AN ARCHITECTURAL FRAMEWORK FOR DELIVERING SIP-AS MULTIMEDIA SERVICES BASED ON JADE/OSGI TECHNOLOGY

Renato B. Cabelino Ribeiro[1], Magnos Martinello[2], Celso Alberto Saibel Santos[3] and Rosane Bodart Soares[4]

[1,4]Departamento de Engenharia Elétrica, IFES, Vitória, Brasil,
[2,3]Departamento de Informática, UFES, Vitória, Brasil

ABSTRACT

This paper proposes a new scalable service-oriented architecture based on Open Service Gateway Initiative (OSGI) technology. A key part of this architecture is its SIP application as a service (SIP-AS). It relies on IMS core network supported by multi agents components implemented using Java Agent DEvelopment (JADE) platform. As a proof of concept, a real testbed/prototype has been developed to validate our approach. The validation process consisted of two phases: (i) configuration of the JADE/OSGi SIP-AS architecture to provide a televoting service and (ii) characterization and analysis of jitter, packet loss, load capacity and CPU utilization of the implemented architecture. Results demonstrate that this televoting service scales up and out enabling the elasticity of the architecture on the processing of concurrent calls and dynamic load balancing.

KEYWORDS

JADE, OSGi, IMS, multimedia services, televoting.

1. INTRODUCTION

SIP (Session Initiation Protocol) has emerged as the signaling protocol designed for supporting IP telephony and unified communications. The standard protocol is supported in a wide variety of IP communications products including IP PBXs, servers, videoconferencing systems as well as desk phones. Typically service providers have replaced their legacy telephony infrastructure by SIP-based solutions and converging voice, video, and traffic onto a common IP backbone, thus they can reduce costs and simplify operations.

SIP trunking services are offered as a cost effective alternatives to conventional PRI (Primary Rate Interface) circuits for PSTN connectivity. By switching to a SIP trunking service, IT organizations can eliminate TDM (time-division multiplexing) gateways, reduce monthly service fees and improve service agility i.e. SIP trunks can be installed and reprovisioned more quickly and easily than conventional PRI circuits.

The scalability of the architecture for providing SIP as a service is a huge challenge. In a centralized topology, SIP is installed in central data centers. External calls are received from the corporate WAN, aggregated, and handed off to the SIP service provider. Inter office calls are carried over the corporate WAN. However, the rate of calls can vary significantly affecting the quality of the provided service. Thus, network planners should evaluate carefully the service architecture in order to determine which model best addresses their organization's specific

DOI : 10.5121/ijngn.2014.6301

requirements. This paper introduces a service-oriented architecture combined with multi-agents systems as a hosting platform for telecommunication supplementary services on IP Multimedia Subsystem (IMS).

The JADE platform, integrated with the OSGi framework, is the proposed agent-based development environment [1][2]. This approach allows for a more flexible and dynamic form of service provisioning over the IMS architecture, allowing services to be negotiated on demand according to the current environment requirements (services rules, QoS requirements, interaction parameters, etc.).

This article presents the implementation of a SIP Application Server (SIP-AS) on OSGi service-oriented architecture, integrated with JADE framework for the creation and provision of multimedia services on IMS. In section 2, the technology is described. The related works are presented in Section 3. Section 4 describes the design and implementation of the televoting service, and the result analysis. Conclusions are described in section 5.

2. TECHNOLOGY BASE

2.1. IP Multimedia Subsystem (IMS)

IMS is an overlay service-provisioning platform, which allows telecommunication operators to utilize Internet technologies to build an open IP-based service infrastructure that enables easy deployment of new multimedia services [3][4].

Moreover, IMS uses the SIP protocol for session control and the RTP/RTSP protocol for media streaming. Network requirements such as roaming, scalability, security and reliability have been defined by 3GPP. The SIP protocol has been standardized by IETF - Internet Engineering Task Force, but not its related architectures.

The layered approach proposed by IMS increases the importance of the application layer as services are designed to work independently of the access network. IMS is designed to bridge the gap between them [4]. It offers more flexibility for telecom operators to manage different services with distinct requisites (e.g.: bandwidth, latency, jitter, etc.). The major IMS elements related to service architecture are the following [3][4][5]:

- S-CSCFs (session management and routing family): the serving call state control functions facilitate the correct interaction between the application servers, media servers, and the Home Subscriber Service (HSS).
- HSS (database): the home subscriber server is the main data storage for all home subscriber and service-related data of the IMS.
- ASs (service functions): the application services are entities that provide value-added multimedia services in IMS, such as presence and Push to Talk Over Cellular.

Figure 1 presents these major elements of the IMS architecture.

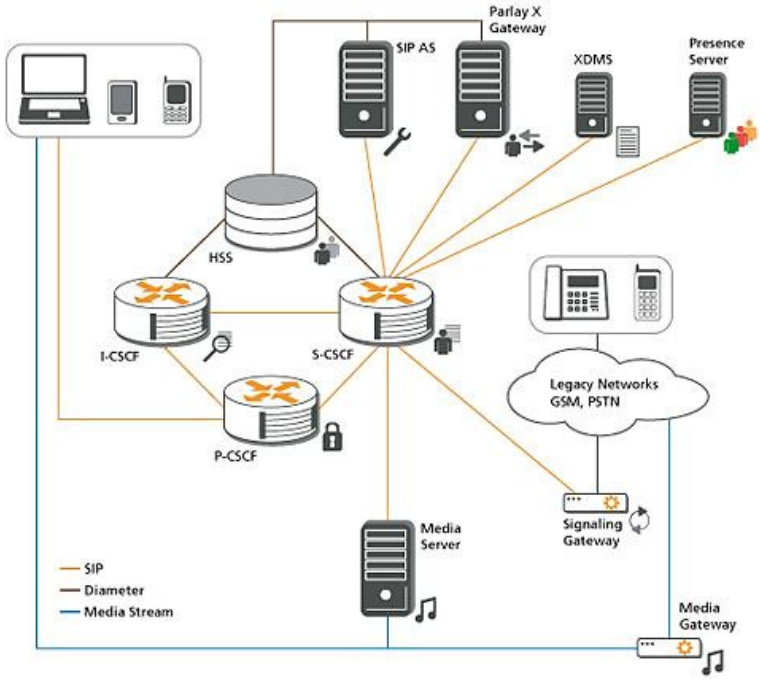

Figure . Minimal IMS based architecture [4].

2.2. JADE/OSGi integration

JADE is a middleware for the development and execution of peer-to-peer applications based on the agent paradigm that can easily work and interoperate on traditional or wireless network environments. JADE internal architecture is currently the only architecture entirely compliant with FIPA standards [6]. According to [7] and [8], the JADE platform can offer the following:

- Graphical interface which allows monitoring, debugging and logging;
- Components which can be distributed over the network;
- Mobility and cloning of agents as well as multi-tasking scheduling;
- Lifecycle management, name and yellow pages services, point-to-point message transport service, speech-act message structure and ontology service;
- Interoperability with other platforms that offer support to FIPA standards.

Along with the agent-based environment, the OSGi framework allows for runtime installing, updating and uninstalling of JAVA modules. The latest specification of the OSGi framework defines the way collaboration among services offered by modules inside a single Java Virtual machine (JVM) occurs.

The service platform offers a standard and open architecture to service providers, developers, software developers, telecommunications operators and equipment manufacturers so that they can develop and manage services in a coordinated and efficient way [9].

The development of applications using OSGi can be accomplished through the combination of collaborative, reusable modules associated with descriptive information on their metafiles which include service-related input that must be instantiated/imported to achieve a consistent execution of the modules [9]. Also, the services provided by OSGi implement a JAVA interface for

registering on local service registries. Through this centralized control model, the modules (or bundles) can verify their service dependencies.

Furthermore, the OSGi services platform offers developers the means to maximize the use of platform independent resources and dynamic updating of JAVA modules, allowing development of services for devices with limited computing resources, widely used in corporate environments. New services registration as well as research and maintenance of pre-existing services (including their uninstallation from the system), services status notifications and follow-ups on bundles lifecycle can be carried out in a simple and efficient way.

3. RELATED WORKS

This section analyzes exclusively SIP approaches that have correlation with customized telecommunications services (for example, additional services, toll free phone services, Televoting, local number portability) or with the framework presented in this work. Oliveira et al [10] propose two approaches for implementing number portability service in IMS networks, tested on an AS (Application Server) according to the standards of the General Regulations of Portability. In the first approach, the AS performs the function of local number portability without call states control, routing every call originated by IMS to AS, or those based on a numeric phone context through an Initial Filter Criteria (IFC). In the second approach, the number portability service acts as a back-to-back user agent (B2BUA), i.e. in a leg termination call aimed to a ported user, configures an IFC to conduct the call routing for an AS, which acts on behalf of the user ported and initiates a new call to the correct destination. For the implementation of AS, SIP Servlet technology was created by the authors.

Munadi et al [11] propose the design and implementation of VoIP services with OpenIMS and ASTERISK, interconnected by an ENUM server which develops numerical mapping function between the two servers. The authors observed the proposed environment according to: (a) performance measures for each server; (b) the Post Dial Delay (PDD) and (c) of the same CPU consumption. The values measured and analyzed in (a) identify the service time consumption on the part of the SIP signaling system. In (b), three scenarios were tested where the Traffic Analyzer WIRESHARK was used in order to capture and analyze traffic from the User Agent Caller from your application until its acceptance by the counterparty in the call, which allowed the analysis of the PDD in each test performed. For (c), were used the TOP utility from the operating system itself in order to obtain the maximum CPU value throughout the experiments.

Li et al [12] implement two IMS services – a chat room (SIP-IM) and Presence services, in a SIP-AS. SIP-AS architecture used is based on Mobicents SIP Servlet component (MSS). In addition, we used the OpenIMS Core to IMS Core Network implementation. The authors have developed a use case diagram and class diagram for the services analyzed. For the test scenarios, XML templates were made to present the requirements and the design of both services.

Franks et al [13] describes the experiences and processes of the performance engineering of SIP applications in Java environment using Java 2 Enterprise Edition (J2EE) as application server. Two scenarios were evaluated: standalone and clustered environment. To build a performance model of each environment, network traces were used. The predicted performance model was calibrated with the data obtained from packet traces analysis to match with real system test data. In both cases, the capacity predicted by the model was close to the capacity measured from prototype system. Moreover, the performance model could help identify the bottlenecks in performance of the system and was presented as a novel way of modeling Ethernet by using two-phase fixed-rate server.

Femminella et al [14] describes a proposal and analysis of original usage of virtualization and parallelization techniques in order to better exploit the computing capabilities of servers hosting Java implementations of SIP application servers (ASs). The authors used a hypervisor to host SIP ASs virtual machines (VMs) and an open source JSLEE AS (Mobicents) running a SIP-based VoIP service, performing several database queries during an call lifetime. The main results shows that the standard deployment of Java-based ASs do not provide the best performance, which is obtained when the server runs in multiple systems in parallel, each one with few CPU cores (running a single AS instance inside VMs, and deploying VMs with 32bit OS and 22/3 CPU cores). With these scenarios the authors were able to evaluate a performance improvement of about 64% in terms of signaling call throughput, with an average setup latency halved.

4. DESIGN AND IMPLEMENTATION

Figure 2 introduces the scenario for implementation Televoting service (or another multimedia service) through the utilization of the JADE platform along with the OSGi framework, running on SIP-AS.

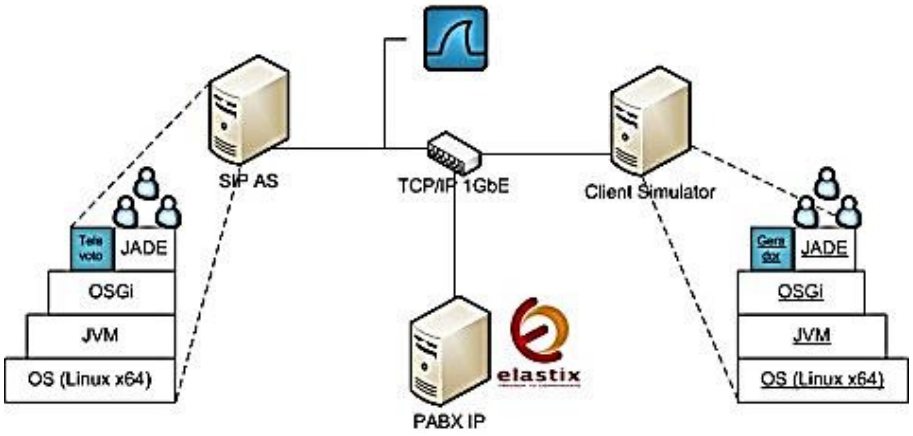

Figure . Network Topology.

4.1. Televoting Service Lifecycle

Upon start, the OSGi Televoting service bundle loads its configuration parameters from a file named televoto.conf and automatically creates a number of attendant agents on the JADE platform. Each agent registers itself on IP PBX as an extension and become operational. All registered agents are grouped in a unique number. When a call arrives at that unique number, the IP PBX redirects it to one of the registered agents and the call is then processed. The rules for redirection (first available, ring all, last called, etc.) depend on the IP PBX distribution. In the proposed scenario the programed rule is first available. As described above, Televoting agent register along IP PBX and became ready to work. Once created, they turn visible on JADE GUI (inside MAIN container) and your control is now managed by JADE framework.

4.2. Televoting Workflow

After a call from a client is placed to a Televoting unique number, which is published on the architecture, the initiation and maintenance of a dialogue are shown on the diagram in Figure 3 and briefly explained:

1) The client places a call to a Televoting service unique number and waits for a response;
2) The IP PBX receives the call and redirects it to an attendant agent according to a configured rule;
3) One of the Televoting agents on the JADE platform receives the redirected call, signaling client and waiting for a response.
4) The Televoting agent accepts the current call (a response to the client request made in item 1).
5) The client acknowledges the call (a response to the Televoting agent request from item 3);
6) The Televoting agent sends a media stream according to session negotiation;
7) At the end of announcement the Televoting agent hangs-up the call, thus ending the session;

The client acknowledges the Televoting agent request. Figure 3 represents these above steps.

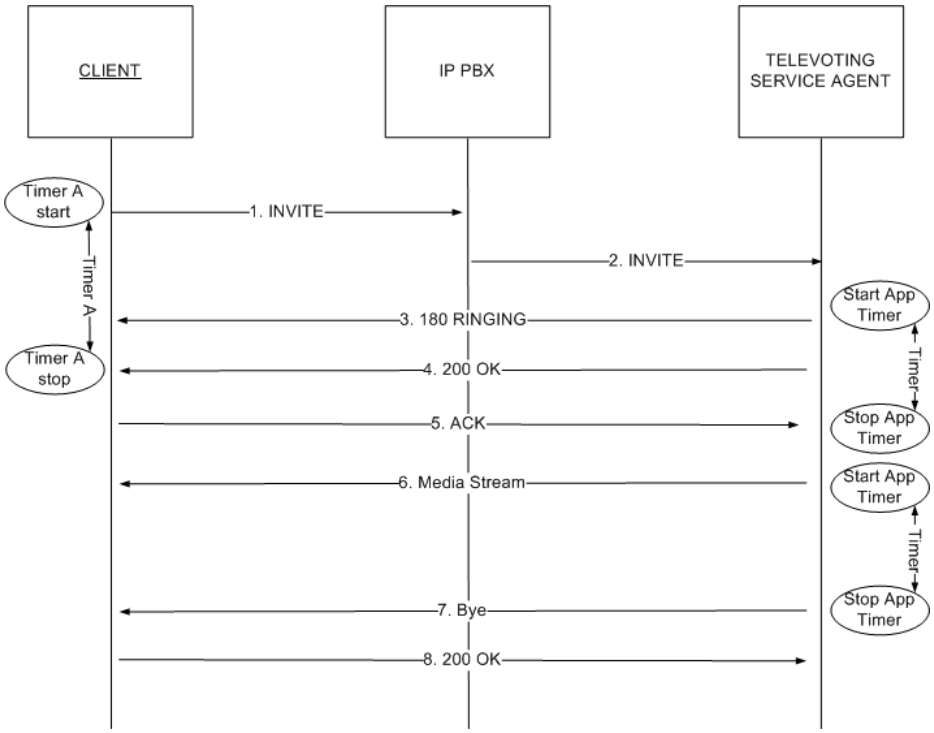

Figure . Network Topology.

To complement the scenario presented, we develop a graph which represents the state machine of the Televoting service, as shown in Figure 4.

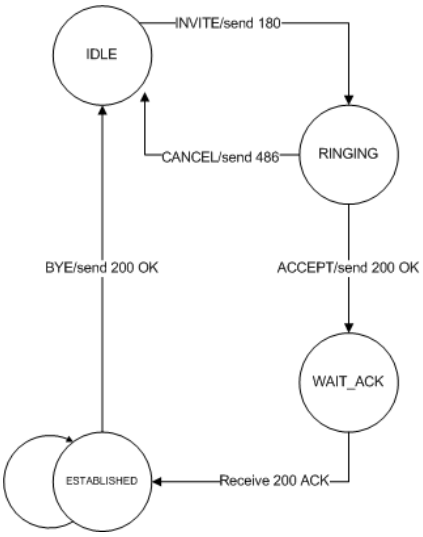

Figure . The Televoting service agent state machine.

Figure 5 presents OSGi console with JADE and televoting bundle started. As described above, televoting agent register along IP PBX and became ready to work. Once created, televoting agent turns visible on JADE GUI (inside MAIN container) and your control is now managed by JADE framework. All log messages are manipulated by an instance of LOG4J bundle and recorded in a file named televoto.log on local hard disk.

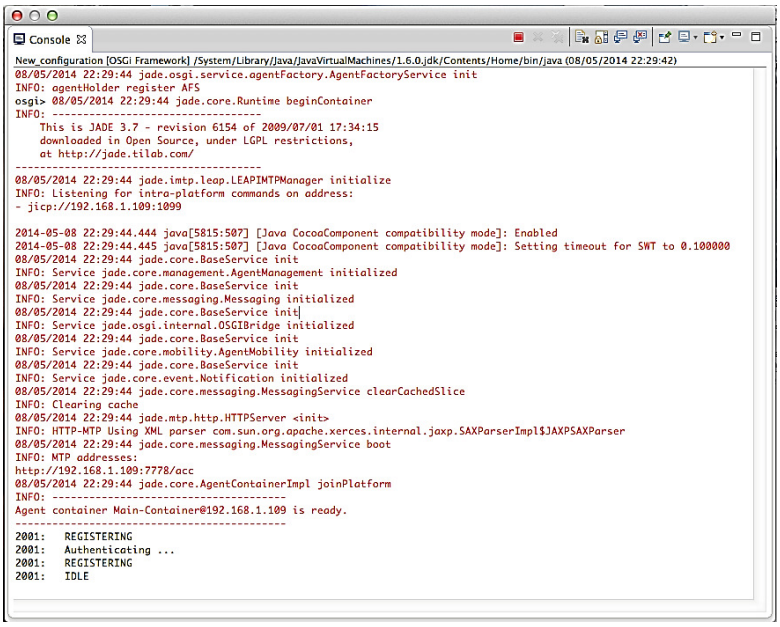

Figure . OSGi console with JADE and televoting agent initialization process.

Inside JADE container the message exchange between televoting agents and JADE architectural components (e.g.: AMS, DF and RMA agents) can be traced by a special JADE agent called SNIFFER Agent (messages must follow FIPA specifications). Also messages can be exchanged between any other agents started on platform too. The number of televoting agent to be created is defined in televoto.conf file. Figure 6 presents JADE GUI.

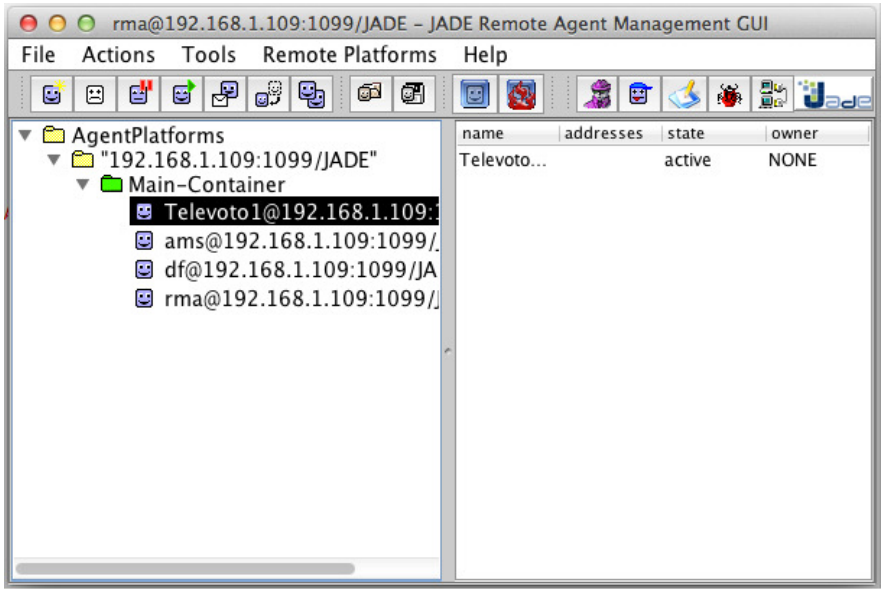

Figure . JADE graphical user interface and agents started on MAIN-Container.

4.3. Validation Tests

We have developed a sequence of tests to validate integration of JADE/OSGi as a SIP-AS. The parameters observed in the experiment are: jitter, variation of jitter, packet loss and CPU load of the SIP-AS. To support this scenario, the equipment presented in Figure 2 and detailed in Table 1 below, were used.

Table . Test equipment's specification.

Item	Equipment
SIP-AS, Client Simulator	Dual Intel Xeon with 4 core/processor, 20 GB RAM, Intel Gigabit Ethernet, OS Linux Server 12.04 x64
IP PBX	Single Intel Xeon with 4 core, 8GB RAM, Intel Gigabit Ethernet, ELASTIX Custom Distro
Switch	H3C-2928 24 ports Gigabit Ethernet
Network Analyzer	Intel Core2 Duo, 4GB RAM, Atheros Gigabit Ethernet, OS Microsoft Windows XP SP3

In each test, the client simulator performed a load of calls to its counterpart in the televoting service (a client for each service agent). This load of calls was parameterized in the configuration file of client simulator such that it is executed one or more times, depending on the amount of redials parameter set.

With this approach, we identified the capacity of the SIP-AS in handling calls faster without the need of integration with the IMS Core. During testing, all network traffic was captured by the Analyzer for subsequent analysis.

The methodology used in the tests was developed according to the following profile: (a) the whole SIP-AS infrastructure is initialized; (b) the packet capture is initialized in the WIRESHARK; (c) the client Simulator is initialized, running 100 concurrent calls to the

Televoting service and (d) at the end, the entire environment is shutdown. In each test, the client Simulator is reconfigured to generate additional concurrent calls as shown in Table 2, up to a total of 1000 calls, in order to identify its impact with respect to jitter and jitter variation. Note that the average value of jitter, as well as your variation (MAXDELTA), increases as the load of concurrent calls grows.

In the jitter column, we observe that even with a high load of concurrent calls, the values remain at an acceptable threshold. However, the same does not occur with the MAXDELTA values, because the higher the value presented more occurrences of gaps in the audio message from the Televoting were perceived on the Wireshark RTP-analysis.

Table . Mean values of Jitter and its MAXDELTA**.**

Concurrent Calls	**Jitter (ms)**	**MAXDELTA (ms)**
100	0,72	111,80
200	1,74	213,07
300	3,03	354,79
400	3,96	401,76
500	4,94	485,36
600	5,93	522,16
700	7,29	629,50
800	9,71	784,57
900	9,91	716,44
1000	11,98	853,66

About CPU usage analysis, Figure 7 presents its load average over the tests. The minimum and maximum load values were between 60 and 80 percent. When added more processing resources to SIP-AS, the average value was reduced by half, leaving around 30 to 35 percent.

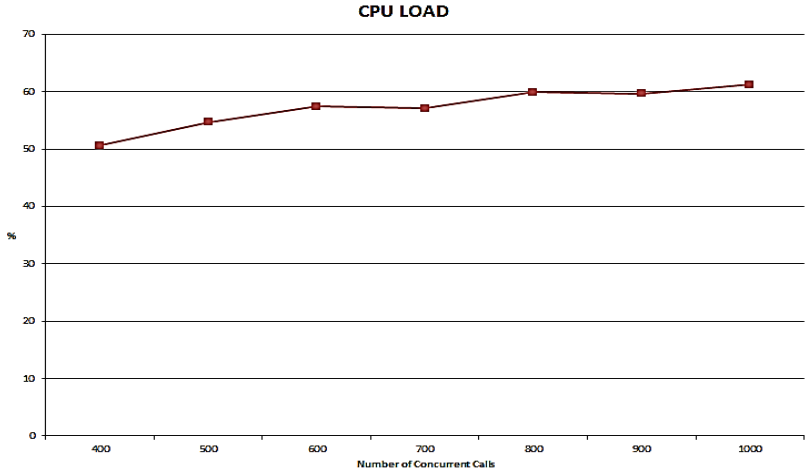

Figure . Average of CPU Load during tests.

The values of packet loss measured were insignificant in all tests and did not influence the communication on media plan of televoting service. As seen in Figure 8, the time spent to attend the n concurrent calls in Televoting service presented a behavior near the linearity as number of concurrent calls grows.

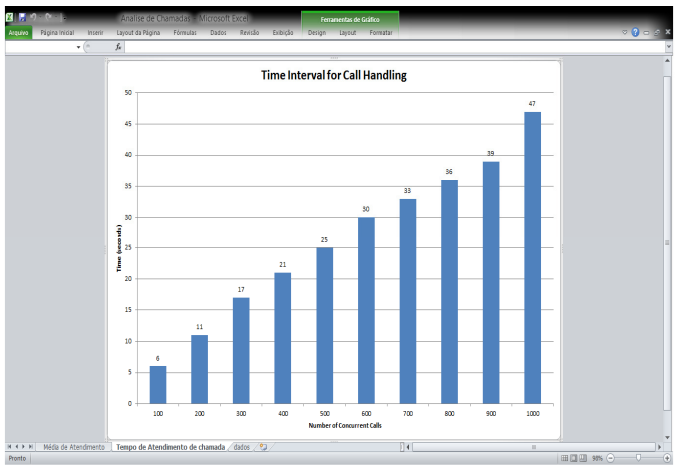

Figure . Time interval for handling n calls in seconds.

Figure 9 depicts the average of calls handled per second on televoting service. We observe an increase in average attendance between 100 and 400 concurrent calls per second on system. From this point, this value remains near the stability between 30 and 32 calls handled.

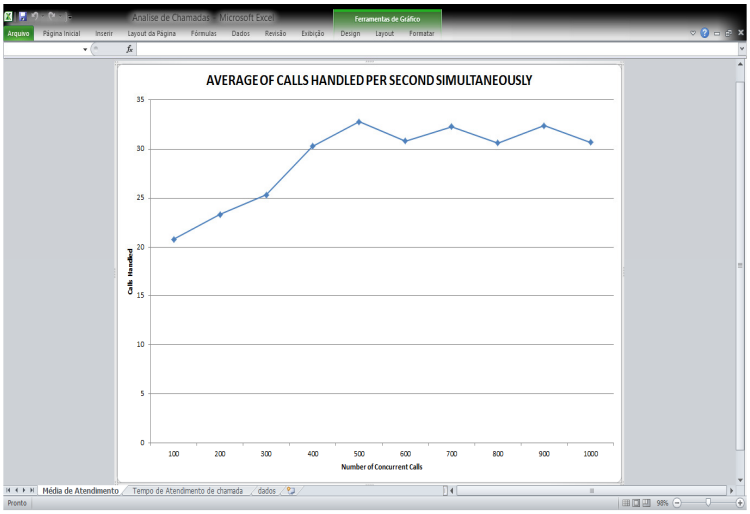

Figure . Time interval for handling *n* calls in seconds.

We believe this is a point that the service needs to be scheduled to a second SIP-AS because we perceive the existence of gaps (hardly noticeable) on audio message transmitted with the same number of concurrent calls. The gaps occur due to the value of jitter variation to be very high, as shown in Table 2.

5. CONCLUSIONS

In this paper, we present the design and the development of a new service-oriented architectural framework to support the provisioning of IP Multimedia as a service. Our approach differs from the previous works by its functional components based on the combination of the OSGi and JADE technologies.

As proof of concept, a prototype of an innovative Televoting service is built and deployed in a real testbed. Results indicate that this architecture is able to support in a scalable way concurrent calls. It allows the calls processing by service offered in PSTN Intelligent Network (IN). When the server CPU consumption is high, it can attenuate with the vertical scaling (i.e. scale up by provisioning of more hardware resources in a local datacenter shown in subsection 4.3) or horizontal (i.e. scale out, distribution of the service JADE/OSGi to the cloud)

.

For future work, we plan to program the system for supporting dynamic load balancing as well as the mobility of service agents and integration with OpenIMS Core.

REFERENCES

[1] Bellifemine, Fabio; Caire, Giovanni; Greenwood, Dominic. Developing MultiAgent Systems with JADE. John Wiley & Sons Ltd. Inglaterra. 2007.

[2] OSGi Alliance. OSGi Service Platform: Core Specification, Release 4, Version 4.2. Technical report, 2009. Disponível em http://www.osgi.org/download/r4v42/r4.core.pdf.

[3] Salchow Jr., Ken. Introduction to the IP Multimedia Subsystem (IMS): IMS Basic Concepts and Terminology. Available in http://www.f5.com/pdf/white-papers/ims-introduction-wp.pdf

[4] Poikselkä, Miikka. Mayer, Georg. THE IMS: IP Multimedia Concepts and Services. 3 ed. John Wiley & Sons, 2009.

[5] Ahson, A. Ilyas, Mohammad. IP multimedia subsystem (IMS) handbook. CRC Press, 2009.

[6] Bellifemine, Fabio; Caire, Giovanni; Poggi, A.; Rimassa, G. JADE White Paper, disponível em: http://jade.tilab.com/papers/2003/WhitePaperJADEEXP.pdf

[7] Wooldridge, Michael. An Introduction to MultiAgent Systems. 2 ed. John Wiley & Sons, 2009.

[8] Bellifemine, Fabio; Caire, Giovanni; Greenwood, Dominic. Developing MultiAgent Systems with JADE. John Wiley & Sons Ltd. Inglaterra. 2007.

[9] Ribeiro, Renato B. C., Soares, Rosane B., The Application of JADE and OSGi Technologies in the Telecommunications Services Architecture, EUROCON 2011.

[10] Oliveira, Rafael G. et all. An Application Server Approach for Number Portability in IMS networks. IEEE - SBrT - ITS 2010 - International Telecommunications Symposium
September 06-09, 2010, Manaus, Amazonas, Brazil.

[11] Munadi, R. et all. Design and Implementation VoIP Service on Open IMS and Asterisk Servers Interconnected through Enum Server. International Journal of Next-Generation Networks (IJNGN) Vol.2, No.2, June 2010.

[12] Li, K. et al. Two Exploring Experiments on IMS Service Based on SIP AS. International Journal of Multimedia and Ubiquitous Engineering, Vol.9, No.4 (2014), pp. 375-386.

[13] Franks, Greg et al. Performance measurement and modeling of a java-based session initiation protocol (SIP) application server. QoSA-ISARCS '11 Proceedings of the joint ACM SIGSOFT conference -- QoSA and ACM SIGSOFT symposium -- ISARCS on Quality of software architectures -- QoSA and architecting critical systems – ISARCS (2011). pp. 63-72.

[14] Femminella, Mauro et al. Performance Management of Java-based SIP Application Servers. 12th IFIP/IEEE IM 2011: Mini Conference. pp 493-500.

Authors

Renato Benezath Cabelino Ribeiro

Graduated in Computer Science from the Fundação de Assistência e Educação (2001). He is currently Professor of 1 and 2 degrees of Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo. Has experience in the area of Computer Science, with emphasis in Telecommunications, acting on the following topics: data communication networks, multimedia applications, telecommunications, JADE/OSGi and NGN.

Magnos Martinello

(magnos@inf.ufes.br) received his B.Sc. Degree in informatics from the Federal University of Paraná (UFPR), Brazil, his M.Sc. Degree in computer and systems engineering from the Federal University of Rio de Janeiro (UFRJ), Brazil, and his Ph.D. degree in computer science from the Institut National Politechnique de Toulouse (INPT), France, in 1998, 2000, and 2005, respectively. In 2008, he joined the Department of Informatics (DI) at the Federal University of Espírito Santo (UFES), Brazil where he currently holds an associate professor position. Since 2011, he had a Research Productivity Fellowship granted by CNPq. His research interests include computer networks, software-defined networks, and performance analysis.

Celso Alberto Saibel Santos

Doctor in Fondamentalle Informatique et parallelism Université Paul Sabatier Toulouse III (1999), with thesis work developed at LAAS / CNRS; Master in Electrical Engineering from the Escola Politécnica da Universidade de São Paulo (1994), with work in LSI / POLI / USP; Electrical Engineering from the Universidade Federal do Espírito Santo (1991). He is currently Adjunct Professor in the Department of Informatics (DI) of Universidade Federal do Espírito Santo (UFES). Has extensive experience guiding and coordinating research projects and innovation, working mainly in the areas of Multimedia, Hypermedia, and Web.

Rosane Bodart Soares

Graduated in Electrical Engineering from the Universidade Federal do Espírito Santo (1981), MS in Electrical Engineering from the Universidade Federal do Espírito Santo (1994) and Ph.D. in Electrical Engineering from the Universidade Federal do Espírito Santo (2003). She is currently Associate Professor at the Universidade Federal do Espírito Santo by the Department of Electrical Engineering. Have experience in Electrical Engineering with emphasis on Telecommunications, acting on the following subjects: NGN, Intelligent Networks, Platform CORBA, Distributed Systems, Multimedia Networks.