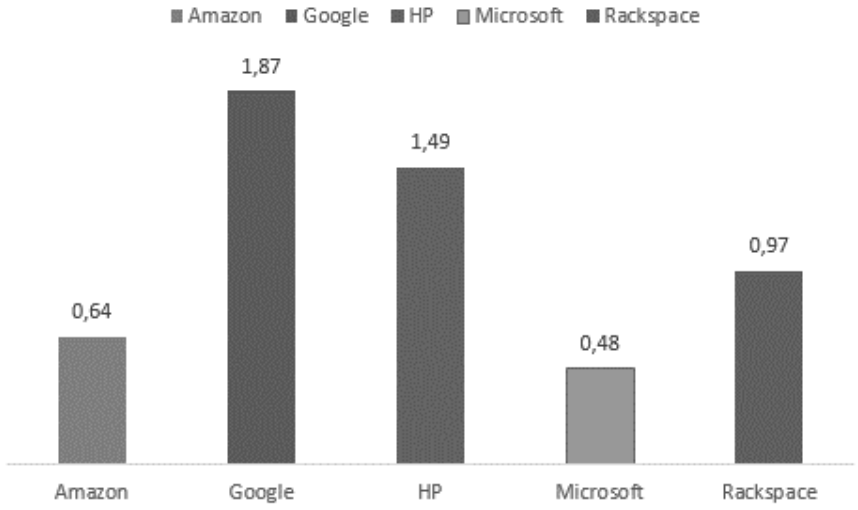

Figure 6. Average response time (in seconds) (Nasuni, 2013)

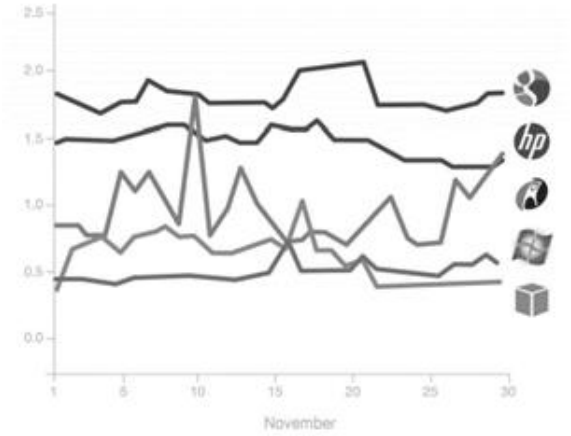

Figure 7. Daily average response time (in seconds) (Nasuni, 2013)

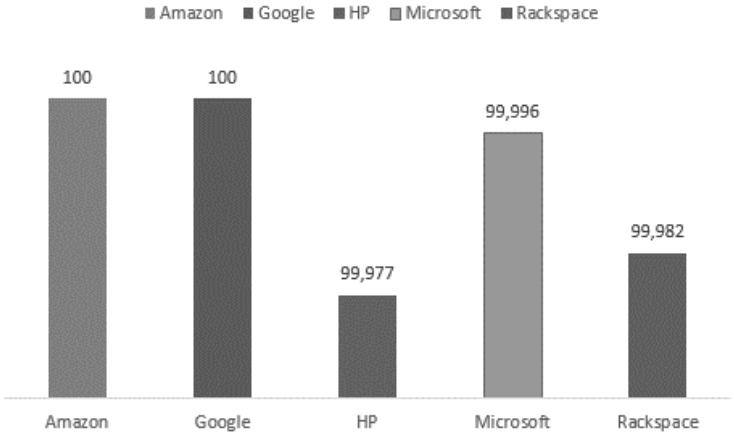

Figure 8. Average time to run (Nasuni, 2013)

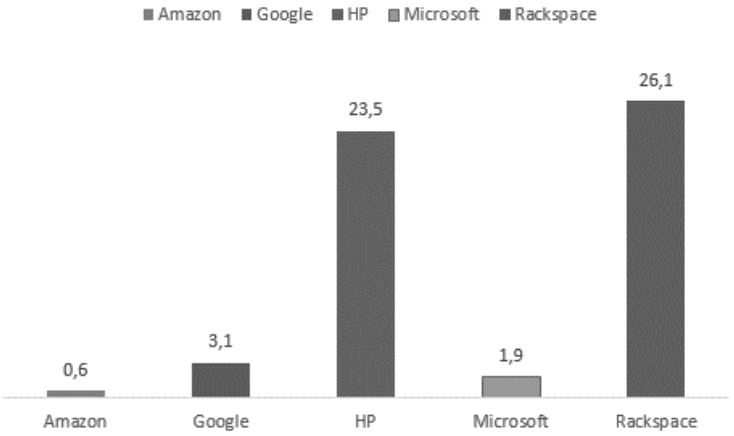

Figure 9. Variance for the scheduling of new objects (% change in writing speed) (Nasuni, 2013)

them. This type of architecture has limitations that are likely to become significant after a few months or even years of use.

In this test, all suppliers were loaded with new objects as quickly as possible up to 100 million objects or 30 days. The variation represents the object's loading speed over time, this variation caused inconsistencies and variability in how the objects are loaded, which can be seen in Figure 9.

Compared to the last results, writing and reading errors during the scalability decrease significantly. For 100 million written, only HP and Rackspace showed clerical errors, as shown in Figure 10.

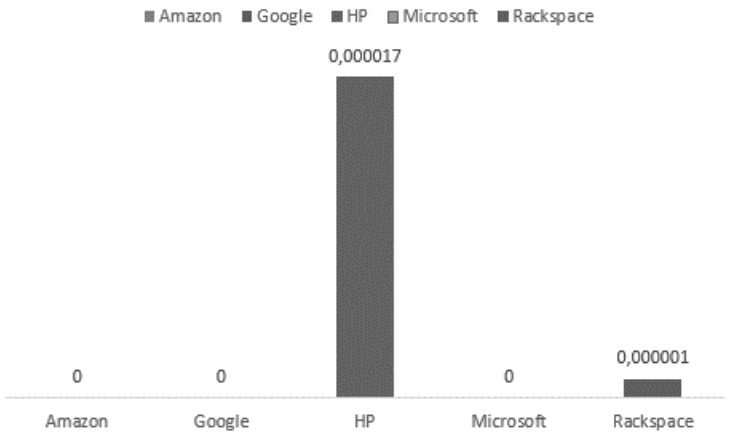

Figure 10. Percentage of writing errors (Nasuni, 2013)

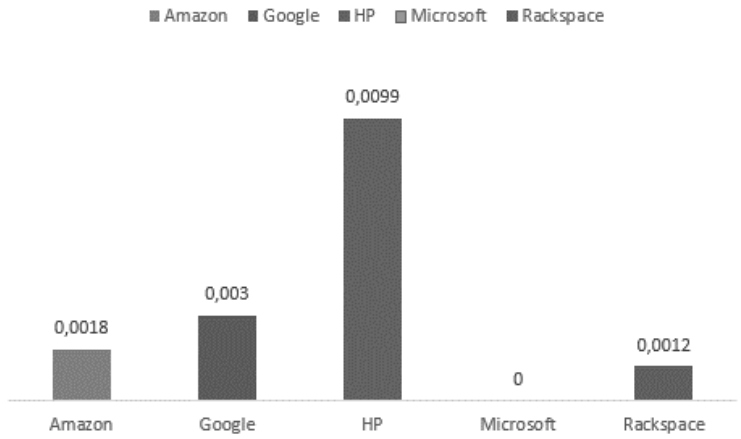

Figure 11. Percentage of reading errors (Nasuni, 2013)

During the readings, only Microsoft doesn't show any error. Amazon comes second as we can see in Figure 11.

f) Conclusion

Consistently, Microsoft showed better performance than any other provider based on our tests, according to the speed by varying the file size, faster response times and lowest error rate. For these reasons, on a storage level, Microsoft surpasses Amazon and achieves the top position in terms of performance of the year 2013.

Another study of Cloud Spectator - A Comparative Analysis of 5 Large Cloud IaaS Providers - highlights that the Azure infrastructure services achieved the highest performance and have an average of three times better price for performance than the Amazon EC2 (Cloud Spectator, 2013) (Microsoft Azure, s.d.).

Based on the studies presented, the choice of cloud service provider went to Microsoft.

Windows Azure

Windows Azure is a cloud platform from Microsoft. It's flexible and allows the creation, deployment and management of applications across a global data centers network, managed by Microsoft itself. Windows Azure allows you to use a range of programming languages that pass through .NET, Node.js, Java, PHP, Ruby and Python to build applications. Runs both Linux and Windows. Libraries are available for the various aforementioned languages and are available in open source and hosted on GitHub (Microsoft Azure, s.d.).

In addition to what has already been said, Windows Azure is scalable and therefore allows you to easily scale applications. It also allows the monitoring of resources and flexibly according to the needs (Microsoft Azure, s.d.).

Its main features pass through computing, web and mobile development, data and storage, analysis tools, network, multimedia and system integration (Microsoft Azure, s.d.).

In the next chapter the reference architecture designed based on the services provided by Windows Azure will be presented and defined.

SYSTEM DESCRIPTION

Before we start the architecture description and operation, it's first important to contextualize the goals for its operation. As has been outlined over the previous chapters, we intend to design an architecture that supports a mobile system that will help bankers in their operations and thus take these same tasks out of bank branches.

Nonetheless, the banking systems have already existing data which must somehow be considered architecture. In terms of access to existing data we can already see two different perspectives: that a full data migration should be allowed to the new structure or that there should provide the creation of new access points.

It is of course difficult for a banking service to isolate themselves from their components because for some these component data migration isn't possible. Therefor it was necessary to think how it would be possible to continue to allow the bank to communicate with on premise services and at the same time use cloud services that offer all the advantages described above. The adopted solution is a hybrid system that allows the two paradigms to work together.

Throughout this chapter there will be described the intended operation and which solution is thought to respond better to the required needs.

Project descriptions

The main goal involves the desmaterialization of operations that an account manager performs in a physical counter. With operations we mean opening bank accounts, banking transactions (eg withdrawals and deposits), credit applications, among other operations that are commonly performed only at a desk. The term dematerialize intends to bring these operations to the customer, in particular, take the bank manager, armed with a mobile device like a tablet, to the clients who need to perform these operations businesses and homes. Initially the concept may seem strange, but if we think about it, we easily realize that generally the services are more and more are closer the client, reducing the time customers have to spend to addressing bureaucratic calls. This was the motto that led to an attempt in transcribing these concepts for a mobile application. This application is not intended for a single financial institution but to a set of entities that wish to use it, supporting multiple institutions / organizations. This practice leads us to the concept of multi-tenant. A multi-tenant application is an application that shares resources but separates institutions, or "tenants". Thus, each user sees the application in their way. Some examples of this are Office 365 and Outlook.com. From the perspective of suppliers, the benefits go mainly for operational efficiency and cost. The same version of the application can handle as numerous customers all in the same system. A multi-tenant application must have some clear objectives and requirements including providing maintenance and monitoring. The benefits for users pass through isolation, availability, scalability and cost (Microsoft Azure).

The use of this concept also extends to how the application data is kept. There is the possibility of using the application with multiple databases; each institution has its own database, or the possibility of only having a single database for all institutions. The solution adopted was to have a single database for all banks, however the data are separated. A single instance can support the different entities as shown in Figure 12.

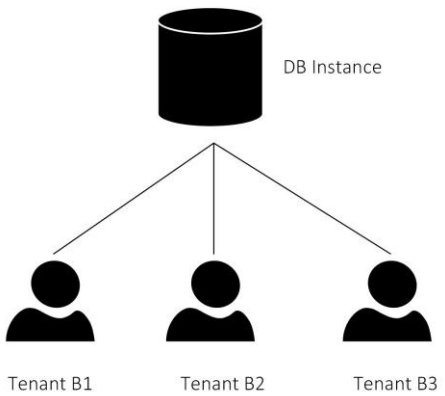

Figure 12. Single instance multi-tentant

Another concept widely used throughout this project is the concept of module. It is to be understood as a module that interconnects with the other modules, thus leading to a modularized architecture, these modules may or may not be in a SaaS environment.

Thus, each bank entity in the system consists of a set of modules, on which such modules may be cloud services or on premise services in with migration isn't possible or is difficult. These modules are set by the financial institution at the time of the first use of the system. We can understand these modules as being the entire customer management service, deposits management, among others. The financial institution has full freedom to configure these modules by connecting them, turning them off and adding new whenever it deems necessary.

Fundamentally, the architecture has to do with how these modules are managed and accessed by the application. When a bank is registered in the system, the data modules are configured, and the architecture relates to the interpretation of this configuration and the access to information wherever it may be, be it in SaaS or in on premise environment.

Modules and services used

It becomes necessary to describe the Azure services that were used to build the architecture in more detail. The used services go from storage features, mobile services to bus services. Then the services and its use will be present with more detail.

Storage - Azure SQL

For this particular case, a SQL Azure database was used. This makes it possible to, if necessary, configure, monitor and scale the database. The Azure SQL database is useful for business applications, cloud based services or hybrid solutions. Data sharing is possible between SQL databases or between a local instance of SQL Server and a SQL Azure database (Microsoft Azure, s.d.).

Given the characteristics presented by Azure SQL database the choice was due to the fact that it is a relational database, which allows communication with other SQL instances outside Azure and that has the flexibility to grow whenever necessary given that the banks manage large volumes of data.

Mobile Services

Mobile services are an Azure service which aims to build and host backbends for any mobile application, whether iOS, Android, Windows or HTML5. It allows the incorporation of push notifications, social integration with leading social networks and is highly scalable. The backend code can be C # or Node.js and allows Single Sign-On authentication.

Storage can be done by SQL, Oracle, SAP, MongoDB, among others. For this particular case the SQL storage was used. When the mobile service is created, it can be associated with the SQL database in Azure. The backend provides integrated support that allows remote applications to store and retrieve data from it - uses an OData format based on JSON - safely. Like all other services of the Azure, this also has the ability to be scaled when needed (Microsoft Azure, s.d.).

These services are fundamental to the architecture. Since this is an architecture for mobile devices, including tablets, these services enable the design of applications that run on any device, cross-platform, as intended, hence the choice to build using HTML5 and JavaScript. The backend is intuitive and in a simple way you can configure a set of functions. It allows you the setting of a custom API where you have HTTP methods such as GET, POST, PUT, PATCH and DELETE. Each function is defined for each method and in it what it will do. In addition to the many features that the mobile services provides, in this project is was primarily used for HTTP methods.

Service bus

The service bus is a cloud-based messaging system that allows you to connect applications, services and devices anywhere. With service bus you can connect applications running locally with applications running on Azure. The integration features such as SQL, storage with message bus ensures smooth operation under heavy load operations and variability over time. It also supports a range of standard protocols such as REST, AMQP, WS * and API.

The bus reroute feature allows the solving of communication problems between local applications and the outside world, as it allows local Web services to design public endpoints. Thus, systems can access these Web services that continue to run locally (Microsoft Azure, s.d.).

As indicated by its characteristics, this service was particularly important because it solves the communication problems between the built Web services and also allows communication with legacy systems that banks have and which they cannot turn off. Thus, the services are all published in public endpoints where they can be easily accessed.

Access Control Service

The access control service is an Azure service that offers an easy way for customer authentication in Web services and applications without having to add authentication logic to applications or services. The Access Control Service (ACS) provides the following services: (Microsoft Azure, s.d.)

- Integration with Windows Identity Foundation;
- Support to Web providers such as Microsoft, Google, Yahoo and Facebook;
- Support to Active Directory Federation Services
- Management portal that allows ACS access.

This service enables authentication services during use. Only with authentication can one access the services and receive the desired content.

The Service Bus uses a security model based on statements implemented through an access control service. The service needs to authenticate itself through the access control service and get a token with a statement in order to be able to open a channel on the Service Bus. When the service and the client are set to use the RelayAccessToken authentication type, the customer needs to acquire a control service access security token that contains a send statement (Send). By submitting the request to the Service Bus the client needs to include the token in the header section of the SOAP request. In turn, the Service Bus will validate the security token and then send the intended message (Salvatori, 2013).

Proposed architecture

With this service package it was possible to design architecture capable of supporting different utilization scenarios as well as different users simultaneously. For a better understanding of the architecture, that has been divided into three sets, allowing for a partial understanding. The first scenario relates to the obtaining of the modules of a particular financial institution, the second scenario aims to show how the access to the cloud modules is done, while the third and last scenario seeks to demonstrate how the access to the on premise local servers modules is done. Next, there will be presented in detail each of the scenarios.

Figure 13 illustrates the modules gathering process of a bank. After the manager authentication on the system, a request is made for the entity registered modules. Let's see the steps:

1. The application sends a request to the Windows Azure Mobile Service API through HTTPS. Through the invokeAPI method set by the Mobile Service calls the service. The service contains a set of methods;
2. These methods call SQL Stored Procedures stored in SQL Azure database. These calls are made through the mssql object, which lets you run Transact-SQL statements that call the stored procedures;
3. The stored procedures send the results requested;
4. The result returns to the application and is processed.

All processes related to bank modules and their respective configurations are entirely in the cloud, and not in local systems. This decision was made taking into account that since this is a first step and the most critical

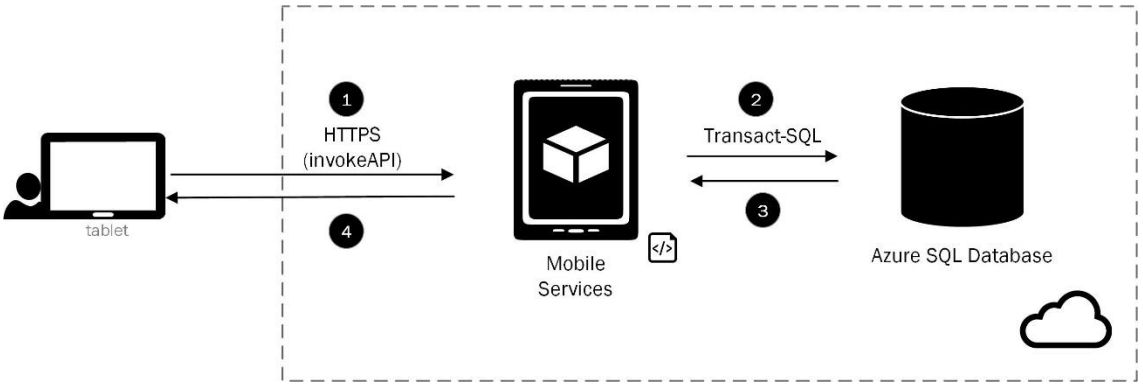

Figure 13. Obtain modules of a bank

to the system functioning, it is important that it is always available and is scalable as the number of modules can grow quickly. As the module management process, the authentication system also lies entirely in the cloud.

As also previously mentioned, when a bank registers a module, it can be found in the cloud through mobile Azure services, or it can be an on premise service. In both cases it is necessary to access the data to perform the desired operations.

Considering now a scenario in which the application intends to access a cloud module, via a mobile Azure service.

Figure 14 depicts the access to a bank cloud module scenario, specifically a mobile service. Briefly, the bank account manager sends a to the application an operations request order, but the application instead of accessing directly the database and returning the results, invokes a line of business that is running on a server. The business line uses a WCF layer to expose their functionalities via SOAP for outdoor applications. More specifically, WFC uses a BasicHttpRelayBinding endpoint to expose their functionalities through the Service Bus.

In order to facilitate understanding of the architecture, we analyze all the steps processed from the order to the results obtainment (Salvatori, 2013).

1. The application calls the mobile service API. The same API contains methods that allow interaction with the data;
2. The API in turn sends a access control service request to acquire the necessary token to authenticate itself in the Service Bus. OAuth authentication protocol is used;
3. The access control service returns a security token;
4. The mobile service uses the API set to extract the wrap_access_token sent the access control service. According to the customer's request the API uses different functions. Each creates a SOAP envelope that invokes a WCF service. The header contains the access token in a format base 64. The body contains the outcome of each call. It is used the node-uuid module to generate a unique id for call security token. Node.js uses the HTTPS module to send the soap envelope to the Service Bus;
5. The Service Bus validates and removes the security token. Then forwards the request to one of the WCF service listeners;

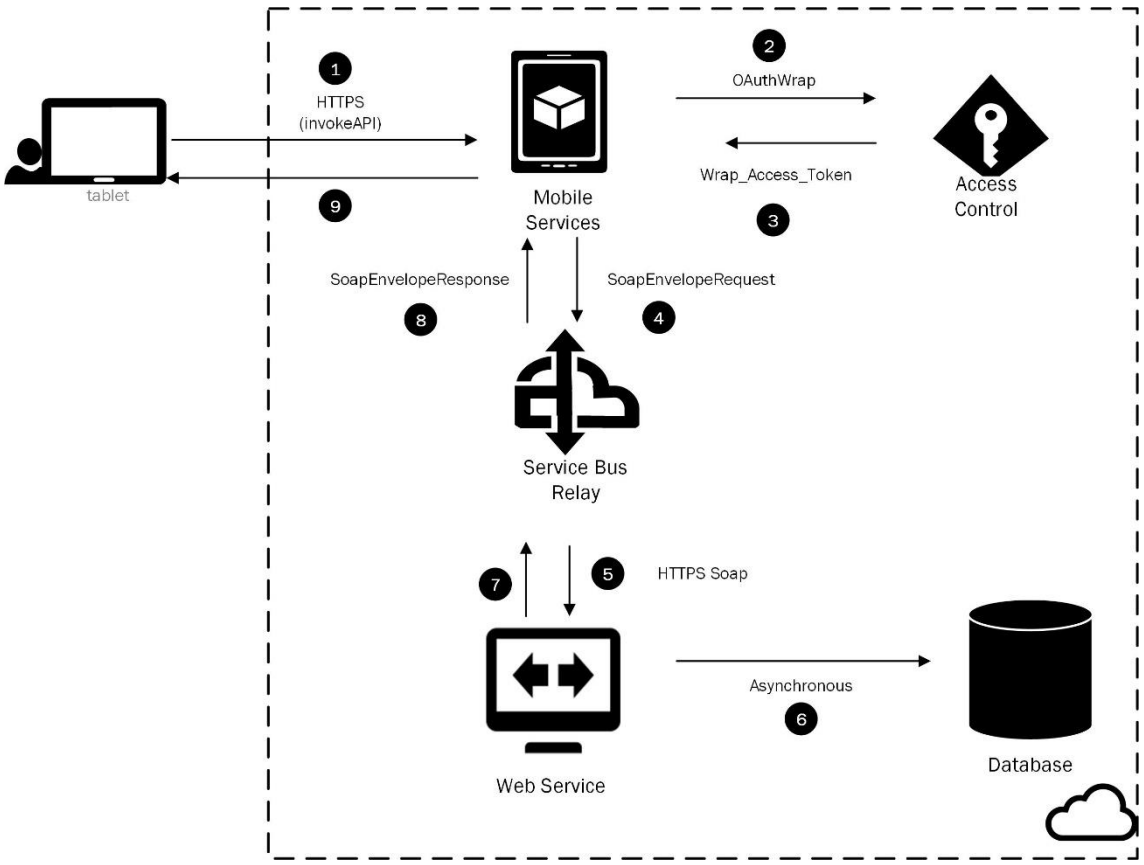

Figure 14. Access scenario to a module in the cloud

6. The WCF service uses asynchronous programming to access data kept in a local database;
7. In turn, the WCF service returns a response to the Service Bus;
8. The Service Bus forwards the message to the mobile service. The API uses the xml2js module to change the format of the SOAP response from XML to JSON. Then flushes the JSON object eliminating unnecessary arrays. Extracts the SOAP representation of data and creates a JSON response;
9. The mobile service returns JSON data to the client application;

This is the application access business line when trying to access a cloud module with a mobile service.

However, as already mentioned, there is also the case of bank registered modules not being in a mobile service but in on premise service. In such case, to be able to support this alternative the architecture has to be slightly different. Let us now analyze the scenario to access a module that is in another server on premise service.

Given the previously presented solution, the main difference to the depicted in Figure 15, it is that it does not use mobile services but only WCF. However this implies that Web services are implemented in data centers. These will later be accessed through the Service Bus Relay as the architecture tries to represent. Briefly, the service to which access is sought is exposed in the Service Bus Relay, when manager making the request, it is first sent to a proxy and there a request to the Service Bus Relay is made. In order to obtain the security token to access to data, the proxy sends a request to Access Control. Now, with the security token it's possible to access the Service Bus Relay where the service is exposed and, in turn, it accesses to the database in order to collect the required data by sending it back to the client.

Let's look in more detail the various steps taken:

1. In order to place an order, the client calls the proxy by sending him the address he wants to access, the method and the input parameters;
2. In the proxy, a request is sent to the access control service to acquire the necessary authentication token to the Service Bus Relay. The authentication protocol used is OAuth WRAP;
3. The access control service returns a security token;
4. Now with the security token, access is made to the Service Bus Relay which subsequently accesses the WCF;
5. Already with access to WCF, this in turn, accesses the SQL database that is on a local server;
6. The data is returned back to the Service Bus Relay;
7. The result of the request is sent back to the proxy;
8. The proxy returns the results to the manager so they can be processed and displayed properly.

Through the architectures shown above, we can see that regardless of the module type the banking entity registers if, it is possible through a hybrid solution to access these same modules. The Service Bus Relay plays an important role by enabling the external services display through the cloud and thus facilitates access to it. The mobile services component also showed a good disclosure, given its backend capacity. As with any system, it is necessary to understand its added value and its shortcomings, well as the advantages and disadvantages of the solution.

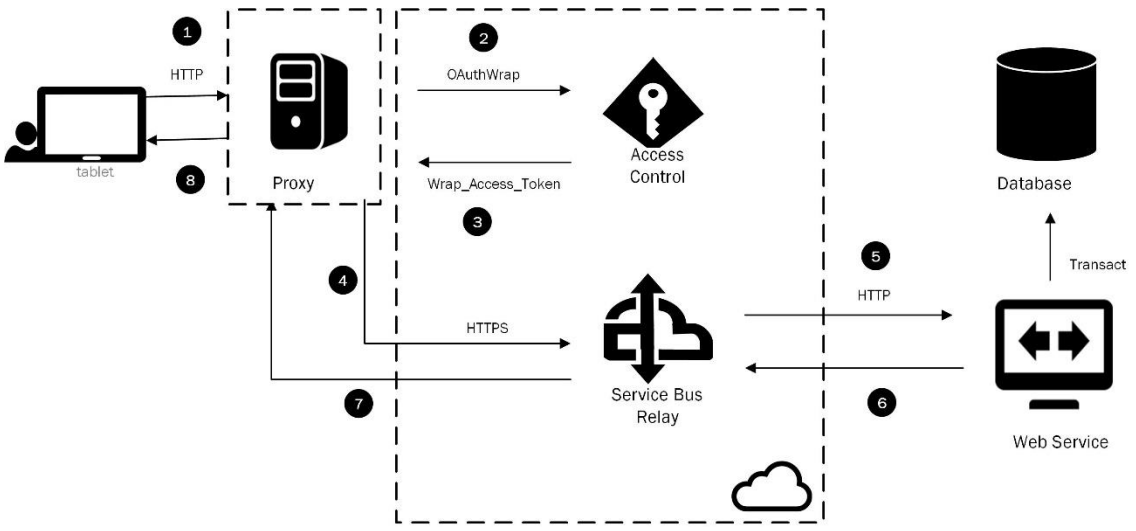

Figure 15. Scenario access to a local module

Advantages

One of the main advantages of this architecture is the fact that it presents clear and well defined business logic. This logic of three layers guarantees the possibility of cloud and local services integration. Through the service bus, it is possible to solve the communication problems between local applications and the outside world. This is advantageous because it allows it to be possible to make use of legacy services.

Another advantage is the multi-tenant capabilities. A multi-tenant application allows resources sharing, separating its users, in this specific case the banks. Thus, each institution sees the application in its own way.

The use of mobile services is an important advantage, as well. Mobile services give the architecture hybrid features since they allow the built of backends for any mobile operating system.

Disadvantages

One of the main disadvantages of the architecture is that the updating the data model is not automatic. The architecture was designed to support a single data model to several banks, where the data only are logically separated. However, if one of the banks needs to extend their line of business, it was not considered any form of automatic update process.

The fact that it's cloud based can also be considered a disadvantage. In addition to the utilization pricing issues, it also adds to the famous vendor lock-in problem (McKendrick J. , 2011). For more than a decade that worked has been done for the construction of solutions based on common standards and protocols that can be built, supported and replaced regardless of their supplier. However, this concept has regressed when it comes to cloud computing. By using a cloud solution based we are using protocols, standards and tools from a particular vendor, making a future migration a costly and difficult task.

CLOSING REMARKS

Conclusions

For some years now the banking sector has bet on electronic solutions in order to provide freedom to their customers so that they can carry out their banking transactions in more convenient way. However, the trend has been changing and has become necessary to develop these same mobile solutions, targeted at business managers. There are already many institutions that provide services for mobile devices, but these, for the most part, are intended solely for customers. Thus, the banking solutions development for commercial management teams has stagnanted, which leads to the continued use of monolithic and isolated systems, which most often than not lead to information silos. These systems eventually entail a distancing between banks and customers because they require the client's presence in a physical desk to perform a wide range of operations.

There are starting to emerge several solutions that claim to provide such services. Many of these are based on SaaS, which in addition to the inherent advantages are also a way for banks themselves to solve some of their problems.

Therefore, a SaaS-based solution was designed, that at the same time could be combined with the advantages of SOA so as to align technology with a structured and well-defined line of business. Some features considered essential for architecture of this kind, such as interoperability issues, scalability and platform usability were drawn. The interoperability issue, with the current banking systems consequent connection was considered crucial, because the disruption with it could in no way be considered. This architecture also accommodates the advantage of being designed to be used simultaneously by multiple banks. The presented architecture ensures that these requirements are considered. It was possible to combine the potential of the cloud and maintain a link with existing systems.

This architecture aims to be the key element for the banking development in emerging countries, bringing the services to the population and allowing growth, as it aids in decreasing the need for physical branches. Clearly there are still many steps to be taken, however it is considered that with TEA the most important step was made: the definition of a reference architecture for the systematization of access by mobile devices, banking systems, in SaaS environments integrated with on-premise solutions.

Future work

The work described throughout this article has still much to develop. In the case of an application with bank implications, it is clear that a rigorous implementation of the concepts presented here is necessary. Thus it is clear that it is necessary an entire security layer in communications within and outside of the architecture.

It will be necessary to ensure that data is not accessible from outside the application, ensuring total safety in the storage and handling of information. Bank's access control should also be monitored and alternative mechanisms for strong authentication should be implemented.

In addition to the security issues, clearly fundamental, there are many other factors that could make this project a strong channel of communication between banks and customers. Some of the challenges include improvement of the ability to access data offline. Often, especially in developing countries, access to the network cannot meet the needs, making that the communication is lost. Thus, it was desirable that it was possible an offline operations execution scenario. However, this scenario also presents competition problems in terms of changes made by banks. It would be necessary to determine the precedence of operations and how it could develop the rollback process to the previous situation.

Another interesting challenge to add is the ability to start a process on a tablet and then continue on another device. This challenge involves the distribution of encryption keys of the stored information.

There are many challenges that can sit on top of this proposed architecture. This is an area that despite all the advances that have suffered, it is always possible to take further and thereby reduce costs such as infrastructure physical counters. Many improvement proposals can be added in order to make a sustainable and possible system.

REFERENCES

Alecrim, E., 2013. Infowester. Retrieved August 2014, from, http://www.infowester.com/cloudcomputing.php

B-SaaS, 2012. Obtido em Dezembo de 2013, http://www.b-saas.com/

Buku, M. W. and Meredith, M. W., 2013. Safaricom and M-Pesa in Kenya: Financial Inclusion and Financial Integrity. Washington Journal of Law, Tecnhology & Arts, 8.

Camhi, J., 2013. Bank Tech. Retrieved December 2013, http://www.banktech.com/core-systems/temenos-acquires-us-saas-core-banking-provider-trinovus/d/d-id/1296255?

Cloud Computing for Financial Markets, 2011. CISCO. Retrieved from http://www.cisco.com/web/strategy/docs/finance/cloud_wp_c112D518876.pdf

Cloud Spectator, 2013. Cloud Server Performance: A Comparative Analysis of 5 Large Cloud IaaS Providers. Retrieved August 2014, from http://cloudspectator.com/2013/06/cloud-server-performance-a-comparative-analysis-of-5-large-cloud-iaas-providers-3/

Cpay360, 2013. Cpay360. (Reliable Software Solutions for Community Banks) Retrieved December 2013, from http://cpay360.com/markets/community-banks

CSC, 2013. Banking Software. Retrieved Dezembro 2013, from http://www.csc.com/banking/offerings/11090-banking_software

ETNA Software, 2011. SaaS & SOA in Retail Banking - Combination for Success. pp. 1-6. Retrieved from http://blog.etnasoft.com/wp-content/uploads/2011/03/ETNA_SaaS.pdf

International Telecommunications Unions (ITU), 2013. Key ICT indicators for developed and developing countries and the world (totals and penetration rates). Geneva: International Telecommunications Unions (ITU).

King, B., 2012. Bank 3.0: Why Banking Is No Longer Somewhere You Go But Something You Do (1ª ed.). Wiley.

King, B., 2013. Branch Today, Gone Tomorrow. Marshall Cavendish Business.

Kshetri, N. and Acharya, S., 2012. Mobile Payments in Emerging Markets. IEEE, pp. 9-13.

McKendrick, J., 2011. Cloud Computing's Vendor Lock-In Problem: Why the Industry is Taking a Step Backward. Forbes.

McKendrick, J., 2011. Cloud Computing's Vendor Lock-In Problem: Why the Industry is Taking a Step Backward. Forbes.

Mell, P. and Grance, T., 2011. The NIST Definition of Cloud Computing. National Institute of Standards and Tecnhology(Recommendations of the National Institute of Standards and Tecnhology), pp. 1-3.

mFoundry. (n.d.), 2013. Mobile Banking & Payments. Retrieved December 2013, from http://www.mfoundry.com/products-services/#mobile-banking

Microsoft Azure. (n.d.). Microsoft Azure. Retrieved from http://azure.microsoft.com/pt-pt/overview/what-is-azure/

Microsoft Azure. (n.d.). Microsoft Azure. (Base de Dados SQL). Retrieved from http://azure.microsoft.com/pt-pt/services/storage/

Microsoft Azure. (n.d.). 2014. Microsoft Azure. Retrieved August 2014, from http://azure.microsoft.com/pt-pt/campaigns/azure-vs-aws/

Microsoft Azure. (n.d.). 2014. Microsoft Azure. Retrieved September 2014, from http://azure.microsoft.com/en-us/documentation/articles/dotnet-develop-multitenant-applications/

Microsoft Azure. (n.d.). 2014. Microsoft Azure. (Serviços Móveis) Retrieved September 2014, from, http://azure.microsoft.com/pt-pt/documentation/services/mobile-services/

Microsoft Azure. (n.d.). 2014. Microsoft Azure. (Barramento de Serviços) Retrieved September 2014, from http://azure.microsoft.com/pt-pt/documentation/services/service-bus/

Microsoft Azure. (n.d.). 2014. Microsoft Azure. Retrieved September 2014, from http://azure.microsoft.com/pt-pt/services/active-directory/

Nasuni, 2013. White Paper: The State of Cloud Storage. pp. 1-16. Retrieved August 2014, from http://www6.nasuni.com/rs/nasuni/images/Nasuni-White-Paper-State-of-Cloud-Storage-2013.pdf

NCR APTRA. (). NCR APTRA Mobile Banking Gateway. Retrieved from http://www.ncr.com/products/banking/mobile-online/aptra-mobile-banking-gateway

Notícias ao Minuto, 2014. Aplicações Bancárias vão Superar Banca Online em cinco anos. Retrieved August 2014, from http://www.noticiasaominuto.com/tech/246870/aplicacoes-bancarias-vao-superar-banca-online-em-cinco-anos

Ribeirinho, V. and Silva, J., 2012. Financial Services - Angola Banking Survey. KPMG.

Salvatori, P., 2013. Microsoft Azure. Retrieved September 2014, from http://code.msdn.microsoft.com/windowsazure/How-to-integrate-a-Mobile-8780500c

Services, BTR., 2013. BTR Services. Retrieved December 2013, from http://www.btr-services.be/our-solutions-services/saas-banking-platform

Silva, J., Teixeira, C. and Pinto, J. S., 2015. Support architecture to desmaterialization of banking operations. 10th Iberian Conference on Information System and Technologies, Aveiro: IEEE. pp. 1-6.

Stevis, M., 2015. The Wall Street Journal. Retrieved February 11, 12, from http://www.wsj.com/articles/african economies-to-grow-4-5-on-average-in-2015-1432544482

Teixeira, C., 2011. Arquiteturas distribuídas. Universidade de Aveiro.