PROVIDING ADAPTIVE TRAFFIC ROUTING BASED ON USER AND NETWORK CONTEXT

Sergio S. Spinola, André L. C. Oliveira, Joatham P. S. Silva, Carlos F. J. Muakad, Paulo N. M. Sampaio

Computing and Systems Graduate Program

Salvador University (UNIFACS)
Salvador, BA - Brazil
email: srgspnl@gmail.com, andreluizoliveira@gmail.com, joathamp@gmail.com, fred.fredjansen@gmail.com, pnms.funchal@gmail.com

Lina M. P.L. Brito

Madeira Interactive Technologies Institute (M-ITI)
University of Madeira
Funchal 9020-105, Portugal
email: lina@uma.pt

KEYWORDS

Adaptive routing, quality of service, quality of experience, context, quality of context, context-based routing.

ABSTRACT

Providing real-time traffic guarantees and fairness based on the availability of network resources has been a major issue presented in the literature. However, due the convergent nature of digital architectures, the increasing demand of upcoming real-time sensitive traffic, such as VoIP, and a higher user´s adaptability (devices, global positioning, content quality, etc.), solutions based on Quality of Service (QoS) turned out to be insufficient in order to meet user´s requirements. Indeed, QoS metrics are network-centered, and mostly related to the dynamic nature of the traffic (such as throughput, delay, jitter, among others). In order to meet the need for a user-centered network, this paper proposes a context-aware solution where the concepts of Quality of Service, Quality of Experience and Adaptive Routing are integrated in order to provide a more dynamic and pro-active approach for the delivery of context-oriented time-sensitive traffic.

INTRODUCTION

The intensive use of the current IP networks requires an optimized management of the available resources of the network infrastructure in order to enable the co-existence of multiple types of traffic. On the last few years there has been a considerable increase in bandwidth availability, which motivated a burst regarding the proposal of new applications, presentation devices, mobile communication, etc. In particular, we should consider emerging applications and services (such as Voice over IP – VoIP – and videoconferencing), which generate an increasing amount of real-time data traffic to the network. Unfortunately, network infrastructure and routing strategies have not evolved at the same pace as data applications. Therefore, network infrastructure is constantly under resources shortage and consequently under congestion.

Different contributions have been proposed in the literature in order to provide real-time traffic guarantees and fairness concerning the utilization of network resources.

Therefore, Quality of Service (QoS) has been the key solution in order to meet user´s requirements. For this purpose, most of the current QoS contributions rely on the concepts introduced by classical paradigms such as Integrated Services (IntServ), Differentiated Services (DiffServ), MPLS/GMPLS and Traffic Engineering (Balci and Sargent 1981).

Currently service providers have deployed these QoS techniques in order to determine configuration strategies, planning and provisioning network services. These techniques are related, in general, to admission and congestion control, buffer management and scheduling. However, regardless of the techniques applied the expected delivery quality has not been achieved completely, generating consumer dissatisfaction with the services offered (El-Gendy et al. 2003). Although Service Level Agreement (SLA) establishes users and infrastructure parameters for the delivery of a particular traffic, the dynamic nature of the user and application´s environment should also be considered.

In order to embody the concept of a user-centered network the notion of context can be applied. Context awareness is understood as an ubiquous and/or pervasive computing paradigm that aims at dealing with changes in the computational system environment (Shaikh and Collange 2010). The implementation of context aware networks can be helpful in order to improve user´s experience and satisfaction when accessing network resources. For this purpose, this work addresses the proposal of a context-aware solution, which relays on three main concepts: Quality of Service, Quality of Experience and Adaptive Routing.

RELATED WORKS

Different issues should be considered when addressing context-based routing. Context is related to both user and communication platform. Therefore, the context-based solution proposed in this paper relies on the definition of Quality of Service (QoS) (Yerima 2011), adaptive routing (Karthiga and Balamarugan 2013) and implementation of Quality of Experience (QoE) (Alreshoodi and Woods 2013). As expected, the contributions related to QoS frameworks focus on the proposal of protocols and mechanisms in order optimize the resources availability related to network equipments. QoE, instead, refers to user's requirements and expectations and how they actually perceived the service delivered. In order to meet user expectations, the implementation of QoS should also be centered on the perspectives of the end users. In general, QoE can be correlated by the measurement of MOS (Mean Opinion Score), whose values range among bad experience, poor, acceptable, good and excellent (El-Gendy et al. 2003).

QoE, instead, refers to user's requirements and expectations and how they actually perceived the service delivered. In order to meet user expectations, the implementation of QoS should also be centered on the perspectives of the end users. In general, QoE can be correlated by the measurement of MOS (Mean Opinion Score), whose values range among bad experience, poor, acceptable, good and excellent (El-Gendy et al. 2003).

CONTEXT AWARE SYSTEMS

Context can be defined as "any information that can be used to characterize the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves. Context is typically related the location, identity and state of people, groups and computational and physical objects" (Dey et al. 2001). The use of context can be applied to entities, persons, places, or even to an object relevant to the application, by the definition of characteristics of individuality, activity that may be involved, location and time and even relationships with other entities (Zimmermann et al. 2007). In order to assess context, and device, the notion of Quality of Context and Quality of Device are also considered.

Quality of Context

Although the contribution of context-aware systems can be expressive, their effectiveness can only be achieved if context information is properly defined and validated. Therefore, the definition of Quality of Context (QoC) is required in order to provide the understanding between QoC, QoS and Quality of the Device (QoD). The latter is related to the hardware components in charge to collect and provide the context information (Nazario et al. 2012), (Bucholz and Shciffers 2003)..

In order to provide QoC some metrics should also be defined, such as (Weiser, 1999): Accuracy of information (Precision); Likelihood the information is correct (Probability of Correctness); Level of trust in sources of information (Trust-worthiness); Resolution of the levels of granularity of information (Resolution); Timeliness of information related to their temporal characteristics (Up-to-dateness).

Quality of Device

Besides Quality of Context (QoC) concerning the characterization of the collected context information, it is also important to consider the Quality of Device (QoD), which is related to the precision of the computing device that will collect the context information. For instance, the Global Positioning System (GPS) of each device can have different levels of precision, or even a particular device that is not able to provide some parameters compared to another due hardware incompatibilities or the lack of ability to collect such information (Mascolo and Museli 2006). Therefore, QoD will provide information on the technical characteristics of each device and its capabilities (Vieira et al. 2009).

Most of these contributions are related to the utilization of context applied mainly to wireless networks. In these studies, the use of context allowed improvements mainly in: stability of the communication link, increased bandwidth (by decreasing overhead), higher batteries autonomy, shorter delay and scalability. These environments differ greatly from wired networks, mainly due to storage capacity and processing constraints, battery life limitation, and in some cases, limited bandwidth.

It is also important the adoption of clear policies in order to provide the correct analysis of contextual information and to be in conformance with QoC (Mascolo and Museli 2006). For this purpose, some contributions in the literature proposed solutions for improving the adoption of QoC policies based not only on the current context, but also on the effects of erroneous context information with low quality and its effects on systems, such as Proteus (Manzoor et al 2009).

PROPOSED CONTEXT AWARE SOLUTION

The conceptual definition of context information and context-aware routing enabled the proposal of a generic context-aware data routing mechanism. Thus, network devices such as routers and switches are able to choose accordingly the routing paths and traffic prioritization based on context information. This section introduces the proposed context model and architecture.

Context Aware Model

The context model applied in the proposed context-aware solution for adaptive routing can be used for both wireless and wired networks. This model is generic allowing the description of different network scenarios and the adaptation based on user´s experience.

The adopted context model describes the state of a particular entity (for instance, a user, router, switch, etc). Therefore, the following features describe this entity (Figure 1):

1. Individuality – which describes a particular information about an entity, such as identification, addressing, protocols, etc.;

2. Time – which describes time information, such as timestamp related to the status of an entity in a given instant;

3. Location – which is related to real or virtual location of an entity, and may be generated by a system such as GPS location, or by referencing information such as home, building, city, a network address, etc.;

4. Activity – which allows the description of explicit goals, tasks and actions performed by an entity, and;

5. Relations – which describes the entity's relationships with other entities, dependencies between entities, connections with objects, people, places, services, etc.

Some other aspects can still enhance the description of an entity, such as:

6. Quality of Experience (QoE) – which describes a group of parameters regarding user´s perspective, which is most of times rated as MOS;

7. Quality of Device (QoD) – which describes a group of parameters regarding devices characteristics such as capabilities, computational power, precision level of data colectors, and;

8. Quality of Service (QoS) – which is related to all the metrics (qualitative/quantitative) considered on an SLA between user and platform, such as (bandwidth, delay, jitter, etc.);

The context model can also be validated according to some metrics, which determine the Quality of Context:

9. Precision – level of information accuracy to assess its relevance;

10. Probability of Correctness – assessment of the probability of the information being correct;

11. Trustworthness – assessment of the level of trust on the information source;

12. Resolution – level of granularity of a given information;

13. Up-To-Dateness – assessing how the information provided is updated

Previous works introduced XML as a representation of data as well as a Relational Database Management System RDBMS implementation for the Context Controller. In this work, in order to represent the context model, MongoDB (MongoDB Architecture, 2015) has been adopted. MongoDB is a document-oriented NoSQL database based on collections adopting a data format able to optimize end-to-end data transfer and management without the need to carry out time-consuming parsing. Therefore, a Javascript Object Notation (JSON and BSON, 2015) description turned out to be useful since it can be handled directly using programming languages, such as Python.

The modular proposal for the context model allows components to be developed independently, even though they apply and manage the same common representation for the context information, as defined in JSON/BSON description presented previously.

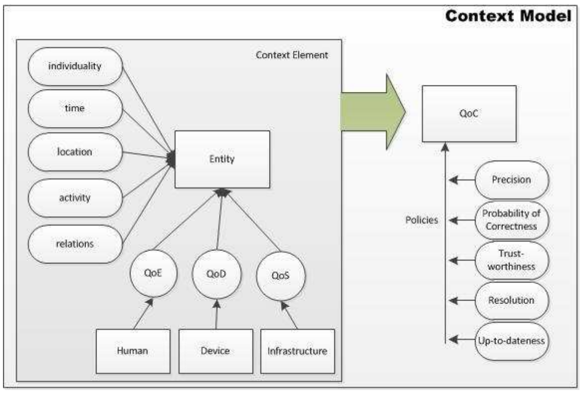

Figure 1: Context Model

The following excerpts from a JSON/BSON document illustrate the description of an user/application, regarding time, location, Network, QoS, QoE, and QoD.

Table 1: MongoDB view of a current representation


```
    "Location": {
      "latitude": -12.9882574,
      "longitude": -38.4586363,
      "altitude": "none",
      "atm_pressure": "none"
    },
    "Network": {
      "src_ip": "192.168.0.21",
      "flow_id": 17,
      "dest_port": "16000",
      "dest_port_low": "16000",
      "dest_port_high": "16038",
      "src_port": "16000",
      "src_port_low": "16000",
      "src_port_high": "16038",
      "dest_ip": "192.168.0.6"
        },
    "QoE": {
      "MOS": 4.409285999999998,
      "R-factor": 93.19999999999997,
      "time": "2015-05-18 15:03:52.632",
      "User_Perception": {
        "urgency": "false",
        "Mood": "false",
        "Back_Noise": "false",
        "MOS_u": 4
      }
    },
    "QoD": {
      "has_gps": "false",
      "gps_precision": "none",
      "process_cores": "two",
  "screen_resolution": "none",
  "processor_overload": "false",
  "resource_fault": "false"
},
    "QoS": {
  "p_loss": 0,
  "round_trip_delay": 93.19999999999997,
  "jitter": 3.4209333333333327
}
```

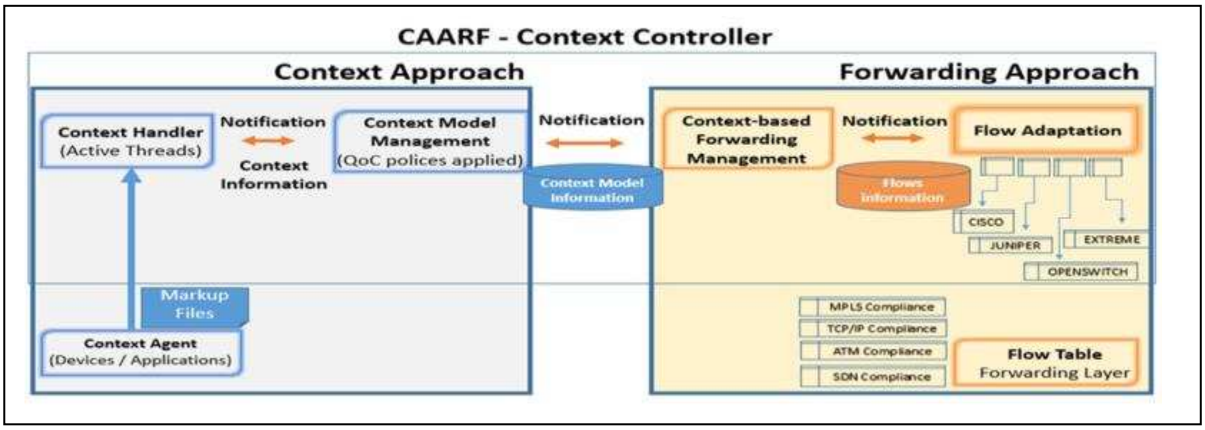

Figure 2: CAARF Context Controller

CONTEXT-AWARE ADAPTIVE ROUTING FRAMEWORK

After having introduced the context model to be applied and the format of the information to be exchanged within the proposed architecture, it is possible to understand how the contextual information is processed and forwarding updates affect the global performance of the network. The architecture of the proposed system was conceived based on the description of two main functional groups:

- Context Management Modules – which are responsible for the analysis and filtering of contextual information, and;
- Forwarding Management Modules – which are responsible for processing contextual information and for the application forwarding rules on network switches.

The main functionalities of the proposed architecture are:

- To collect and share contextual information among network devices
- To centralize storage of contextual information;
- To carry out validation and assessment of contextual information based on Quality of Context policies, and;
- To support the query and utilization of contextual information in order to update routing for context-sensitive services.

The proposed architecture called Context-Aware Adaptative Routing Framework (CAARF) was conceived based on the integration of different functional modules, as depicted in Figure 2:

- **Context Agent:** Responsible to receive contextual information from active communication devices, applications and users. These information is are forwarded to the context management module;
- **Context Handler**: Responsible to receive context information, record them on the context database within context model management module;
- **Context Model Management**: Responsible for storing contextual information, processing quality of context and making available information to context-based forwarding management module;
- **Context-based forwarding management**: Responsible for processing forwarding rules based on contextual information, and;
- **Flow adaptation**: Responsible for the notification of updated forwarding information on the respective active switching/forwarding communication device.

- **Context Agent:** Responsible to receive contextual information from active communication devices, applications and users. These information is are forwarded to the context management module;
- **Context Handler**: Responsible to receive context information, record them on the context database within context model management module;
- **Context Model Management**: Responsible for storing contextual information, processing quality of context and making available information to context-based forwarding management module;
- **Context-based forwarding management**: Responsible for processing forwarding rules based on contextual information, and;
- **Flow adaptation**: Responsible for the notification of updated forwarding information on the respective active switching/forwarding communication device.

Next section illustrates the implementation aspects of the proposed architecture.

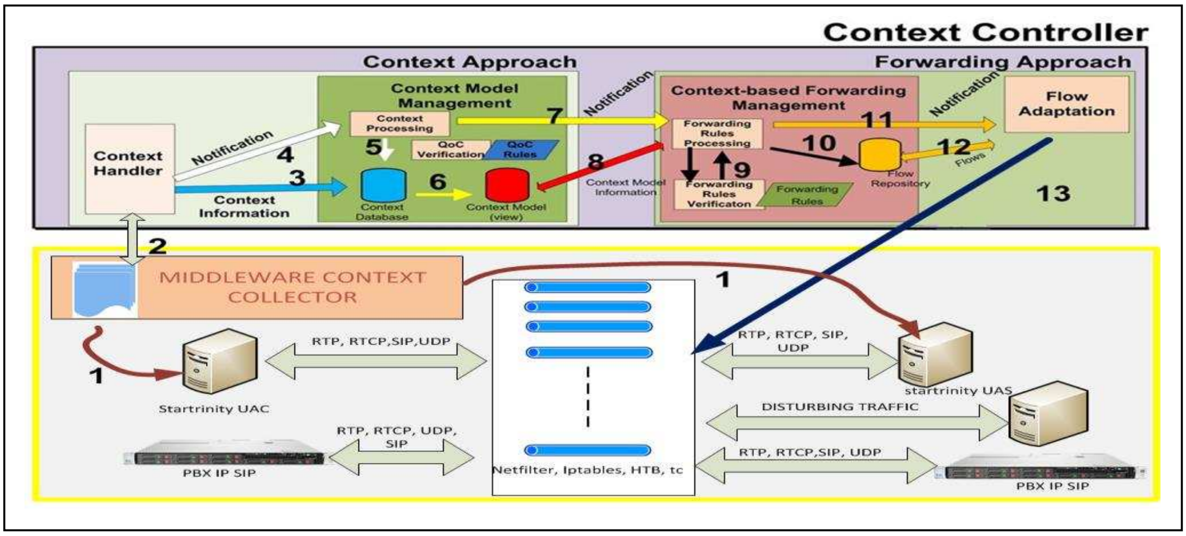

Figure 3: Context Controller Scenario Flows

CAARF: IMPLEMENTATION ASPECTS

In order to illustrate the operational and implementation aspects of the proposed architecture, consider the scenario depicted in Figure 3. As a case study, a pair of IP PBX, compatible to the SIP protocol, located in different networks was implemented. The traffic between both PBXs should pass through a common gateway, which supports traffic shaping. This is one of the cases supported by the Flow Adaptation module. Flow Adaptation supports a number of technologies and protocols such as MPLS, GMPLS, SPB, SDN-Like, Openflow and IP. In this work, we propose the implementation of traffic shaping using Netfilter (Iptables), as well as the utilization of queueing mechanisms and traffic control, htb and tc.

A SIP calling simulation software, Startrinity (Startrinity, 2015), has been applied in order to carry out simultaneous calls from an UAC (User Agent Client) to an IP PBX, in both directions, and complementarily to another SIP Tester Startrinity, as in the role of UAS (User Agent Server).

Initially, the data traffic generated and delivered without interruption will be measured with a respective MOS (Mean Opinion Score), which is related to R-factor, according to ITU-T rec G.107 specifications (Carvalho et al. 2005), meaning that each impairment on voice calls may be computed independently in order to achieve a score result. MOS is a score defined in a grade scale that starts from 1 (Not Acceptable) through 5 (Excellent). Therefore, MOS, will be collected by a Context Handler from data acquired by the Middleware Context Agent. At first, a good MOS reading will not affect routing decisions.

After some time, noisy traffic is generated, such as upstream and downstream traffic related to other concurrent traffic. This noisy traffic will lead to resources shortage, consequently affecting VoIP traffic with MOS violation against the agreed Service Level Agreement (SLA).

This violation should be processed accordingly in order to adapt routing. In order to better understand how the proposed architecture reacts to this scenario, the following events are numbered from 1 through 13:

1. **Middleware Context agents** collect periodically the contextual information from applications and network, such as location, timing, QoS, QoE and QoD, among other parameters;
2. **Context Middleware** prepares JSON contextual information to **context handler,** which will listen to the changes;
3. **Buffer management**, registers all information received in **context database** (repository located within **context model management**);
4. **Notification scheduler**, submodule of **context handler**, notifies **context model management** about the reception of incoming contextual information;
5. **Context Processing,** submodule of **context model management**, receives notification of new contextual information, retrieves information from **context database**, and submit them to the module **Quality of Context (QoC) verification**, which applies pre-defined QoC rules;
6. **Context processing**, submodule of **context model management**, registers a new context within **global context model**;
7. **Context processing**, submodule of **context model management**, notifies the module **context-based forwarding management** the context modification and QoE violations;
8. **Forwarding rules processing**, submodule of **context-based forwarding management**, receives notification

of new context and the results of queries to the **global context model**;

9. **Forwarding rules verification**, submodule of **context-based forwarding management**, queries **global context model** and applies pre-defined forwarding rules;

10. **Forwarding rules processing**, submodule of **context-based forwarding management**, records updated flows within **flow repository**;

11. **Forwarding rules processing**, submodule of **context-based forwarding management**, notifies **flow adaption** about the available updated flow to be applied on the routers;

12. **Flow adaptation** verifies new flow rules on the **current flow view**, repository within **context-based forwarding management**, and;

13. **Flow adaptation** applies new rules on network routers..

CASE STUDY

Considering the scenario depicted in Figure 3, the context database was implemented using the NoSQL MongoDB. Indeed, given the variety of existing context data, situations and context applicable, the description of a context with MongoDB turned out to be more effective in order to describe the extensive number of states and variables that the system should support. Moreover, MongoDB still provides a number of functions in order to describe queues, priority queues and stacks, which allow for an optimized context management and signaling between modules.

The developed system counts with a listening module (event handler) that is responsible for collecting data managed by the Middleware, which is already described using JSON/BSON and its respective MongoDB format. The QoC module is responsible for filtering the QoC events, discarding non-conformant data according to the system´s parameters or to the SLA agreement. For instance, it means outdated data, inaccurate data read from devices, SLA non-conformant data, inconsistent ports, divergent data (or out of scope concerning QoS, QoD), missing data, etc.

As an outcome of the filtering process, besides eliminating the detected inconsistencies, it still preserves the stability of the system, resulting in an event log database that will be processed by a system management module. At last, in this case study, oriented to converging applications a correlation between user/application QoE and SLA for a given service is also carried out.

Context Handler/Management processes the QoC rules as well QoE measurements validation against pre-defined set of user/application definitions, followed by a notification step, in the form of a QoE violation notification queue (Context Model Management → Forwarding Approach). This alert initiates a Flow Adaption routing decision reasoning.

In order to acquire data related to the collecting parameters, some monitoring and evaluation tools are employed. For this purpose, the Middleware Context Agent is responsible to interact with different available tools, being an external layer to the Context Controller that is in charge to collect data from sensors, probing and different measurement tools from multiple vendors.

The main advantage of this approach is the possibility to interact with different tools available among the community without the need to develop new collecting modules from scratch.

The calling Simulator, Startrinity, is able to generate SIP traffic logs and to provide parameters measurements for MOS, R-factor and other QoS parameters, such as jitter, packet loss and RTT. These parameters can be collected by the Handler, resulting in data already formatted by the Middleware collect agent, such as JSON format. In turn, the MongoDB database is made available on the cloud using the MongoLab (MongoLab, 2015) platform.

As a consequence, notifications are generated on the Forwarding module, and based on the routes evaluation map, which in this study case are implemented as forwarding queues according to the Netfilter definitions, and also based on the QoS records for that particular flow, the Forwarding module will modify the forwarding table. These modifications are also recorded on the events log, which will be useful for further generation of flow events reports.

In this experiment, successive rounds comprising fixed blocks of calls between the hosts configured as UAC (caller firing) and UAS (IP PBX role), have been performed. Bandwidth constraints are implemented through HTB rules applied to a Netfilter host, whose queues are defined as a sequence of step values ranging from (256Kbps, 700Kbps, 1400Kbps up to 2400kbps). Concurrent calls are then established in progressive scale, comprising a sequence [3,6,6,12,12,20,20] of simultaneous calls at each round, providing the system with the context data that will be processed by the Context Controller. Experiment rounds and queues changes in response to notifications issued against the Forwarding Module are shown in Table 2, and are related with MOS behavior according to Figure 5.

Table 2: Rounds and Queues Changes

Round	concurrent calls	Queue	BW, Kbps	Notify FWD
1	3	1:20	300	N
2	6	1:20	300	Y
2	6	1:40	700	N
3	12	1:40	700	Y
3	12	1:50	1400	N
4	20	1:50	1400	Y
4	20	1:60	2400	N

Context Handler acquires the data in each round and constantly updates the context database, located in MongoLab, as illustrated in another series of calls in Figure 4, while Context Model Management routines constantly assess the QoC and QoE status in order to meet SLA requirements. Successive QoE updates herein being understood as MOS computed values can trigger notifications towards Context Forwarding Module in the case of requirements violation, shown as a blue arrow in Figure 4.

These notifications will eventually generate readjustments through computing of the necessary bandwidth required by the new context and leading to the choice of a new queue applied on Netfilter/Iptables. In Figure 4 sequencial readings that represent the context data are presented as well part of the MongoDB samplings acquired from four different states associated with their respective time stamps.

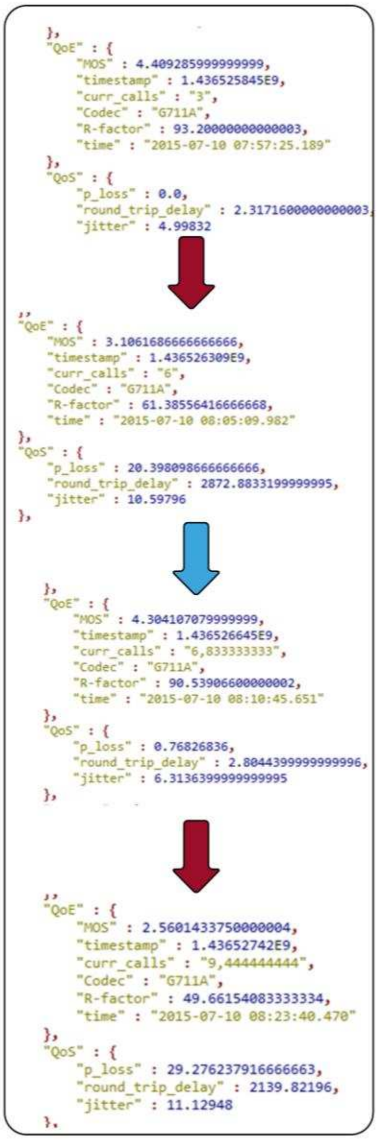

Figure 4: MOS recovery in between rounds

Each round is then related to a different allocated bandwidth, a number of concurrent calls, and MOS measured in each round. Those samples show exerted state views achieved through the action of Context Controller in order to recover the committed QoE level, towards MOS equivalent or above an integer value (4) in this experiment session, according to the user/application requirements settings.

Figure 5 presents the relation between MOS degradation and subsequent recovery achieved by the Context Controller. This graph reflects a series of calls and the action taken by Context Controller in order to recover the MOS based on SLA.

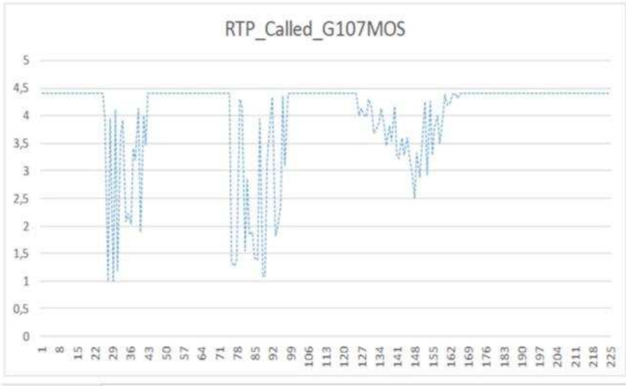

Figure 5: MOS adaptation along successive SIP calls

CONCLUSIONS

This paper discussed the effectiveness of Quality of Experience (QoE) techniques approach as an optimized solution for routing time-sensitive traffic regarding converged networks. In order to propose and focus on a user-centered solution, a solution that is capable of QoE evaluation has been proposed for the context-aware architecture.

The context model adopted is based on the description of the user´s quality of experience, quality of the device and quality of the infrastructure. The main goal is to propose a data model being scalable, flexible and generic and the JSON/BSON representation through MongoDB has been demonstrated as a powerful and flexible architecture that can lead to an extensive representation. Meanwhile, the proposed architecture provides main modules for monitoring and collecting user and network´s status, processing the changes in the status and proposing optimal routes based on the current status as well historical data, usage profile and behavior descriptions.

This work has been validated through a testbed that is capable of inferring results for different stress scenarios. As for future works, the proposed context model and architecture should be validated through the execution of different and more complex scenarios and context-aware update and routing optimization protocols, taking in account, as well, the user perception data acquired from different application usages. A QoC monitoring approach is also under development, meaning that more diverse results and scenarios will be addressed in future works built upon the present framework.

ACKNOWLEDGEMENTS

This project has been supported by FCT – Fundação para a Ciência e Tecnologia, within the Project Scope: **UID/EEA/50009/2013**.

REFERENCES

Alreshoodi, M.; Woods, J. (2013). *Survey on QoE\ QoS Correlation Models For Multimedia Services*. arXiv preprint arXiv:1306.0221.

Buchholz, T.; Schiffers, M. (2003).*Quality of Context: What It Is And Why We Need It*.Paper presented at 10th Workshop of the OpenView University Association (OVUA'03).

Carvalho, L., Mota, E., Aguiar, R., Lima, A. F., & Barreto, A. (2005, June). An E-model implementation for speech quality evaluation in VoIP systems. IEEE, Pages 933-938.

Dey, A. K.; Abowd, G. D.; Salber,D. (2001). *A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications*.Journal Human-Computer Interaction archive Volume 16 Issue 2, Pages 97-166

El-Gendy, M.A.; Bose, A.; Shin, K.G. (2003). *Evolution of the Internet QoS and Support for Soft Real-Time Applications*. Proceedings of the IEEE (Volume:91, Issue: 7)

Karthiga, S; Balamurugan, M. S. (2013). *Traffic Engineering System Based on Adaptative Multipath Routing*. International Journal of Emerging Technology and Advanced Engineering, Volume 3, Issue 2.

Shaikh, J.; Fiedler, M.; Collange, D. (2010). Quality of Experience from user and network perspectives. *annals of telecommunications-annales des télécommunications*, *65*(1-2), 47-57.

Manzoor,A.; Truong,H.; Dustdar, S. (2009).*Quality Aware Context Information Aggregation System for Pervasive Environments*.Published in Advanced Information Networking and Applications Workshops, WAINA '09. Pages 266 – 271.Bradford.

Mascolo, C.; Musolesi, M. (2006).*SCAR: Contextaware Adaptive Routing in Delay Tolerant Mobile Sensor Networks*. Published in international conference on Wireless communications and mobile computing. IWCMC '06.Pages 533-538.

Nazario, D. C.; Dantas, M. A. R.; Todesco, J. L. (2012). *Taxonomia das publicações sobre Qualidade de Contexto*. Published in Sustentable Business InternationalJournal, n.20.

Prechtel,L.; (2002). Are Scripting Languages Any Good ? A validation of Perl, Python, Rexx and Tcl against C, C++ and Jave. A chapter for Advances in Computers, Volume 58. University of Karlsruhe.2002.

Stankiewicz, Rafal, and Andrzej Jajszczyk. "A survey of QoE assurance in converged networks." *Computer Networks* 55.7 (2011): 1459-1473.

Vieira, V.; Salgado, A. C.; Tedesco, P. C. A. R. (2009). *Modelos e Processos para o Desenvolvimento de Sistemas Sensíveis ao Contexto*.PaperpresentedatJAI - XXVIII Jornadas de Atualização em Informática, Bento Gonçalves, RS.

Yasar, A.; Preuveneers, D.; Berbers, Y. (2010).Evaluation framework for adaptive context-aware routing in large scale mobile peer-to-peer systems. Published in Peer-to-Peer Networking and Applications, Volume 4, Issue 1, Pages 37-49.

Yerima, S. (2011). Implementation and evaluation of measurement based admission control schemes within a converged networks QoS management framework," *International Journal of Computer Networks & Communications*, vol. 3, no. 4, pp. 137-152.

Weiser, M. (1999).*The Computer for the 21st Century*.Published in ACM SIGMOBILE Mobile Computing and Communications Review - Special issue dedicated to Mark Weiser, Volume 3 Issue 3, Pages 3-11.

Zimmermann,A.; Lorenz,A.; Oppermann, R. (2007). *An Operational Definition of Context*. Published in 6th international and interdisciplinary conference on Modeling and using context (CONTEXT'07). Pages 558-571.

WEB REFERENCES

JSON AND BSON, available at https://www.mongodb.com/json-and-bson

MongoDB Architecture, available at https://www.mongodb.com/mongodb-architecture

Mongolab: Available at https://mongolab.com/

Startrinity SIP Tester, downloadable at http://www.startrinity.com/

JSON AND BSON, available at https://www.mongodb.com/json-and-bson