Cross-layer QoS-Aware Resource Allocation for IoT-Enabled Service Choreographies

Fábio M. Costa, Nikolaos Georgantas, Raphael Gomes, Ricardo da Rocha, Georgios Bouloukakis

Cross-layer QoS-Aware Resource Allocation for IoT-Enabled Service Choreographies

Fábio M. Costa
Institute of Informatics
Universidade Federal de Goiás
Goiânia, Goiás, Brazil
fmc@inf.ufg.br

Nikolaos Georgantas
Inria Paris
France
nikolaos.georgantas@inria.fr

Raphael de A. Gomes
Instituto Federal de Goiás
Goiânia, Goiás, Brazil
raphael.gomes@ifg.edu.br

Ricardo C. A. da Rocha
Instituto de Biotecnologia
Universidade Federal de Catalão
Catalão, Goiás, Brazil
rcarocha@acm.org

Georgios Bouloukakis
Donald Bren School of Information and Computer Sciences
University of California, Irvine, USA
Inria Paris, France
gboulouk@ics.uci.edu

ABSTRACT

Resource allocation for distributed systems is a well-known approach to deal with quality of service requirements. However, existing approaches do not consider the effects of resource allocation at the different levels of a system, especially when considering the end-to-end behavior of distributed compositions such as service choreographies. Based on our previous research on the optimized and QoS-aware resource allocation for service choreographies and publish-subscribe brokers, we outline a novel approach that integrates the treatment of resource allocation at both levels in order to yield a more precise control over end-to-end QoS. The paper presents an early architecture and discusses some of the main challenges towards realizing the approach.

CCS CONCEPTS

• **Applied computing** → **Service-oriented architectures**; • **Software and its engineering** → **Middleware**;

KEYWORDS

Middleware, Service choreographies, QoS, Resource allocation.

ACM Reference Format:
Fábio M. Costa, Nikolaos Georgantas, Raphael de A. Gomes, Ricardo C. A. da Rocha, and Georgios Bouloukakis. 2018. Cross-layer QoS-Aware Resource Allocation for IoT-Enabled Service Choreographies. In *5th Workshop on Middleware and Applications for the Internet of Things (M4IoT'18), December 10–11, 2018, Rennes, France.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3286719.3286726

1 INTRODUCTION

Large-scale distributed applications in the Future Internet will involve a multitude of heterogeneous services and resources that will need to be integrated and coordinated in a dynamic way to match evolving requirements and changing execution environments. Effective coordination in this context can be achieved through service choreography protocols [1], which enable services to coordinate themselves in a decentralized way to achieve meaningful collaboration. Nevertheless, achieving such distributed coordination in a way that preserves end-to-end QoS and makes efficient use of resources remains an open research problem.

In previous work, we proposed an end-to-end QoS-aware approach for the optimized allocation of resources to the services that take part in choreographies [7]. While the approach is effective in guaranteeing the QoS of service choreographies, its precision is limited due to the fact that it abstracts away the effect of the middleware layer on end-to-end QoS. Middleware components, such as pub/sub brokers, message queues, and service buses are an integral part of the data path, and thus also contribute to performance-related QoS attributes, such as end-to-end delay and throughput. A finer level of control over the QoS of service choreographies can thus be achieved by considering the middleware components, together with the application-level services as first-class entities with respect to resource allocation.

While the problem of QoS-constrained resource allocation for service choreographies (and indeed for service compositions in general) is not new, the approach presented in this paper differs from existing approaches by proposing a novel integration of four complementary architectural elements. Firstly, we adopt a cross-layer view of resource allocation, by considering both the application- and middleware-level entities involved, enabling greater control over the effects of resource allocation on QoS. Secondly, we use a proactive approach, which considers the service demand imposed by choreography clients, together with the capacity and QoS requirements of the services in a end-to-end manner prior to the actual choreography enactment, thus avoiding excessive QoS violations while still allowing runtime adaptation if needed. This is in contrast with reactive approaches, such as in [5] and [4], which start with a initial (default) resource configuration and adapt it as QoS violations are detected. Thirdly, as introduced in previous work [6], resource allocation at the middleware level takes into account the possibility of intermittent connectivity of services, thus contributing to make the approach feasible for service choreographies that involve mobile services and IoT devices. Finally, resource allocation takes into account the fact that services can be shared across multiple choreographies, enabling the optimized allocation of resources [7].

In this position paper, we review our previous work on enabling QoS-aware resource allocation, separately, for the application services layer and for the middleware layer, which in turn provides the building blocks for the integrated cross-layer resource allocation approach that we propose. We discuss a motivating scenario and its handling using the cross-layer approach in Section 2. In Section 3 we describe the foundation elements of the proposed approach, while Section 4 outlines an architectural model to realize it. In Section 5 we present a research agenda in the form of open research questions that need to be answered in order to achieve a concrete realization of the approach. Finally, Section 6 concludes the paper.

2 MOTIVATION

Consider the scenario depicted in Figure 1, which shows an application for controlling urban traffic based on the processing of video streams from cameras distributed across a city area. This application may be implemented as a choreography of IoT devices and services, without a central point of coordination and typically exploiting the flexible cloud-based allocation of resources for services.

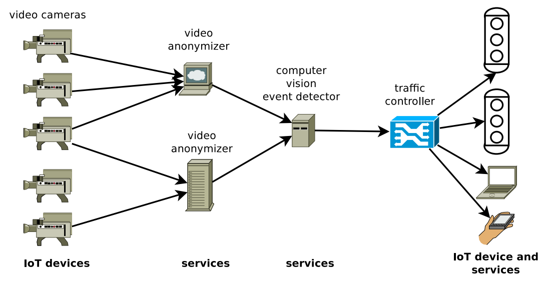

Figure 1: Motivation example of a choreography of services and IoT devices

The diversity and scale in this scenario suggest the adoption of appropriate communication middleware (e.g., pub/sub), in order to deal with heterogeneity, mobility, intermittent connectivity, faults, and scalability. Hence, the communication middleware and its corresponding allocated resources not only impact the effectiveness of message delivery but also its end-to-end QoS, particularly in terms of message delays and throughput.

The allocation of suitable resources for the participating services is a common strategy to improve QoS, and it is relatively well-supported by cloud providers. However, each service is typically considered in isolation, both from other services and from the underlying middleware, thus lacking coordination when allocating resources for an entire choreography of services, and not allowing the system to consider overall goals in terms of end-to-end QoS. Moreover, dealing with the intermittent connectivity of mobile services and IoT devices may require different resource allocation actions at both levels, e.g., to deal with voluntary disconnections at the application services level, and with forced network disconnections at the middleware level. This motivates the need for jointly managing the allocation of resources for both services and their underlying communication middleware (e.g., publish-subscribe brokers), which in turn may require managing the contribution of each service and middleware component towards the end-to-end QoS.

We organize the resource allocation problem in two layers of QoS concerns, depicted in Figure 2: (i) *application layer*, which comprises each service or IoT device that participates in the choreography; and (ii) *middleware layer*, which comprises the components of the middleware infrastructure (e.g., brokers) that enable the communication links among services and IoT devices. We argue for a holistic approach for resource allocation that acts at both layers, taking into account the contribution of each application- or middleware-level component towards the end-to-end QoS. Hence, an effective strategy for resource allocation should carry out coordinated cross-layer actions, interrelating local choreography adaptations with their effect on the global, end-to-end, QoS.

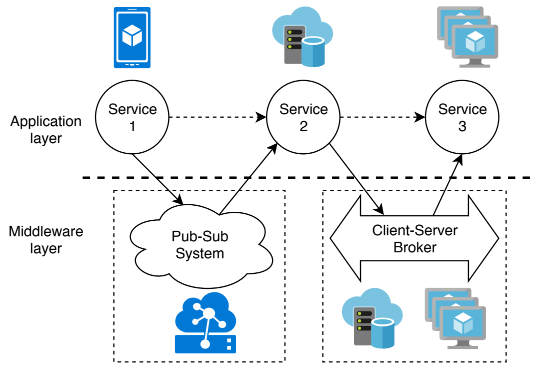

Figure 2: Cross-layer resource allocation

3 EARLY EVIDENCE

From the above discussion, it is clear that accurate QoS-aware resource allocation depends on taking into account both the application and middleware layers. In this section, we outline elements of our previous work that consider each of these levels separately, and which can be used as the basis for a cross-layer approach.

3.1 Resource Allocation for Services

In previous work [7], we addressed the problem at the application layer, by proposing a QoS-aware approach for resource estimation and selection when deploying multiple service choreographies with shared services. Given a high-level description of service choreographies and related constraints, the approach enables autonomic resource allocation in a hybrid cloud environment, with multiple cloud providers, whilst decreasing resource utilization costs.

We formalize the resource allocation problem for service choreographies and solved it using a matching strategy based on the Multiple-choice Multi-dimension Knapsack Problem (MMKP) [9]. We use a graph structure to represent the combined view of all service choreographies, so as to enable the matching between services and available resources. As part of this process, it is necessary to

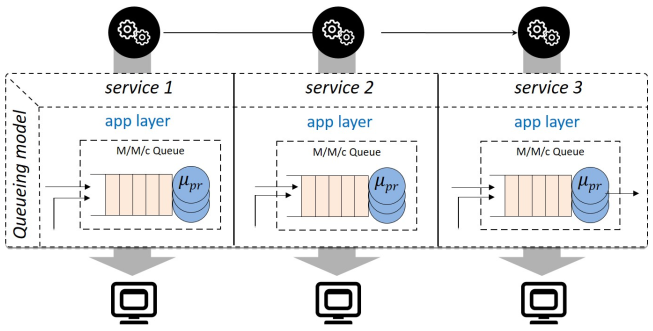

Figure 3: Performance model for the application layer in service choreographies.

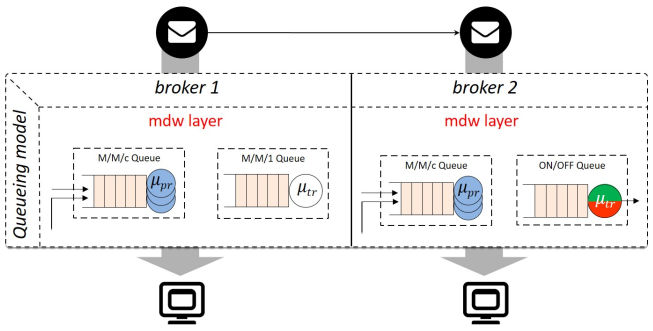

Figure 4: Performance model for the middleware layer in pub/sub systems.

estimate the QoS contribution of each service on all end-to-end paths in the multiple choreographies in which the service takes part. For simplicity, we only considered performance related QoS attributes. In order to estimate the QoS contribution of each candidate resource we rely on *queuing theory* [8].

As illustrated in Figure 3, resource selection is carried out by analyzing each end-to-end interaction flow in the service choreography definitions provided as input. For each pair of service and candidate resource, the expected QoS contribution in the flow is estimated using as input the load from the related choreography, as well as the cumulative load from other choreographies that also use the same service. Moreover, to model the processing rates of messages passing through each service, we use an $M/M/c$ queue – c is the number of processor cores of each candidate resource and μ_{pr} is the rate for processing messages at each core.

In [7], we focused only on application-level services and abstracted away the effects of middleware components on resource utilization as well as on the transmission delay when carrying out the performance analysis. By doing so, we were able to focus on the specificities of service sharing among choreographies. However, we recently extended the proposed approach to target resource allocation for the middleware layer, as described next.

3.2 Resource Allocation for Pub/Sub Brokers

Publish/Subscribe middleware protocols decouple peers (publishers and subscribers) in both time and space. They increase successful

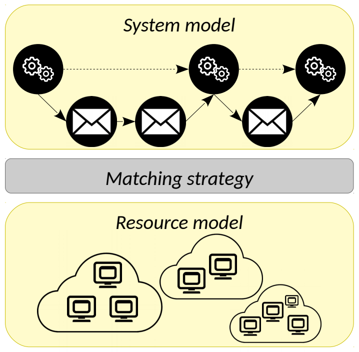

Figure 5: Overall architecture.

interactions especially when peers are intermittently connected. Additionally, they introduce mechanisms for event routing and transmission. To support distributed applications spanning a wide-area, the pub/sub system has to be implemented as a set of independent, communicating brokers, forming a *broker overlay*. An important design-time decision is the deployment of each broker to an appropriate machine (resource allocation). While a number of resource allocation solutions exist, they mainly focus on the satisfaction of local QoS requirements (e.g., employing load balancing strategies for individual jobs) lacking support for end-to-end non-functional properties.

In [6], we provide a QoS-aware solution for allocating resources to pub/sub brokers by taking into account the end-to-end response time between peers, as well as their intermittent connectivity. As shown in Fig. 4, we follow a similar approach as the one described in Section 3.1 by modeling the middleware layer of interacting peers and using a matching strategy. In particular, by relying on [2, 3] we model each broker by using an input and an output queue. The input queue is an $M/M/c$ (μ_{pr} is the core's event processing rate). The output queue can be either an $M/M/1$ or an ON/OFF queue. In both cases the μ_{tr} is used to model the transmission delay of events inside the network. ON/OFF output queues are used to model intermittently connected subscribers, while $M/M/1$ queues are employed to model event transmission between brokers.

It is worth noting that in [6] we ignore possible application layer event processing, which is targeted in the approach proposed here.

4 OVERALL ARCHITECTURE

In our ongoing work, we are combining the building blocks described in the previous section as part of an integrated approach for cross-layer QoS-aware resource allocation for service choreographies. We are currently considering choreographies where services are connected by pub/sub middleware, notably aiming at supporting services that represent IoT devices. We now outline the main elements of the system architecture.

Our initial proposal for the combined approach is to model both levels as a single optimization problem where the goal is to jointly allocate resources for application-level services and for message brokers, subject to common QoS constraints. In doing so, the joint approach matches the system elements (application-level services and middleware components) to resources taking into account the intermittent behavior of communicating peers. As illustrated in Figure 5, the input for the proposed matching strategy includes:

- the system model, which comprises a choreography-like description of the service interaction flows and inter-service communication paths, as well as the queuing models described in the previous section and the related QoS constraints; and
- the resource model, which describes the available resource types in a multi-cloud environment.

The aforementioned approach relies on the fact that the individual solutions presented in Section 3 are essentially based on the same strategy. More precisely, we abstract the application- and middleware-level entities to be deployed/inspected as a group where each entity can participate in multiple contexts, such as multiple service choreographies in the application layer and multiple message routing paths in the middleware layer. However, some open research questions must be answered (as highlighted next), especially if we take into account heterogeneous communication paradigms instead of only pub/sub.

5 OPEN RESEARCH QUESTIONS

An initial concrete realization of the approach proposed in this paper is part of our ongoing work. We are currently investigating the interaction of the queuing models used at the two levels (application and middleware) considering end-to-end paths across multiple service choreographies and the use of pub/sub middleware. Although in previous work we have validated the queuing network model for arbitrary paths on a network of pub/sub brokers, we relied on predefined paths between the communicating entities. Providing a solution that considers the multiple, possibly dynamic paths (and their corresponding loads) between services in a multi-choreography scenario is still an open issue.

We also need to refine the system model to represent the mapping between choreography flows and communication paths at the middleware level. The model should consider the possibility of having multi-hop paths at the middleware level to connect adjacent services in a choreography. Moreover, it must cater for the possibility that these paths may change dynamically.

Another important issue relates to the likelihood that multiple middleware technologies, representing different interaction paradigms (including client-server middleware and message queues), may be needed in large-scale distributed service choreographies. The QoS and resource allocation models need to consider such heterogeneity. For instance, while the middleware layer in the case of pub/sub brokers is typically allocated on separate hosts, for client-server, the middleware components are usually collocated with the application components (e.g., in the form of client and server libraries). Moreover, it may also be necessary to consider the resources requirements of the interoperability mechanisms used to connect the different middleware paradigms.

Finally, while the proposed approach assumes that application-level services and middleware components can be seamlessly treated as first class entities, further research is needed to refine this assumption. Notably, while application services can be shared across multiple choreographies, middleware components can be shared by different services that take part on the same or different choreographies. That is, multiplexing is more complex at the middleware level, and we need to investigate how to address this issue in the system model. Similarly, regarding the treatment of intermittent connectivity proposed in [6], and as suggested in Section 2, we need to investigate whether a simple extension of the use of ON/OFF queues to the application level (to model the voluntary disconnections of mobile services and IoT devices) would be consistent with the treatment of disconnections at the middleware level, especially on an end-to-end basis.

6 CONCLUSION

In this position paper, we leverage on previous research to propose a novel approach for QoS-aware resource allocation for the application and middleware components that take part in distributed service choreographies. The approach takes into account the possibility of having services with intermittent connectivity, thus paving the way to support choreographies that involve IoT devices and mobile services. The existing building blocks were described, followed by an initial architecture that integrates them. We are currently working on a concrete realization in order to validate the approach and shed light on the research challenges identified so far.

ACKNOWLEDGMENTS

The authors would like to thank FAPEG and Inria for funding the ACHOR Associate Team, within which this work was developed.

REFERENCES

[1] A. Barker, C. D. Walton, and D. Robertson. 2009. Choreographing Web Services. *IEEE Transactions on Services Computing* 2, 2 (April 2009), 152–166. https://doi.org/10.1109/TSC.2009.8

[2] Georgios Bouloukakis, Nikolaos Georgantas, Ajay Kattepur, and Valérie Issarny. 2017. Timeliness evaluation of intermittent mobile connectivity over pub/sub systems. In *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering*. ACM, 275–286.

[3] Georgios Bouloukakis, Ioannis Moscholios, Nikolaos Georgantas, and Valérie Issarny. 2018. Simulation-based Queueing Models for Performance Analysis of IoT Applications. In *11th International Symposium on Communication Systems, Networks, and Digital Signal Processing (CSNDSP)*. Budapest, Hungary. https://hal.inria.fr/hal-01797930

[4] Antonio Bucchiarone, Cinzia Cappiello, Elisabetta Di Nitto, Raman Kazhamiakin, Valentina Mazza, and Marco Pistore. 2010. Design for adaptation of service-based applications: Main issues and requirements. In *Service-Oriented Computing. ICSOC/ServiceWave 2009 Workshops*. Springer, 467–476.

[5] T. Furtado, E. Francesquini, N. Lago, and F. Kon. 2014. Towards an Enactment Engine for Dynamically Reconfigurable and Scalable Choreographies. In *2014 IEEE World Congress on Services*. 325–332. https://doi.org/10.1109/SERVICES.2014.64

[6] Raphael Gomes, Georgios Bouloukakis, Fábio Costa, Nikolaos Georgantas, and Ricardo da Rocha. 2018. QoS-Aware Resource Allocation for Mobile IoT Pub/Sub Systems. In *Internet of Things – ICIOT 2018*, Dimitrios Georgakopoulos and Liang-Jie Zhang (Eds.). Springer International Publishing, Cham, 70–87.

[7] Raphael Gomes, Júnio Lima, Fábio Costa, Ricardo da Rocha, and Nikolaos Georgantas. 2016. A Model-Based Approach for the Pragmatic Deployment of Service Choreographies. In *Advances in Service-Oriented and Cloud Computing*, Antonio Celesti and Philipp Leitner (Eds.). Springer International Publishing, Cham, 153–165.

[8] Edward D Lazowska, John Zahorjan, G Scott Graham, and Kenneth C Sevcik. 1984. *Quantitative system performance: computer system analysis using queueing network models*. Prentice-Hall, Inc.

[9] Martin Moser, Dusan P Jokanovic, and Norio Shiratori. 1997. An algorithm for the multidimensional multiple-choice knapsack problem. *IEICE transactions on fundamentals of electronics, communications and computer sciences* 80, 3 (1997), 582–589.