National Technical University of Athens

School of Electrical and
Computer Engineering

Network Management and
Optimal Design Laboratory

Trust-aware Life-cycle Management of Federated Edge Clouds

(ΔΙΑΧΕΙΡΙΣΗ ΚΥΚΛΟΥ ΖΩΗΣ ΟΜΟΣΠΟΝΔΩΝ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΝΕΦΩΝ ΣΤΑ ΑΚΡΑ ΤΟΥ ΔΙΚΤΥΟΥ ΜΕ ΓΝΩΜΟΝΑ ΤΗΝ ΕΜΠΙΣΤΟΣΥΝΗ)

Thesis submitted for the degree of Doctor of Philosophy

of

Konstantinos Papadakis-Vlachopapadopoulos

This dissertation was made possible through a scholarship funded by the Greek State Scholarships Foundation (IKY), that was co-financed by Greece and the European Union (European Social Fund-ESF) through the Operational Programme "Human Resources Development, Education and Lifelong Learning", in the context of the project "Strengthening Human Resources Research Potential via Doctorate Research", 2014-2020 (MIS-5000432)

Athens
October 4, 2021

National Technical University of Athens
School of Electrical and Computer Engineering
Communication, Electronic and Information Engineering
Network Management and Optimal Design Laboratory

Trust-aware Life-cycle Management of Federated Edge Clouds

(ΔΙΑΧΕΙΡΙΣΗ ΚΥΚΛΟΥ ΖΩΗΣ ΟΜΟΠΣΠΟΝΔΩΝ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΝΕΦΩΝ ΣΤΑ ΑΚΡΑ ΤΟΥ ΔΙΚΤΥΟΥ ΜΕ ΓΝΩΜΟΝΑ ΤΗΝ ΕΜΠΙΣΤΟΣΥΝΗ)

Thesis submitted for the degree of Doctor of Philosophy

of

Konstantinos Papadakis-Vlachopapadopoulos

Advisory Committee: Symeon Papavassiliou
Theodora Varvarigou
Ioanna Roussaki

Approved by the seven-member committee on October 7, 2021

.....	.....	.....
S. Papavassiliou	T.Varvarigou	I. Roussaki
Professor NTUA	Professor NTUA	Asst. Professor NTUA
.....	.....	.....
G. Matsopoulos	V. Karyotis	E. Sykas
Professor NTUA	Associate Professor	Professor NTUA
	Ionian University	
	.....	
	D. Kaklamani	
	Professor NTUA	

Athens, October 4, 2021

……

Κωνσταντίνος Παπαδάκης-Βλαχοπαπαδόπουλος

Διδάκτωρ Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών ΕΜΠ

Abstract

In the 5G and IoT era, service delivery and management models become more and more complex, by combining a wide variety of technologies and devices. The complexity of the new service delivery models raises new challenges for the fulfilment and management of the service life-cycle, from service discovery and selection, to orchestration, management and monitoring. In addition, in such context, where unknown users, devices and providers need to interact to achieve service fulfillment, it is critical to provide a sense of trust between users. The motivation for this dissertation is to tackle the new challenges arising in the service life-cycle management and enable trust in such scenarios.

In the first part of this dissertation, our main focus is to assess services over heterogeneous resources, considering Quality of Service (QoS) and Quality of Experience (QoE) by combining monitoring data, the subjective perception of various users and their unique requirements. To achieve this, we aim to quantify the service assessment with the above criteria, as a reputation score value, that represents the performance and reliability of the service in a fair manner, while guaranteeing protection from malicious actors trying to manipulate it. This value can enable trust towards service providers while assisting users in the service selection process taking into account their unique priorities.

To achieve the aforementioned goals, we developed collaborative Service Level Agreements (SLA) and Reputation Trust Management (RTM) frameworks with focus on cloud federations. Cloud Federation is the deployment, management and collaboration of several cloud computing services. It can integrate private, community, and public clouds providing multiple benefits such as scalability, fault tolerance and elasticity to the cloud environment while raising new challenges, namely interoperability issues of heterogeneous clouds and lack of trust among different providers. In this environment, we use SLA, in order to allow users to enforce guarantees about the performance of consumed services. We extend Key Performance Indicators (KPIs) assessed by the SLAs to surpass the limited availability KPI provided by most cloud and service providers at the moment, towards better addressing the specific requirements of each user.

Furthermore, for the RTM aspect of our collaborative framework, we focus on Multi-Criteria Decision Making (MCDM) techniques for the quantification of trust and the computation of the reputation values. We adopt MCDM methodologies due to their capability of receiving as input multiple KPIs combining QoS and QoE metrics. In our research work, we modified and extended two known scalable fuzzy-based MCDM techniques, namely Fuzzy VIKOR and Fuzzy Analytic Hierarchical Process (AHP). The first one is horizontal while the latter one is a hierarchical MCDM system. The key contributions of our extensions in those systems mainly refer to their capability of facilitating inputs, not only limited to fuzzy numbers, but also to include linguistic and binary values. In this way, we can combine a wide variety of QoS and QoE aspects in the computation of the reputation score value, guaranteeing that a high reputation value represents wholesomely the provider's performance and reliability. We also modified those techniques to allow users to provide custom weights in the evaluation process to express different requirements.

Last but not least, we introduce in both approaches credibility mechanisms that compare the user's evaluation with SLA and monitoring measurements in order to detect malicious actors and protect providers from reputation score manipulation attempts.

The implementation and experimental validation and assessment of our novel solutions validated the effectiveness of our SLA techniques and the performance of our proposed RTM solutions. In particular, in comparison with other known frameworks in the literature, our proposed approaches highlighted the importance of mixing several numerical and fuzzy metrics, in contrast to adopting numeric only input, as well as the necessity and effectiveness of our introduced credibility mechanisms.

On the second part of the thesis, we address challenges in the service life-cycle management. Our main focus is to facilitate the life-cycle management of services in a multi-domain federated Edge Cloud environment and enable tenants to either offer services for off-the-shelf leasing or lease a service that suits their needs. Federated edge clouds in the scope of this dissertation, refer to multi-administrative collaboration of multiple providers willing to allow cross-domain communication to allow off-the-self leasing of services developed by either providers or tenants through an Network Service Marketplace (NSM).

In order to fulfill the above objectives, we shift from trust management frameworks to trustless systems to assist the life-cycle management of services in this context. We

use blockchain as our trustless system, as it enables parties without trust between them, to interact without a central trusted authority, communicating in a decentralized fashion, maintaining the same functionality, while all parties can retain the certainty for the outcome of the transactions. We used blockchain technology to develop a fully functional NSM for off-the-self service leasing. We utilize blockchain's distributed ledger as a distributed database between different providers and edge clouds, and we leverage smart contracts to enable the platform and the transactions. Using smart contracts we developed all the necessary interactions and transactions required in an NSM facilitating all steps in a service's life-cycle such as service discovery, selection, leasing, billing and most importantly resource orchestration. We provide a multi-domain architecture aligned with the ETSI-NFV standards, which is highly scalable and requires minimum resource and management and development overheads for new Edge Cloud providers.

Furthermore, in order to fulfill the consumption of a leased service in the NSM we developed a novel Cross-Service Communication (CSC) orchestrator over Network Function Virtualization (NFV) reference architecture. Our orchestrator, assisted with the minimum information required for the operation by our blockchain-based NSM, offers the necessary abstractions and operations for an automated and seamless CSC orchestration. The data stored in the blockchain are the minimum required and contain mostly tags guaranteeing both data efficiency and data privacy regarding the services offered.

The implementation and evaluation of our NSM and orchestrator validated that our proposed solution is highly scalable, and the orchestration time overheads added from blockchain, API and orchestrator interactions are negligible. In addition, the Cross-Communication orchestration requires no extra resources, while other proposals in the literature require intermediary services to function.

Finally, we summarize the conclusions of our thesis and offer interesting ideas for future development and enrichment of our work, along with significant synergies that can occur between the different solutions presented in this dissertation.

Keywords: Trust Management, SLA, Multi Criteria Decision Making, Cloud, NFV, Reputation, Fuzzy Logic, Blockchain

Abstract in Greek

Περίληψη στα Ελληνικά

Στην εποχή των δικτύων πέμπτης γενιάς (5G) και του Διαδικτύου των Αντικειμένων (ΔτΑ - Internet of Things), τα μοντέλα παροχής και διαχείρισης υπηρεσιών γίνονται όλο και πιο σύνθετα, με το συνδυασμό μιας ευρείας γκάμας από τεχνολογίες και συσκευές. Η πολυσυνθετότητα των νέων μοντέλων παροχής υπηρεσιών εγείρει νέες προκλήσεις σχετικά με την εκπλήρωση και τη διαχείριση του κύκλου ζωής μιας υπηρεσίας, από την ανακάλυψη και την επιλογή μιας υπηρεσίας, μέχρι την ενορχήστρωση, τη διαχείριση και την παρακολούθηση της. Επιπρόσθετα, σε ένα τέτοιο πλαίσιο, όπου άγνωστοι μεταξύ τους, χρήστες, συσκευές και πάροχοι απαιτείται να αλληλεπιδρούν για την εκπλήρωση μιας υπηρεσίας, είναι κρίσιμο να παρέχεται μια αίσθηση εμπιστοσύνης στους χρήστες. Κινητήριος δύναμη για την παρούσα διατριβή είναι η αντιμετώπιση των νέων προκλήσεων που εγείρονται στη διαχείριση του κύκλου ζωής υπηρεσιών και η παροχή εμπιστοσύνης.

Στο πρώτο μέρος της παρούσας διατριβής, το επίκεντρο της προσοχή μας εστιάζεται η αξιολόγηση της επίδοσης υπηρεσιών οι οποίες αναπτύσσονται με τη χρήση ετερογενών υποδομών, λαμβάνοντας υπ' όψιν τη Ποιότητα της Υπηρεσίας (ΠτΥ - Quality of Service) και την Ποιότητα της Εμπειρίας (ΠτΕ - Quality of Experience), συνδυάζοντας δεδομένα παρακολούθησης, την υποκειμενική αντίληψη των χρηστών και τις μοναδικές απαιτήσεις τους. Για την επίτευξη αυτού του στόχου, στοχεύουμε να ποσοτικοποιήσουμε την αξιολόγηση της επίδοσης των υπηρεσιών με τα παραπάνω κριτήρια, ως μια τιμή φήμης, η οποία αντιπροσωπεύει τις επιδόσεις και την αξιοπιστία μια υπηρεσίας με δίκαιο τρόπο, με την παράλληλη εγγύηση προστασίας από κακόβουλες οντότητες που επιχειρούν να την τροποποιήσουν προς ιδίων όφελος.

Για την επίτευξη των προαναφερθέντων στόχων, αναπτύξαμε συνεργατικά πλαίσια Συμφωνίας σε Επίπεδο Υπηρεσιών (Service Level Agreement - SLA) και Διαχείρισης της Εμπιστοσύνης με βάση τη φήμη (Reputation Trust Management - RTM) με κύρια μέριμνα τα ομόσπονδα νέφη. Η ομοσπονδία νέφους είναι η ανάπτυξη, διαχείριση και συνεργασία πολλαπλών υπηρεσιών υπολογιστικού νέφους. Η ομοσπονδία νέφους μπορεί να ενσωματώνει ιδιωτικά (private), κοινοτικά (community) και δημόσια (public) νέφη παρέχοντας πολλά οφέλη όπως κλιμακωσιμότητα, ανοχή στα σφάλματα (fault tolerance) και ελαστικότητα στο περιβάλλον νέφους. Παράλληλα όμως, προκύπτουν νέες προκλήσεις όπως η δια-λειτουργικότητα μεταξύ των ετερογενών νεφών και η έλλειψη εμπιστοσύνης μεταξύ των διαφορετικών παρόχων. Σε αυτό το περιβάλλον,

χρησιμοποιούμε SLA ώστε να επιτρέψουμε στους χρήστες να εξασφαλίσουν εγγυήσεις για τις επιδόσεις των υπηρεσιών που καταναλώνουν. Επεκτείνουμε τους καίριους δείκτες απόδοσης (Key Performance Indicators - KPIs) που αξιολογούνται από SLA για να ξεπεράσουμε το πολύ περιορισμένο KPI της διαθεσιμότητας (availability) υπηρεσίας που προσφέρουν οι περισσότεροι πάροχοι υπηρεσιών νέφους, ώστε να μπορούν να εκφραστούν οι μοναδικές απαιτήσεις κάθε ξεχωριστού χρήστη.

Επιπρόσθετα, για το κομμάτι της διαχείρισης εμπιστοσύνης με βάση τη φήμη των συνεργατικών μας πλαισίων, εστιάζουμε σε πολυ-κριτιριακές τεχνικές λήψης αποφάσεων (Multi-Criteria Decision Making - MCDM) για την ποσοτικοποίηση της εμπιστοσύνης και τον υπολογισμό της αντίστοιχης τιμής φήμης. Υιοθετούμε τέτοιες τεχνικές λόγω της ικανότητας τους να δέχονται ως είσοδο πολλαπλά KPIs συνδυάζοντας μετρικές που αντιπροσωπεύουν τόσο την ΠτΥ όσο και την ΠτΕ του χρήστη. Στην ερευνητική μας δουλειά, τροποποιήσαμε και επεκτείναμε δυο γνωστές MCDM τεχνικές που βασίζονται σε ασαφή λογική (Fuzzy Logic), την Fuzzy VIKOR και την Fuzzy Analytic Hierarchical Proccess (AHP). Η πρώτη είναι οριζόντιας δομής ενώ η δεύτερη ιεραρχικής δομής. Οι κύριες συνεισφορές των επεκτάσεων μας στις παραπάνω τεχνικές είναι η διεύρυνση των τύπων δεδομένων που δέχονται ως είσοδο ώστε να μη περιορίζεται σε ασαφής αριθμούς αλλά να περιλαμβάνει και γλωσσικές και δυαδικές τιμές. Με αυτόν τον τρόπο μπορούμε να συνδυάζουμε ένα μεγάλο εύρος κριτηρίων της ΠτΥ και της ΠτΕ στη διαδικασία παραγωγής της τιμής εμπιστοσύνης και να εγγυηθούμε ότι η υψηλή τιμή φήμης αναπαριστά αντικειμενικά την ποιότητα και αξιοπιστία ενός παρόχου. Επιπλέον, τροποποιήσαμε τις τεχνικές αυτές ώστε να επιτρέψουμε στους χρήστες να προσαρμόζουν τα βάρη κάθε κριτηρίου που συμμετέχει στην διαδικασία αξιολόγησης ώστε να μπορούν να εκφράσουν με αυτόν το τρόπο τις μοναδικές τους διαφορετικές απαιτήσεις.

Τέλος, και στις δύο τεχνικές, εισάγουμε μηχανισμούς αξιοπιστίας που αναπτύξαμε και συγκρίνουν τις αξιολογήσεις των χρηστών με δεδομένα SLA και δεδομένων παρακολούθησης προκειμένου να εντοπιστούν κακόβουλες οντότητες και να προστατευτούν οι πάροχοι από απόπειρες χειραγώγησης της τιμής εμπιστοσύνης τους.

Η υλοποίηση και πειραματική επικύρωση και αξιολόγηση των προτάσεων μας, επικύρωσαν την αποτελεσματικότητα των τεχνικών SLA και τις επιδόσεις των προτεινόμενων λύσεων μας για τη διαχείριση της εμπιστοσύνης με βάση τη φήμη. Πιο συγκεκριμένα, η σύγκριση με γνωστές λύσεις από τη βιβλιογραφία ανέδειξε τη σημασία του μείγματος διαφόρων αριθμητικών και

ασαφών δεδομένων, σε σύγκριση με το περιορισμό μονάχα σε αριθμητικές εισόδους. Επίσης αναδείχθηκε η αναγκαιότητα και αποτελεσματικότητα των μηχανισμών αξιοπιστίας.

Στο δεύτερο μέρος της διατριβής μας, συνεχίζουμε την ενασχόληση μας με τις προκλήσεις στη διαχείριση του κύκλου ζωής. Σε αυτό το σημείο εστιάζουμε στη διαχείριση του κύκλου ζωής υπηρεσιών σε ομόσπονδα νέφη στις παρυφές του δικτύου (federated edge clouds) και στο να προσφέρουμε σε χρήστες υπολογιστικών κέντρων νέφους παρυφών τη δυνατότητα, είτε να προσφέρουν τις υπηρεσίες που έχουν αναπτύξει προς μίσθωση είτε να μισθώσουν κάποια που εξυπηρετεί τις ανάγκες τους. Ως ομόσπονδα νέφη παρυφών, στο πλαίσιο αυτής της διατριβής, ορίζουμε την πολύ-διαχειριστική (multi-administrative) συνεργασία πολλαπλών παρόχων οι οποίοι είναι πρόθυμοι να επιτρέψουν την επικοινωνία μεταξύ διαφορετικών διαχειριστικών τομέων (administrative domains), ώστε να γίνει δυνατή η προσφορά για μίσθωση η κατανάλωση μιας υπηρεσίας μεταξύ αυτών μέσα μιας αγοράς δικτυακών υπηρεσιών (Network Service Marketplace - NSM).

Για την επίτευξη των στόχων αυτών, μετακινούμαστε από συστήματα διαχείρισης της φήμης σε συστήματα χωρίς εμπιστοσύνη (trustless systems), για την εξυπηρέτηση του κύκλου ζωής υπηρεσιών. Χρησιμοποιούμε Αλυσίδα Επιβεβαιωμένων Συναλλαγών (ΑΕΣ - Blockchain) ως σύστημα χωρίς εμπιστοσύνη, διότι επιτρέπει σε οντότητες χωρίς εμπιστοσύνη μεταξύ τους, να αλληλεπιδρούν χωρίς τη διαμεσολάβηση κάποιας έμπιστης κεντρικής αρχής. Αντίθετα η επικοινωνία είναι αποκεντρωμένη, διατηρώντας όλες τις λειτουργίες που θα εξυπηρετούσε μια κεντρική αρχή και διατηρώντας παράλληλα την εμπιστοσύνη στην ορθότητα των συναλλαγών και τα αποτελέσματα τους. Χρησιμοποιήσαμε ΑΕΣ για την ανάπτυξη μιας πλήρως λειτουργικής αγοράς δικτυακών υπηρεσιών. Χρησιμοποιήσαμε το κατανεμημένο κατάστιχο (distributed ledger) ως μια κατανεμημένη βάση δεδομένων μεταξύ των διαφορετικών παρόχων και υπολογιστικών κέντρων νέφους παρυφών και αξιοποιήσαμε τα έξυπνα συμβόλαια για την ανάπτυξη της αγοράς και τη διαχείριση των απαραίτητων συναλλαγών. Με την χρήση έξυπνων συμβολαίων αναπτύχθηκαν όλες οι απαραίτητες αλληλεπιδράσεις και συναλλαγές που απαιτούνται σε μια αγορά δικτυακών υπηρεσιών, εξυπηρετώντας κάθε βήμα στο κύκλο ζωής μιας υπηρεσίας, όπως η ανακάλυψη, η επιλογή, η μίσθωση, η χρέωση και το κυριότερο, η ενορχήστρωση της επικοινωνίας μεταξύ υπηρεσιών. Η λύση μας παρέχει μία πολύ-τομεακή αρχιτεκτονική η οποία είναι σύμφωνη με το πρότυπο ETSI-NFV, η οποία έχει πολύ καλή κλιμακωσιμότητα και απαιτεί ελάχιστους πόρους και επιπρόσθετη διαχείριση από νέους παρόχους παρυφών νέφους.

Ακόμη, προκειμένου να επιτευχθεί η κατανάλωση μιας μισθωμένης υπηρεσίας στην αγορά, αναπτύξαμε ένα πρωτότυπο ενορχηστρωτή της επικοινωνίας μεταξύ δικτυακών υπηρεσιών (Cross-Service Communication - CSC) για περιβάλλοντα και αρχιτεκτονικές εικονικοποίησης δικτυακών λειτουργιών (Network Function Virtualization - NFV). Ο ενορχηστρωτής μας, λαμβάνοντας την ελάχιστη απαραίτητη πληροφορία από την βασισμένη σε ΑΕΣ αγορά μας, πραγματοποιεί όλες τις απαραίτητες ενέργειες για την αυτόματη ενορχήστρωση της επικοινωνίας μεταξύ δικτυακών υπηρεσιών. Τα δεδομένα που αποθηκεύονται στην ΑΕΣ είναι τα ελάχιστα απαραίτητα και περιλαμβάνουν κατά βάση ετικέτες. Με αυτόν τον τρόπο εξασφαλίζεται τόσο η αποδοτικότητα των δεδομένων όσο και η ιδιωτικότητα των δεδομένων σχετικά με τις προσφερόμενες υπηρεσίες.

Η υλοποίηση και αξιολόγηση της αγοράς δικτυακών υπηρεσιών και του ενορχηστρωτή μας επιβεβαίωσαν ότι η προτεινόμενη λύση είναι κλιμακώσιμη και ότι οι χρονικές καθυστερήσεις που προσθέτει ο ενορχηστρωτής μας και η επικοινωνία μεταξύ της ΑΕΣ και του ενορχηστρωτή κρίνονται αμελητέες. Επιπλέον, η ενορχήστρωση της επικοινωνίας μεταξύ δικτυακών υπηρεσιών δεν απαιτεί επιπλέον πόρους ενώ παράλληλα άλλες προτάσεις στη βιβλιογραφία απαιτούν ενδιάμεσες δικτυακές υπηρεσίες που διαμεσολαβούν αυτήν την επικοινωνία και απαιτούν επιπρόσθετους πόρους.

Τέλος, συνοψίζουμε τα συμπεράσματα της διατριβής μας και προσφέρουμε μερικές ενδιαφέρουσες ιδέες για μελλοντική ανάπτυξη και εμπλουτισμό της δουλειάς μας καθώς και σημαντικές συνέργειες που μπορούν να προκύψουν μεταξύ των προτάσεων που παρουσιάσαμε σε αυτήν τη διατριβή.

Λέξεις κλειδιά: Διαχείριση εμπιστοσύνης, SLA, Πολύ-κριτιριακές μέθοδοι λήψης αποφάσεων, Υπολογιστική νέφους, Εικονικοποίηση Λειτουργιών του Δικτύου, Φήμη, Ασαφής λογική, Αλυσίδα Επιβεβαιωμένων Συναλλαγών

Contents

List of Figures

Εκτεταμένη Περίληψη στα Ελληνικά

Η καθημερινή εξέλιξη και εξάπλωση του διαδικτύου έχει δημιουργήσει πολλές σχεδιαστικές προκλήσεις καθώς η κλίμακα και οι απαιτήσεις από τα δίκτυα ολοένα και μεγαλώνουν. Αυτό έχει ως αποτέλεσμα να αναζητούνται νέες τεχνολογίες και πρακτικές στη προσπάθεια να καλυφθούν οι απαιτήσεις και να αντιμετωπιστούν τα νέα προβλήματα που προκύπτουν. Υπό την ομπρέλα των δικτύων πέμπτης γενιάς (5G), συνδυάζονται πολλές από αυτές τις νέες τεχνολογίες και πρακτικές προκειμένου να δώσουν συνδυαστικά λύση στις προκλήσεις και τα προβλήματα που προκύπτουν και αφορούν όλα τα στάδια του κύκλου ζωής μιας υπηρεσίας. Μερικές από αυτές, αποτελούν το Διαδίκτυο των Αντικειμένων (Internet of Things – IoT), το υπολογιστικό νέφος (cloud computing), η εικονικοποίηση δικτυακών λειτουργιών (Network Function Virtualization – NFV) και η παροχή υπολογιστικών πόρων στην άκρη του δικτύου (Mobile Edge Computing – MEC). Η συνδυαστική χρήση των παραπάνω ενώ λύνει πολλά προβλήματα δημιουργεί παράλληλα νέες προκλήσεις. Από τη κατάλληλη επιλογή συνδυασμού πόρων και υπηρεσιών, την εγγύηση καλής λειτουργίας και απόδοσης υπηρεσιών, μέχρι την εμπιστοσύνη μεταξύ συσκευών και οντοτήτων σε ένα αχανές διαδικτυακό περιβάλλον και τον τρόπο που ενορχηστρώνονται και αναπτύσσονται υπηρεσίες. Στην επίλυση των παραπάνω προβλημάτων προσπαθεί να συνεισφέρει η παρούσα διατριβή. Αυτό επιχειρείται με δύο τρόπους. Πρώτον, με τη χρήση μηχανισμών εμπιστοσύνης και διαχείρισης φήμης συνδυαστικά με συμβάσεις διασφάλισης επιπέδου ποιότητας υπηρεσιών (Service Level Agreement – SLA). Αυτή η προσέγγιση, μπορεί να παρέχει εχέγγυα επιδόσεων και λειτουργίας στο χρήστη, ενώ ταυτόχρονα με τη διαμόρφωση μιας τιμής εμπιστοσύνης που αντικατοπτρίζει τη συλλογική πεποίθηση των χρηστών για τις καλές επιδό-

σεις ενός παρόχου ή μιας υπηρεσίας, μπορεί να οδηγήσει το χρήστη εύκολα στη σωστή επιλογή συνδυασμού πόρων και υπηρεσιών. Δεύτερον, με τη χρήση λύσεων χωρίς εμπιστοσύνη, όπως η Αλυσίδα Επιβεβαιωμένων Συναλλαγών (ΑΕΣ – Blockchain) όπου λόγω της φύσης των μηχανισμών συναίνεσης που διαθέτει μπορεί να κάνει το χρήστη να εμπιστεύεται μια πλατφόρμα η ένα δίκτυο χωρίς όμως να προϋποτίθεται η εμπιστοσύνη μεταξύ των μελών του. Ταυτόχρονα, τα έξυπνα συμβόλαια (smart contracts) που υποστηρίζει, μπορούν να κάνουν την ΑΕΣ χρήσιμη για πληθώρα σεναρίων. Στην περίπτωση μας, εφαρμόζεται για την παροχή εμπιστοσύνης σε μια αγορά ενοικίασης υπηρεσιών μεταξύ αγνώστων χρηστών και διαφορετικών παρόχων, αλλά και για τη διαχείριση της ενορχήστρωσης και ανάπτυξης της επικοινωνίας μεταξύ υπηρεσιών σε περιβάλλον MEC.

Κεφάλαιο 2

Το Κεφάλαιο 2 είναι αφιερωμένο στη διαχείριση εμπιστοσύνης σε περιβάλλον ομόσπονδων υπηρεσιών νέφους και του κύκλου ζωής υπηρεσιών νέφους. Στο κεφάλαιο αυτό παρουσιάζονται δύο προτεινόμενα πλαίσια για την διαχείριση εμπιστοσύνης και του κύκλου ζωής σε περιβάλλοντα ετερογενών ομόσπονδων υπηρεσιών νέφους τα οποία βασίζονται σε πολύ-κριτηριακούς αλγορίθμους λήψης αποφάσεων (Multi-criteria Decision Making – MCDM) οι οποίοι αξιοποιούν ασαφή λογική (fuzzy logic). Οι δύο αυτές προτεινόμενες λύσεις είναι συνεργατικές, υπό την έννοια ότι αξιοποιούν συνδυαστικά τεχνικές SLA και τεχνικές διαχείρισης εμπιστοσύνης βασιζόμενες στην φήμη. Για την επίτευξη της ποσοτικοποίησης της συλλογικής εμπιστοσύνης των χρηστών στις επιδόσεις και την αξιοπιστία μιας υποδομής ή ενός παρόχου, αξιοποιούνται σαφώς ορισμένα κριτήρια που λαμβάνουν υπ' όψιν και την ποιότητα της υπηρεσίας και την ποιότητα της εμπειρίας του χρήστη. Αυτή η ποσοτικοποίηση έχει στόχο τη παραγωγή μια απλής τιμής φήμης, η οποία θα επιτρέψει στους χρήστες να επιλέξουν εύκολα το καλύτερο συνδυασμό πόρων και παρόχων για τις μοναδικές και συγκεκριμένες απαιτήσεις τους και παράλληλα αντικατοπτρίζει την παρακολούθηση και αξιολόγηση των επιδόσεων πόρων και παρόχων.

Η διαχείριση του κύκλου ζωής μια εφαρμογής σε περιβάλλον ομόσπονδου νέφους έχει πολλές προκλήσεις και χρειάζεται μεγάλη προσοχή στην επιλογή και ενορχήστρωση πόρων καθώς και στην παρακολούθηση και αξιολόγηση των επιδόσεων τους. Όπως φαίνεται στο Σχήμα 2.1, ο πλήρης κύκλος ζωής εφαρμογών νέφους, από τη σκοπιά παρόχων Πλατφόρμας ως Υπηρεσία

(Platfform as a Service - PaaS), περιλαμβάνει τα εξής στάδια:

- **Ανακάλυψη Υπηρεσίας (Service Discovery):** Ο κύκλος ζωής ξεκινάει με τους χρήστες να ανακαλύπτουν στις λίστες παρεχόμενων υπηρεσιών τις καταλληλότερες για τις ανάγκες τους.
- **Αίτημα Υπηρεσίας (Service Request):** Μετά το στάδιο της ανακάλυψης, οι χρήστες αιτούνται τις υπηρεσίες που επέλεξαν.
- **Παροχή Υπηρεσίας (Service Provision):** Μετέπειτα, ο πάροχος κατανέμει και αναθέτει τους πόρους που χρειάζονται για να καλυφθούν οι ανάγκες των χρηστών.
- **Ανάπτυξη Εφαρμογής (Application Deployment):** Οι χρήστες αναπτύσσουν τις εφαρμογές τους αξιοποιώντας τα παρεχόμενα απαραίτητα εργαλεία, λειτουργικά συστήματα κ.ο.κ
- **Διαχείριση Εφαρμογής (Application Management):** Με τη χρήση κατάλληλων εργαλειών, πραγματοποιείται η παρακολούθηση εξυπηρετητών, πόρων και εφαρμογών.
- **Έλεγχος και Χρέωση (Auditing and Billing):** Με την καταγραφή της χρήσης των παρεχόμενων πόρων προκύπτει περιοδικά η χρέωση του χρήστη με βάση το μοντέλο τιμολόγησης.
- **Τερματισμός Εφαρμογής:** Η διαδικασία του τερματισμού επιτρέπει την βέλτιστη ανακατανομή πόρων μετά την παύση ή την απόσυρση μιας υπηρεσίας.

Οι δυο συνεργατικές λύσεις που προτείνονται εμπλέκονται σε διάφορα στάδια των παραπάνω σταδίων. Αρχικά στο στάδιο της ανακάλυψης των υπηρεσιών, η υπηρεσία SLA διαφημίζει τις εγγυήσεις που προσφέρει ο πάροχος, και η υπηρεσία διαχείρισης της εμπιστοσύνης παρέχει την τιμή φήμης για κάθε πάροχο ώστε να διευκολυνθεί η σωστή επιλογή πόρων. Κατά το στάδιο της ανάπτυξης εφαρμογής, ενεργοποιείται το SLA και συνεχώς αξιολογεί τις επιδόσεις της εφαρμογής (στάδιο ελέγχου). Επίσης, η υπηρεσία διαχείρισης της φήμης εμπλέκεται στο στάδιο ελέγχου, προτρέποντας περιοδικά τους χρήστες να υποβάλλουν τις αξιολογήσεις τους για την εφαρμογή. Τέλος, στο τελικό στάδιο, το SLA τερματίζεται και η τελική αξιολόγηση υποβάλλεται στην υπηρεσία διαχείρισης της εμπιστοσύνης.

Και για τις δυο συνεργατικές προτάσεις-πλαίσια που προτείνονται στο κεφάλαιο αυτό η αρχιτεκτονική είναι κοινή. Στο σχήμα 2.2 παρουσιάζεται η αρχιτεκτονική των υπηρεσιών SLA και

διαχείρισης εμπιστοσύνης σε περιβάλλον ομόσπονδων υπολογιστικών νεφών. Η αρχιτεκτονική χωρίζεται σε δύο στρώματα, στο στρώμα της ομοσπονδίας και στο στρώμα των παρόχων. Το στρώμα της ομοσπονδίας, σχετικά με την υπηρεσία SLA, περιλαμβάνει τον συλλέκτη SLA (SLA collector), και το ταμπλό διαχείρισης SLA (SLA dashboard) ενώ το τμήμα διαχείρισης SLA (SLA management module) βρίσκεται στο στρώμα των παρόχων. Το ταμπλό διαχείρισης SLA προσφέρει ένα δικτυακό γραφικό περιβάλλον διασύνδεσης (web-GUI), το οποίο επιτρέπει στους χρήστες να ανακαλύψει τα προσφερόμενα πρότυπα SLA (SLA templates) και στους παρόχους να δημιουργούν τις συμφωνίες SLA (SLA agreements). Επίσης, μέσω αυτού του ταμπλό οι πάροχοι και οι χρήστες μπορούν να ελέγχουν την κατάσταση και τη τήρηση ή μη μιας συμφωνίας. Ο συλλέκτης SLA ενεργεί ως ένα ενδιάμεσο σημείο επικοινωνίας μεταξύ του ταμπλό και του τμήματος διαχείρισης του κάθε παρόχου. Μέσω προγραμματιστικών διεπαφών εφαρμογής (Application Programming Interface - API), ο συλλέκτης υποστηρίζει κάθε διαδικασία που αφορά την υπηρεσία SLA από τη δημιουργία της μέχρι τον τερματισμό της. Επίσης ενημερώνει τα άλλα τμήματα του SLA σε περίπτωση κάποιου γεγονότος, όπως η παραβίαση της συμφωνίας για παράδειγμα. Το τμήμα διαχείρισης SLA, το οποίο βρίσκεται στο στρώμα παρόχου, είναι ο πυρήνας της υπηρεσίας SLA, και είναι υπεύθυνο για την αποθήκευση όλων των απαραίτητων και σχετικών με το SLA πληροφοριών για την αξιολόγηση των ενεργών συμφωνιών. Σχετικά με την υπηρεσία διαχείρισης φήμης, το στρώμα της ομοσπονδίας περιλαμβάνει ένα ταμπλό διαχείρισης φήμης όπου είναι και αυτό ένα δικτυακό γραφικό περιβάλλον διασύνδεσης που προσφέρεται και στους χρήστες και στους παρόχους. Μέσω αυτού, οι χρήστες υποβάλλουν τις αξιολογήσεις τους και αντλούν την πληροφορία για την τιμή φήμης του κάθε παρόχου ενώ ο διαχειριστής μια υποδομής νέφους πραγματοποιεί διαχειριστικές ενέργειες όπως ο καθορισμός νέων καίριων δεικτών απόδοσης (Key Performance Indicators - KPIs) για τον υπολογισμό της τιμής φήμης. Μέσω προγραμματιστικής διεπαφής εφαρμογής, το ταμπλό διαχείρισης φήμης επικοινωνεί με το σύστημα φήμης (Hybrid Reputation Systen - HRS). Το σύστημα φήμης είναι το κύριο τμήμα της υπηρεσίας διαχείρισης της φήμης και είναι υπεύθυνη για τον υπολογισμό και την παραγωγή της τιμής φήμης του κάθε παρόχου. Για τον υπολογισμό αυτών ανακτά πληροφορίες για τις συμφωνίες SLA και δεδομένα παρακολούθησης, από τον συλλέκτη SLA και την διεπαφή δεδομένων παρακολούθησης (Monitoring data API) αντιστοίχως.

Η πρώτη πρόταση του κεφαλαίου αυτού βασίζεται στον αλγόριθμο Fuzzy VIKOR, ο οποίος είναι ένας MCDM αλγόριθμος οριζόντιας δομής. Ο αλγόριθμος αυτός μπορεί να επεξεργαστεί

διαφόρων ειδών δεδομένα. Η προσέγγιση Fuzzy VIKOR λαμβάνει αποφάσεις μετρώντας ταυτόχρονα την εγγύτητα στην καλύτερη και τη χειρότερη εναλλακτική και μπορεί να εφαρμοστεί σε σενάρια όπου οι χρήστες πρέπει να επιλέξουν μεταξύ διαφορετικών παρόχων όπως για παράδειγμα μεταξύ διαφορετικών παρόχων ενέργειας ή στην περίπτωση μας παρόχων υπηρεσιών υπολογιστικού νέφους. Η πρωτότυπη προσέγγιση Fuzzy VIKOR χρησιμοποιεί ρητά μια ομάδα από KPIs ασαφούς λογικής. Για τον υπολογισμό και τη παραγωγή της τιμής φήμης ενός παρόχου υπηρεσιών υπολογιστικού νέφους, επεκτείναμε το Fuzzy VIKOR ώστε να επεξεργάζεται και αριθμητικά KPIs όπως φαίνεται και στο Σχήμα 2.6, όπου η τιμή φήμης προκύπτει απευθείας από τα KPIs. Το επίπεδο με τις κατηγορίες των κριτηρίων δε συμμετέχει στον υπολογισμό της τιμής φήμης και απλά υποδηλώνει τη διαφορετική φύση των των KPIs του από κάτω επιπέδου. Τα μωβ KPIs είναι τεχνικά, αναφέρονται σε μετρικές της ποιότητας της υπηρεσίας και είναι αριθμητικά. Τα ροζ αναφέρονται σε μη τεχνικά KPIs και είναι KPIs ασαφούς λογικής. Τα αριθμητικά δεδομένα μετατρέπονται σε ασαφής αριθμούς με τρόπο που θα περιγράψουμε στη συνέχεια. Για κάθε ζευγάρι από τα KPIs, ορίζεται ένα ασαφές βάρος, που υποδηλώνει τη σχετική μεταξύ τους σημασία για το χρήστη. Στο Πίνακα 2.1 παρουσιάζονται οι γλωσσικοί όροι και οι αντίστοιχες συναρτήσεις συμμετοχής (membership functions) για τα ασαφή βάρη και στον Πίνακα 2.2 παρουσιάζονται οι πληροφορίες για τους ασαφής αριθμούς που χρησιμοποιούνται για την αξιολόγηση των KPIs. Η αξιολόγηση του χρήστη συγκρίνεται με μία τέλεια αξιολόγηση ενός εικονικού χρήστη με βάση την οποία μετριέται η εγγύτητα στη καλύτερη δυνατή επίδοση ενός παρόχου. Τα παρακάτω βήματα περιλαμβάνουν όλους τους απαραίτητους υπολογισμούς και διαδικασίες για τον υπολογισμό της τιμής φήμης ενός παρόχου.

- **Βήμα 1 - Ορισμός των KPIs του υπολογιστικού νέφους:** Ο πάροχος ορίζει όλα τα KPIs ποιότητας της υπηρεσίας και ποιότητας της εμπειρίας που καθορίζουν την επίδοση του νέφους. Επίσης τα KPIs της ποιότητας της υπηρεσίας εντάσσονται σε SLA μεταξύ του παρόχου και του χρήστη.

- **Βήμα 2 - Ορισμός του σχετικού βάρους:** Ο χρήστης αναθέτει γλωσσικούς όρους από τον Πίνακα 2.1 για τα σχετικά βάρη για κάθε πιθανό ζευγάρι από KPIs. Υποθέτοντας ένα νέφος με N KPIs, διατυπώνουμε τον ασαφή κατά ζεύγη πίνακα σύγκρισης βαρύτητας (Pairwise Importance Comparison Matrix - PICM) όπως φαίνεται στην εξίσωση 2.1. Στη συνέχεια τα στοιχεία του πίνακα μετατρέπονται σε απλούς αριθμούς χρησιμοποιώντας

τον τύπο (A.0.6) του παραρτήματος Α. Καθώς τα βάρη προκύπτουν από τις υποκειμενικές προτιμήσεις χρηστών, ο τελικός υπολογισμός της τιμής εμπιστοσύνης μπορεί να βασιστεί σε ασυνεπή και αντικρουόμενα KPIs. Για να αποφευχθεί αυτό, υπολογίζεται η αναλογία συνοχής (Consistency Ratio - CR). Η CR δείχνει το ποσοστό τυχαιότητας στον ορισμό των βαρών μεταξύ πολλών KPIs. Αν η CR έχει τιμή μικρότερη του 0.1 του είναι αποδεκτή και προχωράμε στο επόμενο βήμα, διαφορετικά ο χρήστης πρέπει να διορθώσει τα ορισμένα βάρη.

- **Βήμα 3 - Υπολογισμός του διανύσματος βάρους των KPIs:** Έχοντας τον N-διάστατο πίνακα $PICM = [a_{ij}],\ i, j = 1, \ldots, N$ το διάνυσμα υπολογίζεται ακολουθώντας τα βήματα που περιγράφονται από τις εξισώσεις (2.2), (2.3), (2.4) και (2.5).

- **Βήμα 4 - Αξιολόγηση της επίδοσης υπολογιστικού νέφους:** Όταν το προκαθορισμένο διάστημα χρήσης ενός νέφους τελειώσει, ζητείται από τους χρήστες να υποβάλλουν την αξιολόγηση τους σχετικά με την επίδοση του νέφους. Για την αντιμετώπιση κακόβουλων χρηστών και αξιολογήσεων, λαμβάνεται υπ' όψιν η αξιοπιστία του χρήστη η οποία υπολογίζεται με βάση μηχανισμό αξιοπιστίας που αναπτύξαμε και αναλύεται στην υποενότητα 2.6.2 της διατριβής, και αν χρειαστεί, τροποποιείται κατάλληλα η αξιολόγηση ενός χρήστη για τα KPIs της ποιότητας της υπηρεσίας. Για τα ασαφή KPIs, ο χρήστης αναθέτει γλωσσικές τιμές για την αξιολόγηση του με βάση τον Πίνακα 2.2. Για τα αριθμητικά KPIs, ο χρήστης αναθέτει μια αριθμητική τιμή η οποία μετατρέπεται σε ασαφή βάσει των συναρτήσεων συμμετοχής του Πίνακα 2.2. Έστω $\tilde{x}$ η τροποποιημένη από τον μηχανισμό αξιοπιστίας αριθμητική αξιολόγηση ενός χρήστη, η οποία συμπεριλαμβάνεται σε δύο διαδοχικές γλωσσικές τιμές A και B με μ_A and μ_B οι αντίστοιχες συναρτήσεις συμμετοχής. Τότε η τροποποιημένη γλωσσική τιμή $\tilde{X}$ που αντιστοιχεί στην αριθμητική αξιολόγηση προκύπτει από τη σχέση $\tilde{X} = \mu_A A + \mu_B B$. Χρησιμοποιούμε την αξιολόγηση ενός εικονικού χρήστη με εξαιρετική (Excellent - E) γλωσσική τιμή για όλα τα KPIs η οποία αντιπροσωπεύει τη καλύτερη δυνατή επίδοση του νέφους. Ο ασαφής πίνακας αξιολόγησης (Fuzzy Evaluation Matrix - FEM) υπολογίζεται με βάση την εξίσωση (2.6), όπου η πρώτη σειρά αντιστοιχεί στη τροποποιημένη αξιολόγηση της άποψης του χρήστη (U), ενώ η δεύτερη σειρά αντιστοιχεί στην "τέλεια" αξιολόγησή του εικονικού χρήστη (V).

- **Βήμα 5 - Υπολογισμός και ενημέρωση της τιμής φήμης:** Σε αυτό το βήμα, εφαρμό-

ζουμε την τροποποιημένη μέθοδο Fuzzy VIKOR. Με διάνυσμα βάρους W και τον πίνακα $FEM = [x_{ij}],\ i = 1, 2,\ j = 1, \ldots, N$, καθορίζουμε την καλύτερη ασαφή τιμή $\tilde{f}_j^+$ και τη χειρότερη ασαφή τιμή $\tilde{f}_j^-$. Αφού η εικονική αξιολόγηση είναι η καλύτερη δυνατή, η καλύτερη και χειρότερη ασαφή τιμή θα είναι,

$$\tilde{f}_j^+ = x_{2j},\ j = 1, \ldots, N$$
$$\tilde{f}_j^- = x_{1j},\ j = 1, \ldots, N$$

Η απόσταση του x_{ij} από τη καλύτερη και χειρότερη ασαφή τιμή προκύπτουν από τις σχέσεις (2.7) και (2.8). Έπειτα υπολογίζουμε τις καλύτερες και τις χειρότερες τιμές των $\tilde{S}_i,\ \tilde{R}_i$ όπως προέκυψαν από τις παραπάνω σχέσεις βάσει των (2.9), (2.10). Επομένως, ο δείκτης $\tilde{Q}_i$, ο οποίος συμπεριλαμβάνει την ασαφή τιμή της φήμης που προκύπτει από τον χρήστη που αξιολογεί και τη τιμή που προκύπτει από τον εικονικό χρήστη υπολογίζεται από τη σχέση (2.11), όπου α είναι ο δείκτης της πρόθεσης μας να τιμωρήσουμε μια κακή επίδοση νέφους ή να ανταμείψουμε μια καλή. Προκειμένου να υπάρχει ισορροπία μεταξύ της καλής και της κακής συμπεριφοράς, έχουμε ορίσει $\alpha = 0.4$ διότι ο εικονικός χρήστης έχει πάντοτε τις μέγιστες δυνατές αξιολογήσεις. Χρησιμοποιώντας την (A.0.6), μετατρέπουμε από ασαφή, σε αριθμητική τιμή τις τιμές του $\tilde{Q}_i$. Το στοιχείο Q_i με τη μικρότερη τιμή έχει τη καλύτερη τιμή φήμης. Στην περίπτωση μας λοιπόν ο εικονικός χρήστης έχει την καλύτερη τιμή που είναι πάντα μηδέν ($Q_2 = 0$). Για μια συγκεκριμένη αξιολόγηση η τιμή φήμης R_{exp} ορίζεται από τη σχέση (2.12). Μετά από n αξιολογήσεις, η συνολική τιμή φήμης ενημερώνεται με βάση τη σχέση (2.13).

Η δεύτερη πρόταση βασίζεται στον αλγόριθμο Fuzzy AHP, ο οποίος είναι επίσης ένας MCDM αλγόριθμος ιεραρχικής δομής. Προκειμένου να ανταποκριθεί στις συνθήκες και στις απαιτήσεις ενός περιβάλλοντος ομόσπονδου νέφους, το πλαίσιο που προτείνεται μπορεί να επεξεργαστεί διάφορους τύπους δεδομένων που αντιστοιχούν σε τεχνικά KPIs που αφορούν την ποιότητα της υπηρεσίας και μη τεχνικά KPIs που αφορούν την ποιότητα της εμπειρίας. Τα τεχνικά KPIs αναφέρονται σε αντικειμενικές μετρικές επιδόσεων όπως η καθυστέρηση του δικτύου (network latency) ή η χρήση του επεξεργαστή (CPU utilization), ενώ τα μη τεχνικά KPIs αναφέρονται στην υποκειμενική εμπειρία του χρήστη όπως για παράδειγμα η ικανοποίηση από την υποστήριξη

που προσέφερε κάποιος πάροχος και η ευκολία χρήσης των υποδομών. Με λίγα λόγια, το πλαίσιο αυτό επιτρέπει τη χρήση αριθμητικών, δυαδικών και γλωσσικών τιμών, δίνοντας έτσι την ευκαιρία στους χρήστες να εκφράσουν την υποκειμενική τους άποψη με το καλύτερο και πιο αποδοτικό τρόπο. Κάθε χρήστης έχει μοναδικές ανάγκες και διαφορετικά κριτήρια με τα οποία επιλέγει μια υπηρεσία και αξιολογεί τις συνολικές της επιδόσεις. Για να εξυπηρετηθούν αυτές οι ανάγκες, μέσω της προτεινόμενης λύσης, οι χρήστες μπορούν να αξιολογούν και τις επιδόσεις τόσο της υποδομής νέφους όσο και της εφαρμογής, λαμβάνοντας υπ' όψιν διαφορετικά κριτήρια, προτιμήσεις και προτεραιότητες. Για αυτό το λόγο, στην προσέγγιση μας οι ίδιοι οι χρήστες αναθέτουν διαφορετικά βάρη στα διαφορετικά κριτήρια με βάση τις προσωπικές τους ανάγκες. Το προτεινόμενο σύστημα εμπιστοσύνης βασίζεται όπως προαναφέραμε στο Fuzzy AHP. Το Fuzzy AHP είναι μια μέθοδος κατάταξης με βάση αριθμητικά KPIs ποιότητας της υπηρεσίας και ασαφή KPIs ποιότητας της εμπειρίας. Προκειμένου να μπορέσει να υπολογίσει την τιμή φήμης μιας εφαρμογής νέφους, απαιτούνται διάφορες τροποποιήσεις και επεκτάσεις. Η πρόταση μας έχει τρεις χαρακτηριστικές διαφορές σε σχέση με άλλες προτάσεις βασισμένες στο Fuzzy AHP για την επιλογή πάροχου όπως για παράδειγμα στην [1]. Πρώτον, η πρόταση μας επιτρέπει στους χρήστες να αναθέτουν τα δικά τους βάρη στα διάφορα κριτήρια αξιολόγησης με βάση τις προτεραιότητες και τις ανάγκες τους για τις επιδόσεις μιας εφαρμογής. Δεύτερον, συγκρίνουμε την αξιολόγηση που καταθέτει ο χρήστης για μια εφαρμογή με μια ιδανική αξιολόγηση ενός εικονικού χρήστη. Τέλος, ο υπολογισμός της τιμής φήμης λαμβάνει υπ' όψιν την αξιοπιστία του χρήστη για ν εξασφαλίσει τη δίκαιη κρίση του κάθε παρόχου όπως περιγράφεται στην ενότητα 2.7.2 της διατριβής. Σε αυτό το σημείο θα παρουσιάσουμε τις διαφορετικές φάσεις του προτεινόμενου μηχανισμού.

- **Φάση 1 - Επιλογή των KPIs μιας υπηρεσίας**

  Ο πάροχος της υποδομής νέφους καθορίζει τα τεχνικά και μη τεχνικά KPIs και τα χαρακτηριστικά τα οποία θα χρησιμοποιηθούν για τον υπολογισμό της τιμής εμπιστοσύνης της παρεχόμενης εφαρμογής. Στο Σχήμα 2.10 βλέπουμε ένα παράδειγμα από KPIs και χαρακτηριστικά, σε ιεραρχική δομή, όπου υπογραμμίζεται ποια από αυτά παρέχονται από τους παρόχους. Η διαφορά μεταξύ των KPIs και των χαρακτηριστικών είναι ότι τα KPIs μετράνε μια συγκεκριμένη τεχνική η μη τεχνική μετρική, ενώ ένα χαρακτηριστικό συνοψίζει διάφορα σχετικά μεταξύ τους KPIs. Σε οποιοδήποτε επίπεδο της ιεραρχικής

δομής, τα χαρακτηριστικά μπορούν να αποσυντεθούν περαιτέρω σε αδελφά χαρακτηριστικά (sibling attributes) ή KPIs χαμηλότερου επίπεδο, ενώ τα KPIs δεν μπορούν να αποσυντεθούν περαιτέρω. Υιοθετώντας την προσέγγιση του SMICloud [2], τα αριθμητικά KPIs και χαρακτηριστικά εκφράζονται με αριθμητικές, δυαδικές τιμές και μη διατεταγμένα σύνολα (unordered sets) ενώ τα μη τεχνικά KPIs της ποιότητας της εμπειρίας του χρήστη εκφράζονται με ασαφής αριθμούς της μορφής $A = \{l, m, u\}$ και η συνάρτηση συμμετοχής ορίζεται ως $\mu_A(x)$ (A.0.1). Οι αριθμητικές πράξεις των ασαφών αριθμών ορίζονται στο Παράρτημα Α.

- **Φάση 2 - Υπολογισμός σχετικής σημασίας χαρακτηριστικού**

  Περιοδικά ή κατά τον τερματισμό της εφαρμογής νέφους, ο χρήστης υποβάλει την αξιολόγηση του για τεχνικά και μη τεχνικά KPIs του παρόχου που έχει επιλέξει. Τα τεχνικά KPIs της ποιότητας της υπηρεσίας τροποποιούνται κατάλληλα από τον μηχανισμό αξιοπιστίας που αναπτύξαμε αν χρειάζεται προκειμένου να προστατευθεί το σύστημα μας και οι πάροχοι από κακόβουλες αξιολογήσεις και οντότητες. Η διαδικασία αυτή και ο μηχανισμός αξιοπιστίας περιγράφεται αναλυτικά στην ενότητα 2.7.2. Η τροποποιημένη αξιολόγηση του χρήστη συγκρίνεται με την ιδανική αξιολόγηση του εικονικού χρήστη. Η ιδανική αξιολόγηση χρησιμοποιείται ώστε να μετρηθεί η απόσταση της πραγματικής επίδοσης της εφαρμογής νέφους με την τέλεια επίδοση με βάση τις προτιμήσεις του χρήστη. Αυτό επιτυγχάνεται με τον υπολογισμό του πίνακα σύγκρισης σχετικών χαρακτηριστικών (Relative Attribute Comparison Matrix - RACM) για κάθε KPI του ιεραρχικού μοντέλου. Δεδομένης της ιδανικής αξιολόγησης A_v και της τροποποιημένης αξιολόγησης του χρήστη $\tilde{A_u}$ για το X KPI, ο πίνακας $RACM_X$ ορίζεται ως εξής,

$$RACM_X = \begin{bmatrix} 1 & \tilde{A_u}/A_v \\ A_v/\tilde{A_u} & 1 \end{bmatrix}$$

  Στη περίπτωση που το KPI είναι ασαφής αριθμός, η διαίρεση των στοιχείων του $RACM_X$ πραγματοποιείται με βάση τον ορισμό της ασαφούς διαίρεσης όπως ορίζεται στο (A.0.5). Για τα αριθμητικά δεδομένα η διαίρεση ακολουθεί τις περιπτώσεις του Πίνακα 2.4.

- **Φαση 3 - Υπολογισμός και ενημέρωση της τιμής φήμης**

  Στη περίπτωσή τον αριθμητικών KPIs και χαρακτηριστικών, εφαρμόζουμε την προσέγ-

γιση του extended AHP όπως περιγράφεται στο [2]. Για τα ασαφή KPIs, χρησιμοποιούμε την προσέγγιση Chang. Ο συνδυασμός αυτών των μεθοδολογιών, χρησιμοποιεί τον πίνακα RACM για κάθε KPI και χαρακτηριστικό σε οποιοδήποτε επίπεδο του ιεραρχικού μοντέλου προκειμένου να υπολογίσει τη τιμή του διανύσματος κάθε ενδιάμεσου επιπέδου και εν τέλει στο τελικό επίπεδο τη τιμή φήμης. Για τα ασαφή RACMs, εφαρμόζονται τα ακόλουθα βήματα του [3]. Για N-διάστατο ασαφή πίνακα $RACM_A = [a_{ij}],\ i, j = 1, \ldots, N$, ορίζεται το ασαφές συνθετικό ανάπτυγμα (fuzzy synthetic extent) κάθε σειράς i του $RACM$ από τη σχέση (2.15), όπου ο πρώτος όρος στο πολλαπλασιασμό είναι το άθροισμα των στοιχείων της i^{th} σειράς και ο δεύτερος είναι ο ασαφής αντίστροφος του αθροίσματος των στοιχείων του RACM. Ο ασαφής πολλαπλασιασμός ορίζεται από τη σχέση (A.0.4), ενώ η ασαφής αντιστροφή ορίζεται από τον ορισμό της ασαφής διαίρεσης (A.0.5). Βρίσκουμε το χαρακτηριστικό με το μεγαλύτερο ασαφή συνθετικό βαθμό (fuzzy synthetic degree) υπολογίζοντας ποιος ασαφής αριθμός έχει το μεγαλύτερο βαθμό πιθανότητας (degree of possibility)

$$
\begin{aligned}
&V\left(D_i \geq D_j\right) = hgt\left(D_i \cap D_j\right) = \mu_{D_i}(d) \\
&= \begin{cases}
1 & \text{if } D_{im} \geq D_{jm} \\
\frac{D_{jl} - D_{iu}}{(D_{im} - D_{iu}) - (D_{jm} - D_{jl})} & \text{if } D_{im} \leq D_{jm} \text{ and } D_{jl} \leq D_{iu} \\
0 & \text{otherwise}
\end{cases}
\end{aligned}
$$

Ο βαθμός πιθανότητας είναι μια συγκριτική μέθοδος μεταξύ δύο κυρτών (convex) ασαφών αριθμών. Ορίζεται από την τεταγμένη d από το υψηλότερο σημείο τομής D, Όπως φαίνεται και στο Σχήμα 2.11. Ο βαθμός πιθανότητας ότι το ασαφές συνθετικό ανάπτυγμα D_i είναι μεγαλύτερο από τα υπόλοιπα ασαφή αναπτύγματα του ασαφούς RACM προκύπτει από την σχέση (2.17). Τέλος, το κανονικοποιημένο διάνυσμα σύγκρισης προκύπτει από την σχέση (2.18). Σε οποιοδήποτε επίπεδο του ιεραρχικού μοντέλου του παρόχου νέφους, υπολογίζουμε το διάνυσμα σύγκρισης για κάθε χαρακτηριστικό, από κάτω προς τα πάνω. Με δεδομένα τα βάρη από τη Φάση 2, την αξιολόγηση του χρήστη και την ιδανική αξιολόγηση, ξεκινάμε από το επίπεδο όπου υπάρχει KPI και υπολογίζουμε το διάνυσμα σύγκρισης του χαρακτηριστικού "πατέρα" με τα διανύσματα σύγκρισης των "παιδιών" KPIs και χαρακτηριστικών. Υποθέτοντας έναν χαρακτηριστικό "πατέρα" με *M* υπό-

χαρακτηριστικά και ένα διάνυσμα με τα βάρη με *M* στοιχεία, το διάνυσμα σύγκρισης του χαρακτηριστικού "πατέρα" ορίζεται από τη σχέση (2.19). Φτάνοντας στο ανώτερο επίπεδο του ιεραρχικού μοντέλου, υπολογίζεται το κανονικοποιημένο διάνυσμα σύγκρισης για τη φήμη του παρόχου, $c_{rep} = \left[c_{rep}^{\tilde{u}} c_{rep}^{v}\right]^{\top}$. Το πρώτο στοιχείο του διανύσματος αναφέρεται στην αξιολόγηση της υπηρεσίας, ενώ το δεύτερο στην καλύτερη δυνατή αξιολόγηση του εικονικού χρήστη. Η διαφορά μεταξύ των δύο στοιχείων εκφράζει την απόσταση μεταξύ της πραγματικής επίδοσης όπως την αντιλαμβάνεται ο χρήστης και την ιδανική-τέλεια επίδοση της υπηρεσίας νέφους. Επομένως για την n αξιολόγηση ενός παρόχου η τιμή της φήμης για τη συγκεκριμένη επίδοση υπολογίζεται με βάση τη σχέση (2.20) και με βάση αυτή ενημερώνεται η συνολική φήμη του παρόχου με βάση όλες τις αξιολογήσεις στην ιστορία του σύμφωνα με τη σχέση (2.21).

Τα πειραματικά αποτελέσματα ανέδειξαν ότι η πρόταση βασισμένη στο Fuzzy VIKOR έχει καλύτερα αποτελέσματα και επιδόσεις από γνωστούς αλγορίθμους στη βιβλιογραφία, τη μεγάλη σημασία του συνδυασμού διαφορετικών τύπων δεδομένων, τα οφέλη της αξιοποίησης ασαφούς λογικής καθώς και την μεγάλη αξία των μηχανισμών αξιοπιστίας και της προστασίας που προσφέρουν στα συστήματα αυτά από κακόβουλες οντότητες. Τα πειραματικά αποτελέσματα σχετικά με τη προσέγγιση Fuzzy AHP, δείχνουν ότι οι τιμές φήμης που παράγονται είναι δίκαιες προς τους παρόχους και ότι ο μηχανισμός αξιοπιστίας βελτιώνει κατα 20% τις επιδόσεις της υπηρεσίας φήμης.

Κεφάλαιο 3

Στο **Κεφάλαιο 3** παρουσιάζεται μια πρόταση για τη διαχείριση του κύκλου ζωής υπηρεσιών σε περιβάλλον MEC με βάρος στην ενορχήστρωση της επικοινωνίας μεταξύ διαφορετικών δικτυακών τεμαχίων (network slices). Το βασικό κίνητρο πίσω από αυτό το κεφάλαιο είναι να επιτρέψει μέσω αυτής της επικοινωνίας την προσφορά προς ενοικίαση υπηρεσιών που έχουν αναπτύξει χρήστες υποδομών MEC σε άλλους χρήστες. Για την επίτευξη αυτού του στόχου σε αυτό το κεφάλαιο παρουσιάζεται μια ολοκληρωμένη πρόταση αγοράς δικτυακών υπηρεσιών (Network Service Marketplace – NSM) βασισμένη σε ΑΕΣ. Παρόλο που παραδοσιακά σε αντίστοιχες αγορές χρησιμοποιούνται κεντρικοποιημένες λύσεις, η λειτουργία τέτοιων μοντέλων προϋποθέτει την εμπιστοσύνη των χρηστών στη κεντρική αρχή, κάτι που δεν καθίσταται δυ-

νατό καθώς στο περιβάλλον που εργαζόμαστε, συμμετέχουν άγνωστοι μεταξύ τους χρήστες και πιθανώς πολλαπλοί πάροχοι υποδομών. Για να ξεπεράσουμε αυτό το πρόβλημα, στην πρόταση μας χρησιμοποιούμε ΑΕΣ διότι επιτρέπει λόγο του αποκεντρωμένου τρόπου λειτουργίας της, των μηχανισμών συναίνεσης και του τρόπου που λειτουργούν τα έξυπνα συμβόλαια να εγγυάται το αποτέλεσμα κάθε συναλλαγής και την ακεραιότητα των δεδομένων. Έτσι η εμπιστοσύνη επιτυγχάνεται μέσω των τεχνικών που χρησιμοποιούνται. Η ΑΕΣ έχει υλοποιηθεί με Hyperledger Fabric. Η πρόταση μας είναι σύμφωνη με το πρότυπο ETSI-NFV και υποστηρίζει όλες τις φάσεις και λειτουργίες ενός NSM, συμπεριλαμβανόμενων της εγγραφής υπηρεσίας, διαφήμιση υπηρεσίας, μίσθωση υπηρεσίας, ενορχήστρωση, χρήση και χρέωση υπηρεσίας. Στο Σχήμα 3.2 παρουσιάζεται η αρχιτεκτονική του προτεινόμενου NSM σε υψηλού επιπέδου επισκόπηση. Όπως φαίνεται από το σχήμα, η αρχιτεκτονική απαρτίζεται από τρία στρώματα τα οποία αντιστοιχούν και ανταποκρίνονται σε διαφορετικά λειτουργικά στοιχεία και λειτουργικές φάσεις.

- **Στρώμα χρήστη (User Layer):** Το στρώμα χρήστη παρέχει τις απαραίτητες λειτουργίες για την αλληλεπίδραση των χρηστών με το NSM και το οποίο βλεπουμε στο αριστερό τμήμα του Σχήματος 3.2. Μέσω κάποιας γραφικής διεπαφής αλλά και εργαλεία γραμμής εντολών, πάροχοι και καταναλωτές υπηρεσιών μπορούν να καταχωρήσουν μια νέα υπηρεσία προς μίσθωση, να ανακαλύψουν υπηρεσίες προς μίσθωση και τα χαρακτηριστικά τους, να αιτηθούν τη μίσθωση μιας υπηρεσίας και να εκκινήσουν και να εγκαθιδρύσουν την επικοινωνία μεταξύ της δικτυακής υπηρεσίας του καταναλωτή και του παρόχου (Cross Service Communication - CSC). Επιπρόσθετα, το στρώμα αυτό παρέχει στου χρήστες δυνατότητες παρακολούθησης μιας μισθωμένης υπηρεσίας και την χρέωση και πληρωμή της χρήσης της. Κάθε αποθήκευση η ανάκτηση δεδομένων πραγματοποιείται με την επίκληση του κατάλληλου έξυπνου συμβολαίου μέσω των διεπαφών του στρώματος ΑΕΣ

- **Στρώμα ΑΕΣ (Blockchain Layer):** Το στρώμα ΑΕΣ χειρίζεται μέσω έξυπνων συμβολαίων όλες τις πληροφορίες που αφορούν τους χρήστες, τις υπηρεσίες κ.λ.π. Όπως βλέπουμε στο μεσαίο τμήμα του Σχήματος 3.2, η αρχιτεκτονική του στρώματος ΑΕΣ μπορεί να εφαρμοστεί είτε με ένα υπολογιστικό κέντρο νέφους παρυφών είτε με πολλαπλά ακόμα και διαφορετικής διαχείρισης. Και στις δύο περιπτώσεις, και υπολογιστικό κέντρο θεωρείται ένας ξεχωριστός οργανισμός για το δίκτυο Fabric. Κάθε οργανισμός διαθέτει ένα Fabric Peer με εγκαθιστημένα τα έξυπνα συμβόλαια και μία αρχή έκδοσης πιστοποιητικών (Certificate

Authority - CA) για τη διαχείριση της πρόσβασης και των ιδιοτήτων των χρηστών καθώς και ένα κατάστιχο (ledger). Οι πληροφορίες στο κατάστιχο είναι πανομοιότυπες σε κάθε οργανισμό. Όλοι οι οργανισμοί συμμετέχουν στο ίδιο κανάλι επικοινωνίας του Fabric και τέλος ο παραγγελιοδότης (orderer) του Fabric χειρίζεται των μηχανισμό συναίνεσης της ΑΕΣ, τη δημιουργία νέων συστοιχιών και τη διανομή τους στον Peer κάθε οργανισμού που συμμετέχει στο κοινό κανάλι. Επίσης, το στρώμα ΑΕΣ προωθεί τις αιτήσεις και την απαραίτητη πληροφορία για την εγκαθίδρυση της CSC στο στρώμα NFV. Η διαδικασία αυτή αναπτύσσεται αναλυτικά στην ενότητα 3.5.2 της διατριβής.

- **Στρώμα Εικονικοποίημένων Δικτυακών Λειτουργιών (NFV Layer:)** Το στρώμα NFV, ακολουθώντας το πρότυπο ETSI-NFV, είναι υπεύθυνο για της εγκατάσταση των δικτυακών υπηρεσιών και την εγκαθίδρυση της επικοινωνίας μεταξύ διαφορετικών δικτυακών υπηρεσιών. Στο άνω επίπεδο αυτού του στρώματος βρίσκεται ο πρωτότυπος ενορχηστρωτής υπηρεσιών (Service Orchestrator) που αναπτύξαμε, ο οποίος προσφέρει πρόσθετες λειτουργίες και αφαιρέσεις στο OSM, προκειμένου να επιτευχθεί πλήρως αυτόματα η εγκαθίδρυση της CSC. Ο ενορχηστρωτής πραγματοποιεί τα παραπάνω αλληλεπιδρώντας με της διεπαφές του OSM και η διαδικασία περιγράφεται αναλυτικά στην ενότητα 3.6.2 της διατριβής.

Το στρώμα χρήστη υποστηρίζει την αλληλεπίδραση παρόχων και χρηστών με την αγορά υπηρεσιών και παρέχει τις ακόλουθες λειτουργίες,

- **Έγγραφή:** Κατά την φάση εγγραφής, ένας χρήστης ενός υπολογιστικού κέντρου νέφους, μπορεί να εγγραφεί στην αγορά. Κατά την εγγραφή του καταχωρούνται στα κατάστιχα οι ελάχιστες απαραίτητες πληροφορίες για την δικτυακή υπηρεσία του χρήστη, όπως περιγράφονται στην ενότητα 3.5.2 της διατριβής, για την ενορχήστρωση της επικοινωνίας με άλλες δικτυακές υπηρεσίες. Στην φάση εγγραφής, ένας χρήστης μπορεί να καταχωρήσει για εγγραφή στην αγορά, μια δικτυακή υπηρεσία που έχει αναπτύξει και θέλει να θέσει προς μίσθωση από άλλους χρήστες παρέχοντας και τις κατάλληλες πληροφορίες για τη φύση της υπηρεσίας, τα κόστη μίσθωσης κ.λ.π

- **Διαφήμιση:** Αφού μια υπηρεσία καταχωρηθεί στα κατάστιχα ως υποψήφια προς μίσθωση, ανακαλείται αυτόματα και διαφημίζεται στον κατάλογο υπηρεσιών προς μίσθωση

ώστε να έχουν πρόσβαση στη πληροφορία αυτή όλοι οι χρήστες της αγοράς. Έτσι οι χρήστες μελετώντας τον κατάλογο μπορούν να βρουν τις υπηρεσίες προς μίσθωση, να διαλέξουν την κατάλληλη για αυτούς και να αιτηθούν τη μίσθωση κάποιας από αυτές.

- **Ανακάλυψη:** Στη φάση ανακάλυψης, οι χρήστες προσπελαύνουν τον κατάλογο με τις υπηρεσίες προς μίσθωση και επιλέγουν τη μίσθωση κάποιας υπηρεσίας βάσει των αναγκών τους.

- **Μίσθωση** Μετά την επιλογή κάποιας υπηρεσίας, ο χρήστης αιτείται τη μίσθωση μιας υπηρεσίας για ένα συγκεκριμένο χρονικό διάστημα το οποίο μπορεί να ανανεωθεί αν το επιθυμεί ο χρήστης. Κατόπιν αιτήσεως, η μίσθωση χορηγείται από τα έξυπνα συμβόλαια που αναπτύξαμε και πραγματοποιούνται αυτομάτως όλες οι απαραίτητες ενέργειες για την εγκαθίδρυση της επικοινωνίας μεταξύ της δικτυακής υπηρεσίας του χρήστη και την μισθωμένη υπηρεσία και ο ενοικιαστής μπορεί αμέσως να ξεκινήσει τη χρήση της.

- **Χρήση:** Στη φάση της χρήσης της μισθωμένης υπηρεσίας, αναλόγως και το μοντέλο κοστολόγησης της κάθε υπηρεσίας, ο βαθμός χρήσης της υπηρεσίας παρακολουθείται και αποθηκεύεται στα κατάστιχα και τόσο ο χρήστης όσο και ο πάροχος μπορούν να τον παρακολουθούν.

- **Χρέωση:** Αναλόγως με τον βαθμό χρήσης της υπηρεσίας που προκύπτει από τα δεδομένα παρακολούθησης μιας μισθωμένης υπηρεσίας και το μοντέλο κοστολόγησης, υπολογίζεται η χρέωση της υπηρεσίας από το αντίστοιχο έξυπνο συμβόλαιο για μια δεδομένη χρονική περίοδο και το αντίστοιχο ποσό πληρωμής αποστέλλεται στο χρήστη. Ο χρήστης μπορεί σε οποιαδήποτε στιγμή και όχι μόνο τη στιγμή της κοστολόγησης να παρακολουθήσει το βαθμό χρήσης της υπηρεσίας και τις αναμενόμενες χρεώσεις ανά πάσα στιγμή.

Για να πραγματοποιηθούν όλες οι παραπάνω λειτουργίες είναι κομβική η ύπαρξη και ο σωστός σχεδιασμός μιας αποθήκης δεδομένων (data store). Ως αποθήκη δεδομένων χρησιμοποιούμε το κατάστιχο της ΑΕΣ το οποίο εγγυάται και την εμπιστοσύνη στα δεδομένα, το αμετάβλητο τους, την ακεραιότητα τους και ασφάλεια. Σε αυτό το σημείο θα αναφερθούμε στα δεδομένα που καταχωρούνται στην αποθήκη δεδομένων. Η αποθήκη δεδομένων περιέχει πληροφορίες για τη διαφήμιση και ανακάλυψη μιας υπηρεσίας προς μίσθωση και την ενορχήστρωση της επικοινωνίας μεταξύ υπηρεσιών. Τα δεδομένα που αφορούν την ενορχήστρωση είναι σύμφωνα

με το πρότυπο ETSI-NFV και περιέχουν πληροφορίες σχετικές με τα τεμάχια δικτύου και τις υπηρεσίες με κατάλληλες ετικέτες για την επικοινωνία με το OSM και την ανάκτηση της απαραίτητης πληροφορίας από αυτό από τον ενορχηστρωτή μας. Η Λίστα 2 απεικονίζει τη δομή και τη μορφή των δεδομένων που αποθηκεύονται στην αποθήκη. Το αντικείμενο τεμάχιο δικτύου (Network Slice) αναφέρεται στους εικονικοποιημένους υπολογιστικούς και δικτυακούς πόρους που απαιτούνται για την ανάπτυξη μιας εφαρμογής. Οι τιμές *ID* και *Name* αναφέρονται σε ετικέτες του προτύπου τεμαχίου δικτύου (Network Slice Template Descriptor) όπως περιγράφεται απο το πρότυπο ETSI-NFV και είναι καταχωρημένο και στη βάση δεδομένων του OSM. Το αντικείμενο Tenant αναφέρεται σε κάποιο χρήστη της αγοράς και περιέχει πληροφορίες για αυτόν. Το αντικείμενο Service, αναφέρεται σε υπηρεσία που διαφημίζεται προς μίσθωση και περιλαμβάνει πληροφορίες για τη διαφήμιση της υπηρεσίας και τις κατάλληλες πληροφορίες και ετικέτες για την ενορχήστρωση της επικοινωνίας αυτής με την υπηρεσία κάποιου μισθωτή. Τέλος, το αντικείμενο Lease, καταχωρείται με την έναρξη κάποιας μίσθωσης και περιλαμβάνει πληροφορίες για τη μίσθωση όπως για παράδειγμα το διάστημα μίσθωσης καθώς και πληροφορίες για τους εμπλεκόμενους χρήστες, τις δικτυακές τους υπηρεσίες και τις ετικέτες αυτών. Όσον αφορά τα αντικείμενα Network Slice και Service, αποθηκεύεται μόνο η ελάχιστη δυνατή πληροφορία ώστε να πραγματοποιηθεί η ενορχήστρωση. Αυτό το κάνουμε για δύο λόγους. Πρώτον, για να προστατεύσουμε την ιδιωτικότητα των δεδομένων του χρήστη. Αυτό το επιτυγχάνουμε αποθηκεύοντας μόνο ετικέτες που δείχνουν στην πληροφορία σχετικά με μια υπηρεσία και είναι αποθηκευμένη στο OSM η οποία ανακτάται χωρίς να αποθηκεύεται όποτε χρειάζεται και όχι ολόκληρη τη περιγραφή αυτής που θα εξέθετε πληροφορίες για την εσωτερική δομή και το τρόπο λειτουργίας της υπηρεσίας. Η διαδικασία αυτή και η αναλυτική περιγραφή της λειτουργίας του ενορχηστρωτή περιγράφεται στην ενότητα 3.6.2 της διατριβής. Δεύτερον, για τη βελτιστοποίηση της αποθήκης δεδομένων με κριτήριο και την κλιμακωσιμότητα της λύσης μας ελαχιστοποιώντας τον όγκο των αποθηκευμένων δεδομένων. Η πειραματική αξιολόγηση της λύσης έδειξε την ορθή λειτουργία του ενορχηστρωτή μας και την αποδοτικότητα και τη βιωσιμότητα του όσον αναφορά τις επιδόσεις και τους χρόνους ενορχήστρωσης καθώς και του πόρους που αυτή η επικοινωνία καταλαμβάνει σε σχέση με άλλες γνωστές προτάσεις της βιβλιογραφίας.

Κεφάλαιο 4

Τέλος, το **Κεφάλαιο 4** , συνοψίζει το σύνολο της διατριβής, επιχειρηματολογώντας για τη σπουδαιότητα των εξεταζόμενων ερευνητικών προβλημάτων και των σχεδιαστικών μεθόδων που επιλέχτηκαν για την επίλυση τους, ενώ παράλληλα παραθέτει συγκεντρωμένα τα κύρια συμπεράσματα που ανέκυψαν. Τέλος προτείνονται ανοιχτά ερευνητικά θέματα για μελλοντική εργασία που είτε θα μπορούσαν να αποτελούν την συνέχεια αυτής της ερευνητικής προσπάθειας, είτε μπορούν να εκμεταλλευτούν την αποκτημένη γνώση προκειμένου να την εφαρμόσουν σε νέους τομείς και δραστηριότητες.

Preface

Acknowledgements

Writing this thesis, I would like to deeply thank several people for their love and support during the years which I pursued my Ph.D.

First of all, I would like to thank my advisor, Professor Symeon Papavassiliou, for his guidance and support throughout my Ph.D. studies and for giving me the opportunity to join the NETMODE team and collaborate with many esteemed colleagues.

Special thanks is owed to Dr. Dimitris Dechouniotis. Since the beginning of my Ph.D. studies, Dimitris guided and supported me in every way possible and this dissertation wouldn't be possible without him. I thank him for the productive collaboration and for the privilege to call him a friend.

In addition, I would like to specially thank Giannis for our collaboration in the work presented in this dissertation and his friendship alongside my other good friends and colleagues Marios, Giorgos and Dimitris. I feel honored to have met them and consider them good friends who were always willing to listen to my problems and provide me with the necessary psychological support.

I would also like to thank all the other colleagues in NETMODE laboratory for the precious (and fun) moments we shared the past years.

From the bottom of my heart, I would like to thank my dear friend Eleanna, and my long-time friend and roommate Alex for all the psychological, emotional and even financial support throughout the years and for making my life brighter alongside my ED friends and Theodora and finally my family, my father Vassilis, my mother Elpida and my brother Giorgos, for their unconditional love, understanding and support throughout my life.

Structure

This dissertation is structured as followed.

In Chapter 1 we make a general introduction on the topics that will concern us in our thesis, set the environment we will consider and examine, highlight the key motivating factors and challenges of our work, and finally exhibit the key contributions and observations of this PhD thesis.

The following two chapters constitute the main chapters of the thesis. They present more specific problems that we consider important and we solved during the PhD period.

In both chapters, first a general setting, specific to the problem and related work on the topic is provided, while subsequently our proposed solutions are discussed, by presenting the system models, the mathematical approach to the considered problem and the architecture of our solution. Finally the performance of the proposed frameworks are evaluated in details.

Specifically, in Chapter 2, we will talk about Multi Criteria Decision Making (MCDM) - based Trust management for heterogeneous Cloud Federations. Using and modifying two MCDM systems, we propose two collaborative Service Level Agreement and reputation trust management approaches. Their role is to enable trust between untrusted parties in a multi-user, multi-provider environment, assist the user in cloud selection and assess providers and resources combining multiple QoS and QoE criteria.

In Chapter 3, we talk about trustless systems, and in particular blockchain, as a way to enable trust in a multi-domain federated edge cloud environment. More specifically leveraging blockchain technology and smart contracts, we enable trust in the transactions and use the distributed ledger as a shared database between domains to create a network service marketplace for off-the-shelf leasing. Our proposed marketplace in supported by our custom cross service communication orchestrator in order to automate the communication between services and the consumption of off-the-self leased services.

Finally in Chapter 4, we summarize and conclude the dissertation. In addition, we discuss and present future extensions that are of high research and practical importance, and deserve to be further explored. The thesis ends with some Appendices containing additional complementary and relevant information, that allow the reader to become familiar with some of the approached used in this thesis, which however would otherwise break the flow of

the main text. Whenever this is the case during our thesis, the interested reader is referred to the relevant appendix.

Chapter 1

Introduction

Over the past decades, the generated data traffic has exploded with significant increase the latest years. According to CISCO prediction [4], overall mobile data traffic is expected to grow to 77 exabytes per month by the year 2022. One of the main reason of this data explosion is the outburst of the number of connected devices from smart phones, to vehicles, sensors, home appliances etc., which are estimated by the end of the year to reach up to 50 billions devices [5]. The network connectivity of such devices composes what is called the *Internet of Things (IoT)* . The IoT paradigm aspires to revolutionize every aspect of our life by providing extravagant possibilities for application and service deployment. Smart homes, cities, cars, energy networks, education, agriculture, commerce and logistic are only some examples of the capabilities IoT introduces [6], [7]. While IoT devices open a wide spectrum of new services and applications, they also come with many constraints due to the nature of these devices. Most IoT devices come with computational, storage and power limitations, thus convergence with other technologies is needed in order to deliver high-end services. *Cloud Computing (CC)*, *Edge/Fog Computing (EC)* are key enabling technologies for IoT applications and services [8] by allowing IoT devices to overcome those limitations by sending data for storage, processing and analysis to Cloud and Edge/Fog data centers [9].

1.1 Cloud Computing

Cloud Computing is probably the biggest game changer of the past decade. NIST defines cloud computing as "a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction" [10]. In practice, CC offers access in a pay-as-you-go manner, to an almost unlimited pool of resources with very low cost and without the user's or enterprise's obligation to manage anything. This capability disrupted many industries and through Software as a Service (SaaS) model, has changed the way people work, collaborate or even entertain, with SaaS products that vary from the Google Suite to Netflix.

Conventional cloud architectures favour centralized solutions of huge limited data centers across the globe. These data centers can facilitate simultaneously and scale a very large number of users and services. Although CC has been a huge success in the current internet, its centralized architecture brings certain challenges and limitations to the IoT era. The huge volume and velocity of data accumulation of IoT devices, which is continuously increasing, makes it almost impossible to transfer them to the remote cloud data centers. In addition, due to the large distance between the IoT devices and the data centers, latency issues of end-to-end communications arise for time-critical applications and services [11].

To tackle this challenge the Edge/Fog computing paradigm has started to be adopted. Edge/Fog computing shares some common principles with CC by offering virtualized compute, network and storage resources. The key difference with CC is that the Edge/Fog model favors a large amount of smaller data centers instead of fewer larger ones of CC. This is done mainly to overcome the latency challenge described before, by bringing these small data centers close to the IoT devices. A quick overview and comparison between CC and EC is presented in figure 1.1 [9]. Edge and Fog computing are favoured over the cloud when more location awareness, mobility support, geographical distribution responsiveness and devices are needed by application and services. While the Edge and Fog models provide solutions to many challenges and limitations of the traditional cloud architectures, it is not meant to completely replace it. CC can be a good candidate when reliable connectivity,

Characteristics	Conventional cloud computing	Edge cloud and edge computing
Major applications	Most of the current mainstream cloud-involved applications	Applications on IoT, VR, AR, smart homes, smart cities, smart energy, smart vehicles, etc.
Availability	A small number of large-sized datacenters	A large number of small-sized datacenters
Proximity of services and resources; Data processing location	Usually in remote datacenters and far from users	At the edge close to the users
End-to-end latency	High, due to the distance between the edge and remote datacenters	Low, due to proximity to the users
Backbone network bandwidth consumption	High, since huge data need to be transferred to the datacenters first	Low, since data are locally processed and stored in edge cloud
Scalability	Scalable at center	Scalable both center and edge
Security (e.g., attacks on data enroute)	Data subject to attack due to long-distance transmission; Physical security depends on large facilities	Lower risk for enroute attacks; Physical security varies and different mechanisms needed

Figure 1.1: Brief comparison of CC and EC

large amount of computing power, storage is needed. In many scenarios, depending of the requirements of the application, both will be used as different layers, for different actions and operations along with other enabling technologies. The trade-offs and model selection criteria depending on requirements in a layered architecture between Fog/Edge and Cloud is depicted in figure 1.2 [12].

1.2 5G Networks and Enabling Technologies

As aforementioned, while the era of IoT devices brings new and vast capabilities of services and applications, it also needs the combination of various techniques and technologies to achieve them due to device limitations and by very high service requirements. Following new advancements and innovations in mobile networks and by incorporating various technologies (including CC and Edge/Fog), the fifth generation (5G) technology and network standard, which describes the internet era we are now entering, is defining requirements,

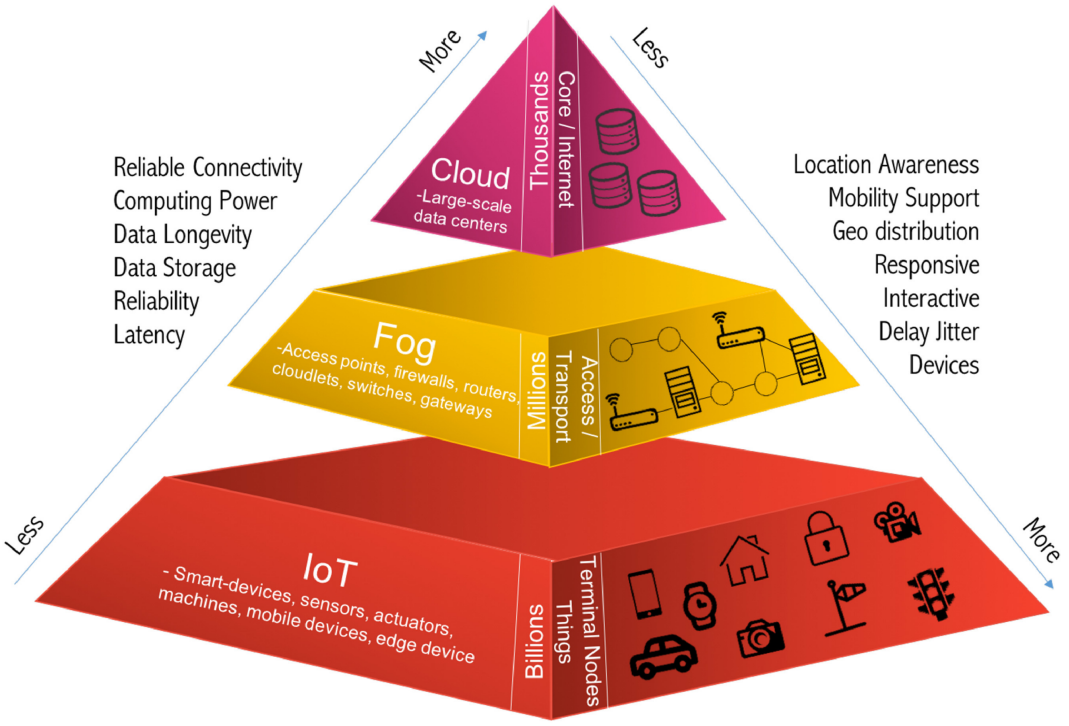


Figure 1.2: Fog/Edge and cloud trade-offs and Layered architecture

architectures and technologies for modern networks in order to unlock the full potential for future service delivery. ITU-R has defined the following main usage scenarios for 5G in their Recommendation [13] to enable such service deployments:

- **Enhanced Mobile Broadband (eMBB)** to deal with hugely increased data rates, high user density and very high traffic capacity for hotspot scenarios as well as seamless coverage and high mobility scenarios with still improved used data rates
- **Massive Machine-type Communications (mMTC)** for the IoT, requiring low power consumption and low data rates for very large numbers of connected devices
- **Ultra-reliable and Low Latency Communications (URLLC)** to cater for safety-critical and mission critical applications

In order to achieve the goals of each category, the following requirements must be met [13] [14] [15] [16]:

- **1-10 Gbps data rates in real networks**: This is almost 10 times increase from traditional LTE network's theoretical peak data rate of 150 Mbps.

- **1 ms round trip latency**: Almost 10 times reduction from 4G's 10 ms round trip time.
- **Enormous number of connected devices**: In order to realize the vision of IoT, emerging 5G networks need to provide connectivity to thousands of devices.
- **Perceived availability of 99.999%**: 5G envisions that network should practically be always available anytime and anywhere.
- **High battery life**: Reduction in power consumption by devices is fundamentally important in emerging 5G networks and IoT scenarios.

To meet all these requirements new technologies and architectures are introduced. Starting with *Radio Access Networks (RANs)* technologies, *Millimeter wave (mmWave)* and *Terahertz (THz)* band promise high data rates, coverage and device density. Millimeter wave provides great amount of available bandwidth, which enables gigabit per second throughput, and it usually refers to frequency bands between 30-300 GHz [17]. Alongside mmWave, which is production ready, THz band has attracted research interest. The band spans the frequencies between 0.3 THz to 3 THz with a broad overlap over mmWave frequencies. While these frequencies are unregulated and currently are a "no man's land" [18], because THz band offers great amount of spectrum resources which can support data rates of more than 100 Gbps or even 1 Tbps and at the same time be backward-compatible with mmWave, thus, THz band is the next frontier of research on wireless communication networks. Last but not least in RANs technologies comes the *Massive MIMO* technology. Massive MIMO is an extension of multiuser MIMO deployed in 4G systems, in which only several tens of antenna components are built on base stations. On the contrary, massive MIMO designs hundreds of antennas at base stations to further increase capacity and system throughput [19]. The benefits of massive MIMO include a huge increase in spectral efficiency, reduced latency, and a scalable air interface structure.

At the networking side, one of the most major innovations is *Software Defined Networking (SDN)*. With SDN, traditional networking devices are replaced with commodity generic hardware and the networks can be dynamically and on the fly configured by software without the need of hardware replacement, addition or repositioning. In [20], SDN is defined as a network architecture where the control plane and data plane are decoupled, removing that

way the control functionality from devices. The control logic is moved to an external entity called the SDN controller, that is basically software running on a commodity server that offers abstractions to facilitate the programming of forwarding devices. Finally the network is programmable through software applications running on top of the SDN controller. SDN offers several advantages compared to legacy networks. Network configuration and programming becomes easy, all applications take advantage of the same global network view, thus leading to more consistent and effective policy decisions. Furthermore, applications can take actions dynamically and reconfigure any part of the network at any time.

Complementary to SDN comes *Network Function Virtualization (NFV)* [21]. NFV enables the virtualization of entire network functions or services that were previously tied to stringent and costly dedicated hardware. Therefore, they can run on commodity hardware and servers such as EC infrastructure. This new paradigm allows the consolidation of network functions and services on virtualized resources, such as virtual machines, on CC or EC infrastructure, saving in this way capital and operational expenditures while allowing flexibility in both data and control plane by scaling up and down the allocated resources dynamically as demands evolve [22]. NFV can benefit complex service deployment by allowing portability, as services and service chains can be easily moved to another infrastructure, enabling federation support for deployment of portable services across interoperable geographically distributed infrastructures and virtual networks and finally through slicing, which means the custom tailored partitioning of network resources for particular applications [23], a concept of high importance which will be analyzed now.

Network slicing is a key concept that can offer an agile networking platform to support services with different functional requirements in an efficient way [24]. Network Slicing comes to life with the combination of several 5G enabling technologies and having in its core EC, SDN and NFV ones. The concept is that from the same infrastructure, with virtualization techniques, several network slices can be deployed, each of them designed and optimized for different specific requirements of their corresponding application or service[25]. A network slice is a logical isolated, self-contained network with custom operation in a multi-tenant environment. An example of different application and services, with different requirements, facilitated by the same infrastructure with slicing techniques is illustrated in figure 1.3 [23]. As we can see, a Mobile Broadband slice that requires high capacity and performance

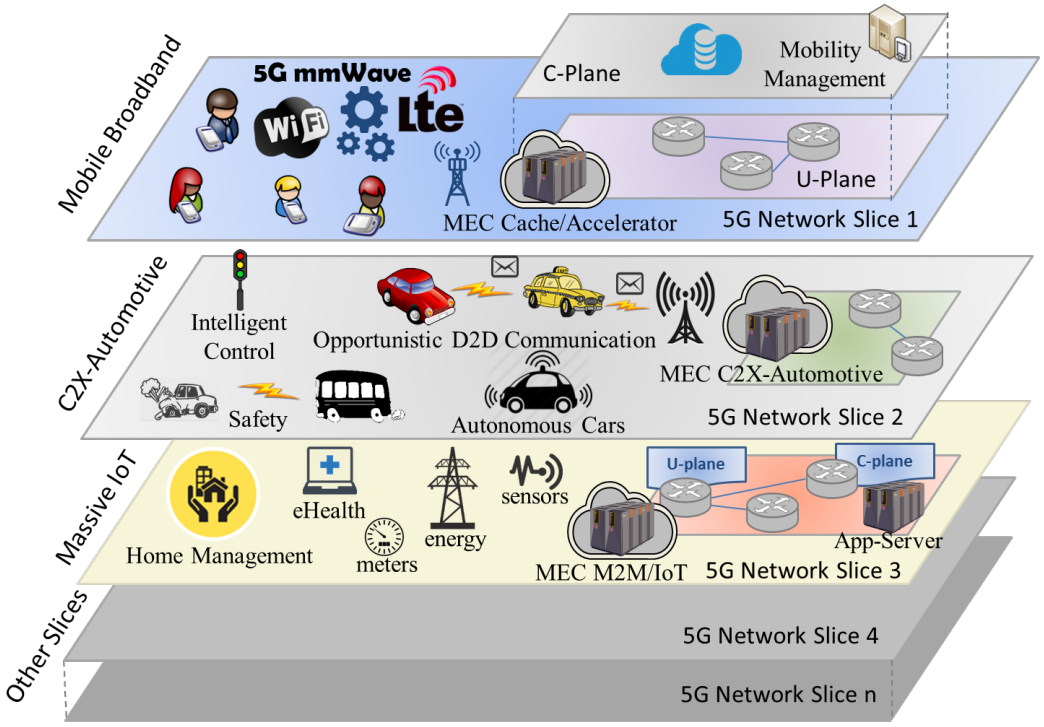


Figure 1.3: Network slicing with EC for different applications

is running simultaneously with an Automotive slice that requires strict latency and and massive IoT slice that requires scalability and handling of huge amounts of small data.

1.3 Service Life-Cycle Management

In the 5G and IoT era, as described and above, services become more and more complex. The traditional client-service model is been replaced by more distributed models consisting of several software components and underlying technologies. Such models can contain cloud federations, EC and CC synergies and multi-domain end-to-end network service chains and slices. The complexity of the new service delivery models raise new challenges for the fulfilment and management of the service life-cycle. The life-cycle of a service may vary a little depending of its nature and the underlying technologies but typically can be summarized by the following stages:

- **Service Discovery:** The user discovers through automated service discovery catalogs various services and providers.

- **Service Selection and Request:** After discovery, the user selects and requests the stack of services according to his/her needs.
- **Service Orchestration:** After service request, the orchestration process begins. By orchestration, we refer to all the necessary steps, from resource provisioning, to virtualization, networking and software deployment, executed in the correct sequence in an automated fashion, even among multiple domains and underlying technologies. After the orchestration process the service should be up and running with no further actions required by the user and ready for consumption [26].
- **Service Management and Monitoring:** Management is responsible for maintaining and healthiness of both service and infrastructure. Its role consists of activities such as software updates, resource optimization, scaling and event detection. Monitoring provides the appropriate data to the previous tasks and provides data to ensure that the agreed *Quality of Service (QoS)* is guaranteed, as it is described by a Service Level Agreement (SLA). In addition monitoring provides usage data for billing and other services.
- **Service Billing:** According to the usage data and the agreed billing plan, the fee for the service is calculated accordingly over the predefined billing time span.

Every stage of this life-cycle has new challenges to be addressed in the 5G era. Since services can be provided by both infrastructure providers or tenants, a marketplace need for such services occurs. This marketplace would involve multiple providers and users unknown to each other and usually with lack of desire to reveal information about their infrastructure or services.

In this context, it is critical to provide a sense of trust between users and providers. Trust is defined as the subjective belief of entity A, that entity B performs a given action [27]. These challenge can be faced with the use of trust management frameworks. In such frameworks, the performance, reliability and trustworthiness of a service and its provider can be quantified and be provided to users. This can be done through trust algorithms with the combination of service monitoring data and user evaluations so that the quantified metric can represent both the Quality of Service and the *Quality of Experience (QoE)* of the users consuming the service. Also, there is need of a fail-safe mechanism to prevent malicious

actors from manipulating trust metrics to either favor or sabotage a provided service. In addition to enabling trusted marketplaces, it is easily concluded that a trust management framework also assists the user in the selection stage.

Another way to tackle the lack of trust is by using trustless systems to handle actions or even whole stages from the service life-cycle. A trustless system is a system that the participants involved do not need to know or trust each other or a third party for the system to function. In such systems, the users trust the system what will always produce a predefined and agreed outcome. This can be achieved with the usage of consensus mechanisms that guarantee the outcome in a secure manner. Trustless mechanisms can be used to perform or assist the discovery and selection stage, the orchestration stage, the management and monitoring stage and the billing stage.

1.4 Challenges & Motivation

As highlighted previously, the IoT and 5G era brings numerous of new capabilities and possibilities with new technologies and service delivery models. Alongside these new possibilities, new challenges and problems arise due to the complexity of the service delivery models. This affects in several ways the life-cycle management of such services. These new possibilities drive us to attempt to address life-cycle management challenges concerning mainly:

- **Enabling trust:** The complex service delivery models of 5G services require multiple providers, users and devices to interact and transact. It is critical in such context to provide a sense of trust in order to utilize such services and service marketplaces.

- **Service assessment over heterogeneous resources:** Since modern services frequently combine several types of devices and resources, the quantification of trust is more challenging as different metrics and requirements must be considered.

- **QoS and QoE defined assessment:** As services with different functional requirements often share the same underlying infrastructure, our work focuses on including in the performance and reliability assessment of the resources and services, the subjective perception of the end user's QoE besides raw monitoring data of an infrastructure's performance.

- **User defined service selection:** Since every user has different needs, different priorities occur about the performance of a given service. Thus, those priorities need to be examined and facilitated in the service selection process.
- **Mitigation of malicious actors:** Since the manipulation of assessment values of the resources has huge economic incentives for adversaries, it is of high importance to provide mechanisms to protect honest providers and users.
- **Service offering, discovery and leasing:** In a multi-tenant multi-provider environment, it is important to provide mechanisms so users can provide services for off-the-self leasing in an automated fashion. This can simplify the creation of custom-tailored service chains and give economic incentives for both providers and tenants.
- **Cross-Service Orchestration:** Although commonly network services are deployed in isolated network slices, it is vital to orchestrate cross-service communication in order to allow the consumption of off-the-self services.

1.5 Contributions

This thesis aims at tackling some of the aforementioned problems that arise in the service life-cycle management in a federated cloud environment. We mainly focus on enabling trust between untrusted users and providers, performance assessment, service selection and monitoring. Our contributions on the above topics are:

1. **Trust and trustless frameworks for federated edge clouds:** We introduce two trust-based frameworks and one trustless approach in order to enable trust between untrusted parties in a multi-tenant provider and device environment.
2. **Reputation-based service assessment:** In order to assess services and quantify the collective trust of their performance, we propose two different reputation based trust management frameworks that can assess services over heterogeneous resources.
3. **Multi-criteria decision making methods for service assessment with user-defined priorities:** In our proposed frameworks, we exploit multi-criteria decision making techniques. This way, our solutions can process simultaneously various types

of data (e.g. binary, numeric, linguistic), while users can adjust weights according to their specific needs and functional requirements.

4. **SLA-based assessment:** *Service Level Agreement (SLA)* is the fundamental mechanism that allows users to enforce guarantees about performance and conformance of services. Our proposed solutions integrate SLA violation metrics for the performance assessment of services.

5. **Credibility mechanism for malicious evaluations filtering:** We introduce a credibility mechanism to secure our frameworks from malicious users. Based on the credibility score of the evaluating user, monitoring data, SLA data and their relation to the user's evaluation, we filter and adjust biased opinions protecting in this way honest users and providers.

6. **Blockchain-based Network Service Marketplace:** We introduce a blockchain-based network service marketplace that can support all steps in the life-cycle of network services, from advertisement, discovery and leasing, to orchestration monitoring and billing.

7. **Novel cross-service orchestrator:** We developed and introduced a custom automated cross-service orchestrator over NFV reference architecture to facilitate communication between services and isolated slices in order to enable the consumption of off-the self services.

8. **Implementation and Evaluation of proposed frameworks with numerical results over real infrastructure:** For all the above contributions mentioned, we performed simulations and experiments in order to capture the effectiveness and the efficiency of the proposed frameworks. The proposed frameworks were implemented and evaluated over actual edge cloud resources.

Chapter 2

MCDM-based Trust Management of Heterogeneous Federated Clouds

2.1 General Setting

As mentioned in Chapter 1, in the modern era, the composition and deployment of services require the combination and synergy of various heterogeneous resources and technologies. Furthermore, many applications are geographically dispersed and multi-cloud architectures or cloud federations are utilized, which involve the interaction between private and public clouds controlled by different providers. In federated cloud scenarios, a cloud acquires and/or offers spare capacity to a set of providers and/or users. Cloud federations enable scalability, fault tolerance and elasticity to the cloud environment,

The application life-cycle in such cloud environment has several phases including authentication, resource discovery, booking, provisioning and application deployment, monitoring, management and retirement [28]. The utilization of heterogeneous resources poses complex requirements at every stage of the life-cycle. Two of the most important management services, which are involved in various phases of cloud application's life-cycle, are SLA and trust management.

SLA is the fundamental mechanism that allows users to enforce guarantees about performance and conformance of single or federated Cloud services [29]. Service Level Agreements specifically establish the consensus on the characteristics of the service to be provided between the service provider and the cloud service user.

The second life-cycle management service addressed in this chapter is trust management. In the cloud application life-cycle, trust management facilitates the resource selection and the performance evaluation of the cloud application or provider. In a federated environment, many providers offer similar resources and applications and the selection of the appropriate ones is a non-trivial task.

Considering this setting and the challenges mentioned above and in Chapter 1, in this chapter we present two Multi-criteria decision making based trust management frameworks for heterogeneous cloud federations. Both solutions have a collaborative SLA and reputation-based trust management approach which we call Hybrid Reputation Systems. Our proposed solutions use well-defined Quality of Service (QoS) and Quality of Experience (QoE) metrics to quantify the collective confidence on the cloud application's or provider's performance and enable users to properly select those who fulfill their specific requirements facilitating both performance monitoring and evaluation as resource selection. Both our solutions are protected with credibility mechanisms from malicious evaluations and attacks.

2.2 Related Work

A cloud federation provides various types of resources, such as network, storage, compute resources. Thus, in this section, we present and classify, the most interesting trust management and reputation approaches on different types of services, resources and networks in literature as also in Cloud SLA management.

2.2.1 Cloud SLA Management

The SLA current operational approaches in cloud providers are primarily limited to availability [30] (e.g. Amazon [31], Rackspace [32]). Beyond this, in research environments, SLAs and more broadly Service Level Management frameworks take often more complex forms in managing QoS in cloud distributed environments. They are mainly motivated by

the fact that the managed resources belong to different administrative domains. In these approaches, it is common that negotiation phase is implemented, since uttering expected QoS is not an acceptable possibility. Multiple projects [33], specifically addressing Cloud brokerage, have studied this problem from diverse perspectives. Cloud4SOA [34] project produced a framework facilitating dynamic SLA negotiation on multi-cloud Platform as a Service environments. It's SLA framework permits publication of offerings, as well as SLAs enactment for agreed QoS terms. Specifically, it considers business dynamics through business performance related SLA metrics. OPTIMIS [35] considered diverse deployment and run-time configuration scenarios. These included private, bursting, federated and multi-cloud deployments. OPTIMIS studied SLA negotiation and management in these scenarios. The QoS terms considered in OPTIMIS SLA's included operational Cloud capabilities. In addition to these, additional SLA terms were considered: service or provider's risk, trust, ecological or cost levels, as well as, legal requirements (related to personal data management) [36]. A significant number of these works are based on standardization efforts performed in Grid computing environments in WS-Agreement [37]. WS-Agreement is a full recommendation of the Open Grid Forum. WS-Agreement provides protocol and specific language in order to generate SLAs.

Other models besides WS-Agreement has been proposed. SLA@SOI[38] developed the SLA(T) model, that enables the description of both functional and non-functional characteristics of a service. The model allows providers to describe the offered services, and customers to describe their requirements and discover matching offers. Also, it allows multi-layered SLAs, which can be composed along functional and organizational domains. CONTRAIL[39] adopted the SLA(T) model and extended it to offer elastic PaaS services over a federation of IaaS clouds, while dealing with QoS and SLA management. The scenario considered in CONTRAIL is focused on cloud federations and automated generation of SLA offerings. In CONTRAIL, the user negotiates a SLA with the cloud federation, and the federation satisfies it by negotiating SLAs with the federated providers on behalf of the user. A different scenario was proposed in MODAClouds [40], which considers that three actors take part in cloud SLAs; end users, Application Providers and Cloud Service Providers. Within this context, Application Providers lease resources from Cloud Service Providers to offer services to end users. Then, MODAClouds devises a two-level SLA system building up an

aggregation of WS-Agreement SLAs. The first level describes the QoS to be offered by the Application Provider to the end users, incurring in penalties in case of SLA violations; this SLA only monitors for observable metrics by the end user (e.g., availability, response time). The second level describes the expected QoS from the Cloud Service Provider to the Application Provider for each of the resources, resulting in one agreement per VM; this second level is not an actual agreement, but just monitors the service offered by the Cloud Service Provider in order to be able to react and enforce the first level SLA.

With regards to security SLAs, SPECS[41] delivered an open source framework that provides SLA life-cycle, automatic negotiation and monitoring of security parameters specified in the SLAs. The proposed model is based on the WS-Agreement Standard. MUSA [42] presents a solution to SLA-based security assurance for multi-cloud applications, whose components are deployed in distributed cloud services. It enables the automatic creation of the offered Security SLA of the multi-cloud application, but it also enables to monitor at runtime the security service level objectives specified in the SLA. The proposed SLA composition adopts the SPECS cloud Security SLA model.

2.2.2 Trust Management for Web Services

Wahab et. al [43] present a complete survey on trust and reputation systems for three types of web services, named single, composite and communities. Authors of [44] proposed a Bayesian network reputation and trust model for single web services that is based on direct user feedback, the recommendation of other users and QoS data. Furthermore, a credibility mechanism for the users is provided. RATEWeb [45] is a trust framework for selecting and composing web services. The reputation score is computed with a statistical method that utilized the credibility of the users, their personalized references, the immediate knowledge and the temporal sensitivity. In [46], a statistical approach was proposed to provide trust value to the parts of composite services. An on-line expectation maximization algorithm is used to assign trust to the individuals behind the service according to their contribution to the overall performance. A game-theoretic model for composite services was proposed by Yahyaoui [47]. A Bayesian model was used to derive the trust value of each service for possible collaboration with other services. This value was used to compute a trust-based cost in order to find the winner service, which was eventually allocated with tasks.

2.2.3 Trust Management for Ad-Hoc Networks

Several approaches were proposed for reputation and trust management in wireless sensor and ad-hoc networks. In [48], a reputation framework for data integrity in wireless sensor networks was presented, where each node evaluated the past activities and predicted the future behavior of other nodes by maintaining reputation values. A Bayesian formulation was adopted for reputation representation and evolution. Furthermore, a consensus-based outlier scheme was used as credibility mechanism of data reading. Ren et. al [49] presented a trust management approach for unattended wireless sensor networks based on Subjective Logic. This study aims at providing trusted data storage and generation. Also authors used trust similarity function to detect outliers and protect form trust pollution attacks. Authors of [50] proposed a information-theoretic trust model for ad-hoc networks in order to provide secure routing and malicious user detection. Four axioms defined the concept of trust and the rules of trust propagation and a entropy-based and a probabilistic model are used for computing the level of trustworthiness by concatenation and multipath propagation. CATrust [51] is a context-aware trust management framework for service-oriented ad-hoc networks, such as Internet of Things (IoT) networks. CATrust used logistic regression to compute the trust value of mobile nodes based on their service behavior patterns in response to context environment changes. Complementary, a threshold-based recommendation filtering mechanism was designed to effectively filter out dishonest recommendations. ART [52] aimed at detecting malicious attacks and evaluating trustworthiness of mobile nodes and data in vehicular ad-hoc networks. In this study, node trust has a two-dimension meaning in terms of fulfilling a functionality and recommendation to other nodes. Dempster-Shafer theory of evidence is used for data analysis and these evidences are utilized to derive the trustworthiness of node and data. The recommendation trust of nodes are evaluated by collaborating filtering.

2.2.4 Trust Management for Cloud Computing

In cloud computing environment, reputation and trust management systems have broadly used for provider selection or security. CloudArmour [53] is a reputation-based trust management framework that focused on availability and security. A credibility model was proposed

to detect feedback collision and Sybil attacks while an availability model spreads the trust management service nodes in order to manage the users' feedback in a decentralized manner. Manuel [54] proposed a trust model for resource selection in heterogeneous cloud resources based on past credentials and present capabilities of a cloud resource provider focusing on four parameters. Those can be availability, reliability, turnaround efficiency and data integrity. The proposed solution also leveraged from combined usage of SLA and reputation. CloudRec [55] is a recommendation mechanism designed for mobile cloud services. It is based on adaptive QoS management and monitors the performance of cloud services and recommends the ideal one to users according to their contextual information. Yan et. al [56] proposed a data access control scheme for cloud computing based on individual trust and public reputation values. These values are used to apply Attribute-Based Encryption and Proxy Re-Encryption. Hatman [57] is a reputation based trust management framework for Hadoop based clouds. It is based on EigenTrust [58] approach to improve the data integrity of distributed cloud computations.

2.2.5 Trust Management for SDN and NFV

With the advent of 5G technologies, many researchers focused on Network Function Virtualization (NFV) and SDN. The trust management of this type of networks is an open challenge. In [59], the authors proposed a network function virtualization infrastructure trust platform that was responsible for the QoS guarantee of a virtual network function (VNF) fulfilled the user's requirements. The reputation of VNF is quantified by local monitoring data and from trust information of other devices. Giotis et. al [60] proposed a reputation threshold-based mechanism for cooperative SDN domains in order to mitigate the distributed denial of service attacks. The reputation score is based on a Bayesian method. FlowBroker [61] is a brokering agent architecture suitable for the coordination of distributed SDN controllers. The broker's reputation is based on metrics of the end-to-end delay, the max link utilization ratio and the packet loss ratio and is used by other agents in order to accept flow rule changes and peer broker forwarding updates. A machine learning method, named Linear Discriminant Analysis, is adopted for the quantification of the reputation of each broker.

2.2.6 Trust Management for Peer to Peer Networks

An interesting category of trust and reputation systems are systems designed for peer to peer (P2P) networks. The following trust management frameworks are also applied to other distributed systems. EigenTrust algorithm [58] was designed to protect P2P network users from downloading malicious or inauthentic files. It calculated and assigned a global trust value to every peer by using recursive method and the users opinions. The algorithm was based on power iteration and can be implemented both as a centralized and distributed service. PBTrust [62] is a priority based trust model oriented for service selection and consists of the Request Module, the Reply Module, the Priority-based Trust Calculation Module and the Evaluation Module. The Priority-based Trust Calculation Module computed the reputation scores considering the following factors; the similarity between the priority distribution of the requested and referenced services, the suitability of the potential providers for the requested service, the providers experience of service, the ratings of third party actors and the timestamp of the third party ratings to eliminate out of date opinions. The ROCQ mechanism [63] proposed a reputation based trust management system that produced reputation values in order to represent the trustworthiness of peers in P2P networks. Evaluation of peers are provided after the end of each transaction between peers. The ROCQ system can be implemented in a distributed manner and the reputation value was based on the user's opinion, the user's credibility produced by his/her previous evaluations and the quality that represents the confidence of a peer on the accuracy of his/her evaluation.

2.2.7 Trust Management for Heterogeneous Resources

There are very few studies that proposed a reputation or trust management approach for different types of resources. FTUE [33] is a reputation based trust management framework for federated testbeds. It utilized user feedback and monitoring data to compute the reputation metric per service per testbed. Four different scenarios are used to characterize the user's credibility which was considered on the reputation score. Brinn et. al [64] proposed an approach for federated trust in the context of GENI project. The concept of trust had three different meaning, named credibility, endorsement and reliance. GENI provided authentication and authorization mechanism to realize trust operation between the federated

entities. Zhu et. al [65] proposed an authenticated reputation and trust management system for integrated cloud and wireless sensor networks. This systems focused on cloud and sensor providers protection from malicious attacks and enabling users to select the proper providers based on their reputation and trust values. The trust value derived by the processing, data privacy and transmission capability of the cloud provider, while for sensor providers data collection, network lifetime, response time and data transmission metrics were used. The reputation score for both types of providers derived by feedback of previous SLAs about a service.

The overwhelming majority of the aforementioned approaches have focused only on a specific type of resources. In a federated environment, the calculation of reputation and trust value is more challenging, because there are many different metrics and requirements that must be considered.

2.3 Contribution & Outline

Specifically, to overcome the aforementioned limitations, we introduce two *Multi Criteria Decision Making (MCDM)*-based trust management frameworks that take simultaneously into account various QoS, QoE and SLA data from different type of resources and cloud providers in a scalable way. Briefly, the basic contributions and differences of our proposed approach and frameworks in this chapter are summarized as follows:

1. We present an architecture for collaborative SLA and *Reputation-based Trust Management (RTM)* frameworks for cloud federations. Our architecture is scalable and can support numerous cloud providers and various cloud federation architectures.

2. SLA offerings are defined based on cloud application's *Key Performance Indicators (KPIs)* and provide reports on SLA violations. The SLAs support more complex KPIs and QoS metrics, such as cloud application response time while most cloud providers offer only availability.

3. We introduce two MCDM-based trust management frameworks. MCDM methodologies can take into account multiple KPIs combining QoE and QoS input. Using such methodologies our frameworks provide Reputation-based Trust Management, which

utilizes user ratings of cloud providers in order to produce a reputation score which reflects the cloud's performance and reliability. Consequently, the reputation score facilitates the selection of the appropriate services and resources by future users according to their needs.

4. Credibility mechanisms are developed to protect the output of the RTM frameworks from biased evaluations and attacks. They measure the deviation of the customer's rating from objective SLA and monitoring values, and their output is considered by the RTM frameworks to mitigate malicious evaluations.

5. Both proposed frameworks have been implemented and evaluated in Cloud Federation-like environment. In addition SLA validation is performed and detailed numerical results are provided that demonstrate the reputation score calculation process of the proposed frameworks and the effectiveness and adaptability of the credibility mechanisms.

2.4 Cloud Application Life-Cycle and Collaborative SLA-RTM architecture

Cloud computing is composed of three service models; Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). With IaaS, a provider supplies the basic computing, storage and networking infrastructure along with the hypervisor (the virtualization layer) and all the installation and configuration process depends on the users. With PaaS, a provider offers more of the application stack than IaaS providers, adding operating systems, middle-ware (such as databases) and other run-times into the cloud environment. With SaaS, a provider offers an entire application stack that the user simply consumes. In this thesis, we focus mainly on PaaS, the cloud computing model in which a third-party provider delivers hardware and software tools, hosted by the provider on its own infrastructure and alleviates users from having to install in-house hardware and software to develop or run a new application. The biggest added value of PaaS is that developers are completely abstracted from the lower-level details of the environment, so they can fully focus on what there are really good at (rapid development and deployment)

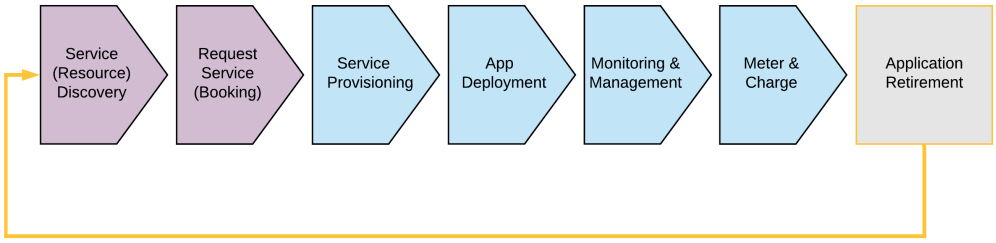


Figure 2.1: Cloud Application Life-Cycle in Federated Clouds

and not worry about things like scalability, security and more that are fully managed by PaaS.

2.4.1 Cloud Application Life-Cycle Management

The proposed collaborative SLA and RTM solutions focus on cloud federations. In such complex environments, the application's deployment is not a trivial task and needs careful resource selection, orchestration and performance monitoring. As it is shown in Figure 2.1, the complete life-cycle of cloud applications from the perspective of federated PaaS providers includes the following stages:

- **Service Discovery:** The cloud life-cycle starts with a user discovering the stack of the provided services in order to find the necessary services that satisfy his/her needs.
- **Request Service:** After the discovery phase, the user initiates the request for the corresponding services.
- **Service Provisioning:** Afterwards, the cloud provider allocates and assigns the resources to match user's usage demands.
- **Application Deployment:** Application is deployed by the user leveraging the necessary OS, applications and tools, which are provided.
- **Application Management:** Utilizing off-the-shelf tools, monitoring supervises the servers, the resources and the running software components. Another important management problem is the resource allocation of co-hosted cloud applications, which can be static or dynamic. In the latter case, this can be done by an automatic optimization tool or by a custom-made scheme.

- **Auditing and Billing:** Assessment tools report the resource usage (metering) and a periodic billing information is created.

- **Application Termination:** High utilization of resources is primary goal of cloud providers. This functionality enables the optimal resource reallocation after an application decommissioning, the pausing or the retirement of the application.

The SLA and RTM collaborative solution is involved in several of the above stages. Initially, both of them are required in service discovery. The SLA service advertises the offerings defined by the provider, while the RTM service provides a reputation value for every provider, enabling the proper selection of resources. Both RTM and SLA services are based on the same KPIs. During the application deployment, the SLA is activated and continuously evaluates the application performance (auditing stage). Also, the RTM service is involved in the auditing process, since it periodically prompts customers to submit their ratings about the application's status. Finally, at the final stage, the SLA terminates and the final rating is sent to RTM service.

2.4.2 Collaborative SLA and Reputation Architecture

In this subsection, we present the architecture of our collaborative SLA and RTM solution. Figure 2.2 demonstrates the high-level architecture of the SLA and RTM services in a cloud federation environment. The architecture is separated in two layers; namely the federation layer and the provider layer. Regarding the SLA service, the federation layer includes SLA Collector and Dashboard components, while the SLA Management Module lays in the provider layer.

SLA Dashboard offers a web-based graphical user interface (GUI), which enables customers to discover available SLA templates and providers to create the agreements. Furthermore, the SLA Dashboard allows providers and customers to check the status of the existing agreements.

SLA Collector acts as the intermediate communication point between the SLA Dashboard and the SLA Management module of each cloud provider. Through a REST API, this component supports every SLA process from creation to termination of a cloud application. Additionally, it provides a subscription mechanism, which allows other components

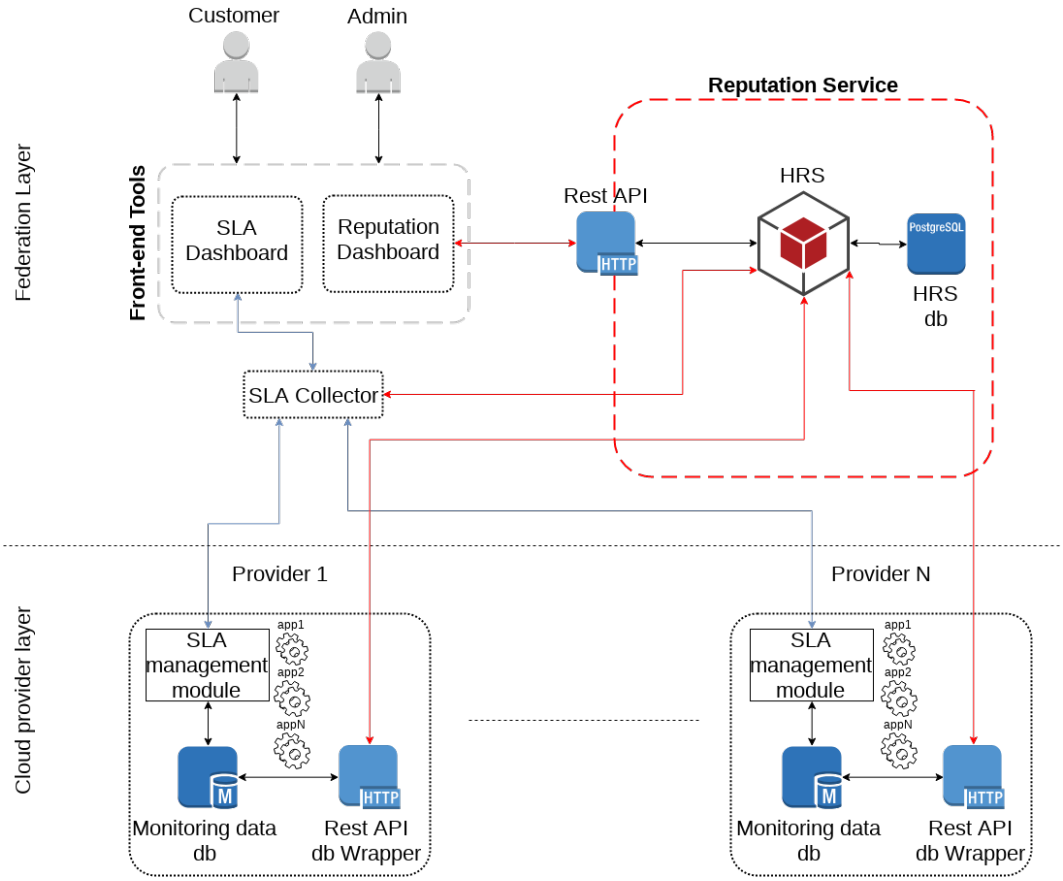


Figure 2.2: Architecture of SLA and RTM Service in cloud federation

to receive SLA events (e.g., SLA violations).

SLA Management Module, placed at the provider layer, is the core component for the SLA management process. It is responsible for storing all the SLA-related information and evaluating the active agreements, as described in 2.5.

As far as the RTM Service is concerned, the Reputation Dashboard offers a GUI to both customers and administrators. Through the Reputation Dashboard, the customers submit their rating and retrieve information about the reputation of each provider, while the administration of a cloud domain performs administrative tasks such as the definition of new KPIs for the computation of the reputation score. Through a REST API, the Reputation Dashboard communicates with the Hybrid Reputation System (HRS). HRS is the core component of the reputation service and is responsible for computing the reputation score of each provider. It retrieves SLA information and monitoring data through the SLA

collector and the monitoring data REST API respectively.

2.5 Service Level Agreement Management

This section presents the SLA Management concepts, components and functionalities that were used and leveraged in both our proposed approaches for MCDM-based Trust Management. Following the taxonomy of [29], we mainly focus on Access and Dependability aspects, which is the current practice for public cloud providers. Service (or Node) availability is defined as the degree of up-time for the service (or node) and expresses the Access perspective, while SLA penalties and violations of specific QoS metrics refer to Dependability notion. In this section, we describe the SLA life-cycle in a federated cloud environment were our proposed solutions focus.

2.5.1 SLA components

In this section we will present the components and functionality of the **SLA Management Module**, which is the core component for the SLA management process. It is responsible for storing all the SLA-related information and evaluating the active agreements. Figure 2.3 demonstrates the internal architecture of the SLA Management Module. Analytically, the most important components are:

- **Repository** is the database that stores any SLA entity, i.e., templates, agreements, violations and penalties.
- **REST service** is the REST interface of the SLA Management to external components. It provides CRUD (Create, Read, Update, Delete) operations to manage the SLA entities and change the state of an assessment.
- **Assessment** is the component responsible for the evaluation process of the SLA agreements. Utilizing monitoring data as input, it detects violations and generates penalties whenever a performance degradation occurs.
- **Monitoring adapter** makes queries to the monitoring database and parses the data for the the assessment of the active agreements.

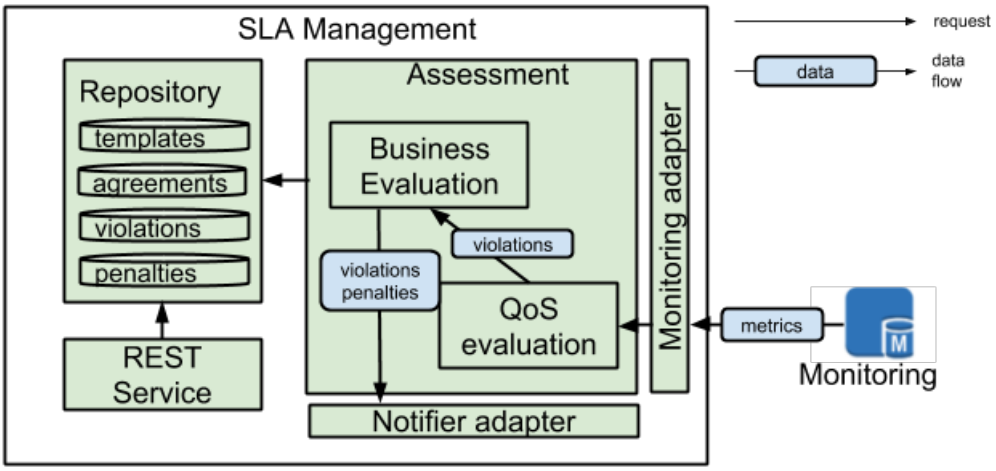


Figure 2.3: SLA Management Module

- **Notifier adapter** is a customized component in charge of communicating events (e.g. violations and penalties) to interested entities through the SLA collector.

2.5.2 Overview of the SLA Life-cycle

Figure 2.4 illustrates an overview of the SLA life-cycle, which includes several phases. Initially, at the **service publishing** phase, the cloud providers publish their services by creating offerings for specific performance indicators, e.g. "99.99% of the deployed application's availability" or "the application response time is always lower than T ms". Furthermore, the penalties are described here. This information is formally described in an SLA template, a document structured as a WS-Agreement template [37]. The SLA templates are stored in the Service Registry. At the **service discovery** phase, the customers are able to discover the available offerings advertised in the Service Registry in a centralized way through the SLA Dashboard. The discovery can be as simple as a keyword filtering, taking advantage of the extendable nature of the WS-Agreement standard to store the service keywords in the templates. Complex matchmaking mechanisms might be developed, but it is out of this thesis. Next, at the **service request** phase, the customer selects an offering and the corresponding agreement - based on the offering's SLA template - is created, containing

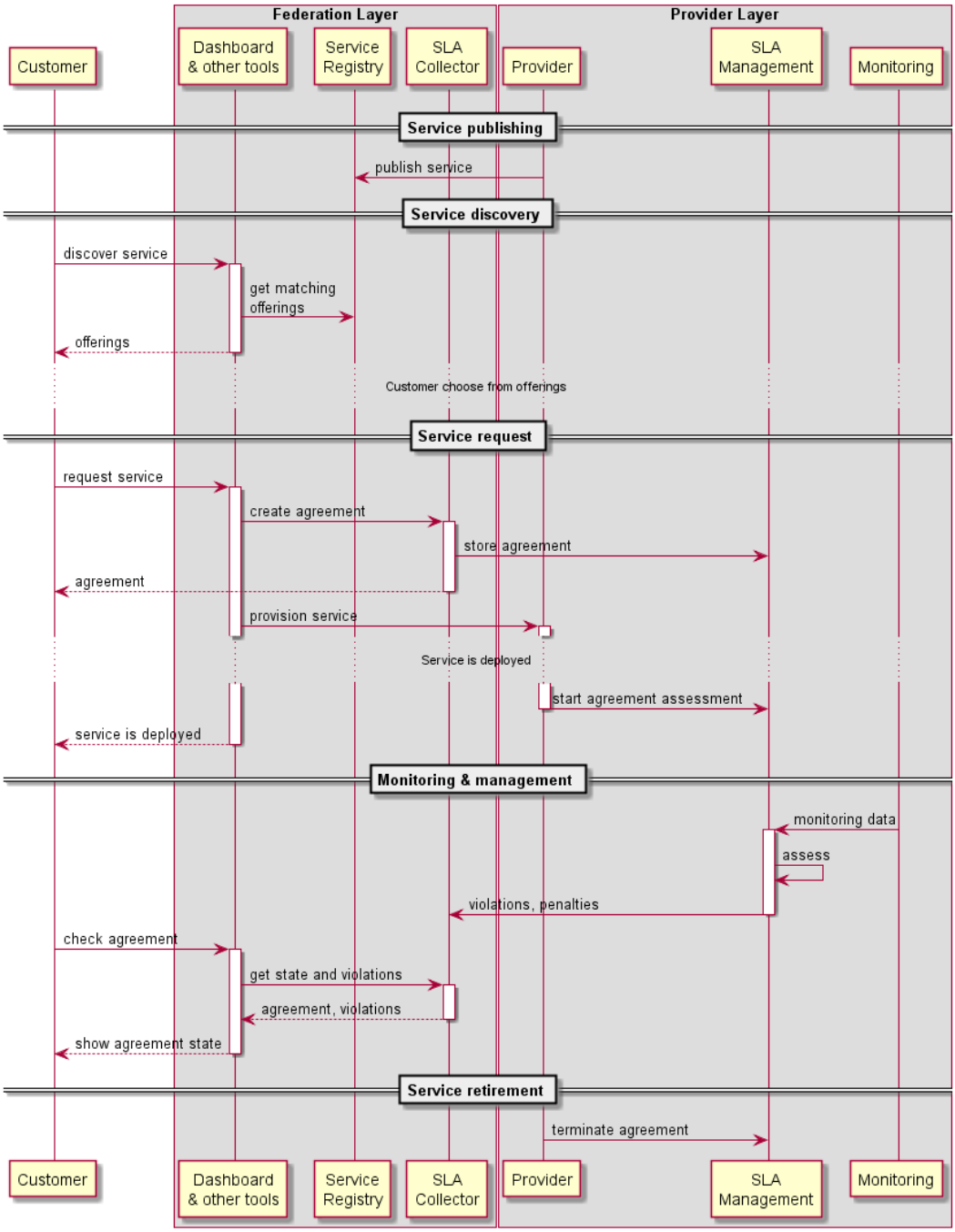


Figure 2.4: SLA Life-cycle sequence diagram

the information of the customer, the provider, the performance indicators and the expected penalties. Sequentially, the SLA Collector forwards the agreement to the SLA Management module of the selected provider in order to be internally stored and evaluated until the cloud

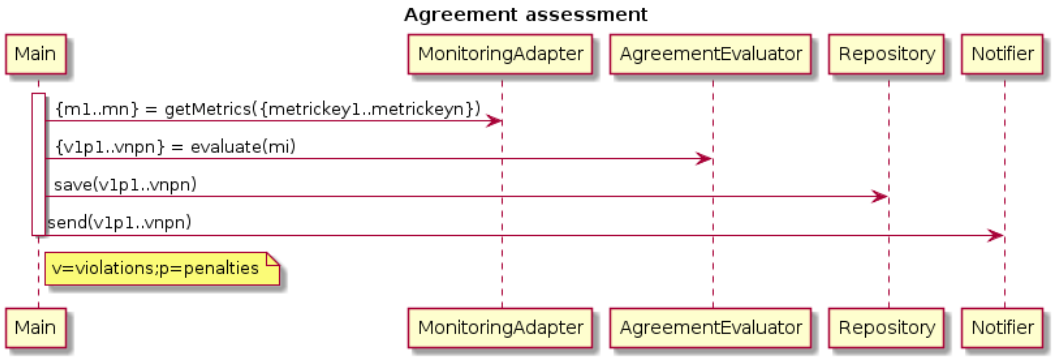


Figure 2.5: SLA assessment sequence diagram

application's termination. After the application instantiation, the assessment of the agreement starts and the customer is notified. At the **monitoring and management phase**, the essential monitoring values for the agreement assessment are provided by the provider's monitoring service. The detected violations are stored internally and forwarded to the SLA Collector component, which includes a notification/subscription service to communicate violations to any interested entity, i.e. customer, or service such as the reputation service. Finally, the agreement is terminated with the service retirement.

2.5.3 SLA Assessment

The importance of SLA assessment is twofold. First, it provides a clear view on the cloud application's performance, while the produced violations are used to compute the provider's reputation score. A high-level sequence diagram of the SLA assessment is shown on Figure 2.5. Its main process is executed periodically (e.g., every minute), getting in first place the necessary metric values to evaluate each guarantee term from the MonitoringAdapter, which is in charge of making the requests to the actual Monitoring component (not shown in the diagram) and transforming the data from the Monitoring domain to the SLA Management module domain, hence allowing the use of various monitoring tools in a transparent way. Then, the AgreementEvaluator checks if the retrieved values of each guarantee term satisfy the guarantee term constraint. In the case of an unsatisfied constraint, a violation is produced. Also, a penalty is produced only if it was defined by the agreement. The output of the AgreementEvaluator is the collection of generated violations and penalties, which are then stored in the internal Repository and sent to the Notifier, a plugin component intended

to push notifications of the agreements status to interested observers. Depending on the nature of the cloud application, there is flexibility on the assessment's configuration and execution. The evaluation of a guarantee term can be chosen to be performed periodically (e.g per hour, per day, etc) or at the termination of a service, lease, etc. Additionally, the evaluation can be done using single values (e.g. response time for a particular timestamp) or by aggregated values (e.g. average availability of the service).

2.6 Fuzzy VIKOR based Trust Management

This section presents our fisrt approach of MCDM-based trust management. We present a fuzzy reputation-based trust management system for federated clouds and a credibility mechanism for the users' ratings. The fuzzy VIKOR reputation frameworks has horizontal structure, which can easily scale up, and is actually multi-criteria decision technique. It can process simultaneously various types of data, e.g. binary, numeric and linguistic values. This allows us to use numeric QoS and SLA data combined with linguistic QoE data, that are appropriate to express the vague and subjective user preferences. Each cloud provides a set of services. The services of a cloud depend on the type of the available resources and refer to computing or network key performance indicators (KPIs), e.g., node/link/server availability or network delay, bandwidth and packet loss ratio. The proposed framework considers various QoS and QoE criteria from SMICloud [66]. In a federated environment, several users can use the same cloud with different goals. Thus in our approach, each user is able to adjust the weight of criteria according to his/her needs. The consistency of the weights is checked in order to provide meaningful rates and discourage malicious evaluators.

Our framework is based on fuzzy VIKOR (Visekriterijumsko Kompromisno Rangiranje), which is a multi-objective decision making approach. This technique is applicable for cases where the best provider must be selected among different alternatives. For instance, the fuzzy VIKOR approach is used for renewable energy planning [67]. The fuzzy VIKOR method handles only fuzzy inputs. Thus, we proposed a modification of fuzzy VIKOR that considers both numeric and linguistic values. Also the user can assign the weight of each criterion according to his/her requirements. The reputation system modifies the reputation score of each cloud after an certain usage period. Furthermore, a credibility mechanism

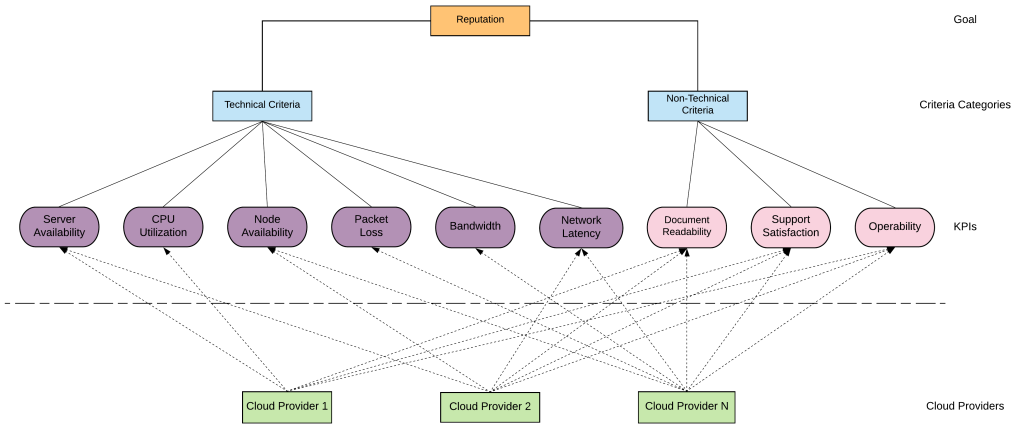


Figure 2.6: Modified Fuzzy VIKOR Reputation Model for Federated Clouds

compares the user's opinion with SLA and monitoring measurements in order to protect the reputation of cloud and cloud providers from malicious users. We choose this multi-objective methodology to investigate the effectiveness of using both numerical and fuzzy inputs on the reputation score of the federated clouds. The evaluation of our framework in Section 2.6.3 showcases the importance of using fuzzy criteria for the QoE metrics. Appendix A presents the basic information on fuzzy numbers and sets.

2.6.1 Fuzzy VIKOR

Fuzzy VIKOR is a multi-criteria decision making approach that simultaneously measures the closeness to the best and worst alternative. It can be applied to any scenario that a user has to select among alternative providers, such as cloud services [68] and renewable energy resources [67]. The original fuzzy VIKOR approach uses explicitly a group of fuzzy KPIs. For the computation of the reputation score of a cloud provider, we extend this approach in order to process also numeric KPIs, as shown in Figure 2.6, where the reputation value is directly derived from the KPIs. In Figure 2.6, the level of criteria categories does not contribute to the computation of the reputation value but it indicates only the different nature of the underlying KPIs. The purple technical KPIs refer to QoS metrics and they are numeric, while the pink non-technical KPIs correspond to fuzzy KPIs. These numeric inputs are converted to fuzzy numbers, as explained in the following subsection. For each pair of KPIs, an assigned fuzzy weight indicates the relative importance between them.

Table 2.1: Linguistic Terms and Membership Functions of Fuzzy Weights

Linguistic Term	Membership Function
Absolutely strong (AS)	$(2, 5/2, 3)$
Very strong (VS)	$(3/2, 2, 5/2)$
Fairly strong (FS)	$(1, 3/2, 2)$
Slightly strong (SS)	$(1, 1, 3/2)$
Equal (E)	$(1, 1, 1)$
Slightly weak (SW)	$(2/3, 1, 1)$
Fairly weak (FW)	$(1/2, 2/3, 1)$
Very weak (VW)	$(2/5, 1/2, 2/3)$
Absolutely weak (AW)	$(1/3, 2/5, 1/2)$

Table 2.2: Linguistic Terms and Membership Functions of Fuzzy KPIs

Linguistic Term	Membership Function
Extremely Poor (EP)	$(0.1, 1, 2)$
Very Poor (VP)	$(1, 2, 3)$
Poor (P)	$(2, 3, 4)$
Medium Poor (MP)	$(3, 4, 5)$
Fair (F)	$(4, 5, 6)$
Medium Good (MG)	$(5, 6, 7)$
Fair Good (FG)	$(6, 7, 8)$
Good (G)	$(7, 8, 9)$
Very Good (VG)	$(8, 9, 10)$
Excellent (E)	$(9, 10, 10)$

Table 2.1 presents the linguistic terms and the corresponding membership functions for the fuzzy weights, while Table 2.2 presents the information about the fuzzy numbers used for the KPIs' evaluation. The user's evaluation is compared with the perfect evaluation of a virtual user in order to acquire a quantitative measure of the closeness to the best cloud's performance. The following steps include all the necessary computations of the reputation score of a federated cloud provider.

Step 1 - Definition of the cloud KPIs: The cloud provider defines all the QoS and QoE KPIs that determine the performance of the cloud. Furthermore, the QoS KPIs are included in an SLA between the provider and the user.

Step 2 - Definition of relative importance weights: The user assigns the linguistic term for the relative importance weight of all possible KPIs pair from Table 2.1. Assuming a cloud with N KPIs, we formulate the fuzzy pairwise importance comparison matrix (PICM),

as follows:

$$PICM = \begin{array}{c} \\ K_1 \\ K_2 \\ \vdots \\ K_N \end{array} \begin{array}{c} \begin{array}{cccc} K_1 & K_2 & \dots & K_N \end{array} \\ \begin{bmatrix} 1 & FS & \cdots & VW \\ FW & 1 & \cdots & E \\ \vdots & \vdots & \ddots & \vdots \\ VS & E & \cdots & 1 \end{bmatrix} \end{array} \tag{2.1}$$

Then the PICM's elements are defuzzified using (A.0.6) of Appendix A. Since the weights are derived from the subjective preferences of individuals, the final computation of reputation can be based on inconsistent and conflicting KPIs. Thus, in order to avoid such inconsistencies, the Consistency Ratio (CR) [69] is calculated for each group of sibling attributes. The CR is the degree of the randomness in the weight assignment between several sibling attributes. CR values less than 0.1 are acceptable to continue to the next phase, otherwise the user must correct the assigned weights.

Step 3 - Computation of the KPIs weight vector: The fuzzy Analytical Hierarchical Process (AHP) is a methodology of determining the relative importance of the selection criteria. In our approach, the extended analysis on fuzzy AHP, as proposed by Chang [3], is adopted to determine the KPIs weight vector.

The following steps of extent analysis on fuzzy AHP are applied. Let the N-dimension fuzzy $PICM = [a_{ij}]\,,\ i, j = 1, \dots, N$, the fuzzy synthetic extent is defined by,

$$D_i = \left(D_i^l, D_i^m, D_i^u\right) = \sum_{j=1}^{N} a_{ij} \otimes \left(\sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} \right)^{-1} \tag{2.2}$$

We find the attribute with the higher fuzzy synthetic degree by computing the degree of

possibility for a fuzzy number to be greater than other one,

$$V\left(D_i \geq D_j\right) = hgt\left(D_i \cap D_j\right) =$$
$$= \begin{cases} 1 & \text{if } D_i^m \geq D_j^m \\ \frac{D_j^l - D_i^u}{\left(D_i^m - D_i^u\right) - \left(D_j^m - D_j^l\right)} & \text{if } D_i^m \leq D_j^m and D_j^l \leq D_i^u \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

The degree of possibility that a fuzzy synthetic extent D_i is greater than the rest synthetic fuzzy extents of the fuzzy PICM is,

$$d_i = V\left(D_i \geq D_k, \forall k = 1, \ldots, N, k \neq i\right) = \min V\left(D_i \geq D_j\right) \tag{2.4}$$

Finally the normalized weight vector of KPIs is obtained,

$$W = [w_1 \ldots w_N] \text{ where } w_i = \frac{d_i}{\sum_{k=1}^{N} d_k} \tag{2.5}$$

Step 4 - Evaluation of Cloud Performance: When the defined usage period of a cloud has expired, the users are prompted to submit their judgment on the performance of the used clouds. In order to mitigate the effect of malicious ratings, the user's credibility is considered. Thus, for the QoS KPIs, the user's opinion x is properly modified to $\tilde{x}$ by the credibility mechanism of the following subsection. For the fuzzy KPIs, the user evaluates the clouds using the linguistic values of Table 2.2. For the numeric KPIs, the user assigns crisp values, which are converted to fuzzy numbers using the membership functions of Table 2.2. Assume that the numeric evaluation $\tilde{x}$, modified by the credibility mechanism, corresponds to two adjacent linguistic values A and B and μ_A and μ_B are the respective membership functions. Then the modified linguistic value $\tilde{X}$ that corresponds to the numeric evaluation is obtained by $\tilde{X} = \mu_A A + \mu_B B$. We use a virtual user's rating with excellent linguistic values (E) for all KPIs in order to represent the best possible performance of the cloud.

Thus, the obtained fuzzy evaluation matrix (FEM) for the expired usage time is,

$$FEM = \begin{array}{c} \\ U \\ V \end{array} \begin{array}{c} \begin{array}{cccc} K_1 & K_2 & \cdots & K_N \end{array} \\ \begin{bmatrix} \tilde{x_1} & \tilde{x_2} & \cdots & \tilde{x_N} \\ E & E & \cdots & E \end{bmatrix} \end{array} \tag{2.6}$$

where the first row of FEM corresponds to the modified user's (U) opinion, while the second row correspond the perfect ratings of the virtual user V.

Step 5 - Computation and update of reputation: In this step, the modified fuzzy VIKOR method is actually applied. Let the weight vector W and the fuzzy evaluation matrix $FEM = [x_{ij}],\ i = 1, 2,\ j = 1, \ldots, N$, we determine the fuzzy best value $\tilde{f}_j^+$ and the fuzzy worst value $\tilde{f}_j^-$. Since the virtual user ratings are perfect the fuzzy best and worst values are,

$$\tilde{f}_j^+ = x_{2j},\ j = 1, \ldots, N$$
$$\tilde{f}_j^- = x_{1j},\ j = 1, \ldots, N$$

Then, the separation measure of x_{ij} from the fuzzy best and worst value are obtained by,

$$\tilde{S}_i = \frac{\sum_{j=1}^{N} w_j \left(\tilde{f}_j^+ - x_{ij}\right)}{\tilde{f}_{j.u}^+ - \tilde{f}_{j.l}^-} \tag{2.7}$$

$$\tilde{R}_i = \max_j \left[\frac{w_j \left(\tilde{f}_j^+ - x_{ij}\right)}{\tilde{f}_{j.u}^+ - \tilde{f}_{j.l}^-} \right] \tag{2.8}$$

Next, the best and worst values of $\tilde{S}_i,\ \tilde{R}_i$ are calculated,

$$\tilde{S}^+ = \min_i \tilde{S}_i,\ \tilde{S}^- = \max_i \tilde{S}_i \tag{2.9}$$

$$\tilde{R}^+ = \min_i \tilde{R}_i,\ \tilde{R}^- = \max_i \tilde{R}_i \tag{2.10}$$

Then, the evaluation index $\tilde{Q}_i$ contains the fuzzy reputation score of the cloud user and the

virtual user and is computed as,

$$\tilde{Q}_i = \alpha \frac{\tilde{S}_i - \tilde{S}^+}{\tilde{S}^- - \tilde{S}^+} + (1-\alpha)\frac{\tilde{R}_i - \tilde{R}^+}{\tilde{R}^- - \tilde{R}^+} \tag{2.11}$$

where α is a index of our willingness to penalize the poor cloud performance or reward the good one. In order to have a balance between good and poor behavior. We set $\alpha = 0.4$, because the virtual user has always excellent ratings. We defuzzify the elements of $\tilde{Q}_i$, using (A.0.6) to get the crisp reputation value of the experiment Q_i. The element Q_i with the minimum value has the best reputation score. Thus, in our case the virtual user has the best score that is always zero ($Q_2 = 0$). For a single expired usage period, the reputation score R_{exp} is defined as,

$$R^T_{exp} = (1 - Q_1)\, 100\% \tag{2.12}$$

After the n user evaluations, the overall reputation value of the cloud is updated as,

$$R^T_n = \frac{(n-1)R^T_{n-1} + R^T_{exp}}{n} \tag{2.13}$$

2.6.2 User's Credibility

The modified fuzzy VIKOR considers the credibility of the user for computing the reputation score in order to prevent malicious users from giving misleading evaluations. The QoE KPIs are excluded from our credibility mechanism, since they are subjective opinions of the user and cannot be compared with any real measurement. On the contrary, the QoS KPIs can be compared with objective SLA and monitoring data, which consist the ground truth of every cloud's performance. Some reputation mechanisms, i.e., FTUE [33], define different categories of users based on the comparison between their past evaluations and monitoring data and their credibility varies accordingly. In our case, we do not assume any category of user's behavior. Our proposed mechanism leverages SLA and monitoring data to infer and update the credibility of users in the federated environment.

Algorithm 1, presented below, shows how the user's credibility is calculated for a cloud used in a certain usage period. If multiple clouds were used, the credibility value is sequentially calculated for every cloud. The inputs of the credibility algorithm are the user opinion vector $UO = [UO_i]^\top,\ i = 1, \ldots, k$ containing the evaluations of the k QoS KPIs

Algorithm 1 User Credibility Mechanism

Inputs: UO, SD, MD
Outputs: CR, $\widetilde{UO}$
for $\forall UO_i \in UO$ **do**
  $CASE1 \equiv (SD_i \geq UO_i) \ \wedge \ (SD_i \geq MD_i)$
  $CASE2 \equiv (SD_i \leq UO_i) \ \wedge \ (SD_i \leq MD_i)$
  $CASE3 \equiv (SD_i > UO_i) \ \wedge \ (SD_i \leq MD_i)$
  $CASE4 \equiv (SD_i \leq UO_i) \ \wedge \ (SD_i > MD_i)$
  $e_M = \frac{|MD_i - SD_i|}{SD_i}$, $i = 1, \ldots, k$, Monitoring Relative Error
  $e_O = \frac{|UO_i - SD_i|}{SD_i}$, $i = 1, \ldots, k$, Opinion Relative Error
  $e_D = \frac{|UO_i - MD_i|}{SD_i}$, $i = 1, \ldots, k$, Relative Distance
  $C = [c_i]^\top$, $i = 1, \ldots, k$, Correction Vector
  if $CASE1 \vee CASE2$ **then**
    $c_i = 1 - e_O$
  else if $CASE3 \vee CASE4$ **then**
    $c_i = 1 - (e_{Mi} + e_{Oi})$
  end if
  $c_i = max(c_i, 0)$
end for
$\hat{c} = avg(c_i)$
$CR_n = \frac{(n-1)CR_{n-1} + \hat{c}}{n}$
for $\forall UO_i \in UO$ **do**
  $\widetilde{UO}_i = UO_i$
  if $CASE1 \vee CASE2$ **then**
    if $UO_i \leq MD_i$ **then**
      $\widetilde{UO}_i = MD_i + e_{Oi} CR_n$
    else if $UO_i < MD_i$ **then**
      $\tilde{UO}_i = MD_i - e_{Oi} CR_n$
    end if
  end if
  if CASE3 **then**
    $\widetilde{UO}_i = MD_i - min(e_{Mi}, e_{Oi}) CR_n$
  end if
  if CASE4 **then**
    $\widetilde{UO}_i = MD_i + min(e_{Mi}, e_{Oi}) CR_n$
  end if
end for

of the cloud, the SLA data vector $SD = [SD_i]^\top$, $i = 1, \ldots, k$ and the monitoring data vector $MD = [MD_i]^\top$, $i = 1, \ldots, k$, which contain the respective SLA and monitoring

values for these KPIs. The output of the algorithm are the updated user's credibility CR and the modified user opinion vector $\widetilde{UO} = [\widetilde{UO}_i]^\top,\ i = 1, \ldots, k$ (lines 1-2). For all KPIs of an involved cloud, four possible cases are identified (lines 4-7). In the first case, named $CASE1$, the user's opinion and the monitoring value for a specific KPI are smaller than the SLA value while in the second case, CASE2, both user's opinion and monitoring data satisfy the SLA. In CASE3 and CASE4, there is a significant deviation between the user's opinion and the monitoring data in respect with the predetermined SLA value. These cases correspond to suspicious ratings. More specifically, in CASE3 the user's opinion is lower than the SLA value while the monitoring data show that the SLA is satisfied. In CASE4 the monitoring data are lower than the SLA value while the user's rating is higher than the SLA value. Then, the relative errors of the monitoring and opinion values and the relative distance between user's opinion and the actual monitoring value regarding to the SLA are defined (lines 8-9). The elements of the correction vector, $C = [c_i]^\top,\ i = 1, \ldots, k$, are actually the credibility value of each KPI. The values of elements of C depend on which of the above cases is satisfied. The user's credibility for the evaluation conducted at the moment is calculated as the average value of the correction vector. Then, the overall user's credibility is updated (lines 11-20).Furthermore the user' opinion is updated according to previous defined cases (lines 21- 35). The modified opinions $\widetilde{UO}$ are used in Step 4 of the fuzzy VIKOR.

2.6.3 Evaluation

The operation of the proposed reputation algorithm has been tested and evaluated in a real Future Internet Federation of testbeds in the context of HELNET project. HELNET project [70] provides a federation of heterogeneous testbeds aiming to facilitate and promote test driven research for the future internet. The federation offers experimentation services in the fields of 4G/3G communications, WiFi networking, Software Defined Networking and Software Defined Radios. For the evaluation of the proposed reputation-based trust framework introduced here, the wireless NETMODE [71] and NITOS [72] testbeds are used in particular for demonstration purposes acting as different cloud providers.

Initially, we demonstrate the evaluation of the proposed reputation algorithm for the two federated wireless testbeds with four KPIs. Three QoE KPIs are also utilized, i.e., Document

Readability (K1), Support Satisfaction (K2) and Operability (K3) focus on non-technical aspects, while Node Availability (K4) is the QoS KPI that measures the average availability of all reserved wireless nodes during an experiment. The user submits his rating on the reputation service and in the following we show the step-by-step computation of the cloud's performance reputation score, using the modified fuzzy VIKOR, and the credibility value using Algorithm 1. In the second use case, one hundred users are assumed to have used the NETMODE testbed. This data set contains a mix of random, honest and malicious ratings. This scenario demonstrates the key role of credibility mechanism and the effect of α parameter in Step 5. Finally, our proposed solution is compared against an existing reputation based trust framework, named FTUE [33], that is designed for federated facilities of FED4FIRE. As mentioned before FTUE uses only crisp QoS and QoE KPIs to infer the reputation of a testbed and the experimenter's credibility. In the following subsections, the higher score translates to better reputation for both reputation systems. Consequently, since the the corresponding reputation values are obtained as combination of the perceived users' experience and the usage of technical KPIs, we argue that high reputation score is a strong indication that the testbed, as a cloud provider, is better as a whole.

2.6.4 Fuzzy VIKOR Evaluation

In order to evaluate the fuzzy reputation system, the ratings of K1-K3 are obtained by linguistic terms of Table 2.2, which are mapped onto triangular fuzzy numbers (A.0.1). The ratings of K4 is numeric and converted to fuzzy number according to Step 4 of modified fuzzy VIKOR methodology. In the following paragraphs, we demonstrate the computations of each step of Fuzzy VIKOR methodology for NETMODE testbed. The computations for NITOS testbed are similar and they are omitted. According to Step 2, the experimenter assigns the pairwise importance for each KPI with respect to the others, so we obtain the

PICM,

$$PICM = \begin{array}{c} \\ K_1 \\ K_2 \\ K_3 \\ K_4 \end{array} \begin{array}{c} \begin{array}{cccc} K_1 & K_2 & K_3 & K_4 \end{array} \\ \begin{bmatrix} 1 & SW & SW & VW \\ SS & 1 & SW & FW \\ SS & SS & 1 & FW \\ VS & FS & FS & 1 \end{bmatrix} \end{array}$$

$$= \begin{bmatrix} (1,1,1) & (\frac{2}{3},1,1) & (\frac{2}{3},1,1) & (\frac{2}{5},\frac{1}{2},\frac{2}{3}) \\ (1,1,\frac{3}{2}) & (1,1,1) & (\frac{2}{3},1,1) & (\frac{1}{2},\frac{2}{3},1) \\ (1,1,\frac{3}{2}) & (1,1,\frac{3}{2}) & (1,1,1) & (\frac{1}{2},\frac{2}{3},1) \\ (\frac{3}{2},2,\frac{5}{2}) & (1,\frac{3}{2},2) & (1,\frac{3}{2},2) & (1,1,1) \end{bmatrix}$$

The Consistency Ratio of the defuzzified PICM, $CR = 0.018$, is acceptable. As described in Step 3 of section 2.6.1, using the fuzzy extended analysis, we obtain the weight vector for the KPIs,

$$W_j = \begin{bmatrix} 0.109 & 0.199 & 0.233 & 0.46 \end{bmatrix}, \quad j = 1, \ldots, 4$$

Then, the experimenter evaluates the fuzzy KPIs by using the linguistic variables in Table 2.2. For the numeric KPI, assume that the credibility mechanism modifies the user's opinion to 0.90, the SLA value was set at 0.80, and the monitoring data for this KPI is 0.85. The triangular membership function of the modified linguistic value $\widetilde{UO}$ is $\mu_{\widetilde{X}} = (7.9, 8.9, 9.9)$. According to the proposed method, we compute the testbed's reputation score for this experiment. The first row of FEM contains the experimenter's ratings for the KPIs, while the ideal ratings of the virtual user are in the second row,

$$FEM = \begin{array}{c} \\ E \\ V \end{array} \begin{array}{c} \begin{array}{cccc} K_1 & K_2 & K_3 & K_4 \end{array} \\ \begin{bmatrix} E & E & E & \widetilde{UO} \\ E & E & E & E \end{bmatrix} \end{array}$$

The fuzzy best value and fuzzy worst value are determined and the separation measures are calculated according to (2.7)-(2.10). The evaluation indexes Q_i are calculated by (2.10); $Q_1 = 0.179$ and $Q_2 = 0$. Finally the reputation score for this experiment is computed by

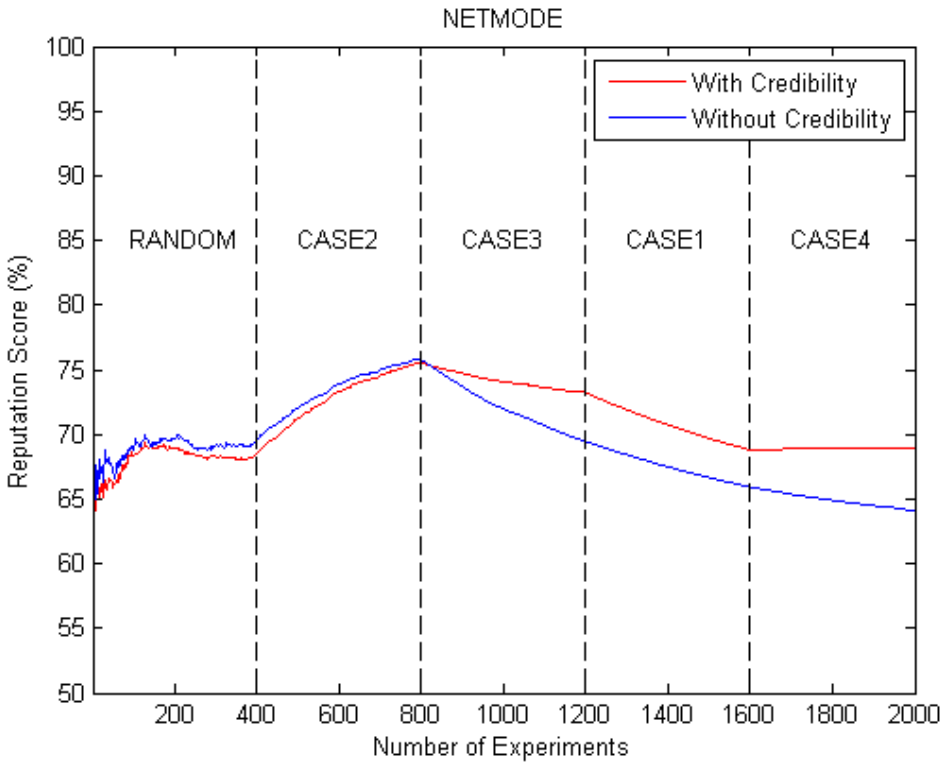


Figure 2.7: Credibility mechanism effect in fuzzy VIKOR reputation system

(2.12),

$$R_{exp}^{T} = (1 - Q_1)\,100\% = 82.1\%$$

The following example illustrates the performance of the reputation system and the credibility mechanism in the case of a suspicious evaluation. Assuming that $SD = 0.8$, $MD = 0.9$, $\widetilde{UO} = 0.6$, the credibility mechanism modifies the user's opinion to $\widetilde{UO} = 0.8055$ with $\mu_{\widetilde{UO}} = (7.05, 8.05, 9.05)$ and the user's credibility decreases from 0.8 to 0.756. The final reputation value is computed to 73.9%. This case shows that the user is possibly malicious and his credibility is reduced while the testbed's reputation score is not significantly affected. The credibility mechanism is important to alleviate the effect of misleading ratings. In the above example, if there was no credibility mechanism, the reputation value would be 66.7%. Finally, it is remarkable that the reputation system based on modified fuzzy VIKOR is scalable considering the number of the KPIs, because the reputation score is computed by simple fuzzy mathematical formulas.

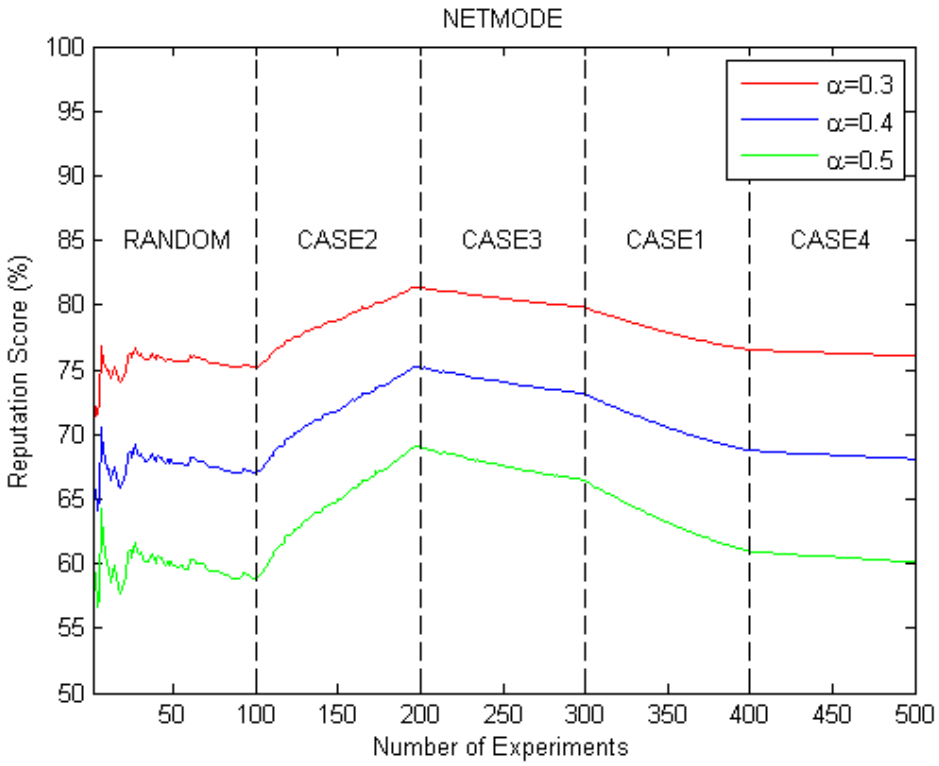
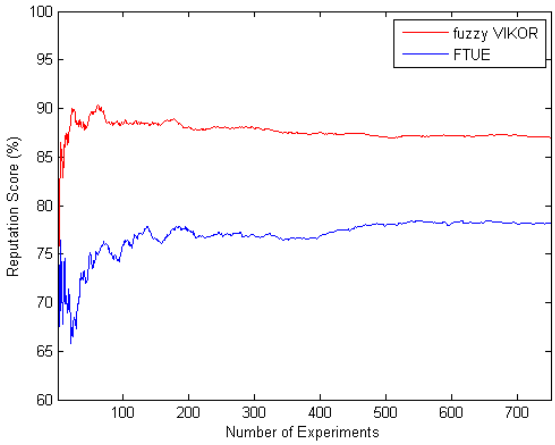


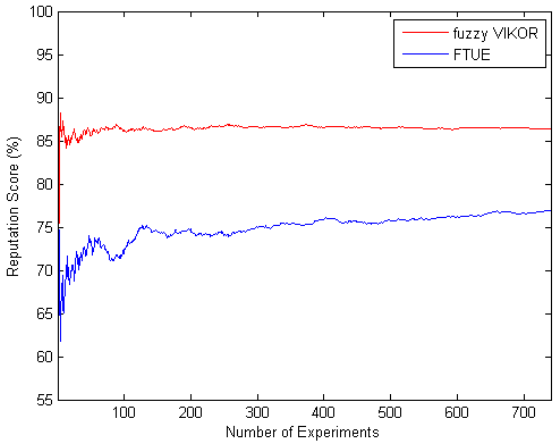
Figure 2.8: The effect of α parameter on the modified fuzzy VIKOR method

In the second scenario, a dataset of two thousand usage periods and evaluations, considers any possible case of the user's behavior to highlight the effect of credibility mechanism and the resilience of the proposed reputation system against malicious users and their ratings. The initial reputation score of the testbed is set to 0.5. In Figure 2.7, it is shown which case, CASE1 - CASE4 of Algorithm 1, is valid for each experiment of the dataset. The first part of the dataset corresponds to experiments where the ratings are random with respect to SLA and monitoring values.

As shown in Figure 2.7, at the first part, named RANDOM, the small fluctuations of the reputation score are due to the random difference between monitoring value and the user's opinion. The implementation of the credibility mechanism improves 1% the reputation score. Then, the reputation score increases rapidly, almost 6%, because the users are honest and their opinion agree with the monitoring data, as in CASE2 of Algorithm 1. In CASE3, the decrease of the reputation score is not steep, only 2%, because contrary to the user's opinion, the monitoring value is higher that the SLA. This case reflects the behavior of malicious



(a) NETMODE Testbed



(b) NITOS Testbed

Figure 2.9: Comparison of fuzzy reputation system with FTUE framework

users, who try to damage the reputation of a testbed. In CASE1, the users are rightly unsatisfied with the testbed's performance, thus its reputation score decreases 5%. The last part of Figure 2.7 corresponds to CASE4 and some biased users try to unfairly enhance the testbed's reputation. However, the increase of reputation score is negligible due to the fact that monitoring value violates the SLA value. Also, Figure 2.7 indicates that the credibility mechanism plays a key role in the robustness of the reputation system. More specifically, CASE3 and CASE4 illustrate that the reputation system without credibility mechanism is not adequate against the malicious users. In CASE3 and applying the credibility mechanism, the reputation score is 4% higher than without this mechanism. Similarly, in CASE4 and

using Algorithm 1, the output of the reputation system is 5% higher than the opposite case.

The most important parameter of the modified fuzzy VIKOR method is parameter α of Step 5. Large values of α mean that we penalize the poor testbed performance, while small values mean that we are lenient with the worse solution. It should be noted here that we compare our testbed performance with the ideal rating of a virtual user, which means that that the experimenter's evaluation is always worse than the virtual one. Figure 2.8 demonstrates the performance of the fuzzy reputation system for three different values of α parameter. Generally, the smaller values of α produce higher reputation scores. Thus, for $\alpha = 0.5$ the produced reputation score is too small and do not encourage users to select a testbed even if it has actually good performance. On the contrary, in the case of $\alpha = 0.3$, the value are over-optimistic and cannot depict the real performance of the testbed. Furthermore, the reputation system ignores bad evaluations and is more stiff. Thus, the selected $\alpha = 0.4$ is a good trade-off that offers realistic reputation score and enhance the sensitivity of the reputation system.

2.6.5 Comparison with FTUE framework

As mentioned earlier, FTUE [33] is a reputation-based trust framework for federated testbeds that uses numerical QoS and QoE KPIs and also provides a credibility mechanism. FTUE framework assumed four types of users with respect to the difference between the monitoring data and the user opinion, and the user's credibility is updated accordingly. The reputation score per service per testbed is the aggregation of user opinions, weighted by the credibility and the confidence of the user for his evaluation. FTUE framework does not take into account any SLA information.

We compare our proposed reputation system with FTUE framework following the experimental settings of [33]. Eighty experimenters are truthful and twenty experimenters are malicious in disguise, who reserve several testbeds and give biased evaluations only for one specific testbed. The nodes of one or both wireless testbeds are reserved by the users to conduct ten experiments and submit the respective ratings.

Figures 2.9a-2.9b demonstrate the reputation score of the two approaches for NETMODE and NITOS testbed respectively. For both testbeds, the reputation score computed by FTUE framework is 10% lower than the fuzzy VIKOR approach. Three major remarks can

be discussed on this result. First, the numerical evaluation of FTUE framework is based on five-star scale for evaluation, that provides coarse rating comparing with the fine-grain numerical and fuzzy values of the proposed approach. Secondly, FTUE credibility value depends heavily on the Mu parameter, which is testbed specific. For the results of Figure 2.9, we follow the experimental set-up of set [33] and we set $Mu = 0.75$. For this value, we produce the best reputation score for both testbeds. Finally, the credibility mechanism of the compared reputation systems have two important differences that play a key role in their performance. First, our proposed mechanism leverages SLA data to quantify the user's requirements and check if they are actually satisfied by comparison with the monitoring data. FTUE did not exploit any SLA data. Secondly, FTUE's credibility mechanism is based on four specific types of experimenter's behavior, which are quantified by the difference between user's opinion and monitoring values. On the contrary, the proposed reputation system does not assume any specific categorization of user's behavior like FTUE framework. In lines 4-10 of Algorithm 1, four cases, CASE1-CASE4, and the necessary relative errors are defined in order to measure exactly the deviation of the user's opinion from the SLA and monitoring value without assuming any specific behavior and covering any possible scenario.

2.7 Modified Fuzzy AHP based Trust Management

In this section, our second approach for MCDM-based trust management is presented. The proposed framework is actually a scalable MCDM system with hierarchical structure. In order to respond to the conditions of a federated cloud environment, this hybrid framework is able to process various types of data, which correspond to technical QoS and non-technical QoE KPIs of each provider. The technical KPIs refer to objective performance metrics, e.g, network latency and CPU utilization, while the non-technical KPIs correspond to subjective user's experience metrics such as Support Satisfaction and Usability. In a nutshell, this framework enables the use of numeric, binary and linguistic values, giving the customer the opportunity to express his or her subjective opinion in the best possible and effective way. Each customer has unique needs and different criteria about service selection and overall performance. To accommodate this, the users through the proposed HRS can evaluate both cloud infrastructure and application performance considering different criteria, preferences

and priorities. For this reason, in our approach, the customer himself/herself is capable of assigning different weights to the criteria according to his/her personalized service deployment goal.

The proposed reputation system is based on the principles of Fuzzy Analytic Hierarchical Process (FAHP) [3]. FAHP is widely used in various cases, such as product design and operational research [73]. FAHP is a ranking method based on numeric QoS and fuzzy QoE KPIs, such as node availability and support satisfaction respectively. However, in order to compute the reputation score of cloud applications, several modifications and extensions are required. There are three key differences with respect to FAHP between our HRS and the provider selection use cases, such as in [1]. First, our approach allows the customers to assign their weights on the criteria according to their application's performance criteria. Secondly, we compare the application's evaluation with an ideal rating of a virtual user, which contains the best values of all criteria. Finally, the computation of the reputation score takes into account the customer's credibility, as it is described in Section 2.7.2, in order to ensure the fair judgment of each cloud provider. In the following, the phases of the proposed HRS mechanism are described in detail.

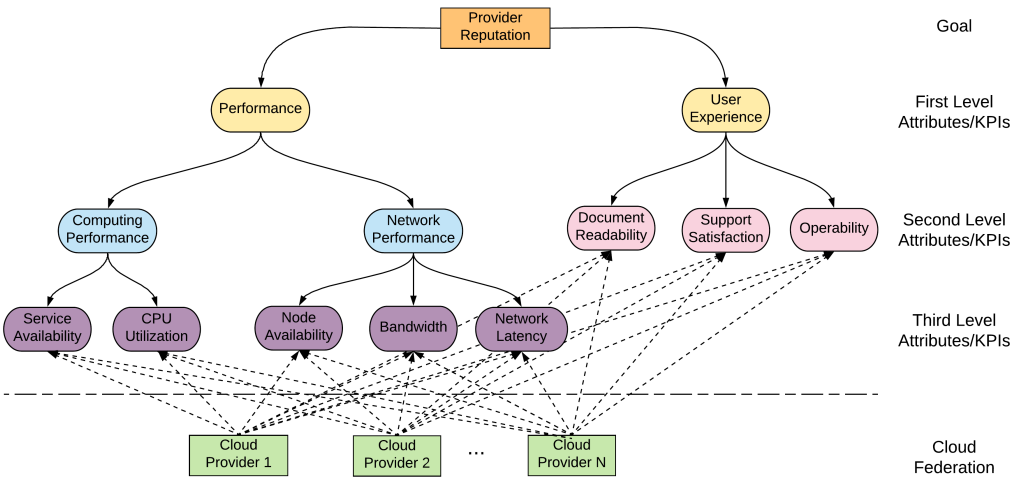


Figure 2.10: HRS Model for Federated Clouds

Table 2.3: Linguistic Terms and Membership Functions of Fuzzy Numbers

Linguistic Term	Fuzzy Numbers
Very Poor (VP)	$(1, 2, 3)$
Poor (P)	$(3, 4, 5)$
Medium (M)	$(4, 5, 6)$
Good (G)	$(5, 6, 7)$
Very Good (VG)	$(6, 7, 8)$
Excellent (E)	$(7, 8, 9)$

2.7.1 Modified Fuzzy Analytical Hierarchical Process

Phase 1 - Selection of service KPIs

The cloud provider determines the technical (QoS) and the user experience (QoE) KPIs and attributes that are used in the computation of the reputation score of the provided application. Figure 2.10 shows an example of KPIs and attributes in a hierarchical structure and highlights which of them are provided by the cloud providers. The difference between KPIs and attributes is that a KPI measures a specific technical or experience metric, while an attribute summarizes several KPIs of relevant metrics. At any level, the attributes can be further decomposed into the sibling attributes or KPIs of the lower level, while the KPIs cannot be decomposed further. Adopting SMICloud approach [2], numerical KPIs and attributes are represented by numeric, boolean, unordered sets and range values. On the contrary, the QoE KPIs are represented by fuzzy numbers. In Figure 2.10, pink (right) and purple (left) colored KPIs refer to fuzzy and numerical attributes respectively. In this study, we utilize triangular fuzzy numbers of the form $A = \{l, m, u\}$ and the membership function is defined as $\mu_A(x)$ (A.0.1). Furthermore, the arithmetic operations on fuzzy numbers are defined in Appendix A.

Table 2.3 contains the linguistic terms and the membership functions of the fuzzy numbers used for QoE attributes.

Phase 2 - Computation of relative attribute importance

Periodically or after the termination of a cloud application, the customer submits his rating for the QoS and QoE KPIs of the selected providers. The ratings of QoS KPIs are modified by the credibility mechanism, as it is analyzed later. The modified customer's ratings are compared against the ideal rating of a virtual user. The ideal rating is used to measure

Table 2.4: Relative ranking model for the four types of numerical values [2]

Numeric KPI: $A_i/A_j = \begin{cases} v_i/v_j & \text{if higher is better} \\ v_j/v_i & \text{if lower is better} \\ w_q & \text{if } \nexists v_i \\ 1/w_q & \text{if } \nexists v_j \end{cases}$	Boolean KPI: $A_i/A_j = \begin{cases} 1 & \text{if } v_i = v_j \\ w_q & \text{if } v_i = 1 \wedge \\ & \quad \wedge v_j = 0 \\ 1/w_q & \text{if } v_i = 0 \wedge \\ & \quad \wedge v_j = 2 \end{cases}$
Unordered Set KPI: For essential attributes $A_i/A_j = \frac{size(v_i)}{size(v_j)}$ For non-essential attributes $A_i/A_j = \begin{cases} \frac{len(v_i \cap v_r)}{len(v_i \cap v_r)} & \text{if } v_i \cap v_r \neq \emptyset \wedge \\ & \quad \wedge v_j \cap v_r \neq \emptyset \\ 1 & \text{if } v_i \cap v_r \equiv \emptyset \wedge \\ & \quad \wedge v_j \cap v_r \equiv \emptyset \\ w_q & \text{if } v_i \cap v_r \neq \emptyset \wedge \\ & \quad \wedge v_j \cap v_r \equiv \emptyset \\ 1/w_q & \text{if } v_i \cap v_r \equiv \emptyset \wedge \\ & \quad \wedge v_j \cap v_r \neq \emptyset \end{cases}$	Range KPI: For essential attributes $A_i/A_j = \frac{len(v_i \cap v_r)}{len(v_i \cap v_r)}$ For non-essential attributes $A_i/A_j = \begin{cases} \frac{len(v_i \cap v_r)}{len(v_i \cap v_r)} & \text{if } v_i \cap v_r \neq \emptyset \wedge \\ & \quad \wedge v_j \cap v_r \neq \emptyset \\ 1 & \text{if } v_i \cap v_r \equiv \emptyset \wedge \\ & \quad \wedge v_j \cap v_r \equiv \emptyset \\ w_q & \text{if } v_i \cap v_r \neq \emptyset \wedge \\ & \quad \wedge v_j \cap v_r \equiv \emptyset \\ 1/w_q & \text{if } v_i \cap v_r \equiv \emptyset \wedge \\ & \quad \wedge v_j \cap v_r \neq \emptyset \end{cases}$

the distance between the actual performance of a cloud application and its perfect performance according to the customer's preferences. This is achieved by computing the Relative Attribute Comparison Matrix (RACM) for each KPI of the hierarchical model. Given the ideal rating A_v and the modified customer's rating $\tilde{A_u}$ for the X KPI, $RACM_X$ is defined as follows,

$$RACM_X = \begin{bmatrix} 1 & \tilde{A_u}/A_v \\ A_v/\tilde{A_u} & 1 \end{bmatrix} \tag{2.14}$$

In the case of fuzzy KPIs, the division of $RACM_X$ elements corresponds to the fuzzy division of (A.0.5). For numerical KPIs, the division follows the cases of Table 2.4, leveraging the values of the corresponding KPIs v_i, v_j and the size of the set v_r.

Phase 3 - Computation and update of reputation score

In the case of numerical KPIs and attributes, the extended AHP approach is applied as described in SMICloud [2]. For the fuzzy KPIs, the extended analysis on FAHP is adopted according to Chang's approach [3]. The combination of these methodologies uses the RACM

of each KPI and attribute at any level of the hierarchical model in order to calculate the score vector of all intermediate attributes and the top level reputation attribute. For the fuzzy RACMs, the following steps of extent analysis on FAHP [3] are applied. Let the N-dimension fuzzy $RACM_A = [a_{ij}]\,,\ i,j = 1,\ldots,N$, the fuzzy synthetic extent of each row i of $RACM$ is defined by,

$$D_i = (D_{il}, D_{im}, D_{iu}) = \sum_{j=1}^{N} a_{ij} \otimes \left(\sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} \right)^{-1} \tag{2.15}$$

where the first term of the fuzzy multiplication is the sum of the elements of the the row and the second terms is the fuzzy inverse of the sum of the RACM's elements. The fuzzy multiplication is defined by (A.0.4), while the fuzzy inverse is defined using the fuzzy division of (A.0.5). We find the attribute with the higher fuzzy synthetic degree by computing the degree of possibility for a fuzzy number to be greater than another one,

$$\begin{aligned} &V\left(D_i \geq D_j\right) = hgt\left(D_i \cap D_j\right) = \mu_{D_i}(d) \\ &= \begin{cases} 1 & \text{if } D_{im} \geq D_{jm} \\ \frac{D_{jl} - D_{iu}}{(D_{im} - D_{iu}) - (D_{jm} - D_{jl})} & \text{if } D_{im} \leq D_{jm} \text{ and } D_{jl} \leq D_{iu} \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{2.16}$$

The degree of possibility is a comparison method between two convex fuzzy numbers. It is defined by the ordinate d of the highest intersection point D, as it is shown in Figure 2.11. The degree of possibility that a fuzzy synthetic extent D_i is greater than the rest synthetic fuzzy extents of the fuzzy RACM is,

$$d_i = V\left(D_i \geq D_k, \forall k = 1,\ldots,N, k \neq i\right) = \min V\left(D_i \geq D_j\right) \tag{2.17}$$

Finally the normalized comparison vector is obtained,

$$c = [c_1 \ldots c_N]^\top \text{ where } c_i = \frac{d_i}{\sum_{k=1}^{N} d_k} \tag{2.18}$$

At any level of the cloud provider's hierarchical model, we calculate the comparison vector for each attribute with the following bottom-up procedure. Given the weights of

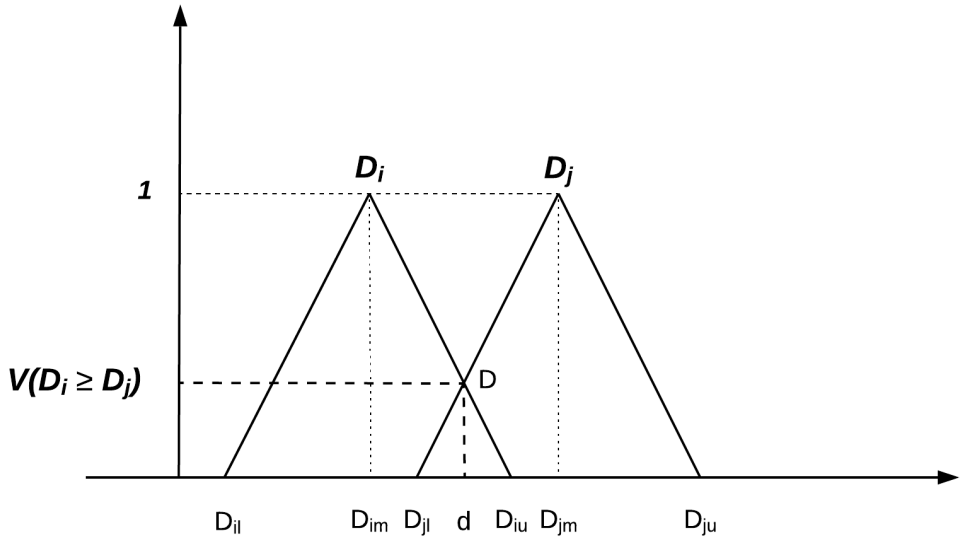


Figure 2.11: Graphical Presentation of the degree of possibility.

Phase 2, the ratings of the customer and the ideal rating, we start from the level where KPIs exist, and compute the comparison vector of the parent attribute by the comparison vectors of the sibling KPIs or attributes. Assuming a parent attribute with M sub-attributes and the weight vector with M elements, the comparison vector of the parent attribute is defined,

$$c_{par} = \begin{bmatrix} c_{sub1}^{\tilde{u}} & \cdots & c_{subM}^{\tilde{u}} \\ c_{sub1}^{v} & \cdots & c_{subM}^{v} \end{bmatrix} \begin{bmatrix} w_{sub1} \\ \vdots \\ w_{subM} \end{bmatrix} = \begin{bmatrix} c_{par}^{\tilde{u}} \\ c_{par}^{v} \end{bmatrix} \tag{2.19}$$

Reaching the top level of the hierarchical model, the normalized comparison vector for the provider's Reputation attribute is computed, $c_{rep} = \left[c_{rep}^{\tilde{u}} c_{rep}^{v}\right]^{\top}$. The first element of this vector refers to the service evaluation, while the second corresponds to the best possible rating of the virtual user. The difference between the two elements indicates the distance between the actual performance as interpreted by the customer, and the perfect performance of the cloud service. Thus, for the n^{th} submitted rating, the cloud provider's reputation score

is computed by,

$$R_{exp}^{T} = \frac{c_{rep}^{\tilde{u}}}{c_{rep}^{v}} 100\% \tag{2.20}$$

After n customers' evaluations, the provider's reputation value is updated,

$$R_{n}^{T} = \frac{(n-1)R_{n-1}^{T} + R_{exp}^{T}}{n} \tag{2.21}$$

2.7.2 Credibility Mechanism

In our approach, the notion of credibility express the user's ability to provide objective evaluation for QoS KPIs. With this capacity, the credibility mechanism aims at reducing the impact of malicious users in the computation of reputation score. It takes into account the QoS KPIs and the respective SLA values. Essentially, the customer's subjective opinion, the predefined SLA and the monitoring values are compared in order to check the divergence between the user's rating and the cloud's actual performance. In this process the non-technical QoE KPIs are excluded due to their subjective nature.

Algorithm 2 shows the steps that define the user's credibility calculation process. The represented process concerns the use of one cloud provider. Nevertheless, in an real life scenario, customers have the opportunity to use more than one cloud providers. In that case the customer's credibility value is calculated utilizing the customer's ratings for all QoS KPIs of all providers. Considering the customer's opinion (ratings) for every QoS KPI, as the vector $UO = [UO_i]^{\top},\ i = 1, \ldots, k$, the monitoring data vector, $MD = [MD_i]^{\top},\ i = 1, \ldots, k$ and the SLA data vector, $SD = [SD_i]^{\top},\ i = 1, \ldots, k$, the algorithm updates the credibility value for the specific customer and a vector with the updated ratings for the QoS KPIs as those modified by the credibility mechanism, $\widetilde{UO} = [\widetilde{UO_i}]^{\top},\ i = 1, \ldots, k$ respectively. For each KPI of the cloud provider, the threshold and correction vectors are initialized (lines 4-5). The elements of the threshold vector express the tolerance against an opinion and is based on the deviation of the monitoring data from the SLA reference value (lines 6-13). The elements of the correction vector are actually the credibility values for each KPI. The customer's credibility for the current evaluation is computed as the average value of the elements of the correction vector. Then, the overall customer's credibility is updated

Algorithm 2 Credibility Mechanism

1: **Inputs:** UO, SD, MD
2: **Outputs:** CR, $\widetilde{UO}$
3: **for** $\forall UO_i \in UO$ **do**
4: $E = [e_i]^\top, e_i = 0.1,\ i = 1, \ldots, k$, Threshold Vector
5: $C = [c_i]^\top,\ i = 1, \ldots, k$, Correction Vector
6: **if** $|MD_i - SD_i| \geq e_i$ **then**
7: $e_i = |MD_i - SD_i|$
8: **end if**
9: **if** $|MD_i - UO_i| \leq e_i$ **then**
10: $c_i = 1$
11: **else**
12: $c_i = \frac{e_i}{|MD_i - UO_i|}$
13: **end if**
14: **end for**
15: $\hat{c} = avg(c_i)$
16: $CR_n = \frac{(n-1)CR_{n-1} + \hat{c}}{n}$
17: **for** $\forall UO_i \in UO$ **do**
18: **if** $|MD_i - UO_i| \geq e_i$ **then**
19: **if** $UO_i < MD_i$ **then**
20: $\widetilde{UO}_i = MD_i - e_i CR_n$
21: **else**
22: $\widetilde{UO}_i = MD_i + e_i CR_n$
23: **end if**
24: **else**
25: $\widetilde{UO}_i = UO_i$
26: **end if**
27: **end for**

according to lines 15-16. For each KPI, we adapt the customer's opinion if the difference between the opinion and the monitoring data is greater than the respective threshold value. The modified opinion is based on the monitoring value, the updated customer's credibility and the threshold value (lines 17-27). The modified opinions on KPIs are used in Phase 3 of HRS.

2.7.3 Evaluation

The collaborative SLA and RTM solution has been deployed, tested and evaluated in the FED4FIRE+ [29] platform. FED4FIRE+ initiative is the largest testbed federation in Europe, designed to facilitate experimentally driven research in the context of Future Internet Research and Experimentation (FIRE). Currently, the federation consists of sixteen core testbeds, offering wired, wireless, OpenFlow and cloud testbeds, recently extended to sup-

port big data experimentation. Furthermore, FED4FIRE+ is federated with other initiatives worldwide, i.e., GENI [74] and CloudLab [75]. The federation allows the experimenter to book and utilize resources from different testbeds at the same time in order to provide real life networking conditions for Future Internet experimentation, thus, it is suitable for evaluating the cloud applications' performance in federated environment. In this work, for demonstration purposes, Virtual Wall [76] and NETMODE [77] testbeds - both being part of the FED4FIRE+ federation - are used in order to test, evaluate and validate the proposed SLA and RTM services. The Virtual Wall testbed offers cloud resources and its SLA template includes two offerings; Service Availability and Response Time, while the NETMODE wireless testbed defines Node Availability (the degree of up-time of a wireless node) as SLA performance indicator. Regarding the RTM service, the above SLA offerings are used as QoS metrics of both testbeds and they are combined with four QoE KPIs; namely Support Satisfaction, Documentation Readability, Usability and Operability, in order to compute the reputation score of each testbed - provider.

The collaborative SLA and RTM solution is validated through two illustrative use cases. Scenario A architecture is depicted in Figure 2.12. Scenario A uses three physical nodes from Virtual Wall. Two of the physical nodes, named *Node0* and *Node1*, act as cloud providers, while the third one, *Node2*, hosts the RTM Service. Both *Node0* and *Node1* offer a dummy cloud application that just returns a 200 OK HTTP status code. The response time of the web server can be regulated for the purpose of the experiment. So, for *Node0*, it is configured to produce responses between 100ms and 400ms (stable node), while for *Node1* varies between 100ms and 650ms (unstable node). These values are selected for demonstrating the case of a stable cloud provider, which never breaches the SLA, and an unstable provider, which does breach the SLA, in order to validate the SLA mechanism and show the exploitation of SLA assessments by the RTM Service. In the Scenario A, the two SLA offerings guarantee: (i) the response time of the service should be lower than 500 ms, (ii) the service availability, measured as the average of availability values in ten minute intervals, should be higher than 90%. Finally, the Monitoring adapter obtains the values of the response time and the availability metrics, whose values are generated every ten seconds and ten minutes respectively, for *Node0* and *Node1*.

Scenario B emulates the deployment of an application within a Mobile Edge Computing

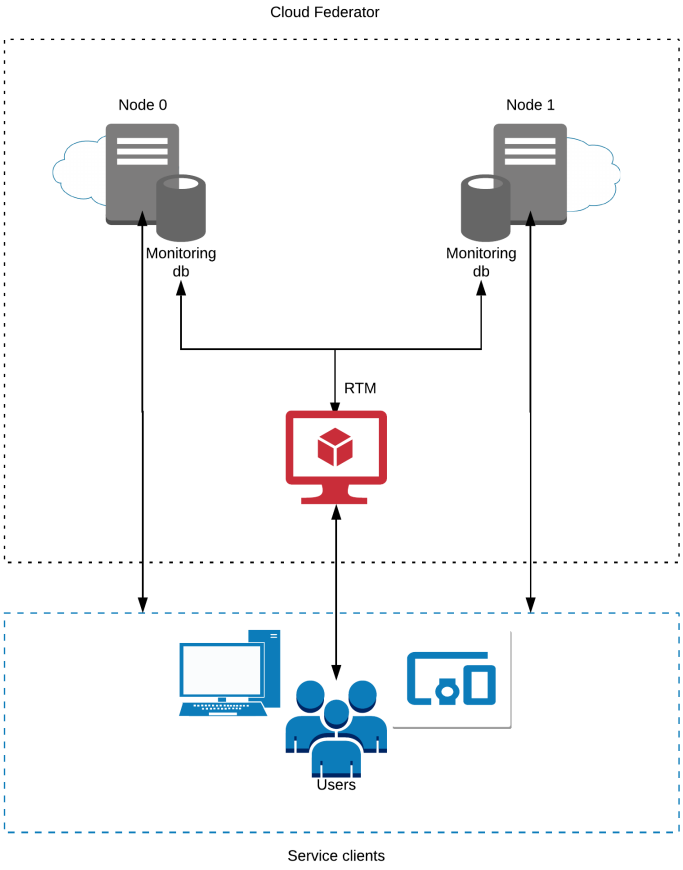


Figure 2.12: Scenario 1 Architecture

service delivery paradigm, which requires the orchestration of wireless and cloud resources, and illustrates the applicability and the efficiency of the proposed collaborative solution to any type of resources. The architecture of this scenario is illustrated in Figure 2.13. Three Raspberry Pi devices are connected on the wireless nodes of NETMODE testbed and generate requests of a cloud application. These requests are directed to a physical node of Virtual Wall testbed, which act as cloud provider and hosts the cloud application, which is identical to Scenario A and the same SLA offering is used. The cloud application is instantiated every half hour and 750 customers' ratings are submitted. In order to highlight the importance of the credibility mechanism, the 20% of these ratings are biased while the rest customers submits objective ratings.

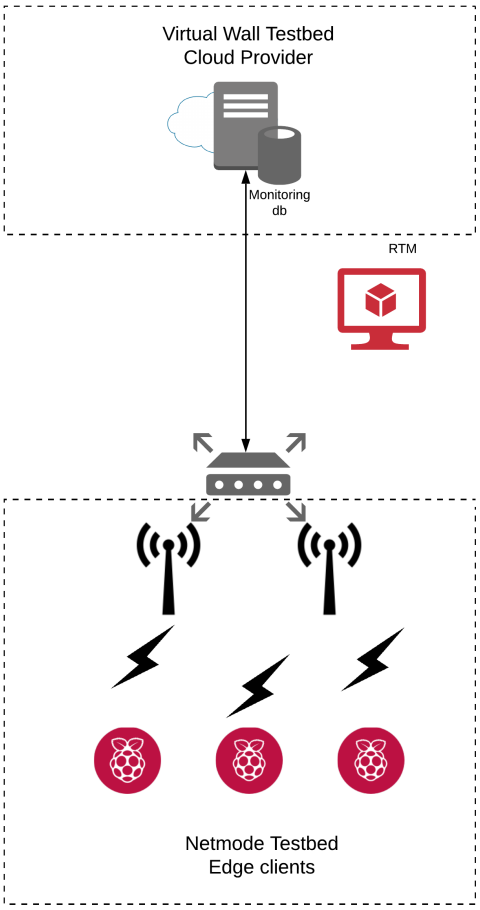


Figure 2.13: Scenario 2 Architecture

2.7.4 Proof of Concept

SLA Validation

Scenario A provides an SLA validation where two federated providers offer identical services and guarantee terms to the customer. As mentioned before, *Node1* is parameterized to produce SLA violations, while *Node0* fulfills the performance requirements, according to the agreement defined above. The cloud application runs for one and half hour. Each provider has a monitoring process, which stores the measured values into a KairosDB database, and the SLA Management module retrieves these monitoring data using a specific KairosDB Adapter. The SLA assessment process runs every minute and detects possible violations on the response time and service availability in different time scales. Regarding the first KPI,

Table 2.5: Evaluation data

	ratings	previous reputation	evaluation	monitoring data	modified opinion	updated reputation
Node0	7	82.1432	65	100	90	83.3227
Node1	6	74.5615	55	72.7777	55	73.4554

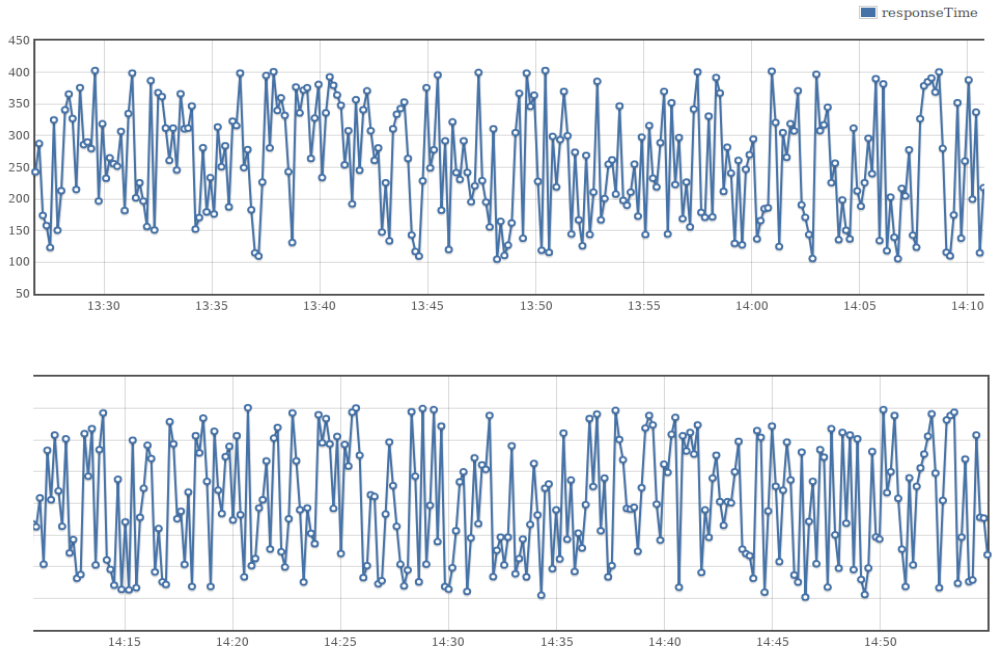
six values of response time are used in each assessment process, since it is polled every ten seconds. The monitoring value of the service availability is updated every ten minutes, thus the assessment is executed accordingly. Figures 2.14a and 2.14b show the response times and the corresponding violations for both cloud providers. As it was expected, there are no violations for the application running on *Node0*, while several violations are generated for *Node1*. On Figure 2.14b, the threshold value of the agreement is marked with green colour, while the red line highlights the generated response time violations. The created violations are forwarded to the cloud provider and the customer. Also, they are used by the HRS for the computation of the reputation score.

Reputation Example

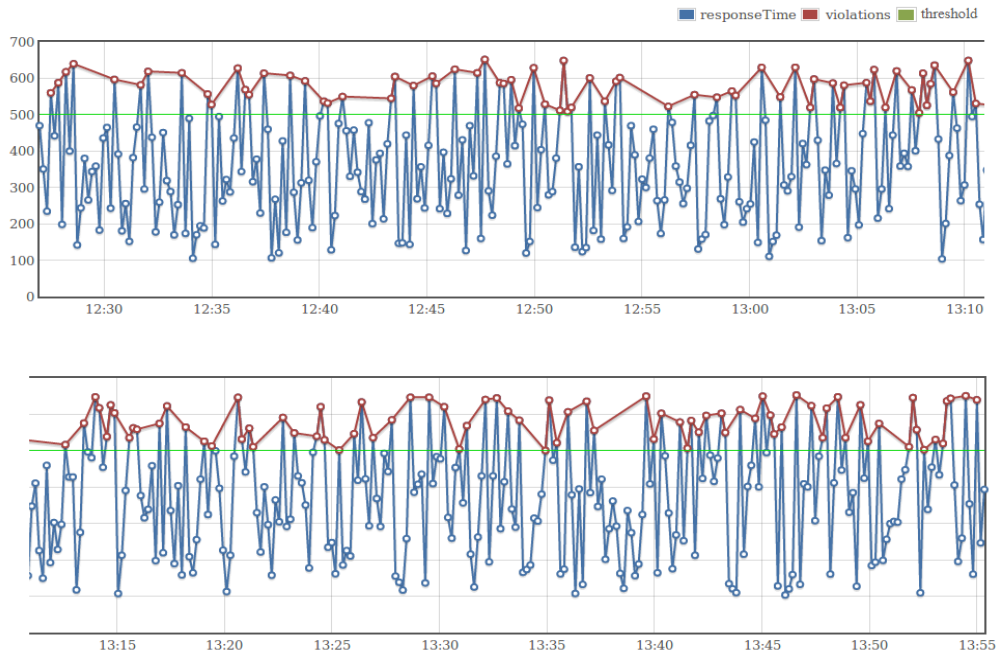
In this section, we demonstrate a simple example of the behaviour of the reputation algorithm to biased evaluations. After the end of the experiment described above, we compute the reputation score of the two nodes-providers. As described in the section above, *Node1* violates the agreed standards, contrary to *Node0*, which fulfills completely the agreement. Since the inputs of the credibility mechanism are normalized values, for the response time KPI, we utilize the SLA violations to define an alternative KPI, which actually measures the fragment of time interval where no violation occurs, with the following formula,

$$rt = (1 - \frac{violations}{samples}) \cdot 100$$

As it shown in Table 2.5, we assume that the submitted rating is the seventh and sixth sequential experiment for *Node0* and *Node1* respectively, while the respective reputation score is 82.1432 and 74.5615. In this scenario, the customer is not satisfied with the overall application's performance, thus, he submits poor evaluations, 65 and 55 respectively, for both nodes. The rating of *Node0* is unfair while the evaluation of *Node1* is closer to its actual performance. Leveraging the monitoring data, the credibility mechanism modifies the customer's opinion, thus the rating of *Node1* has changed to 90, while the rating of the



(a) Node0: SLA Assessment



(b) Node1: SLA Assessment

Figure 2.14: Application's Response Time

second node remains the same. This implies that the credibility mechanism fairly mitigates the effectiveness of misleading ratings on the reputation score of a provider. For the service

Table 2.6: Ratings for Node0

Top Level	First Level	Second Level	Experimenter	Virtual User
Reputation	Computing Performance (0.8)	Service Availability (0.5)	1	1
		Response Time (0.5)	0.65	1
	User Experience (0.25)	Support Satisfaction (0.25)	[VG]	[E]
		Operability (0.2)	[VG]	[E]
		Usability (0.25)	[VG]	[E]
		Document Readability (0.25)	[VG]	[E]

availability KPI, for both nodes, the ratings and the monitoring values are 100% and are not modified by the credibility mechanism. Thus, they are omitted from the Table 2.5. Additionally both nodes receive the "VERY GOOD" linguistic value for all the QoE KPIs.

Following the phases of Section 2.7.1, we calculate the updated reputation score of both nodes. We demonstrate the step by step calculation of the proposed method, for *Node0*. Table 2.6 shows the structure of KPIs in the hierarchical model of HRS. The user assigned weights are written next to every attribute and KPI. The third column contains the user's evaluation, while the ideal rating of the virtual user lies in the last column. As described in Phase 2 and 3 of subsection 2.7.1, the bottom-up procedure with the intermediate computations will be presented briefly. For example, the fuzzy RACM of the Support Satisfaction KPI is computed using the user's value $\tilde{A}_u = (6, 7, 8)$ and the virtual user's rating $\tilde{A}_v = (7, 8, 9)$,

$$RACM_{SupSat} = \begin{bmatrix} (1,1,1) & (0.67,\ 0.87, 1.14) \\ (0.87,\ 1.14, 1.5) & (1,1,1) \end{bmatrix}$$

Then, the fuzzy synthetic extent is computed for the virtual and the actual user is computed according to $RACM_{SupSat}$ and (2.15),

$$D_1 = (0.36, 0.47, 0.60)$$

$$D_2 = (0.40, 0.53, 0.71)$$

Following the procedure in Phase 3 and using (2.16) and (2.17), we get the degree of possibility for the experimenter and the virtual user, $d_1 = 0.75$ and $d_2 = 1$ respectively. Finally, the normalized comparison vector is obtained (2.18),

$$c_{SupSat} = [\ 0.43\ 0.57]^\top$$

Similarly, for the rest of the QoE KPIs, we calculate the following comparison vectors are computed,

$$c_{Oper} = [\ 0.43\ 0.57]^\top$$
$$c_{Usab} = [\ 0.43\ 0.57]^\top$$
$$c_{DocRead} = [\ 0.43\ 0.57]^\top$$

Then, the comparison vector for the User Experience attribute is obtained,

$$c_{UsExp} = \begin{bmatrix} 0.43 & 0.43 & 0.43 & 0.43 \\ 0.57 & 0.57 & 0.57 & 0.57 \end{bmatrix} \begin{bmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{bmatrix} = \begin{bmatrix} 0.429 \\ 0.571 \end{bmatrix}$$

For the QoS KPI Service Availability the comparison vector is,

$$c_{Avail} = [\ 0.5\ 0.5]^\top ,$$

as the rating is equal to the perfect evaluation of 1 (100%). The QoS KPI Response Time is modified according to the user's credibility. So the modified rating for Response Time is 0.9. The comparison vector of Response time is,

$$c_{RespTime} = [\ 0.474\ 0.526]^\top ,$$

Following the same procedure with the QoE KPIs, we compute the comparison vector for the Computing Performance attribute,

$$c_{CPerf} = \begin{bmatrix} 0.5 & 0.474 \\ 0.5 & 0.526 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 0.428 \\ 0.512 \end{bmatrix}$$

At the top level, the comparison vector for the Reputation attribute is,

$$c_{Reput} = \begin{bmatrix} 0.428 & 0.429 \\ 0.512 & 0.571 \end{bmatrix} \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix} = \begin{bmatrix} 0.475 \\ 0.525 \end{bmatrix}$$

Then, the reputation score for this experiment is computed by (2.20),

$$R_{exp}^{T} = \frac{c_{Rep}^{\tilde{u}}}{c_{Rep}^{v}} 100\% = 90.4\%$$

So, as that was the seventh evaluated experiment, the total Reputation Score for the Node0 is updated according to (2.21),

$$R_{7}^{T} = \frac{6 * R_{6}^{T} + R_{exp}^{T}}{7} = 83.32\%$$

2.7.5 Effect of Credibility Mechanism

As it is mentioned above, scenario B illustrates the effectiveness of the RTM service on federated heterogeneous resources and focuses on the effect of credibility mechanism on the computation of the reputation score. For both testbeds and 750 ratings, the reputation score is computed for two different cases. As depicted in Figure 2.15, in the first one the credibility mechanism has been taken into account (red line), while in the second (blue line) the update of the reputation values is performed without the use of the credibility mechanism. For both cases, the output of HRS is drawn in Figure 2.15. As it is shown, for both testbeds, the activation of the credibility mechanism leads to 20% increase of the reputation score. The credibility mechanism leverages SLA and monitoring data and compares them with the subjective user's opinion. If a significant deviation is observed, the user's opinion is modified and the credibility value is decreased. Thus the influence of the malicious evaluation on the reputation score is minimal. On the other hand, when the reputation value is updated without the use of the credibility mechanism, malicious users and evaluations seem to significantly affect the HRS performance. In that case, the reputation score is generally lower and rapid fluctuations are observed, as shown in Figure 2.15.

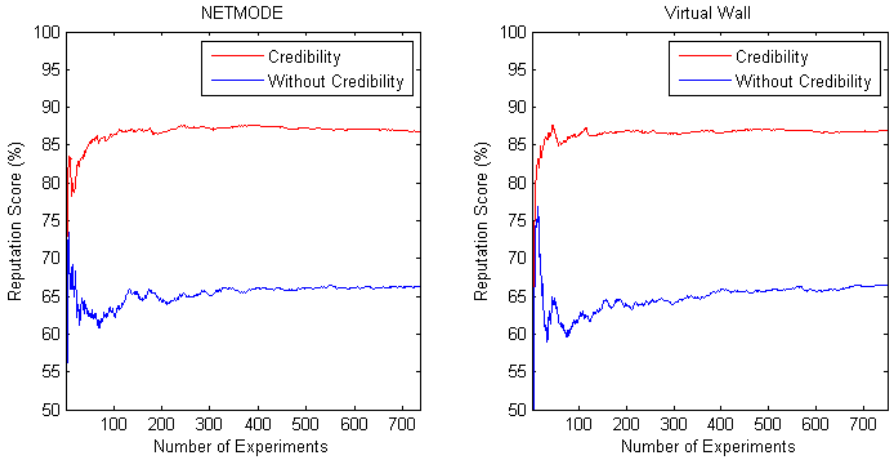


Figure 2.15: Credibility Mechanism's Effect in HRS

2.8 Conclusions

In this chapter we presented two MCDM-based trust management frameworks for heterogeneous federated clouds. Both frameworks are supported by our scalable architecture that can support any cloud federation. Both frameworks basically provide a collaborative SLA and trust management platform for federated cloud providers. The SLA service enables cloud providers to describe with specific indicators the performance of the deployed cloud application and provide the necessary tools for the assessment of the agreement. In case of performance degradation, SLA violations are generated and the interested entities are notified. Complementary, reputation-based trust management services are developed in order to fairly depict the provider's reliability to provide specific resources and services. This services are based on both QoS and QoE KPIs, and are suitable for federated environments since it scales easily.

The reputation-based trust management service is what mostly differentiates our two approaches presented in this chapter. The difference occurs by the methodology used to produce the reputation score of a provider. In our first approach we modified fuzzy VIKOR, which is an horizontal MCDM approach. The second approach we modified fuzzy AHP, which is a hierarchical MCDM system. Both approaches accommodate every users unique needs by letting them to assign different weights to different criteria according to their requirements.

Alongside, a credibility mechanism is introduced in both approaches, that leverages SLA

and monitoring data to mitigate the effect of biased customers' evaluations. Apart from the evaluation of the cloud application performance, the reputation-based trust management services facilitates future customers to select the appropriate providers and resources for deploying their applications.

Both presented approaches were implemented in cloud federation-like environment and experimentally evaluated. As far as the modified VIKOR approach is concerned, is shown to outperform the FTUE reputation framework, which is designed for experimental federated environment based only on numerical QoS and QoE metrics. This comparison underlines the importance of mixing several numerical and fuzzy metrics in the computation of the reputation score and the significance and robustness of the credibility mechanism.

The experimental results of the modified AHP approach, demonstrate that the approach's reputation output fairly represent the provider's reputation. Also, it was shown that the utilization of the credibility mechanism improves 20% the performance of the reputation service.

Chapter 3

Blockchain-Based Resource Orchestration on Edge Clouds

3.1 General Setting

As mentioned in Chapter 1, with the advent of the Internet of Things (IoT) and 5G era, modern time- and mission-critical applications with stringent requirements have been developed for every section of human activity. These complex end-to-end applications consist of network services that form a structured service chain. However, cloud resources cannot guarantee the delay requirements, and this motivates the need for the infrastructure operators to provide computing capabilities closer to end users. Thus, presently, the Edge Computing service delivery model is the most promising one to meet the requirements of applications derived from 5G verticals, Industry 4.0 [78] and smart cities [79, 80]. The resource orchestration at the edge of the network relies on the Network Function Virtualization (NFV) [22] and Software-Defined Networking (SDN) [81] concepts that facilitate network slicing. This allows infrastructure providers to lease customized bundles of computing, storage and network resources, to their tenants allowing the addressing of the service-specific (non-)functional requirements in an isolated fashion. Many standardization initiatives, i.e., ETSI Multi-access Edge Computing (MEC) [82] and ETSI-NFV [21] aim to describe and standardize the specifications on the automated resource orchestration and support of the

entire lifecycle of network services.

The evolution of 5G ecosystems and verticals requires the deployment of end-to-end network services over multiple domains or multi-administrative domain. Under this multi-tenant and multi-provider setting, the isolation of network slicing, which is its greatest advantage till now, impedes the cross-service communication (CSC). Thus, next-generation network service marketplaces (NSMs) aspire to overcome this obstacle and enable tenants to create tailor-made service chains using off-the-shelf network services that can consume other services in a secure manner. With this capacity, a multi-tier NSM should provide various functionality elements for both service provider and consumer. At the top layer of the marketplace, functionalities such as easy registration, enriched description and on-boarding of services should be provided to service providers. Furthermore, a service discovery and lease mechanism are essential for the tenants, who search for the network service to be included in their custom service chain. The next downward layer is responsible for handling the high-level information of every service chain and automating the resource orchestration of the network services. At the bottom layer, the service chain is instantiated by the resource orchestrator.

Although a centralized trusted authority and a trust management mechanism provide the desired level of trust, this is usually performed through non-automated functionality and requires a lot of human intervention. Furthermore, the complexity of trust management significantly increases in the case of multi-domain or multi-administrative resource orchestration. Contrary to centralized solution, Blockchain enables entities in trustless environment to make transactions in a decentralized manner and guarantees the integrity of the outcome of the transactions to all participants of the Blockchain network [83]. The transactions are stored in a ledger in the form of blocks and the order of the blocks is based on cryptographic hashes. The process that the blocks are produced ensures the immutability, data integrity, non-repudiation and security of the Blockchain ledger. Furthermore, Blockchain platforms such as Hyperledger Fabric [84], provide smart contracts that enable an event-based execution of transactions based on predefined rules and conditions. This is of high importance in incorporating any business logic as automated functionality in the Blockchain network.

3.2 Related Work

In this section, the most recent and interesting studies and projects relevant to the cross-service, multi-domain and Blockchain-based orchestration are presented.

On top of the open-source MANO (OSM) [85], MESON orchestration platform enables the cross-slice communication within an Edge Point of Presence (EPoP) [86]. Aligned with ETSI-NFV [87] and ETSI MEC [88] architecture, MESON platform provides centralized service discovery through a service registry, EPoP selection and establishment of cross-slice communication. Based on functional and non-functional requirements, the EPoP selection is performed by a multi-criteria EC selection methodology that determines the most suitable EC for the slice deployment [89]. Then, through OSM, an automated mechanism establishes the actual cross-slice communication that aims to minimize the generated intra-EC network traffic and allocated resources [90]. Also aligned with ETSI-NFV architecture, FENDE marketplace [91] enables developers to offer their VNF-based services, while users can select Virtual Network Functions (VNFs) from the service and VNF catalogue, compose custom service chains and instantiate them in public or private cloud infrastructures. NECOS marketplace is a broker-based platform that enables the resource discovery, negotiation and selection for deploying network slice over multiple administrative domains [92], while the 5GEx platform focuses on cross-domain orchestration of network services [93]. With respect to the latter, a hierarchical architecture of multi-domain orchestrators enables the exchange of business-level information with customers, the service discovery and placement and supports the VNF configuration and monitoring.

Rathi et al. proposed a Blockchain-enabled multi-domain orchestrator at the edge of the network [94]. The objectives of this approach are the automated orchestration of network slices over heterogeneous networks and the Service Level Agreement (SLA) assessment for 5G services using smart contracts. Nubo is a virtual service marketplace created to connect providers and consumers of such services in an edge/cloud computing environment [95]. Nubo is basically a web portal built over and extending Saranyu [96], which is a decentralized application (DApp) built on top of Quorum private Blockchain [97]. Saranyu is used for the management of cloud tenants and services, providing identity management, authentication and charging functionality with the use of smart contracts. The authors

in [98] present an experimental prototype for a multi-administrative domain federation of 5G services using distributed ledger technologies. Its objective is to provide a distributed federation solution where different administrative domains and providers participate in a permissioned Blockchain network with a single node. A single generic Federation Smart Contract acts as a distributed authority to enable secure and trusted interactions for functionalities such as registration to the federation, service announcement/discovery and service auctioning. The 5GZORRO project proposes an architecture focused on cross-domain security and trust resource orchestration mechanisms, by coupling Blockchain with AI-driven operations and service lifecycle automation in multi-tenant and multi-stakeholder environment [99]. Specifically, smart contracts are used to support multilateral agreements among all parties that are involved in the end-to-end service delivery. In [100], a Blockchain-based resource brokering mechanism for end-to-end slice deployment is proposed to secure transactions in the resource auctioning procedure. Upon a slice request, the slice is divided in sub-slices and an auction mechanism facilitates the resource providers to make bids for every sub-slice. The slice owner selects the best offers for each sub-slice. The authors of [101] proposes BSec-NFVO, a Blockchain-based system, to enable secure orchestration operations in virtualized networks ensuring auditability, non-repudiation and integrity. This mechanism is built on top of the Open Platform for Network Function Virtualization (OPNFV) [102]. The solution ensures security in delivering and orchestrating end-to-end NFV service chains with small performance overheads. In [103], a proof of concept Blockchain-based implementation using distributed applications is presented in the context of operational phases to support multi-administrative domain networking for multi-domain service orchestration. Also, the authors analyze three use case scenarios (MEC, 3GPP and ETSI-NFV standards) discussing standardization opportunities around Blockchain-based DApps as enablers for multi-domain network services.

3.3 Contribution & Outline

Our work complements and extends the above approaches and creates a Blockchain-based cross-service mechanism for ECs that alleviate the orchestration overhead and facilitates the establishment of NSMs. Leveraging the capabilities of smart contracts, the service

providers and consumers can easily trigger and monitor the lifecycle operations of their services. Furthermore, based on the power of the smart contracts and OSM, the proposed Service Orchestrator enables the automated establishment of the cross-service communication with the minimum resource requirements and instantiation overhead. Briefly, the basic contributions of our proposed solution in this chapter are summarized as follows:

- At the user layer, we specify the essential functionalities, i.e., registration, service discovery, service lease and billing, for both network service providers and consumers to become stakeholders of the marketplace.
- At the Blockchain layer, we update the asset definitions of [104] to facilitate cross-service orchestration with minimum possible data volume on the distributed ledger and preserve privacy of the users regarding their Network Slice information and structure.
- At the NFV layer, we provide an analytical description of our novel Service Orchestrator that establishes CSC orchestration leveraging Blockchain capabilities.
- We provide proof of concept results that demonstrate the applicability and efficiency of the proposed approach in terms of deployment time and reserved resources.

3.4 Network Service Marketplace Architecture

3.4.1 NFV-Related Definitions

Towards the establishment of CSC and NSM, it is important to describe the basic NFV assets that are taken into account in the proposed solution. Aligned with the ETSI-NFV architecture [87], a service chain corresponds to the network slice, which describes the main interactions as well as the network communications between the services. Based on OSM information model [85], a service chain is described in the Network Slice Template (NST). An example of such NST is shown in Listing 1. The main fields of the NST are the *netslice-subnet* and the *netslice-vld*. The *netslice-subnet* objects are the declared services of the slice. For each service, the slice owner determines the corresponding network service descriptor - which in OSM model is defined as *nsd* in the *nsd-ref* key, the name of the service and a Boolean key about enabling sharing for a specific service with other slices, in the *is-shared-nss* field. Therefore, a slice owner can share any of the services that is included in

an NST. The *netslice-vld* objects correspond to the networking management of the service chain. A *netslice-vld* object consists of various parameters. These parameters refer to both Management and Orchestration (MANO) level information (e.g., the identifier, name of the virtual network - vld) and to the Virtual Infrastructure Manager (VIM) level, such as configuration about VIM networking. Also, the connection points for each service, which is attached to that vld, are included. A network service consists of one or more VNFs, in which the necessary functionality of the service is implemented. Each VNF constitutes of a specific instance at the VIM level.

Considering the above features of the OSM Model and based on the ETSI-NFV architecture, we can design appropriate architectures and implement efficient mechanisms, in order to facilitate the cross-service interaction between services of different slice owners, leading to the network service marketplaces establishment.

An example of CSC is illustrated in Figure 3.1, where two network slices are deployed, with one shared NS between them. The *network-slice-1* consists of three NSs: *network-service-1-1, network-service-1-2*, and *network-service-shared* and two Virtual Networks: *vld1-1* and *vdl1-2*. In the descriptor of the shared NS, three connection points are defined, while the slice owner sets as *true* the *is-shared-nss* parameter in the NST descriptor. Then, another slice owner is interested for deploying a new slice (NST ID *network-slice-2*). This slice will include two new NSs, namely *network-service-2-1* and *network-service-2-2*, and the shared NS of the first network slice, *network-service-shared*. A connection point of the shared NS is necessary to create a virtual link between the shared NS and the *vld-2*, the virtual network of the second network slice. In such a way, the CSC is established without creating a new instance of *network-service-shared* and the instantiation time, the consumed resources, and the monetary cost are reduced for both slice owners. Beyond the basic set up of CSC, further functionalities, such as shared policies and billing, must be realized in order to provide a secure communication scheme in this multi-domain environment.

3.4.2 System Architecture

Our proposed Blockchain-based service marketplace architecture is designed in alignment with the ETSI-NFV standardization activities, and supports all phases and functionalities of an NSM, including registration, advertisement, leasing, orchestration, usage and billing.

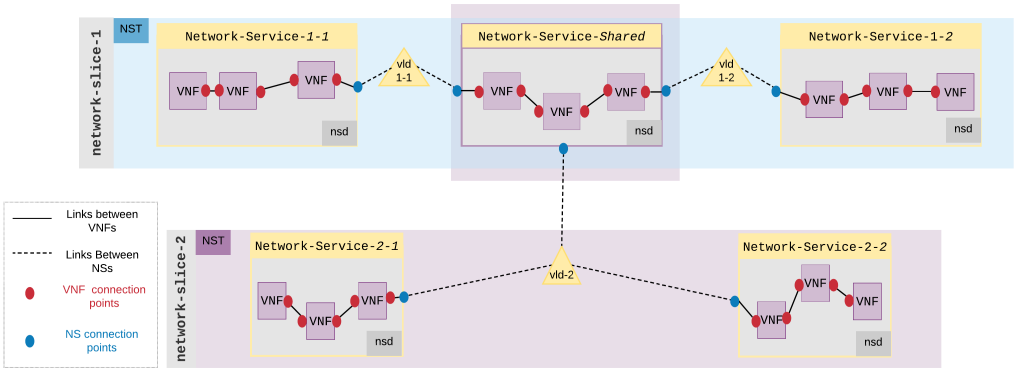


Figure 3.1: Cross-Slice Communication Using a Shared Network Service

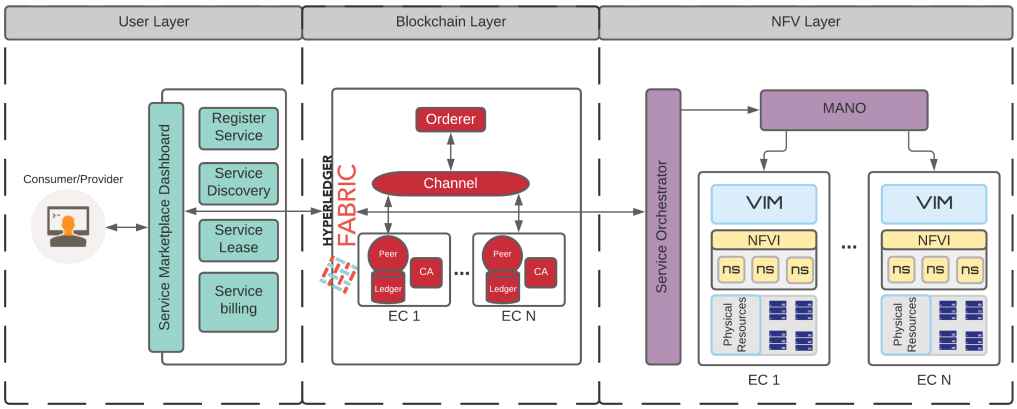


Figure 3.2: Network Service Management Architecture

Figure 3.2 provides a high-level overview of the components of the NSM architecture. As it is shown, the architecture consists of three layers corresponding to different functionality elements and operational phases.

User Layer

The User layer provides the essential functionality required for the interaction of the users with the NSM, as it is shown in the left part of Figure 3.2. Through a client, portal and/or some dashboard, both service providers and consumers can register a service available for leasing, discover service features, request service lease and trigger service instantiation, and establish CSC. In addition, this layer provides to the users monitoring and billing functionalities for the leased network services. Any data storage or retrieval is performed

Listing 1 Provider's NSTD

```
# NST descriptor example of a Service Provider. The slice consist of 2 Services.
# The second of them is shared among other slice tenants.
nst:
-   id: provider_slice_nstd
    name: provider_slice_nstd
    SNSSAI-identifier:
        slice-service-type: eMBB
    quality-of-service:
        id: 1

    netslice-subnet:
    -   id: provider-service-1
        is-shared-nss: 'false'
        description: NetSlice Subnet (service) composed by 1 vnf with 2 cp
        nsd-ref: provider-service-1_nsd
    -   id: provider-service-2
        is-shared-nss: 'true'
        description: NetSlice Subnet (service) composed by 1 vnf with 3 cp
        nsd-ref: provider-service-2_nsd


    netslice-vld:
    -   id: slice_vld_mgmt
        name: slice_vld_mgmt
        type: ELAN
        mgmt-network: 'true'
        nss-connection-point-ref:
        -   nss-ref: provider-service-1
            nsd-connection-point-ref: nsd_cp_mgmt
        -   nss-ref: provider-service-2
            nsd-connection-point-ref: nsd_cp_mgmt
    -   id: slice_vld_data
        name: slice_vld_data
        type: ELAN
        mgmt-network: 'false'
        nss-connection-point-ref:
        -   nss-ref: provider-service-1
            nsd-connection-point-ref: nsd_cp_data
        -   nss-ref: provider-service-2
            nsd-connection-point-ref: nsd_cp_data
```

by invoking the appropriate smart contracts through the Blockchain layer's interfaces.

Blockchain Layer

The Blockchain layer handles through smart contracts all information required regarding users, services etc. This proposition leverages the Blockchain functionality, the smart contracts and the components described in [104]. Additionally, as it will be described in the following sections, we update and extend the above components.

As shown in the middle part of Figure 3.2, the Blockchain layer architecture refers either to a single EC deployment or a multi-domain EC environment, which can be either single

or multi-administrative. In both scenarios, each EC is considered to be an individual organization in the Fabric Network. Each organization has instantiated a Fabric Peer with the smart contracts, a Certificate Authority to manage identities and a Ledger. The information on the Ledger is identical in every EC. All organizations join a common Fabric channel and finally the orderer handles the consensus mechanism and is responsible for ordering transactions, creating new blocks and distributing a newly created block to all peers in a channel. The Blockchain Layer also forwards all cross-service orchestration requests and information to the Service Orchestrator of the NFV layer, which is thoroughly analyzed in Section 3.5.2.

NFV Layer

Aligned with ETSI-NFV architecture, the NFV layer is responsible for the network service instantiation and the establishment of the cross-service communication. On top of this layer our extended Service Orchestrator offers additional functionality elements and abstractions to OSM, in order to enable fully automated CSC. The Service Orchestrator handles the above by interacting with OSMs APIs. Section 3.6.2 provides analytical description of the developed Service Orchestrator.

3.5 Network Service Marketplace Operations

In this section, we describe the concepts and phases of the proposed Network Service Marketplace. The main concept and ambition of the NSM is to enable EC tenants, even from different providers or domains, to act as providers of their own services to candidate customers, or being consumers of off-the-shelf services instead of having to develop and deploy them on their own. Such services could be for example image recognition services, media caches etc. For that purpose, we implement a distributed Blockchain-based marketplace using Hyperledger Fabric. The Blockchain solution enables untrusted parties to interact and perform transactions in a trustworthy and secure environment, through smart contracts without any intermediaries involved, and use the Blockchain distributed ledger as a datastore for our marketplace. Additionally, as the proposed NSM is aligned with the ETSI-NFV architecture, any ETSI-NFV-based EC can join the marketplace by just installing a Fabric Peer with its ledger and the smart contracts instantiated, accompanied with a certificate

authority. For the case of EC, this process requires a very small amount of resources and minimum administrative overhead from the providers. Below we present in detail all functionality of all the layers of the proposed NSM.

3.5.1 Marketplace Functionality

The User layer supports the interaction of both providers and users with the NSM and it provides the following functionalities,

- **Registration**: In the Registration phase, a tenant of an EC registers to the marketplace. Also, some minimum information described in 3.5.2 is provided about the tenant's Network Slice for orchestration purposes. In the registration phase, a tenant can also register a service for leasing providing information about the service to be leased, quotas pricing, etc.

- **Advertisement**: After a service is registered for lease and committed to the datastore, it is automatically retrieved and advertised in a service catalogue for the rest marketplace users, in order to browse and select it for lease in an automated fashion.

- **Discovery**: In the discovery phase, the users of the marketplace can browse and select a service for lease according to their needs through the service catalogue.

- **Lease**: After service selection, users request the leasing of a service for a specific time period, which can be renewed. Upon request, the leasing is granted by the smart contract and then the appropriate establishment of cross-service communication is performed automatically, and the leased service is ready to be consumed.

- **Usage**: In the usage phase of a leased service, depending on the pricing model of the service, the usage of the service is monitored and stored in the data store for logging, while the user can track the usage of the leased service.

- **Billing**: Depending on the usage monitoring data of a leased service and the pricing model defined by the service provider, billing is calculated by a smart contract and the appropriate fee is sent to the consumer for a specific predefined time period. Also, the users can monitor their usage and expected fees at any moment.

As an example, we present the workflow of the Register Lease function of the smart contract that corresponds to one the Lease functionalities of the user layer. This function grants a requested service lease, commits the Lease object in the data store as described in 3.5.2 and supports the CSC orchestration process. As depicted in Figure 3.3, when a user requests a lease, the client invokes the register function of the Smart Contract. All attributes of the Lease object are provided as input. Then, the Smart Contract checks if the requested Service for lease exists in the ledger. If the Service exists, then the Smart Contract checks the received service object from the ledger to validate that the service requested belongs to the grantor specified in the request. Lastly, the Smart Contract checks if the lease requested has already been granted and already exists in the ledger. If it does not exist, the lease is granted and committed to the ledger and the client is informed for the lease success. If any of the above validations fails, the smart contract returns to the client the appropriate error message. The above functionality corresponds to the Request Lease operation as depicted in Figure 3.4. More details about all developed smart contracts can be found here [105]

3.5.2 Service Data Store-Blockchain Functionality

As above mentioned, we leverage Blockchain technology and smart contracts to support the network service lifecycle of the NSM and use the distributed ledger as the NSM data store, while ensuring trust, security, authenticity, integrity and immutability. In this subsection, we focus on the data stored in the data store. The data store is one of the most important components of the marketplace as it facilitates the service advertisement, leasing and orchestration functions of the marketplace. The stored information on the distributed ledger contains data both for service advertisement and discovery, and cross-service orchestration. The data required for the orchestration operations are fully aligned with the ETSI-NFV standard and refer to information about Network Slices and Services linking to information also stored in OSM database. Delving into more details, Listing 2 illustrates the structure and format of the stored data.

Initially, the **Network Slice** object refers to the set of virtualized computing and network resources required to deploy an application. The *ID* and *Name* attributes refer to the Network Slice Template descriptor of the slice as described by ETSI-NFV, and the *Tenant* attribute points to the object stored in the ledger that contains all information about the

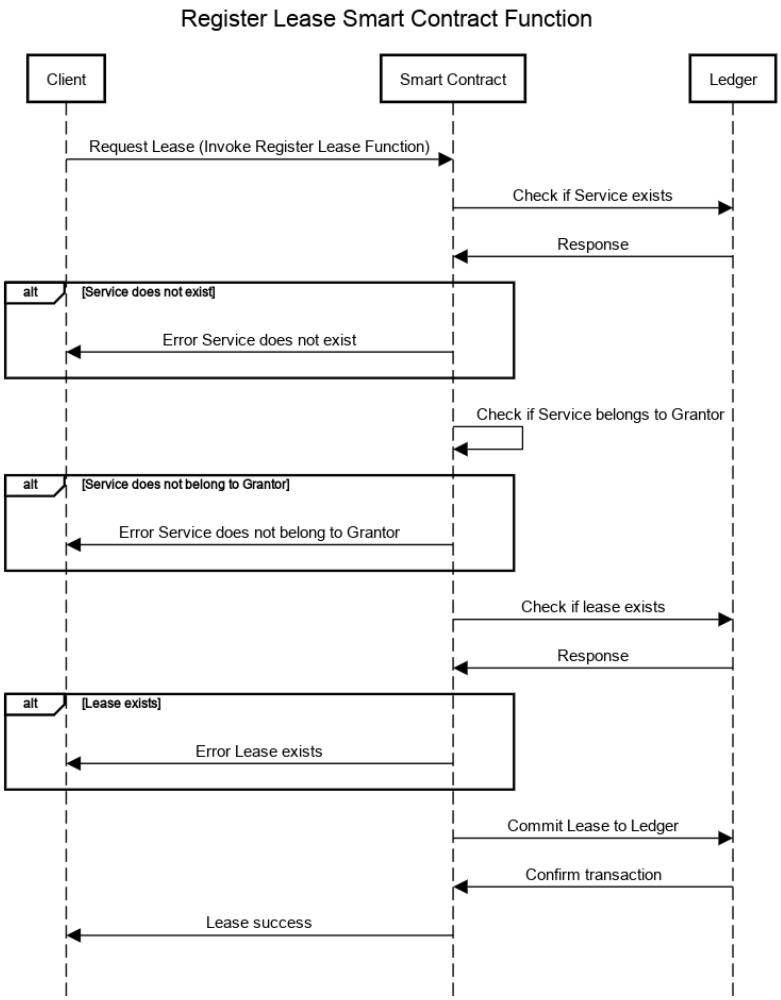


Figure 3.3: Register Lease Smart Contract Function

slice owner.

The **Service** object refers to the advertised network services for lease in the marketplace. As it is shown in Listing 2, it contains the following attributes; *ID, Name, Shortname, Description, NsdName and Provider*. The attributes of *Name, Shortname* and *Description* are used solely for advertisement and discovery purposes and contain information and functionality description of the service for lease. The attributes of *ID, NsdName* and *Provider* are used for orchestration, management and billing operations. In particular, the *NsdName* argument indicates the name of the service's Network Slice descriptor that contains all information and descriptions for the slice deployment. The *ID* attribute indicates the unique identifier of the service's Network Slice instance. The *Provider* attribute points to a Network

Slice object in the ledger that contains the information of the Network Slice of the provider and the tenant to which it belongs.

Listing 2 Data store formats.

```
//NetworkSlice data struct def
type NetworkSlice struct {
        ObjectType string `json:"docType"`
        ID         string `json:"id"`
        Name       string `json:"name"`
        Tenant     Tenant `json:"tenant"`
}

//Service data struct def
type Service struct {
        ObjectType  string `json:"docType"`
        ID          string `json:"id"`
        Name        string `json:"name"`
        ShortName   string `json:"short_name"`
        Description string `json:"description"`
        NsdName     string `json:"nsd_name"`
        Provider    string `json:"provider"`
}


//Lease data struct def
type Lease struct {
        ObjectType string   `json:"docType"`
        Grantor    string   `json:"grantor"`
        Recipient  string   `json:"recipient"`
        Service    string   `json:"service"`
        Issue      int32    `json:"issue"`
        Expiry     int32    `json:"expiry"`
        Revokers   []string `json:"revokers"`
}
```

Finally, the **Lease** object contains the stored information in the distributed ledger whenever a lease is granted. The *Grantor* and *Recipient* attributes refer to the Network Slice descriptors of the Provider and the Consumer respectively for orchestration purposes. The *Service* argument contains the ID of the service to be leased as described above, while the *Issue* and *Expiry* attributes indicate the duration of the granted lease. Lastly, the *Revokers* attribute is an array containing the identities of the users or entities that can revoke the lease.

Regarding the **Network Slice** and **Service** objects, we aim at maintaining only the essential information for orchestration in the ledger for the following two reasons. Initially, preserving the users' data privacy is one of the top priorities of the NSM. With this capacity, we only store the name of the Network Slice's descriptor in the data store and not the whole descriptor which would reveal internal details and structure of a deployed slice. When or-

chestration is requested, our Service Orchestrator can retrieve the whole descriptor securely from the OSM's API to perform the required operations to enable CSC without storing any data. This procedure and all related operations are thoroughly presented in Section 3.6.2. Secondly, storing only the name of the descriptor, instead of the whole or partial information of it, we preserve the minimum amount of data in the data store.

3.6 Network Service Lease and Orchestration

In this section, we describe the lifecycle and component interaction of an automated cross-service orchestration. At first, we discuss the interaction lifecycle of the three layers of the architecture depicted in Figure 3.2 and then we present, step by step, the operations and interactions performed in the NFV layer within this lifecycle.

3.6.1 Network Service Lease

In this subsection, we briefly describe the lifecycle of a network service lease and cross-service orchestration. As Figure 3.4 illustrates, a client or dashboard sends a lease request to the Blockchain layer invoking the smart contract. If the lease is successful, its information is stored in the distributed ledger and the client/dashboard receives the appropriate message. After the acknowledgement of the successful leasing, the client/dashboard requests the appropriate CSC orchestration from the NFV layer by contacting the Service Orchestrator. Upon completion of the orchestration, the NFV layer returns a confirmation message to the client and invokes the smart contract, so that the establishment of the CSC is committed and stored to the distributed ledger mainly for logging purposes.

3.6.2 Network Service Lease Orchestration

The main interactions between the Service Orchestrator component of the proposed architecture, the MANO, the Dashboard and the Blockchain layer are presented in Figure 3.5. In this subsection, we describe these interactions, from the scope of the NFV layer, in order to clarify the key functionality of the proposed Service Orchestrator. As a first step, the interested client for CSC instantiation, posts a request on the Service Orchestrator API. The request consists of the NST descriptor (NSTD) name, which is onboarded in the OSM and

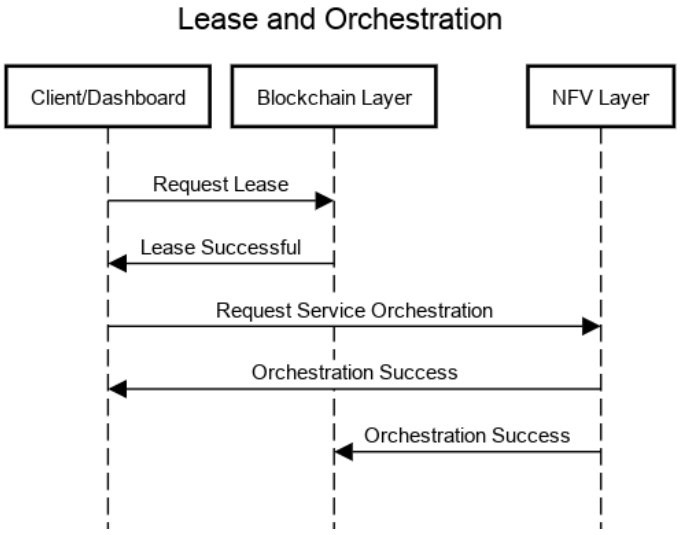


Figure 3.4: Lease and Orchestration Lifecycle

describes the service chain of the client, alongside with the identifier of the specific service of the provider, for which the user is interested in cross-service communication, among the set of the registered services in the data store. At the second step, the Service Orchestrator requests the provider's NSTD for the CSC-available service and extracts the necessary data from the descriptor file. Looking back on Listing 1, where an example of a NSTD of a service provider is depicted, we assume that a client is interested in the network service with identifier *"provider-service-2"*. In order to update the client's NSTD, we need the information stored in the *netslice-subnet* object, which matches with the *id* attribute, and the *netslice-vld* objects for the corresponding connection points of the desired service. Listing 3 illustrates the extracted data in a JSON file for the *"provider-service-2"* service. After retrieving the mandatory data from the provider's NSTD, the Service Orchestrator requests the onboarded client's slice NSTD. Then, using the extracted data from the provider's NSTD, the Service Orchestrator updates the client's NSTD accordingly, in order to be able to access the shared service. The updates in the descriptor refer to the addition of the shared-service object of the *netslice-subnet* field in the JSON file, and the corresponding connection points attachment at a management-network and a data network of the client's slice. The additions take place in the YAML NSTD file of consumers, in the corresponding fields. Finally, the updated NSTD is used for the instantiation request. For the instantiation, the Service Orchestrator constructs a configuration object, with parameters that refer to the VIM layer, in

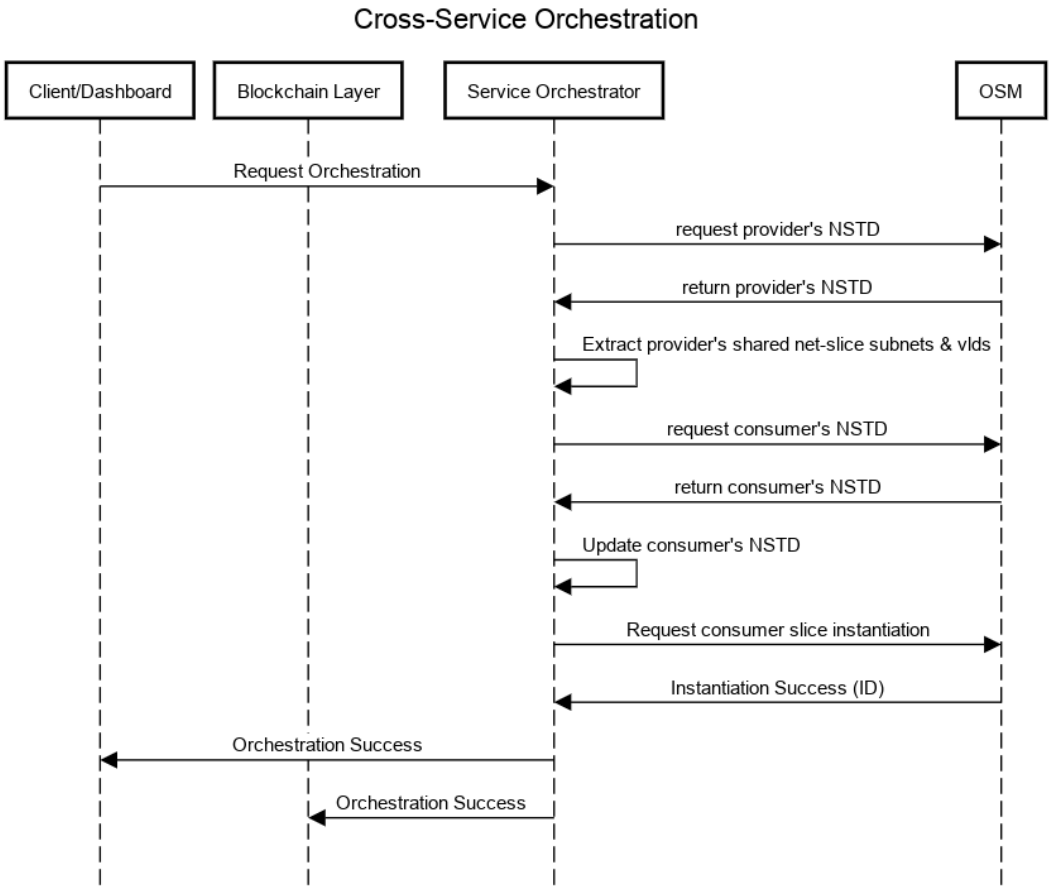


Figure 3.5: CSC interactions and operations

order to enable the network connections between the services of the client and the provider. After a successful instantiation, every component of the proposed architecture receives an orchestration success message.

Listing 3 Provider's extracted shared info.

```
{
        "netslice-subnet":
                {
                "id":"provider-service-2",
                "is-shared-nss": true,
                "description": "Provided service for cross-service interaction",
                "nsd-ref": "provider-service-2_nsd"
                },
        "mgmt-connector":
                        {
                                "nss-ref": "provider-service-2",
                                "nsd-connection-point-ref": "nsd_cp_mgmt"
                        },
        "data-connector":
                        {
                                "nss-ref": "provider-service-2",
                                "nsd-connection-point-ref": "nsd_cp_data"
                        }

}
```

3.7 Experimentation and Results

In this section, initial evaluation results are presented, aiming to demonstrate the benefits and feasibility of our proposed approach. For the assessment, the following experimental setup has been used. A virtual machine instance of 4 vCores and 8 GB of memory running Ubuntu 18.04 was used to deploy a test Hyperledger Fabric network v1.4 with two Fabric Peers with all instantiated smart contracts. OSM v8.0.4 as the MANO component, Openstack 5.4.0 as the VIM and the developed Service Orchestrator were deployed in a Virtual Machine (VM) with the same specifications. The Service Orchestrator is developed in Python [106] and the API was built using the Flask framework. The client is implemented as a python script interacting with the Blockchain and NFV layer as depicted in Figures 3.4 and 3.5. Our evaluation scenario is performed using a single EC. We evaluate our solution in terms of orchestration time overheads and resource consumption compared to known practices in the literature for cross-service communication.

In terms of time overheads, OSM takes an average of 38 s to perform the cross-service orchestration (consumer slice instantiation) while all API and Blockchain interactions add an average of 2-s overhead. Thus, the interaction between the client, the Blockchain layer, the Service Orchestrator and the OSM corresponds to the 5% of the total cross-service orchestration time, which is considered insignificant for practical purposes. As far as scaling

is concerned, depending on the version of Fabric and the programming language the Smart Contracts are developed, Hyperledger Fabric can perform from 194 transactions per second, up to 592 transaction per second (commit to ledger operation) for our assets size [107] , while Openstack can process up to 5 transactions per second for network creation [108], which is the operation needed to enable CSC. Taking also into account that Openstack is one of the industries standards for Cloud, Edge and NFV infrastructures, it is evident that the service instantiation at the NFV layer is the bottleneck in terms of scaling for the establishment of CSC.

Additionally, the proposed architecture focuses on the minimization of the required computing and network resources for cross-service orchestration. For example, in typical cross-slice communication solutions, an intermediary network slice is deployed to establish the communication between the provider's slice and the consumer's one [86]. In terms of resources, this solution demands extra computing and network resources, such as VCPUs, Memory and Connection Points, in the VIM, where the slices are located. Considering the simplest scenario, which requires an intermediate network slice just for traffic forwarding, a VM with a minimal Linux distribution allocates at least one VCPU and 512MB RAM. Furthermore, the more complex requirements the CSC establishment has the more resources the intermediate slice consumes. However, the resources of an EC are limited and the existence of many slices with CSC peers leads to the allocation of significant number of resources for the essential intermediate network slices. Furthermore, the CSC intermediary Slice instantiation from the MANO layer introduces an additional time overhead. On the contrary, the Blockchain-based CSC is minimal and only the existing connection points are routed and connected in the virtual network through the NST descriptor updates.

3.8 Conclusions

In this chapter, we presented a Blockchain-based network service marketplace and a resource orchestration mechanism to enable cross-service communication in edge clouds. Based on the ETSI-NFV architecture, the proposed solution allows the stakeholders of the marketplace to interact trustfully in a multi-domain or multi-administrative environment. Furthermore, we introduce a novel Service Orchestrator that offers abstractions and functionality to automat-

ically orchestrate cross-service communication for service consumption to enable off-the-shelf leasing of services. We presented this process following a step-by-step approach, from lease to service usage by the consumer. Our proposed solution brings promising results, as the essential operations to enable CSC add only an insignificant overhead to the total orchestration time. The rest of the time is consumed by the operations made by OSM and the according VIM to instantiate the CSC. In addition, compared to other proposed solutions for CSC that require intermediary slices, our work does not require extra resources to establish the cross-service communication. These metrics indicate that the proposed architecture and solution provides efficient, fully automated and seamless CSC orchestration.

Chapter 4

Conclusions & Future Work

In this Chapter summarize the conclusions of our dissertation we will talk about our thoughts that could lead us in interesting future research directions based on this thesis.

4.1 Conclusions

Throughout this dissertation, we discussed and addressed challenges in the life-cycle management of services in federated edge-cloud environment, while enabling trust between involved parties in such context.

In chapter 2, we introduced two reputation-based trust management frameworks for cloud service and service provider performance assessment and evaluation. Both solutions are collaborative, in the sense that they also use data from SLAs that are offered to assess and compute the reputation value of a service based on modified MCDM methodologies, as described in chapter 2. In both frameworks, we introduce credibility mechanisms that protect our system from malicious actors and evaluations. Our proposed frameworks through reputation values quantify the subjective collective opinion of a service's performance. In this way they enable trust, while at the same time they facilitate the service selection, monitoring and performance assessment stages of a service life-cycle. Experimentation on real infrastructure and simulations demonstrate the better performance compared with other proposed in literature, the importance and robustness of the credibility mechanisms and the high importance of mixing several numerical and fuzzy metrics in the computation of the

reputation score.

In chapter 3, we propose a blockchain-based NSM for off-the-self leasing of services provided by federated edge cloud providers and tenants along side with a novel CSC orchestrator. Blockchain, as a trustless system, enables trust in towards the platform and it's transactions through its consensus mechanism and the way smart contracts operate. Our proposed multi-domain architecture is highly scalable, new edge computing providers can be added at any time with minimum administrative and resource overhead. This solution handles or assists every step in service's life-cycle and it is aligned with the ETSI-NFV standards. Our novel CSC orchestrator offers abstraction's over OSM and can automatically assisted by blockchain to handle the orchestration with insignificant time overheads and no extra resources compared to other propositions in the literature as shown by implementation and experimentation.

4.2 Future work

Based on the work in this dissertation, we would like to contribute some interesting ideas for future development and enrichment of our work and significant synergies that can occur between the different solutions presented.

Firstly, our trust-based solutions could be enriched with more sophisticated QoS and QoE KPIs in order to be able to express even more accurately the diversity of the requirements of the unique user. In addition and most importantly, exploiting the reputation scores and the ability to provide custom weights for each criteria, the reputation system could be complemented by a recommendation system that will process the information and navigate the user through selection instead of the users evaluating provided services through their processing of the reputation values.

Secondly, for our trustless blockchain-based solution, while it's architecture is multi-domain and highly scalable our novel orchestrator, mostly due to infrastructure restrictions in the development phase, is developed and tested for services in the same edge cloud provider. Incorporating the same principles and abstractions it could be easily extended to handle orchestration incorporating multiple VIMs and by extension multiple providers.

Last but not least, with the above extensions of the two approaches separately in mind,

by combining both approaches a truly complete solution for service life-cycle management in federated edge-cloud environment could occur. With the trust management frameworks on top, providing trust towards the performance and services, handling recommendation in the selection process and providing service monitoring performance assessment and SLA guarantees. The blockchain solution below it, providing trust in all transactions required in an NSM, enabling the federation of edge clouds through the marketplace, handling orchestration and facilitating every step of a service's life-cycle from automated service advertisement and discovery throughout billing.

Appendix A

Preliminaries on Fuzzy Sets

The basic concepts of fuzzy numbers and their mathematical operations are presented in the following. Zadeh defined the fundamental concepts of fuzzy logic and sets in [109]. A fuzzy number $A =$ is a fuzzy set and its corresponding membership function $\mu_A(x)$ must hold the following properties,

- $\mu_A(x) : \mathbb{R} \rightarrow [0, 1]$, which means that is continuous and normalized fuzzy set.

- For exactly one element x_0, $\mu_A(x_0) = 1$.

- $\mu_A(x)$ is a convex fuzzy set.

In our proposed fuzzy approaches, we use positive triangular membership functions (TMF), as shown in Figure A.1, that are defined as,

$$\mu_A(x) = \begin{cases} \frac{x-l}{m-l} & \text{if } x \in [l, m] \\ \frac{u-x}{u-m} & \text{if } x \in [m, u] \\ 0 & \text{otherwise} \end{cases} \tag{A.0.1}$$

Assuming that $l \leq m \leq u$, a fuzzy number is denoted as the triplet $A = (l_A, m_A, u_A)$. The following mathematical operation between fuzzy numbers are defined according to [3],

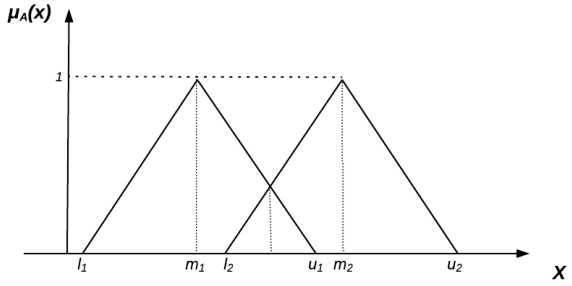


Figure A.1: Triangular Membership Functions

$$A \oplus B = (l_A + l_B, m_A + m_B, u_A + u_B), \tag{A.0.2}$$

$$A \ominus B = (l_A - u_B, m_A - m_B, u_A - l_B), \tag{A.0.3}$$

$$A \otimes B = (l_A * l_B, m_A * m_B, u_A * u_B), \tag{A.0.4}$$

$$A \oslash B = (l_A / u_B, m_A / m_B, u_A * l_B). \tag{A.0.5}$$

The comparison of fuzzy numbers is not straightforward. The most common comparison method is the defuzzification of these numbers; converting them into crisp values. Many defuzzification methods were proposed in literature. Adopting the defuzzification approach of [67], the crisp value $\hat{A}$ of the fuzzy number A is defined as,

$$\hat{A} = \frac{l + 4m + u}{6} \tag{A.0.6}$$

Appendix B

Author's Publications

International Peer Reviewed Journals

- Dechouniotis, D., Dimolitsas, I., **Papadakis-Vlachopapadopoulos, K.** and Papavassiliou, S., 2018. Fuzzy multi-criteria based trust management in heterogeneous federated future internet testbeds. Future Internet, 10(7), p.58.
- **Papadakis-Vlachopapadopoulos**, K., González, R.S., Dimolitsas, I., Dechouniotis, D., Ferrer, A.J. and Papavassiliou, S., 2019. Collaborative SLA and reputation-based trust management in cloud federations. Future Generation Computer Systems, 100, pp.498-512.
- **Papadakis-Vlachopapadopoulos**, K., Dimolitsas, I., Dechouniotis, D., Tsiropoulou, E.E., Roussaki, I. and Papavassiliou, S., 2021, March. On Blockchain-Based Cross-Service Communication and Resource Orchestration on Edge Clouds. In Informatics (Vol. 8, No. 1, p. 13). Multidisciplinary Digital Publishing Institute.

International Conferences

- Kalatzis, N., Avgeris, M., Dechouniotis, D., **Papadakis-Vlachopapadopoulos**, K., Roussaki, I. and Papavassiliou, S., 2018, June. Edge computing in IoT ecosystems for UAV-enabled early fire detection. In 2018 IEEE International Conference on Smart Computing (SMARTCOMP) (pp. 106-114). IEEE.
- **Papadakis-Vlachopapadopoulos, K.**, Dimolitsas, I., Dechouniotis, D., Tsiropoulou,

E.E., Roussaki, I. and Papavassiliou, S., 2020, December. Blockchain-Based Slice Orchestration for Enabling Cross-Slice Communication at the Network Edge. In 2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C) (pp. 140-147). IEEE.

- Spatharakis, D., Avgeris, M., Kakkavas, G., **Papadakis-Vlachopapadopoulos, K.**, Dimolitsas, I., Dechouniotis, D., Karyotis, V. and Papavassiliou, S., 2021, September. In IEEE International Mediterranean Conference on Communications and Networking. Demo Session

Bibliography

[1] Ioannis Patiniotakis, Stamatia Rizou, Yiannis Verginadis, and Gregoris Mentzas. Managing imprecise criteria in cloud service ranking with a fuzzy multi-criteria decision making method. In Kung-Kiu Lau, Winfried Lamersdorf, and Ernesto Pimentel, editors, *Service-Oriented and Cloud Computing*, pages 34–48, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[2] Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. Smicloud: A framework for comparing and ranking cloud services. In *Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on*, pages 210–218. IEEE, 2011.

[3] Da-Yong Chang. Applications of the extent analysis method on fuzzy AHP. *European journal of operational research*, 95(3):649–655, 1996.

[4] GMDT Forecast. Cisco visual networking index: global mobile data traffic forecast update, 2017–2022. *Update*, 2017:2022, 2019.

[5] Dave Evans. The internet of things: How the next evolution of the internet is changing everything. *CISCO white paper*, 1(2011):1–11, 2011.

[6] Li Da Xu, Wu He, and Shancang Li. Internet of things in industries: A survey. *IEEE Transactions on industrial informatics*, 10(4):2233–2243, 2014.

[7] In Lee and Kyoochun Lee. The Internet of Things (IoT): Applications, investments, and challenges for enterprises. *Business Horizons*, 58(4):431–440, 2015.

[8] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials*, 17(4):2347–2376, 2015.

[9] Jianli Pan and James McElhannon. Future edge cloud and edge computing for internet of things applications. *IEEE Internet of Things Journal*, 5(1):439–449, 2017.

[10] Peter Mell, Tim Grance, et al. The NIST definition of cloud computing. 2011.

[11] Hesham El-Sayed, Sharmi Sankar, Mukesh Prasad, Deepak Puthal, Akshansh Gupta, Manoranjan Mohanty, and Chin-Teng Lin. Edge of things: The big picture on the integration of edge, IoT and the cloud in a distributed computing environment. *IEEE Access*, 6:1706–1717, 2017.

[12] Ashkan Yousefpour, Caleb Fung, Tam Nguyen, Krishna Kadiyala, Fatemeh Jalali, Amirreza Niakanlahiji, Jian Kong, and Jason P Jue. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *Journal of Systems Architecture*, 98:289–330, 2019.

[13] M Series. IMT Vision–Framework and overall objectives of the future development of IMT for 2020 and beyond. *Recommendation ITU*, 2083, 2015.

[14] Jeffrey G Andrews, Stefano Buzzi, Wan Choi, Stephen V Hanly, Angel Lozano, Anthony CK Soong, and Jianzhong Charlie Zhang. What will 5G be? *IEEE Journal on selected areas in communications*, 32(6):1065–1082, 2014.

[15] GSMA Intelligence. Understanding 5G: Perspectives on future technological advancements in mobile. *White paper*, pages 1–26, 2014.

[16] Shanzhi Chen and Jian Zhao. The requirements, challenges, and technologies for 5G of terrestrial mobile telecommunication. *IEEE communications magazine*, 52(5):36–43, 2014.

[17] Yong Niu, Yong Li, Depeng Jin, Li Su, and Athanasios V Vasilakos. A survey of millimeter wave communications (mmWave) for 5G: opportunities and challenges. *Wireless networks*, 21(8):2657–2676, 2015.

[18] Ian F Akyildiz, Josep Miquel Jornet, and Chong Han. Terahertz band: Next frontier for wireless communications. *Physical Communication*, 12:16–32, 2014.

[19] Massive MIMO for next generation wireless systems, author=Larsson, Erik G and Edfors, Ove and Tufvesson, Fredrik and Marzetta, Thomas L. *IEEE communications magazine*, 52(2):186–195, 2014.

[20] Diego Kreutz, Fernando MV Ramos, Paulo Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76, 2014.

[21] Etsi - standards for nfv - network functions virtualisation | nfv solutions. `https://www.etsi.org/technologies/nfv`. (Accessed on 05/14/2021).

[22] Juliver Gil Herrera and Juan Felipe Botero. Resource allocation in NFV: A comprehensive survey. *IEEE Transactions on Network and Service Management*, 13(3):518–532, 2016.

[23] Tarik Taleb, Konstantinos Samdanis, Badr Mada, Hannu Flinck, Sunny Dutta, and Dario Sabella. On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration. *IEEE Communications Surveys & Tutorials*, 19(3):1657–1681, 2017.

[24] NGMN Alliance. Description of network slicing concept. *NGMN 5G P*, 1(1), 2016.

[25] Ibrahim Afolabi, Adlen Ksentini, Miloud Bagaa, Tarik Taleb, Marius Corici, and Akihiro Nakao. Towards 5G network slicing over multiple-domains. *IEICE Transactions on Communications*, 100(11):1992–2006, 2017.

[26] Nathan F Saraiva De Sousa, Danny A Lachos Perez, Raphael V Rosa, Mateus AS Santos, and Christian Esteve Rothenberg. Network service orchestration: A survey. *Computer Communications*, 142:69–94, 2019.

[27] Diego Gambetta et al. Can we trust trust. *Trust: Making and breaking cooperative relations*, 13:213–237, 2000.

[28] Kellyn Pot'vin, Anand Akela, Gokhan Atil, Bobby Curtis, Alex Gorbachev, Niall Litchfield, Leighton Nelson, and Pete Sharman. Cloud lifecycle management. In *Expert Oracle Enterprise Manager 12c*, pages 153–186. Springer, 2013.

[29] Ana Juan Ferrer and Enric Pages i Montanera. The role of SLAs in building a trusted cloud for Europe. In *IFIP International Conference on Trust Management*, pages 262–275. Springer, 2015.

[30] SLALOM project. `http://slalom-project.eu/`.

[31] Amazon Compute Service Level Agreement. `https://aws.amazon.com/compute/sla/`.

[32] Rackspace Cloud Service Level Agreement. `https://www.rackspace.com/information/legal/cloud/sla`.

[33] Aggelos Kapoukakis, Stella Kafetzoglou, Georgios Androulidakis, C Papagianni, and Symeon Papavassiliou. Reputation-Based Trust in federated testbeds utilizing user experience. In *2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, pages 56–60. IEEE, 2014.

[34] Eleni Kamateri, Nikolaos Loutas, Dimitris Zeginis, James Ahtes, Francesco D'Andria, Stefano Bocconi, Panagiotis Gouvas, Giannis Ledakis, Franco Ravagli, Oleksandr Lobunets, et al. Cloud4SOA: A semantic-interoperability PaaS solution for multi-cloud platform management and portability. In *European Conference on Service-Oriented and Cloud Computing*, pages 64–78. Springer, 2013.

[35] Ana Juan Ferrer, Francisco HernáNdez, Johan Tordsson, Erik Elmroth, Ahmed Ali-Eldin, Csilla Zsigri, RaüL Sirvent, Jordi Guitart, Rosa M Badia, Karim Djemame, et al. OPTIMIS: A holistic approach to cloud service provisioning. *Future Generation Computer Systems*, 28(1):66–77, 2012.

[36] Wolfgang Ziegler, Ming Jiang, and Kleopatra Konstanteli. OPTIMIS SLA framework and term languages for SLAs in cloud environment. *OPTIMIS Project Deliverable*, 2, 2011.

[37] Open Grid Forum (OGF), Web Services Agreement Specification (WS-Agreement). `http://ogf.org/documents/GFD.192.pdf`.

[38] M. Comuzzi, C. Kotsokalis, G. Spanoudakis, and R. Yahyapour. Establishing and monitoring slas in complex service based systems. In *2009 IEEE International Conference on Web Services*, pages 783–790, July 2009.

[39] RobertoG Cascella, Lorenzo Blasi, Yvon Jegou, Massimo Coppola, and Christine Morin. Contrail: Distributed application deployment under sla in federated heterogeneous clouds. In *The Future Internet*, pages 91–103, 05 2013.

[40] Danilo Ardagna, Michele Ciavotta, Giovanni Paolo Gibilisco, Riccardo Benito Desantis, Giuliano Casale, Juan F. Pérez, Francesco D'Andria, and Román Sosa González. *QoS Assessment and SLA Management*, chapter 4, pages 35–46. Springer International Publishing, Cham, 2017.

[41] V. Casola, A. D. Benedictis, M. Rak, and U. Villano. Sla-based secure cloud application development: The specs framework. In *2015 17th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC)*, pages 337–344, Sep. 2015.

[42] Erkuden Rios, Massimiliano Rak, Eider Iturbe, Wissam Mallouli, et al. Sla-based continuous security assurance in multi-cloud devops. 2017.

[43] Omar Abdel Wahab, Jamal Bentahar, Hadi Otrok, and Azzam Mourad. A survey on trust and reputation models for Web services: Single, composite, and communities. *Decision Support Systems*, 74:121–134, 2015.

[44] Hien Trang Nguyen, Weiliang Zhao, and Jian Yang. A trust and reputation model based on bayesian network for web services. In *2010 IEEE International Conference on Web Services*, pages 251–258. IEEE, 2010.

[45] Zaki Malik and Athman Bouguettaya. Rateweb: Reputation assessment for trust establishment among web services. *The VLDB Journal*, 18(4):885–911, 2009.

[46] Chung-Wei Hang, Anup K Kalia, and Munindar P Singh. Behind the curtain: Service selection via trust in composite services. In *2012 IEEE 19th International Conference on Web Services*, pages 9–16. IEEE, 2012.

[47] Hamdi Yahyaoui. A trust-based game theoretical model for web services collaboration. *Knowledge-Based Systems*, 27:162–169, 2012.

[48] Saurabh Ganeriwal, Laura K Balzano, and Mani B Srivastava. Reputation-based framework for high integrity sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 4(3):1–37, 2008.

[49] Yi Ren, Vladimir I Zadorozhny, Vladimir A Oleshchuk, and Frank Y Li. A novel approach to trust management in unattended wireless sensor networks. *IEEE Transactions on Mobile Computing*, 13(7):1409–1423, 2013.

[50] Yan Lindsay Sun, Wei Yu, Zhu Han, and KJ Ray Liu. Information theoretic framework of trust modeling and evaluation for ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 24(2):305–317, 2006.

[51] Yating Wang, Ray Chen, Jin-Hee Cho, Ananthram Swami, Yen-Cheng Lu, Chang-Tien Lu, and Jeffrey JP Tsai. CATrust: Context-aware trust management for service-oriented ad hoc networks. *IEEE Transactions on Services Computing*, 11(6):908–921, 2016.

[52] Wenjia Li and Houbing Song. ART: An attack-resistant trust management scheme for securing vehicular ad hoc networks. *IEEE transactions on intelligent transportation systems*, 17(4):960–969, 2015.

[53] Talal H Noor, Quan Z Sheng, Lina Yao, Schahram Dustdar, and Anne HH Ngu. CloudArmor: Supporting reputation-based trust management for cloud services. *IEEE transactions on parallel and distributed systems*, 27(2):367–380, 2015.

[54] Paul Manuel. A trust model of cloud computing based on Quality of Service. *Annals of Operations Research*, 233(1):281–292, 2015.

[55] Wei Tang and Zheng Yan. Cloudrec: A mobile cloud service recommender system based on adaptive qos management. In *2015 IEEE Trustcom/BigDataSE/ISPA*, volume 1, pages 9–16. IEEE, 2015.

[56] Zheng Yan, Xueyun Li, Mingjun Wang, and Athanasios V Vasilakos. Flexible data access control based on trust and reputation in cloud computing. *IEEE transactions on cloud Computing*, 5(3):485–498, 2015.

[57] Safwan Mahmud Khan and Kevin W Hamlen. Hatman: Intra-cloud trust management for Hadoop. In *2012 IEEE Fifth International Conference on Cloud Computing*, pages 494–501. IEEE, 2012.

[58] Sepandar D Kamvar, Mario T Schlosser, and Hector Garcia-Molina. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th international conference on World Wide Web*, pages 640–651, 2003.

[59] Zheng Yan, Peng Zhang, and Athanasios V Vasilakos. A security and trust framework for virtualized networks and software-defined networking. *Security and communication networks*, 9(16):3059–3069, 2016.

[60] Kostas Giotis, Maria Apostolaki, and Vasilis Maglaris. A reputation-based collaborative schema for the mitigation of distributed attacks in SDN domains. In *NOMS 2016-2016 IEEE/IFIP Network Operations and Management Symposium*, pages 495–501. IEEE, 2016.

[61] Dan Marconett and SJ Ben Yoo. Flowbroker: A software-defined network controller architecture for multi-domain brokering and reputation. *Journal of Network and Systems Management*, 23(2):328–359, 2015.

[62] Xing Su, Minjie Zhang, Yi Mu, and Kwang Mong Sim. Pbtrust: A priority-based trust model for service selection in general service-oriented environments. In *2010 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, pages 841–848. IEEE, 2010.

[63] Anurag Garg and Roberto Battiti. The reputation, opinion, credibility and quality (ROCQ) scheme. 2004.

[64] Marshall Brinn, Nicholas Bastin, Andy C Bavier, Mark Berman, Jeffrey S Chase, and Robert Ricci. Trust as the Foundation of Resource Exchange in GENI. *EAI Endorsed Trans. Security Safety*, 2(5):e1, 2015.

[65] Chunsheng Zhu, Hasen Nicanfar, Victor CM Leung, and Laurence T Yang. An authenticated trust and reputation calculation and management system for cloud and sensor networks integration. *IEEE Transactions on Information Forensics and Security*, 10(1):118–131, 2014.

[66] Saurabh Kumar Garg, Steve Versteeg, and Rajkumar Buyya. Smicloud: A framework for comparing and ranking cloud services. In *2011 Fourth IEEE International Conference on Utility and Cloud Computing*, pages 210–218. IEEE, 2011.

[67] Tolga Kaya and Cengiz Kahraman. Multicriteria renewable energy planning using an integrated fuzzy VIKOR & AHP methodology: The case of Istanbul. *Energy*, 35(6):2517–2527, 2010.

[68] Hamzeh Mohammd Alabool and Ahmad Kamil Mahmood. Trust-based service selection in public cloud computing using fuzzy modified VIKOR method. *Australian Journal of Basic and Applied Sciences*, 7(9):211–220, 2013.

[69] Geoff Coyle. The Analytic Hierarchy Process (AHP). Practical Strategy: Structured Tools and Techniques. Open Access Material, 2004.

[70] Helnet. `http://helnet.eu/index.php/home-2/`. (Accessed on 04/22/2021).

[71] Netmode testbed. `http://www.netmode.ntua.gr/main/index.php?option=com_content&view=article&id=103&Itemid=83`. (Accessed on 04/22/2021).

[72] Nitos - nitlab - network implementation testbed laboratory. `https://nitlab.inf.uth.gr/NITlab/nitos`. (Accessed on 04/22/2021).

[73] S. Kubler, J. Robert, W. Derigent, A. Voisin, and Y. Le Traon. A state-of the-art survey & testbed of fuzzy ahp (fahp) applications. *Elsevier Expert Systems with Applications*, 65:398–422, 2016.

[74] Geni. `http://www.geni.net/`.

[75] CLoudLab Project. `https://www.cloudlab.us/#top`.

[76] Virtual Wall - FED4FIRE+. `https://www.fed4fire.eu/testbeds/virtual-wall/`.

[77] NETMODE - FED4FIRE+. `https://www.fed4fire.eu/testbeds/netmode/`.

[78] Dechouniotis, Dimitrios and Athanasopoulos, Nikolaos and Leivadeas, Aris and Mitton, Nathalie and Jungers, Raphaël M and Papavassiliou, Symeon. Edge Computing Resource Allocation for Dynamic Networks: The DRUID-NET Vision and Perspective. *Sensors*, 20(8):2191, 2020.

[79] Avgeris, Marios and Spatharakis, Dimitrios and Dechouniotis, Dimitrios and Kalatzis, Nikos and Roussaki, Ioanna and Papavassiliou, Symeon. Where there is fire there is SMOKE: a scalable edge computing framework for early fire detection. *Sensors*, 19(3):639, 2019.

[80] Ahsan Adeel, Mandar Gogate, Saadullah Farooq, Cosimo Ieracitano, Kia Dashtipour, Hadi Larijani, and Amir Hussain. A survey on the role of wireless sensor networks and iot in disaster management. In *Geological disaster monitoring based on sensor networks*, pages 57–66. Springer, 2019.

[81] Nick Feamster, Jennifer Rexford, and Ellen Zegura. The Road to SDN: An Intellectual History of Programmable Networks. *SIGCOMM Comput. Commun. Rev.*, 44(2):87–98, April 2014.

[82] Etsi - multi-access edge computing - standards for mec. `https://www.etsi.org/technologies/multi-access-edge-computing`. (Accessed on 07/06/2021).

[83] Steve Huckle and Martin White. Socialism and the blockchain. *Future Internet*, 8(4):49, 2016.

[84] Linux Foundation. Hyperledger Fabric. =https://github.com/hyperledger/fabric, Jun 2020.

[85] ETSI. Open Source MANO. `http://osm.etsi.org`.

[86] G. Papathanail, A. Pentelas, I. Fotoglou, P. Papadimitriou, K. V. Katsaros, V. Theodorou, S. Soursos, D. Spatharakis, I. Dimolitsas, M. Avgeris, D. Dechouniotis, and S. Papavassiliou. MESON: Optimized Cross-Slice Communication for Edge Computing. *IEEE Communications Magazine*, 58(10):23–28, 2020.

[87] ETSI GS NFV 002. Network Functions Virtualisation (NFV); Architectural Framework. https://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_NFV002v010101p.pdf, 2013.

[88] ETSI. Mobile-Edge Computing (MEC); Service Scenarios. = https://www.etsi.org/deliver/etsi_gr/MEC/001_099/017/ 01.01.01_60/gr_MEC017v010101p.pdf, 2015.

[89] I. Dimolitsas, D. Dechouniotis, V. Theodorou, P. Papadimitriou, and S. Papavassiliou. A Multi-Criteria Decision Making Method for Network Slice Edge Infrastructure Selection. In *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, pages 1–7, 2020.

[90] I. Fotoglou, G. Papathanail, A. Pentelas, P. Papadimitriou, V. Theodorou, D. Dechouniotis, and S. Papavassiliou. Towards Cross-Slice Communication for Enhanced Service Delivery at the Network Edge. In *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, pages 22–28, 2020.

[91] L. Bondan, M. F. Franco, L. Marcuzzo, G. Venancio, R. L. Santos, R. J. Pfitscher, E. J. Scheid, B. Stiller, F. De Turck, E. P. Duarte, A. E. Schaeffer-Filho, C. R. P. d. Santos, and L. Z. Granville. FENDE: Marketplace-Based Distribution, Execution, and Life Cycle Management of VNFs. *IEEE Communications Magazine*, 57(1):13–19, 2019.

[92] P. D. Maciel, F. L. Verdi, P. Valsamas, I. Sakellariou, L. Mamatas, S. Petridou, P. Papadimitriou, D. Moura, A. I. Swapna, B. Pinheiro, and S. Clayman. A Marketplace-based Approach to Cloud Network Slice Composition Across Multiple Domains. In *2019 IEEE Conference on Network Softwarization (NetSoft)*, pages 480–488, 2019.

[93] Riccardo Guerzoni, Ishan Vaishnavi, David Perez Caparros, Alex Galis, Francesco Tusa, Paolo Monti, Andrea Sganbelluri, Gergely Biczók, Balasz Sonkoly, Laszlo Toka, Aurora Ramos, Javier Melián, Olivier Dugeon, Filippo Cugini, Barbara Martini, Paola Iovanna, Giovanni Giuliani, Ricardo Figueiredo, Luis Miguel Contreras-Murillo, Carlos J. Bernardos, Cristina Santana, and Robert Szabo. Analysis of end-to-end multi-domain management and orchestration frameworks for software defined infrastruc-

tures: an architectural survey. *Transactions on Emerging Telecommunications Technologies*, 28(4):e3103, 2017.

[94] V. K. Rathi, V. Chaudhary, N. K. Rajput, B. Ahuja, A. K. Jaiswal, D. Gupta, M. Elhoseny, and M. Hammoudeh. A Blockchain-Enabled Multi Domain Edge Computing Orchestrator. *IEEE Internet of Things Magazine*, 3(2):30–36, 2020.

[95] James Kempf, Sambit Nayak, Remi Robert, Jim Feng, Kunal Rajan Deshmukh, Anshu Shukla, Aleksandra Obeso Duque, Nanjangud Narendra, and Johan Sjöberg. The nubo virtual services marketplace. *arXiv preprint arXiv:1909.04934*, 2019.

[96] Sambit Nayak, Nanjangud C Narendra, Anshu Shukla, and James Kempf. Saranyu: Using smart contracts and blockchain for cloud tenant management. In *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pages 857–861. IEEE, 2018.

[97] Github - consensys/quorum: A permissioned implementation of ethereum supporting data privacy. `https://github.com/ConsenSys/quorum`. (Accessed on 01/08/2021).

[98] Kiril Antevski and Carlos J Bernardos. Federation of 5g services using distributed ledger technologies. *Internet Technology Letters*, page e193.

[99] G. Carrozzo, M. S. Siddiqui, A. Betzler, J. Bonnet, G. M. Perez, A. Ramos, and T. Subramanya. AI-driven Zero-touch Operations, Security and Trust in Multi-operator 5G Networks: a Conceptual Architecture. In *2020 European Conference on Networks and Communications (EuCNC)*, pages 254–258, 2020.

[100] Boubakr Nour, Adlen Ksentini, Nicolas Herbaut, Pantelis A Frangoudis, and Hassine Moungla. A blockchain-based network slice broker for 5g services. *IEEE Networking Letters*, 1(3):99–102, 2019.

[101] Gabriel Antonio F Rebello, Igor D Alvarenga, Igor J Sanz, and Otto Carlos MB Duarte. Bsec-nfvo: A blockchain-based security for network function virtualization orchestration. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2019.

[102] Home - opnfv. `https://www.opnfv.org/`. (Accessed on 06/14/2021).

[103] R. V. Rosa and C. E. Rothenberg. Blockchain-Based Decentralized Applications for Multiple Administrative Domain Networking. *IEEE Communications Standards Magazine*, 2(3):29–37, 2018.

[104] Konstantinos Papadakis-Vlachopapadopoulos, Ioannis Dimolitsas, Dimitrios Dechouniotis, Eirini Eleni Tsiropoulou, Ioanna Roussaki, and Symeon Papavassiliou. Blockchain-based slice orchestration for enabling cross-slice communication at the network edge. In *2020 IEEE 20th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pages 140–147. IEEE, 2020.

[105] Konstantinos papadakis / blockchain based network service marketplace · gitlab. `https://gitlab.com/cpapad/bcpaper`. (Accessed on 06/17/2021).

[106] Github - jdimol/cs_osm_client: Osm client for enabling cs interaction. `https://github.com/jdimol/cs_osm_client`. (Accessed on 06/17/2021).

[107] Performance - hyperledger blockchain performance reports. `https://hyperledger.github.io/caliper-benchmarks/fabric/performance/`. (Accessed on 06/17/2021).

[108] 6.19. openstack load testing report — performance_docs 0.0.1.dev196 documentation. `https://docs.openstack.org/developer/performance-docs/test_results/openstack_load/index.html#create-neutron-networks`. (Accessed on 06/17/2021).

[109] Lotfi A Zadeh. Information and control. *Fuzzy sets*, 8(3):338–353, 1965.