Architectural Issues for Self-adaptive Service Migration Management in Mobile Edge Computing Scenarios

Vittoria De Nitto Personè and Vincenzo Grassi
Dept. of Civil Engineering and Computer Engineering
Università di Roma Tor Vergata
Roma, Italy
denitto@ing.uniroma2.it, vincenzo.grassi@uniroma2.it

***Abstract*—We present a simple analytical model as a first step toward a thorough investigation of the effectiveness of different architectural solutions for the management of service migration in mobile edge computing scenarios.**

***Keywords*-mobile edge computing; service migration; self-adaptation; software architecture.**

I. Introduction

Mobile edge computing (MEC) architectures are widely recognized as a fundamental infrastructure to effectively augment the capabilities of personal portable devices (e.g., smartphones), to make them able to fully support advanced and personalized services like augmented and virtual reality, speech recognition, machine learning. Indeed, with respect to traditional cloud architectures based on relatively few large data-centers, they appear more suitable to meet latency, responsiveness and privacy requirements of these advanced services [4]. However, once a personal device has connected itself to the service(s) offered by a nearby cloud node, user mobility could bring the device far from that node, thus conflicting with the claimed advantage of MEC infrastructures. Service migration has been proposed as a way to overcome this problem, where virtualized service instances migrate between the nodes of the MEC infrastructure to remain close to the devices connected to them [3], [5], [7], [10]. To make this approach really effective, the advantages of service migration (maintaining minimal distance between the mobile device and the external services) should not be outweighed by its costs (e.g., temporary service unavailability, consumption of infrastructure resources). In this respect, some recent papers focus on the definition of migration policies aimed at determining when and where a migration should actually occur, to dynamically adapt the service(s) location to the mobility of personal devices [2], [5], [6], [8], [9], [12]. However, little attention is given to issues concerning how the resulting self-adaptive service migration management system is actually architected. These issues include, for example, how the main responsibilities are divided among a suitable set of components, where these components are located (e.g., single or distributed locations), and how they coordinate their activities (e.g., centralized or decentralized coordination). Different choices for these issues could lead to architectures that have different impacts on the overall effectiveness of a service migration system for MEC infrastructures.

Within this framework, this work intends to be a first step toward a thorough investigation of architectural issues in the design of a self-adaptive service migration management system for MEC infrastructures. To this end, in Section II we make explicit the underlying architectural choices of existing proposals, and suggest an alternative architecture. Then, we present in Section III a simple yet sufficiently expressive modeling approach that allows us, through a simple ”back of the envelope” analysis, to get some first insight on the impact of some possible architectural solutions, thus providing the basis for a future more articulated and refined investigation.

II. Architectural Issues

Our focus is on the management subsystem of self-adaptive service migration systems for MEC scenarios. To investigate architectural issues involved in its design, we refer to the well-established MAPE-K pattern [1], and on some of its refinements, in particular those discussed in [11].

As a first step, we explicit the respective roles of the four components of the basic MAPE-K feedback loop in the MEC service migration scenario we are considering. The Monitor (M) component collects information that certainly includes the distance between the mobile devices and the nodes hosting the services they are connected to. Possibly, it could collect other information, e.g. the load state of different nodes, to allow informed decisions about service migration. Based on this information, Analyze (A) and Plan (P) components decide whether some service should be migrated to bring it closer to the device connected to it. In the positive case, the migration is actuated by the Execute (E) component.

We observe that existing proposals implicitly adopt one of the following refinements of the basic MAPE model:

- *partially distributed architecture with centralized control* [6], [8]: the managing subsystem is based on a hierarchical architecture, with a centralized controller that implements the A and P components, while the

M and E components are distributed among the MEC infrastructure nodes (mobile devices and cloud centers);

- *fully distributed architecture with decentralized control* [2], [5], [9], [12]: the managing subsystem is fully distributed among the MEC infrastructure nodes with each node hosting dedicated instances of all components of a MAPE loop; each MAPE loop controls the service migration only for a single $\langle$service - mobile device$\rangle$ pair.

Both these models can be matched with patterns for the control of self-adaptive systems identified in [11]. In particular, the former model closely corresponds to the *master-slave* pattern, while the latter corresponds to the *information sharing* pattern, in particular its variant where information sharing about the system state is only indirectly achieved by M components through their independent observations of state variations. According to the discussion in [11], these two patterns present complementary pros and cons with respect to scalability, flexibility and possibility of achieving global quality goals. We refer to [11] for their thorough presentation.

As an alternative architecture, not yet considered in literature, we suggest the *regional planning* pattern proposed in [11]. According to it, M and A components are replicated at each mobile device. As result of their analysis, a service migration request could be issued towards a centralized P component. The P component collects these requests, and triggers the activation of a subset of them, based on global optimization and fair use objectives.

With respect to the master-slave architecture formerly considered, this latter architecture should in principle allow a substantial reduction of the data traffic caused by the managing subsystem operations, thanks to the locally performed monitoring and analysis. On the other hand, the presence of a centralized planning component should allow to avoid most of the problems caused by uncoordinated local adaptation actions that characterize the fully distributed architecture.

III. Modeling and Analysis

In this section we present a simple analytical modeling approach aimed at giving some first insights about the effectiveness of different architectural choices for the self-adaptive management of service migration. For space reasons, we focus on the *fully decentralized* architecture, and the *regional planning* architecture we are suggesting.

System model: We consider a MEC infrastructure consisting of a set D of homogeneous mobile devices, and a set N of homogeneous network-connected edge nodes offering cloud-based services. Each mobile device $d \in D$ has a *personal service* PS_d associated with it, deployed at a node $n \in N$, which augments the mobile device capabilities. The quality of the service delivered by PS_d degrades for increasing $distance$ between d and the node hosting PS_d. To this end, we adopt a coarse-grained distance model consisting of only two possible states, used to discriminate between *optimal* (e.g. one hop) and *nonoptimal* distance between d and PS_d. This distance model can be considered as a simplification of more detailed models presented in literature, where all states representing distances greater than the optimal one have been collapsed into a single state.

Mobility model: We assume a discrete time model, where events of interest (e.g., device mobility, service migration) may take place at the end of each time slot. Let $dist_i^d \in \{opt, nonopt\}$ denote the distance state of a device d with respect to the node currently hosting its PS_d, at time slot t_i, $i \in \{0, 1, \dots\}$. The following simple mobility model defines how $dist_i^d$ possibly changes in the next time slot:

- $p_{out}^d = Pr\{dist_{i+1}^d = nonopt | dist_i^d = opt\}$
- $p_{back}^d = Pr\{dist_{i+1}^d = opt | dist_i^d = nonopt\}$

Hence, p_{out}^d is the probability that, because of its mobility, d is no longer at optimal distance from its PS_d in the next time slot, while p_{back}^d is the probability that d returns at optimal distance from PS_d. As an example of mobility patterns that can be captured by this model, $p_{back}^d \approx 0$ could model "constrained" mobility towards a destination (e.g. a vehicle on a road), while values of p_{back}^d closer to 1 could model a more "wandering" behavior. Different values of p_{out}^d could instead be used to model different degrees of "stationarity".

Cost model: The generic $cost$ we refer to in the following should be intended as a measure of some quality-related penalty, e.g. service latency, service downtime, etc. We assume that for each time slot a mobile device d spends at distance $nonopt$ from its PS_d, it accumulates a *cost* c_D. This cost is instead zero when PS_d is at optimal distance.

When the distance between d and PS_d is *nonopt*, a decision can also be taken (we discuss later who and how will take this decision) to migrate PS_d at *opt* distance from d. In this case however, d accumulates a cost c_M caused by the migration (e.g., caused by the temporary unavailability of PS_d). Moreover, the migration of PS_d causes also the accumulation of a *system cost* c_S, which is intended to measure quality-related penalties that do not directly affect d, but could have an impact on the overall system (e.g., overutilization of shared network resources).

Migration control model: To carry out our "back of the envelope" analysis, we consider two simplified instances of a MEC service migration system, respectively architected according to the fully decentralized and regional planning pattern. In the *fully decentralized case*, all the four components of a MAPE loop are fully instantiated at each mobile device and control the migration of its PS_d only. In particular, we assume that the A and P components operate according to a *threshold-based policy*, where they keep track of the number of consecutive time slots d spends at $nonopt$ distance from PS_d; when this number reaches a threshold value K (whose optimal value is locally calculated), a migration directive is issued and immediately executed, which moves PS_d at *opt* distance from d. In the *regional planning case*, the M and

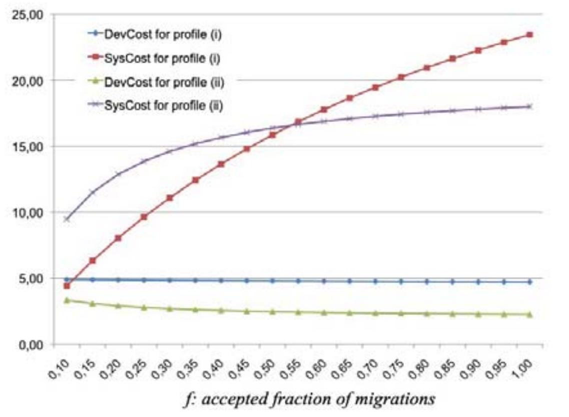

Figure 1. Device and system costs.

A components are still instantiated at each mobile device d, and operate according to the same threshold-based policy defined above. However, differently from before, when the number of consecutive time slots spent at $nonopt$ distance from PS_d becomes equal or greater than K, this event is only reported to the centralized P component without triggering a migration. P collects these notifications and issues actual migration directives to only a subset of the "migratable" PS_d instances, with the goal of avoiding an uncontrolled use of system resources. To this end, P accepts only a fraction f ($0 \leq f < 1$) of the migration requests that result active in a time slot. A mobile device d whose migration request is not fulfilled keeps it holding in the next time slots, unless its mobility brings it again at *opt* distance from its PS_d; in that case, the migration request is canceled.

From the model defined above, we can easily derive by using basic probability and renewal theory arguments the following closed-form expressions for the average cost per unit time experienced by a mobile device or by the system:

$$DevCost(f) = \frac{c_D \overline{T}_{nonopt}}{\overline{T}_{opt} + \overline{T}_{nonopt}} + c_M(1-p_{back}) f \pi_K(f) \quad (1)$$

$$SysCost(f) = |D| c_S (1 - p_{back}) f \pi_K(f) \quad (2)$$

where $\overline{T}_{nonopt}(f)$, $\overline{T}_{opt}(f)$ and $\pi_K(f)$ denote, respectively, the average time spent in states $nonopt$, opt between two consecutive visits to state opt, and the steady-state probability of spending K or more consecutive time slots in state $nonopt$ (closed-form expressions for them are not shown here for space reasons). We have evidenced in (1) and (2) the dependence on f. When $f < 1$ they apply to the regional planning architecture, while for $f = 1$ they apply to the fully decentralized architecture. Fig. 1 shows results obtained from this model, for two application and mobility profiles: (i) $p_{back} = 0.01, p_{out} = 0.9, c_D = c_M = c_S = 5$ (e.g., light-state application and vehicle moving along a highway), and (ii) $p_{back} = 0.01, p_{out} = 0.1, c_D = 5, c_M = c_S = 20$ (e.g., heavy-state application and person commuting among a few places). In both cases we see that, with respect to the fully decentralized architecture ($f = 1$), the migration regulation ($f < 1$) allowed by the regional planning architecture can drastically reduce the system cost, causing only a slight increase of the mobile device cost. These results thus suggest that it is worth considering the latter architecture, and give first indications about scenarios where it could be advantageous. As next steps of this work, we plan to extend the model to the master-slave architecture, and then to use it to get basic insights about the effectiveness of the three architectures under different mobility and cost scenarios. These results will then drive a thorough investigation on a more detailed model.

References

[1] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *IEEE Computer*, 36(1):41–50, 2003.

[2] A. Ksentini, T. Taleb, and M. Chen. A markov decision process-based service migration procedure for follow me cloud. In *IEEE ICC 2014* , pp. 1350–1354, 2014.

[3] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis. Live service migration in mobile edge clouds. *Wireless Commun.*, 25(1):140–147, Feb. 2018.

[4] M. Satyanarayanan. The emergence of edge computing. *IEEE Computer*, 50(1):30–39, 2017.

[5] T. Taleb, A. Ksentini, and P. Frangoudis. Follow-me cloud: When cloud services follow mobile users. *IEEE Trans. on Cloud Computing*, page 1.

[6] R. Urgaonkar, S. Wang, T. He, M. Zafer, K. S. Chan, and K. K. Leung. Dynamic service migration and workload scheduling in edge-clouds. *Perform. Eval.*, 91:205–228, 2015.

[7] K. Wang, M. Shen, J. Cho, A. Banerjee, J. Van der Merwe, and K. Webb. Mobiscud: A fast moving personal cloud in the mobile network. In *5th Workshop AllThingsCellular '15*, pp. 19–24, NY, USA, 2015. ACM.

[8] S. Wang, R. Urgaonkar, T. He, K. Chan, M. Zafer, and K. K. Leung. Dynamic service placement for mobile micro-clouds with predicted future costs. *IEEE Trans. Parallel Distrib. Syst.*, 28(4):1002–1016, Apr. 2017.

[9] S. Wang, R. Urgaonkar, T. He, M. Zafer, K. S. Chan, and K. K. Leung. Mobility-induced service migration in mobile micro-clouds. *IEEE Mil. Comm. Conf.*, pp. 835–840, 2014.

[10] S. Wang, J. Xu, N. Zhang, and Y. Liu. A survey on service migration in mobile edge computing. *IEEE Access*, 6:23511–23528, 2018.

[11] D. Weyns, B. R. Schmerl, V. Grassi, S. Malek, R. Mirandola, C. Prehofer, J. Wuttke, J. Andersson, H. Giese, and K. M. Göschka. On patterns for decentralized control in self-adaptive systems. In R. de Lemos, H. Giese, H. A. Müller, and M. Shaw, eds., *Software Engineering for Self-Adaptive Systems II* , vol. 7475 *LNCS*, pp. 76–107. Springer, 2010.

[12] C. Zhang, and Z. K. Zheng. Task migration for mobile edge computing using deep reinforcement learning. *Future Generation Comp. Syst.*, (96):111–118, 2019.