CASE: A Context-Aware Storage Placement and Retrieval Ecosystem

GANESH CHANDRASEKARAN1, BOOPATHI RAMASAMY2, PRANAY DHONDI1, PANKAJ BHIMRAO THORAT1, AND RAJESH CHALLA1, (Member, IEEE)

1Samsung Electronics Research and Development Institute India, Bangalore (SRIB), Bengaluru 560037, India
2Samsung Electronics, Suwon 16677, South Korea
Corresponding author: Ganesh Chandrasekaran (ganesh.c@samsung.com)

This work was supported in part by Samsung R&D Institute India, Bangalore (SRIB), and in part by Samsung Electronics, HQ, Suwon.

ABSTRACT Emerging cloud-native technologies, such as container-runtime and container-orchestrator offer unprecedented agility in developing and running applications, especially when combined with microservice-style architecture. Several existing 5G-Telecom network products such as element Management System (e-MS), 5G-Core and 5G-Access are being redesigned to fit the microservice paradigm. Cloud environment allows enterprises to scale their application on-demand with minimum cost; however, it is often difficult to use containers without sacrificing many benefits that cloud-native technology offers. The e-MS is characterized to orchestrate 5G network elements (5GNEs) deployed nationwide, and systematically store terabytes of stateful data per second. Containers are characterized to have an ephemeral state, hence ‘statefulness’ aspect of e-MS makes orchestration complex. In this paper, different challenges around stateful storage selection, content placement and content retrieval operations within e-MS microservices are described. To overcome these challenges, to this end, we propose CASE - A Context-Aware Storage placement and retrieval Ecosystem - which enables context-based operations to be intrinsically supported by the underlying e-MS application. Our approach has been designed to maintain the location-independent philosophy of cloud-native by associating context information directly to 5GNE rather than fixed storage entities, thereby ensuring scalability. Through simulation with real data-set obtained from one of the world’s largest terrestrial telecom operator, we show that based on such location-independent context information, CASE with e-MS can facilitate high performance despite dynamic 5GNE count agility, stateless e-MS replication and stateful storage scaling, while not posing a significant signaling burden on the cloud environment.

INDEX TERMS 5G networks, cloud native, microservices, containers, context-aware services, communication networks and telecommunication network topology.

I. INTRODUCTION
Recently, container-based microservice architecture has gained substantial attention among several 5G telco vendors and operators. Many challenges of traditional monolithic architecture applications are tackled by microservices paradigm [1]. However, to leverage the benefits of the microservices style, one needs to use technologies aligned with the characteristics of microservices for its deployment [1]. The cloud native, container-runtime and container-orchestrator has become a popular deployment format for microservice applications.

Several commercial 5G Telecommunication network products including element management system (e-MS), radio access central unit (CU), radio access distributed unit (DU) and 5G core network functions are already being redesigned to fit the microservice paradigm as containers [2]–[4]. These also align with the 5G standardization bodies such as 3GPP and European Telecommunication Standards Institute (ETSI). Based on 5G standards in 3GPP release-15 [5], 3GPP defined service-oriented architecture (SOA) for the control plane of 5G core. Furthermore, on release 17 [5], 3GPP expanded the SOA to the 5G core user plane, which changed the interface between the user plane and the control plane to a service-based interface (SBI). This new approach to 5G core would allow the user plane network functions to leverage the benefits of cloud native [5]. In addition,
ETSI group on Network Functions Virtualization (NFV) unveiled its first specification enabling containerized network functions (CNF) to be managed in an NFV framework on ETSI GS NFV-IFA 040 [6]–[8]. The ETSI-NFV continues its efforts to make further progress in the specifications for the NFV release-4 feature on “cloud-native network functions and container infrastructure management” [7].

Microservice architecture is often conflated with SOA, as both of them involve in building reusable and scalable individual components that can be consumed by other applications. Irrespective of any chosen architecture, on-demand dynamic scaling and high availability are important non-functional requirements that the carrier-grade service providers must fulfill [8]. The stateless microservice applications deployed as independent container, allow dynamically on-demand replication. However, the same is not true for stateful microservices. Containers are characterized by having an ephemeral state and hence deploying a replicated set of stateful microservice applications requires coordination between replicas to keep them synchronized for retrieval and update i.e. the “state” aspect introduces more complex to the system [9].

The e-MS is characterized to monitor, orchestrate and manage key functionalities such as fault, configuration, accounting, performance and security (FCAPS) of 5G Network Elements (5GNEs) in RAN and core domain deployed nationwide. In cloud environments, monitoring by centralized management system (i.e. e-MS) is critical for operational efficiency. With telco-specific products such as e-MS pursuing cloud-based deployment using microservices architecture, these solutions will have to tackle multiple problems than typical-based monolithic software [9]. Based on data originating from 5GNEs, e-MS systematically handles and stores several Terabytes of stateful data per second. The e-MS handling such scale of stateful data, and aligning to fit with the microservice paradigm has called for new research issues and architectural design changes. In telco-grade solutions, strict reliability, scaling and availability are critical, so finding the right solution for the new architecture is a challenging and essential task [9].

In this paper we introduce CASE — Context-Aware Storage placement and retrieval Ecosystem — which enables context-based operations to be intrinsically supported by the underlying application for data placement and retrieval functions. Our approach has been designed to maintain the location independent philosophy of cloud native by associating context information directly to 5GNE rather than to explicit fixed storage node. In addition, to support any changes in the number of 5GNEs/storage sets, our proposed scheme ensures location-independent context information exchanged among e-MS replicas. Contextual profile information on storage node such as, traffic load, compute load, geographical location, and information on change in number of managed 5GNE are exchanged when and where necessary. This, in turn, reduces context information staleness, reduces operational complexities and thereby enabling dynamic scaling of stateless e-MS application on demand. CASE framework also facilitates the possibility to add and remove stateful storage nodes onto existing live e-MS without hindering the scope of current state with zero downtime, thereby, opportunity to expand service logic and functionalities independently as required for the operator’s business need.

The rest of this article is structured as follows. A discussion of some important background information and related work is presented in Section II. We illustrate our problem and elaborates on the CASE framework in Section III. We also elaborate on the data placement and resolution in the same section. In Section IV, we propose our context publication strategy employed on the e-MS. We also define and formulate the problem on the data storage selection and efficient bloom filter construction. Strategy for context retrieval is discussed in Section V. Implementation and experimental results are depicted in Section VI to evaluate our CASE approach and prove the efficiency of our model against competing solutions. Finally, in Section VII, we conclude our work and open the door for future research directions.

II. RELATED WORK

Microservice architecture has significant benefits in terms of flexibility and scalability, compared with the traditional monolithic architecture. Microservice has been drawing great attentions in designing the software solution due to faster delivery, greater autonomy, and improved scalability in developing and deploying the applications in the containerized environment [10]–[16]. In terms of deployment of a stateful microservice, which stores information externally to some data store, needs additional consideration such as placement of microservice and data stores for effectively reducing the latency involved in redirecting application traffic to its destined microservice to data retrieval from the data store. There are few proposals that focuses [17]–[19] on the microservice placement to achieve latency reduction. However, these approaches focus only on the deployment of stateless microservices closer to the edge of the network for latency reduction.

Intelligent container placement schemes have been proposed that assist in selection and distribution of services towards the edge data centers [20]. Unlike traditional 5G telco data centers, these are characterized by sporadic resources availability, mobility, and increased flexibility. Proposals made on distributed redundant placement frameworks, resource management algorithms and server selection algorithm [21] and [22] are primarily done to bring microservice applications on cloud closer to the devices i.e. network edge. The aforementioned algorithms and design are most suitable with fog/edge computing especially when combined with Internet of Things type of data. These proposals [20]–[22] focus on enabling ability to have stateless services deployed on the fly but do not share light on how heavy stateful services can be managed on a containerization and micro-service 5G telco cloud environment.
Context based storage and retrieval have been studied for some time by the research community, spanning a wide spectrum such as, Content Distribution Networks (CDNs), Information/Content Centric Networking (ICN/CCN) and other areas. However, our work with CASE is one of the first attempt to introduce such context based framework to microservice architecture principles in cloud native domain. Looking onto other related works, in [11] author investigates microservice coordination among multiple clouds to enable seamless and real-time responses to service requests from mobile users. The objective of this work is to devise the optimal microservice coordination scheme which can reduce the overall service delay with low costs. However, no focus was given to stateful information retrieval and processing with these microservices. In [12] author proposes, predictive auto-scaling orchestration system for cloud-native telecom microservices. Such predictive mechanism with workload forecasting was proposed to enable auto-scaling and thereby improve the performance of the orchestration system. However, in this paper, the investigation was only performed for stateless replicas. Stateful replication management is one of key hurdle when aligning with microservice architecture. In [13], authors report about context based retrieval that can be improved by getting user interventions such as user feedback, preferences and machine context based data, but this loses its efficiency with millions of users and context profiles start being part of the system.

All of the above-mentioned approaches utilize either user behavior or user preferences or both to construct a contextual profile, and some form of user intervention is required. In addition, none of these approaches discuss how to effectively perform context based stateful storage and retrieval on a large-scale distributed system. However, in this paper, contextual profile on storage node such as, traffic load, compute load, geographical location etc. is exchanged and used without any user intervention for effective storage and retrieval operations. With the existing frameworks, it is not possible to expand the system components independently without downtime. System components include stateful storage, stateless processing elements and support for various types of input (e.g. 5G network elements types). In the remainder of this paper we discuss CASE framework for contextual information retrieval that potentially addresses all the challenges mentioned above.

### III. CASE FRAMEWORK

In any software architecture, application needs to be able to handle stateful request towards storage data nodes. It is common practice to have a single unified data store in an application architecture, however, there can be cases when the scope of the application is too large or due to business needs, arises the need for multiple data storage node (DSN) with isolation. One such instance is characterized with e-MS, where the application needs to be able to handle multiple DSN and resolve request as with specific 5GNE. Each 5GNE is identified uniquely across 5G infrastructure by its identifier (xID). In 5G infrastructure, there can be a large number of 5GNEs based on the size of coverage area and the number of cells, resulting in the need to handle massive stateful dataset. It then becomes practically impossible to store and handle all 5GNE’s datasets on a single DSN, therefore arises a need to isolate and handle the entire volume of stateful datasets on multiple DSNs as per 5GNE’s xID.

Generic method of manually stitching xIDs with specific fixed DSN will hinder microservice and cloud native paradigm. Thereby, complicating 5GNE count agility, optimal data placement on DSNs and dynamic scaling of e-MS on demand. However, e-MS aligned with CASE will be able to seamlessly handle the content request of xID towards the respective DSN thereby maintaining location independent philosophy of cloud native by associating context information directly to 5GNE’s xID rather than to explicit fixed storage node.

Fig. 1 shows the architecture of 5G telco in two-tiers, infra tier and orchestration tier. 5G infra tier deployed nationwide hosts elements that are part of access and core domain as Virtual Network Functions (VNF), Containerized Network Function (CNF) and Physical Network Functions (PNF). The e-MS in orchestration tier is responsible for life cycle management and monitoring of VNFS, CNFS and PNFs deployed nationwide. Due to the distributed nature of 5G Infra elements and the need to handle large traffic volume in a short time span, e-MS has several DSN spread geographically with various dimensions and scale (Fig. 1(a) shows the DSN address in which the location is referenced as AZ-Arizona and NJ-New Jersey etc.) Each DSN on its own supports redundancy, high availability and stores unique stateful information on various 5GNE infra elements. The e-MS as microservice architecture is composed of stateful (for storage) and stateless (for logic execution) services. DSN pool containers belong to stateful service while the remaining e-MS are stateless. Unlike stateful services, stateless services can be used interchangeably and can easily be scaled on demand. Seamless arrival of requests from various 5GNEs in infra tier to e-MS on orchestration tier for stateful processing adds complexity. CASE ecosystem alignment with e-MS can perform context-based operations to be intrinsically supported by the underlying application for data placement and retrieval functions. Two main functions of the CASE ecosystem are:

#### A. DATA PLACEMENT

When new 5GNE (e.g. among RAN or core domain) with unique xID is added on to e-MS management pool, the request is load balanced to a specific e-MS container replica and thereby two types of action are performed. First, gathering context information on all DSN in its storage pool such as, traffic load, compute load, geographical location etc., based on which ideal DSN for 5GNE is selected. Second, e-MS container replica does dynamic stitching between specific 5GNE xID and selected DSN by sending out Updated Context Publication (UCP) message as in Fig. 1(b) to all other e-MS replicas. In contrast, when a new DSN is added, it is
simply added to e-MS storage pool for future 5GNE content placement considerations. With both the above actions such as, information gathering and publication, context based processing thrives as key differentiator for data placement and resolution operations.

B. DATA RESOLUTION

Request originating from existing 5GNE xID towards e-MS is load balanced to a specific container replica. This e-MS container replica then processes the request with content management table (CMT) as in Fig. 1(a) and determines the stateful operation towards the correct DSN with very high accuracy. This stateful operation request can lead to any further basic persistent storage operations such as create, read, update, and delete (CRUD) towards DSN.

There could be several replicas of e-MS microservice containers scaled based on demand, each request from 5GNE is load-balanced to one replica for processing. Each e-MS container replica has an identical CMT. The CMT contains information associating xIDs with one DSN node address through which content requests can be routed towards a DSN holding the requested data. As in Fig. 1(a) CMT table has two columns, the DSN address and its associating bloom-filter. CMT is updated based on UCP messages received from other replicas, a new UCP message is generated only when a 5GNE is added or removed from e-MS’s management pool. The aim of the UCP is to eliminate managed 5GNE infra pool staleness and disseminate any variation in knowledge about 5GNE’s information on a DSN, without revealing explicit information about 5GNE/storage node’s identities.

CASE is designed to facilitate accurate decision-making during data resolution/retrieval, so as to ensure that 5GNE’s request with a xID always reaches its specific DSN. We follow a probabilistic data structure approach to xID’s data placement and resolution. The main focus and novelty of our approach lies in the stateful handling on cloud native environment in two stages, and which together serves to facilitate context-aware resolution. The following sections elaborate on these two stages. In a later section, we explain how the DSN selection during 5GNE count agility is used to intelligently position storage, in such a way that ensures well-balanced network load while minimizing the handling time.

IV. UPDATED CONTEXT PUBLICATION

As depicted in Fig. 1, the 5G infra structure tier has 5GNEs in the form of VNF, CNF and PNF distributed among both access and core domain. A change in the number of VNF and CNF elements can happen for various reasons, such as, provisioning new 5G service, de-provisioning existing 5G service, scaling domain specific VNF elements based on traffic load, network slice commissioning, network slice decommissioning and physical infrastructure failure etc. Similarly, change in number of PNFs can be due to interference optimization, improve coverage area, network upgrades and physical failure etc.

Any change in the state of 5GNEs will be notified to e-MS replica in the form of an event or alarm. An UCP with a modified bloom filter message is sent out by an e-MS microservice replica whenever it detects a change in the number of 5GNEs being managed. In contrast, when a new
DSN is added, it is simply appended to the e-MS storage pool for future 5GNE content placement considerations. Thereby to take effect, UCP message with new bloom filter field is sent out to all other e-MS replicas.

As depicted in Fig. 1(b), the UCP message contains three fields: ‘message type,’ ‘DSN address,’ and ‘bloom filter.’ The ‘DSN address’ simply indicates a unique reference to the specific data node i.e. DSN. The ‘message type’ indicates that the message is a ‘UCP.’ The last field in the UCP is a bloom filter [14], which is a probabilistic data structure that allows for a set of elements to be represented by a single space-efficient bit string. Computationally-efficient logic-based set membership queries can then be performed on it to determine if an element is a member of the set it represents.

In our case, the set of elements represented by the bloom filter is the set of xIDs that have their storage node selected as specific DSN address. Set membership for queries are performed on 5GNE xIDs with bloom filter along each row in CMT to determine respective DSN address. As can be seen, 5GNE identifier is not sent in the UCP, thus completely decoupling DSNs from 5GNE identities. The following sections elaborate on the DSN selection, bloom filter construction and UCP processing employed on the e-MS replicas.

### A. DSN SELECTION

When a new 5GNE (e.g. among access or core domain) with unique xID is added on to e-MS management pool, the request is load balanced to a specific e-MS container replica and thereby DSN selection process is triggered. DSN selection process involves gathering context information on all DSNs in its storage pool such as, traffic load, and geographical location etc., based on which ideal DSN for 5GNE is selected. DSN selection algorithm also considers various context information of 5GNE such as its locality, user density and capacity etc. DSN selection algorithm aims to assign 5GNE to DSN with decreased data access latency and to optimally distribute traffic load across the 5G tier.

Data Access latency (DAL) as in Eq.1 represents the effective time delay taken for 5GNE to access or change data in DSN and assuming e-MS identified a suitable DSN for a request ‘i,’ then let \( d_{ij} \) indicate delay in stateful request \( req_i \) that is made from 5GNE to DSN \( j \).

\[
DAL_i = \sum_j x_{ij} \cdot d_{ij} \tag{1}
\]

where, \( x_{ij} \in \{0,1\} \) is a binary variable. If 5GNE’s request \( req_i \) data belongs to DSN \( j \) then \( x_{ij} = 1 \), otherwise \( x_{ij} = 0 \). \( \sum_j x_{ij} = 1 \) guarantees that \( req_i \)’s stateful data lies on one of the DSNs.

The problem can be formulated as to minimize DAL with representation below as shown in Eq.2.

\[
\min \sum_i DAL_i \tag{2}
\]

For a large-scale telco operator, e-MS has a massive number of 5GNEs in its management pool. Depending on the size of telco infrastructure, number of cells can range over 3,000,000 in number and each 5GNE in access domain can oversee one or more cells (up to 32) as per various local factors. One cell can generate up to 4GB of data per day, i.e. each 5GNE has sizable unique dataset and configurations to be handled on-demand. This implies a massive volume of stateful requests to be handled at the orchestrator domain i.e. e-MS. Managing data on a single DSN will lead to traffic overload, increase data access latency and thereby affecting 5G business performance. Context-based optimized distribution of data across multiple DSN will decrease traffic per DSN, thereby decreasing data access latency. Low latency data access is one of the key performance metric that help in achieving the 5G telco requirements. The algorithm that distributes 5GNE traffic across multiple DSNs, is crucial as uneven distribution can directly affect latency. In Algorithm.1 we propose an algorithm that aim to efficiently distribute incoming 5GNE traffic across a group of Data Store Node (DSN) servers.

In the 5G telco infrastructure, the number of incoming requests depends on the number of consumers, type of consumer, and capacity of 5GNE. Using the population density of an area and market share, we can estimate consumer request count. Assume \( F(X) \) denotes the population density of a location. Not all consumers use the services equally. Data usage per user is different across rural and urban areas, so depending on the locality (e.g. village, town, city) the traffic can vary. As consumers are mobile, the number of consumers change over time, therefore, considering the type of location (e.g. residential, mall, cinema hall, bay area, tech parks, highway, etc.) can help in estimating traffic for each 5GNE (e.g. RAN) over duration of a day. Parameters like locality and location type can help estimate traffic pattern realistically. Thereby, \( G(X) \) and \( H(X) \) denotes serving locality and location type respectfully. In addition, traffic handling capability is directly proportional to 5GNE capacity. Let \( K(X) \) denote 5GNE capacity (e.g. Macro or Pico as part of 5G access domain). Using the above functions, we can estimate the approximate 5GNE traffic per hour. The approximate load per day for 5GNE can then be calculated by aggregating the hourly data and represented as \( M(X) \) below Eq.3.

\[
M(X) = p_1 \cdot \delta_1 \cdot F(X) + p_2 \cdot \delta_2 \cdot G(X) + p_3 \cdot \delta_3 \cdot H(X) + p_4 \cdot \delta_4 \cdot K(X) \tag{3}
\]

\( p_1, p_2, p_3, p_4 \) are constants that convert function values to traffic per day. \( \delta_1, \delta_2, \delta_3, \delta_4 \) are weightage to each function that adds to 1.

Actual traffic for 5GNE may be different compared to calculated approximate traffic in Eq.3. To estimate more realistic request arrival rate, we use a compound function using previous traffic details as shown in below Eq.4. Assuming \( L(X) \) denotes average traffic for 5GNE per day,

\[
FutureTrafficEstimate = (1 - q) \cdot M(X) + L(X) \cdot q \tag{4}
\]

where, ‘q’ is a weightage variable and changes over time. Initially, the value of ‘q’ is one and gradually decreases and
Algorithm 1 DSN Selection Algorithm

Assumptions:
F(X): Consumer density
G(X): 5GNE Locality (e.g. Village/Town/Type)
H(X): 5GNE Location proximity (e.g. Residential/Mall/Highway)
K(X): 5GNE Type (e.g. Macro/Micro/Pico)
L(X): Average request handling load of 5GNE per day
M(X): Estimated 5GNE load per day
T: Total number of DSNs
D(X, Y): Denote network delay between 5GNE X and DSN Y
D(X, Y): Network delay between 5GNE X and DSN Y

where, qi changes over time as traffic from 5GNE is learned.

Assuming 'c' number of 5GNEs are already allocated to DSN. Then, 5GNE that minimizes composite function as below Eq.6,

\[ \delta * A(i) + (1 - \delta) * D(X, i) \]  

where, \( \delta \) is function weightage that depends on 5GNE type. After e-MS replica decides a suitable DSN for the new 5GNE, UCP preparation takes place. The first step in UCP preparation is bloom filter construction, the following section elaborates on the bloom filter construction process.

B. BLOOM FILTER CONSTRUCTION

In order to produce the bloom filter message, each element, \( e_i \), \( 1 \leq i \leq n \), in the set, \( S \), is hashed \( k \) times using \( k \) independent hash functions. The resulting bloom filter bit array representing \( S \) is formed of \( m \) bits, that is given by A. Broder et al. [15] as Eq.7 below.

\[ m \geq \frac{n \log_2(1/f_p)}{\ln(2)} \]  

where, \( f_p \) is the false-positive probability, i.e., the probability that a test for element membership of element \( e_j \notin S \) is positive when it should be negative. In Fig. 2 we show a simple example of the construction of a bloom filter by e-MS during UCP creation, and also the testing of elements for membership during content retrieval operations. In this example, the bloom filter, \( b_{f1} \), is constructed by performing the bitwise OR operation on various 5GNEs i.e. \( xID_1 \), \( xID_4 \) and \( xID_5 \). If e-MS replica wants to add a new 5GNE onto this list, then a UCP message is created with the resulting bloom filter and is disseminated to all active replicas. Each e-MS replica receiving the UCP checks, and updates the information on its CMT. When a request originating from 5GNE \( xID_4 \) reaches an e-MS replica, element membership testing is done with all entries on CMT to find its corresponding DSN address. This is achieved by performing a bitwise AND operation with \( b_{f1} \). In the example, the result of the bitwise AND operation of \( xID_4 \) and \( b_{f1} \) yields \( xID_4 \), indicating that \( xID_4 \) is a member element of the bloom filter \( b_{f1} \). Thereby based on CMT e-MS identifies that all stateful content related to \( xID_4 \) is stored with DSN1 (\( xID_1 \_AZZ271Ha \)). If some other element \( xID_7 \) were to be tested for membership in \( b_{f1} \), the result would be negative, since the bitwise AND operation of \( xID_7 \) with \( b_{f1} \) does not yield \( xID_7 \). In this case, e-MS will move onto performing element membership testing with other entries on CMT until a match is found. From A. Broder et al. work [15] we know that the optimal number of hash functions, \( k \), is given by below Eq.8.

\[ k = \log_2 \left( \frac{1}{f_p} \right) \]  

Whereby optimality implies that \( m \) in Eq.7 is minimized subject to meeting the target false-positive probability, \( f_p \). From the above equations Eq.1-7, we can observe two key characteristics of bloom filters that make them ideally suited for our application:

1) The size of a bloom filter is independent of the size of the elements. This means it is possible to use very long xIDs i.e. support for large number of 5GNEs...
without increasing the size of the bloom filter. For instance, in each DSN entry, to create a bloom filter for 100,000 xIDs, given that the optimal number of hash functions is used, then to achieve a false-positive probability of 2%, approximately 8 bits per element will need to be used. Thereby, giving a total bloom filter size of approximately 100 kB. If the size of each 5GNE xID is assumed to be 512-bits, then without bloom filters, conveying information about 100,000 xIDs would require approximately 6.4 MB of space.

2) The number of hash functions used has a bearing on the computational complexity of the bloom filter, since the number determines the bits that need to be read to test for membership. From Eq.7 we can deduce that the optimal number of hashes grows only linearly with the number of bits per element, b, where b is given by the ratio m/n.

In very rare case where a false-positive occurs, the e-MS replica identifying this issue need to repeat the element membership testing with entries of CMT. We anticipate that this won’t have an adverse effect on the performance of our proposed mechanism.

C. UCP PROCESSING STRATEGY

In an e-MS microservice architecture, communication between various services is characterized into synchronous and asynchronous interaction. With synchronous communication when an e-MS replica receives an UCP message, it will first confirm receipt by sending back an acknowledgement. Since each e-MS replicas are independent of each other, the request can be sent out in parallel to all neighbor replicas. In asynchronous communication, all replicas listen to the message broker to receive the UCP message and then process it accordingly. When an e-MS replica creates/uploads an existing UCP it publishes the messages to a message broker. Instead of sending direct messages, it will send event details to the message broker along with the altered UCP payload. Consumer e-MS replicas will identify what the event is and take action accordingly. This brings in loose coupling and high-availability among stateless e-MS replicas, any scaling performed on e-MS or on e-MS restart can still reconstruct its CMT based on reading all events from message broker. With CASE we do not specifically enforce or recommend a specific method of communication, making this discussion out of our scope.

When a new DSN is added to e-MS’s storage pool, it creates a new UCP message (with a new bloom filter field) and is sent out to all other stateless e-MS microservice replicas. After which, this new DSN will be considered for any further placement decision made for adding new 5GNEs onto management pool. Update on existing UCP (i.e. add or remove elements from existing bloom filter) is done when the number of managed 5GNEs change. UCP arriving onto all neighboring e-MS first checks if CMT has an entry with DSN’s address that is same as the arrived UCP. If the match was not found, it will then proceed to create a new record in the CMT, copying the DSN address and the bloom filter information from UCP. In case the entry was found then
arrived UCP’s bloom filter is updated on existing entry of CMT. As a result, the content retrieval process (discussed in the next section) from any previous 5GNEs will still be able to route requests towards DSN holding 5GNE stateful data. Handling the removal of 5GNE from the e-MS management pool is not discussed here in detail, but one could imagine using a counting bloom filter [15] to handle deletions from the bloom filters.

V. CONTEXT RETRIEVAL
When a request originates from 5GNE, by default it reaches onto the cloud’s load balancer or domain name system. Depending on the policy of load balancer the request is then forwarded onto a stateless e-MS replica for processing. The stateless e-MS replica depending on the processing strategy might need to perform CRUD operations on its stateful information held on a specific DSN. Stateless e-MS replica find the respective DSN address based on the requested 5GNE ID. This e-MS container replica processes the request with content management table (CMT) as in Fig. 1(a) and determines a specific stateful operation towards the correct DSN with high accuracy.

All stateless e-MS container replica has identical CMT. The CMT contains bloom filter information associating xIDs with only one DSN node. As in Fig. 1(a) CMT table has two columns, DSN address and its associating bloom-filter. As in Fig. 2, Element membership testing is done with the request’s 5GNE ID and all entries on CMT to find a positive outcome. Once a corresponding DSN address is determined, a respective stateful operation is performed on DSN by a stateless e-MS replica.

VI. PERFORMANCE EVALUATION
In this section, we evaluate CASE and verify how it can perform in a distributed system to provide optimal 5GNE traffic distribution among DSNs, with minimal overhead and minimum time to resolve a 5GNE’s request. In our first use case, we validate CASE by assuming the request arrival from each 5GNE to follow mathematical distributions, in specific uniform distribution and normal distribution. In these cases, the physical location of 5GNEs across the country are spread out at random. In our second use case, the request arrival is based on a real data-set obtained from one of the world’s largest terrestrial mobile phone network operator measured by the number of subscriptions. In this case, the physical location of 5GNEs and DSNs are placed based on the strategic planning and positioning of real operator data-set.

A. SIMULATION SCENARIO
Our simulator used for evaluation consist of two parts, one is responsible for sending 5GNE events towards the orchestration domain and other is a server-side orchestrator application that takes the role of e-MS. Here the e-MS handles the requests by load balancing it to a stateless replica and eventually to a respective DSN based on the type of evaluating framework. The following section describes various type of the evaluation frameworks. Our simulation was run on 2 bare-metal servers composing 80CPU and 500GB memory each. The server side e-MS component has several stateless replicas scaled based on demand, each stateless replica has 100 threads to process the incoming requests from the load-balancer. The load balancer simply balances the requests across multiple e-MS instances using its buffer queue size and current active processing threads. Load balancer increases e-MS instances i.e. processing thread count if the sum of load is greater than 90% of capacity for consecutive 5 monitor cycles. It also automatically scales down the replica if sum of the load is less than 20% of capacity for 5 monitoring cycles. Depending on the type of framework under evaluation, the number of stateful DSN is decided.

For evaluation of CASE, we have identified several key performance indicators (KPIs) such as average data access latency, efficient utilization of DSNs, average overhead signaling and compute resource consumption of e-MS. Reduced data access latency and efficient distribution of traffic across DSNs are the two key metrics that our DSN-selection algorithm aimed to solve. Metrics like signaling overhead and compute resource consumptions are aimed at challenging the CASE framework as whole. Many times in a realistic telco infrastructure, requests arise from 5GNE that need to be processed with a predetermined Quality of Service (QoS) (e.g. latency) else the request would be marked as failed. A ratio of failed requests is one such metric that can be captured based on the average data access latency KPI described earlier.

For frameworks evaluation and comparison, we have considered a set of realistic frameworks that is used to evaluate against CASE.

1) Unified DSN framework (UDF), is similar to the existing monolithic approach where all of e-MS’s stateful traffic is handled on a single DSN. Depending on the load of request arrival and processing duration, e-MS will queue up the excess requests for future processing. 2) Range based DSN framework (RBDF), assuming the maximum number of 5GNEs in the 5G infra is already known, a fixed set of 5GNE ranges are pre-determined for each DSN. Such range-based configuration information is pre-fed into all stateless e-MS replicas, thereby, based on request arrival from a specific 5GNE the resolution is successfully performed by each e-MS stateless replica. This RBDF would be implemented by the operational team, who has sufficient knowledge on forecasting the subscribers traffic and plan the telco infra structure accordingly. This increases the operational cost substantially and would also not comply with cloud native principles.

3) Centralized relational positioning DSN framework (CPDF), here each 5GNE gets assigned a DSN based on a specific policy (e.g. round-robin) or based on user invention. However, the mapping information of relation between 5GNE’s xID and DSN’s address is centrally positioned on either one of fixed DSN or memcache or other storage location. This relation information will be centrally accessed by all stateless e-MS replicas for every 5GNE request. CPDF is similar to using the state of the art commercial database
proxies, this increases the overhead and do not take advantage of context information to make a more sophisticated decision.

As discussed earlier, part of simulator is responsible for sending 5GNE events towards orchestration domain, in addition to generating random events it also provisions new 5GNE and adds it to the e-MS’s orchestration domain. With UDF, provisioning of new 5GNE and de-provisioning of existing 5GNE doesn’t affect the system in anyway because it has only one DSN node, and all the requests from 5GNE use the same DSN. In the case of RBDF, it is assumed that all mapping between 5GNEs and DSNs are preconfigured.

Unlike RBDF, for the CPDF, during 5GNE provisioning the simulator assigns DSN in a round robin fashion. However, for every request from 5GNE, the e-MS stateless replica retrieves DSN mapping information from centralized location and only then directs the requests to DSN. CASE framework as mentioned above stores 5GNE-DSN mapping in bloom filter. To construct bloom filter, parameters like the number of hashes, maximum 5GNE count per DSN, false positivity value are needed. For simulation of bloom filter hashes, maximum 5GNE of 10 million and false positivity of 0.0001 is used. For any new provisioning and de-provisioning of 5GNE, CASE sends UCP messages to each of e-MS replicas asynchronously. In the next section, we depict the actual performance of CASE against various frameworks.

### B. EVALUATION

Performance evaluation of CASE with simulator tries to emulate and use real-world telco deployment conditions. We assume that the entire network is divided into multiple circles and each circle has several 5GNEs. Our simulator creates a pool of scheduled threads for each circle and schedules various event from 5GNE based on delay. For such event generation, the simulator simulates various types of distributions, such as, uniform distribution events, normal distribution events and also re-uses real data trace obtained from one of world’s largest operator to generate events.

1) EVALUATION BASED ON UNIFORM DISTRIBUTION AND NORMAL DISTRIBUTION

Our simulator supports having a uniform time differences between two events of a 5GNE and provides uniform distributed time-interval to each simulated 5GNE. Similarly, it supports 5GNE event generation type (i.e. time differences between two events) based on normal distribution and also it provides normal distribution required mean and standard deviation value for creating time delay between events from 5GNE.

Fig.3 (a-f) depicts various KPIs of CASE against other frameworks. As in Fig. 3 (a-b), the average latency of data access, CASE is performing over 150% better than other frameworks. This is because CASE can identify the DSN address for a 5GNE by performing membership testing on the same e-MS stateless replica. Similarly, in the case of UDF and RBDF, the DSN addressed is also resolved on the same e-MS stateless replica but the traffic load is not distributed optimally. All the 5GNE traffic with UDF goes to the same DSN, which leads to queuing up of requests, as the number of 5GNE requests increases. For this reason,
it can be observed that UDF performs better than CASE with lower number of 5GNEs in the network. RBDF also has the 5GNE and DSN mapping predetermined, this would mean that requests are not always send to the closest DSN. Fig 3 (b) depicts a case where the request arrival follows a normal distribution, hence the frequency of 5GNE request arrival get large during a specific period time when the bell curve of normal distribution goes up. For this reason, in other frameworks the effective latency of data access is lower than uniform distributed traffic.

KPI capturing effective utilization of DSNs in the network is depicted in Fig.3 (c-d). This metric is computed by taking Standard Deviation on the load of each DSN in the network. It is a measure of the amount of variation or dispersion of a set of values, i.e. lower value would mean that traffic load is well distributed among DSNs. From Fig.3 (c-d) it can be seen that for both uniform and normal distribution the UDF is always zero. This is due to the fact that UDF has only one DSN for the entire 5GNE. Comparing with other frameworks, it can be seen that CASE is performing over 500% better than RBDF and over 100% better than CPDF. Irrespective of the number of 5GNE or request load, CASE always provides an optimized load distribution. Whereas with RBDF and CPDF frameworks, because of the absence of context information during 5GNE positioning the distribution of load is uneven, the load distribution for RBDF and CPDF is worst when the traffic in the network is less (i.e. small 5GNE) similarly, when the number of 5GNEs increase, both RBDF and CPDF starts performing well, this is because with higher request load to the network the effective utilization of DSNs increase.

Fig.3 (e-f) captures the compute resource utilization of each framework. CASE consumes more resources in terms of both CPU and memory usage, as element membership testing for every 5GNE request has to be done with rows of CMT. Additional compute resources are required for the dissemination of UCP and management of CMT. The number of rows in CMT is equal to the number of DSN, in a realistic telco system there is only a small finite number of DSN hence this doesn’t affect the data access latency. This metric captures maximum memory utilization in each framework. With UDF there is no additional data stored in the stateless replica so this can be considered as base value and it can be seen that CASE utilization is comparable with UDF and others.

The average overhead signaling metric captures the additional time taken in adding a new 5GNE/DSN into management pool and thereby also resolving request from 5GNEs. Average overhead with various framework is tabulated below as shown in Table 1. For UDF and RBDF the overhead is close to zero because for every new 5GNE/DSN addition or for every 5GNE request processing, the stateless replica does not need to communicate with any external entity. Thereby, no additional context information is exchanged among replicas leading to zero overhead. However, in CPDF, for both new 5GNE/DSN addition and for 5GNE request processing the stateless replica need to communicate with external entity. For this reason, with CPDF, the overhead introduced is directly proportional to the number of requests to the system. In CASE, overhead is only introduced for new 5GNE/DSN addition.

**TABLE 1. Average overhead introduced with provisioning of 5GNEs and resolution of requests.**

| Overhead (msec) | Number of 5GNEs |
|----------------|-----------------|
|                | 5000 25000 75000 100000 |
| CPDF           | 18.2 25.98 24.83 27.984 |
| CASE           | 1.1 0.829 0.448 0.2077 |
addition thereby performance of CASE is over 15 times better than the CPDF.

2) EVALUATION BASED ON OPERATOR DATA-TRACE
FROM FIELDS
The request arrival is based on the real data-set obtained from one of the world’s largest terrestrial 4G mobile operator. Here the physical location of 5GNEs and DSNs are placed based on the strategic planning and positions extracted from the real operator data-set. The dataset used only includes 530 5GNEs over a span of 155,707 km², spread across multiple cities, and has different localities and types. The macro, micro and pico type 5GNEs are distributed in 62:29:9 ratios. The real dataset includes varying velocities and volumes of data across time. The events are spread over 24 hours and includes, distribution of events e.g., skewed nights, rush morning hours, calm afternoon and chaotic evenings. Over 80 types of events are sent out from 5GNE to the e-MS orchestration system with various types of notification.

Some common notification categories include events such as, optimization events, reallocation events, cell refresh events, configuration events, performance and failure events. The total number of events adds up to over 1.1M, across 24 hours’ span. The resources thread of DSN is minimized due to lesser 5GNEs in the trace. The KPI evaluation for real-data trace is depicted in Fig 4(a-d). For average latency, CASE performs far better than context unaware frameworks, as shown in Fig. 4 (a). One of the benefits of CASE efficacy is its optimal network traffic distribution to DSNs, reflected by its lower deviation of DSN utilization in comparison to the context-unaware approaches, as shown in Fig. 4(b). Furthermore, as shown in Fig. 4(c-d), for compute utilization, CASE is still reasonably comparable with other frameworks.

VII. CONCLUSION
In this paper, we have discussed the feasibility of applying the CASE ecosystem onto cloud-native microservices, in which location-independent context information is efficiently distributed among e-MS replicas to facilitate better decisions during 5GNE request resolution. Bloom filters are periodically constructed to efficiently convey the latest management state of 5G infra tier elements, to which up-to-date context information is used to make decision on a specific DSN and thereby disseminated to other e-MS replicas.

Through simulation with real data-set obtained from one of the world’s largest terrestrial telecom operator, we have shown that CASE can achieve optimized network utilization and data access latency in a 5G telco infrastructure, particularly when 5GNE request arrival is at the critical level. Furthermore, the use of bloom filters ensures that UCP messages do not pose a significant overhead to the network, from the perspective of adding a new 5GNE/DSN into the e-MS pool.

Towards future work, we will extend CASE and further improve the KPIs by introducing cache at stateless replicas, and also we plan to introduce further slicing of 5GNE requests into multiple components. Current slicing of traffic is only based on 5GNE’s xID, with introduction of additional parameters to slice such as, QoS requirement and data type, CASE can resolve 5GNE requests in a more realistic way e.g. based on the type of 5GNE’s event (e.g. performance event, configuration event and also based on type of the DSN (e.g. Cassandra DB, Maria DB etc.).

ACKNOWLEDGMENT
The authors gratefully acknowledge our mentors and colleagues at Samsung, SRIX Network Divisions who gave meticulous and valuable comments on this paper and the anonymous reviewers who spent the valuable time in reviewing our paper.

REFERENCES
[1] N. Dragoni, S. Guilleronzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, “Microservices: Yesterday, today, and tomorrow” in Present and Future Software Engineering. Cham, Switzerland: Springer, 2017, pp. 195–216.
[2] Samsung 5G Core White Paper, Cloud Native 5G Core. Accessed: May 26, 2021. [Online]. Available: https://images.samsung.com/iss/content/samsung/samsung/s5/global/business/networks/insights/white-paper/cloud-native-5g-core/Cloud-Native-Samsung-5G-Core-Volume-2.pdf.
[3] HPE Technical White Paper; Container Platform for Telcos. Accessed: May 26, 2021. [Online]. Available: https://hpe.com/iss/content/hpedan/s5000343enw
[4] Nokia. Blog Containers and the Evolving 5G Cloud-Native Journey. Accessed: May 26, 2021. [Online]. Available: https://www.nokia.com/blog/containers-and-the-evolving-5g-cloud-native-journey/
[5] The 3rd Generation Partnership Project, 3GPP Standard Release 15 and Release 17. Accessed: May 26, 2021. [Online]. Available: https://www.3gpp.org/release-15 and https://www.3gpp.org/release-17
[6] European Telecommunications Standards Institute. ETSI. Accessed: May 26, 2021. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/ NFV-IFA/001_099/040/01_01_601/gs_NFV-IFA040v040101p.pdf
[7] European Telecommunications Standards Institute. ETSI. Release-4. Accessed: May 26, 2021. [Online]. Available: https://docbox.etsi.org/STC/ NFV/Open/Other/ReleaseDocumentation/NFV/21/0000025_NVF_Release _4_Definition_v0_3_0.pdf
[8] Samsung 5G Core White Paper, Cloud Native 5G Core. Accessed: May 26, 2021. [Online]. Available: https://images.samsung.com/iss/content/ samsung/s5/global/business/networks/insights/white-paper/cloud-native-5g-core/Samsung-5G-Core-Vol-2-Cloud-Native-5G-Core.pdf
[9] R. Morabito, “Virtualization on Internet of Things edge devices with container technologies: A performance evaluation,” IEEE Access, vol. 5, pp. 8835–8850, 2017.
[10] A. Balalai, A. Heydarnoori, and P. Jamshidi, “Microservices architecture enables devOps: Migration to a cloud-native architecture,” IEEE Softw., vol. 33, no. 3, pp. 42–52, May/Jun. 2016.
[11] D. Wang, Y. Guo, N. Zhang, P. Yang, A. Zhou, and X. Shen, “Delay-aware microservice coordination in mobile edge computing: A reinforcement learning approach.” IEEE Trans. Mobile Comput., vol. 20, no. 3, pp. 939–951, Mar. 2021.
[12] D.-H. Luong, H.-T. Thieu, A. Outtagarts, and Y. Ghamri-Doudane, “A framework for contextual information retrieval from the WWW,” IASSE, 2005.
[13] B. H. Bloom, “Space/time trade-offs in hash coding with allowable errors,” Commun. ACM, vol. 13, no. 7, pp. 422–426, Jul. 1970.
[14] A. Broder and M. Mitzenmacher, “Network applications of Bloom filters: A survey,” Internet Math., vol. 1, no. 4, pp. 636–646, 2002.
[15] A. Sill, “The design and architecture of microservices,” IEEE Cloud Comput., vol. 3, no. 5, pp. 76–80, Sept./Oct. 2016, doi: 10.1109/MCC.2016.111.
[17] T. Bahreini and D. Grosu, “Efficient placement of multi-component applications in edge computing systems,” in Proc. 2nd ACM/IEEE Symp. Edge Comput., Oct. 2017, pp. 1–11.

[18] S. Wang, M. Zafer, and K. K. Leung, “Online placement of multi-component applications in edge computing environments,” IEEE Access, vol. 5, pp. 2514–2533, 2017.

[19] G. Tato, M. Bertier, E. Rivière, and C. Tedeschi, “Split and migrate: Resource-driven placement and discovery of microservices at the edge,” in Proc. OPODIS, vol. 153, 2019, pp. 1–16.

[20] H. Sami and A. Mourad, “Dynamic on-demand fog formation offering on-the-fly IoT service deployment,” IEEE Trans. Netw. Service Manage., vol. 17, no. 2, pp. 1026–1039, Jun. 2020, doi: 10.1109/TNSM.2019.2963643.

[21] A. Mseddi, W. Jaafar, H. Elbaze, and W. Ajib, “Joint container placement and task provisioning in dynamic fog computing,” IEEE Internet Things J., vol. 6, no. 6, pp. 10028–10040, Dec. 2019, doi: 10.1109/JIOT.2019.2935056.

[22] H. Zhao, S. Deng, Z. Liu, J. Yin, and S. Dustdar, “Distributed redundancy scheduling for microservice-based applications at the edge,” IEEE Trans. Services Comput., early access, Aug. 3, 2020, doi: 10.1109/TSC.2020.3013600.

GANESH CHANDRASEKARAN received the B.E. degree in electronics and communication from the Anna University of Technology, Tamil Nadu, India, in 2011, the M.S. degree in telecommunication from University College London, U.K., in 2013, and the Ph.D. degree from the 5G Innovation Centre, University of Surrey, U.K., in 2016. Since 2017, he has been a Staff Engineer with the Samsung Electronics Research and Development Center, Suwon, South Korea, and Bangalore, India. He has also been an Active Reviewer of IEEE Access, from 2018 to 2021. His research interests include 5G/6G networks, 5G end to end network slicing, network function virtualization, software-defined networking, information-centric networking, software-defined networks, and D2D communication.

BOOPATHI RAMASAMY received the B.E. degree in computer science and engineering from Bharathiar University, Tamil Nadu, India, in 2004. From 2004 to 2007, he worked in embedded system at Flextronics, Bengaluru, India. From 2007 to 2013, he worked on shaping various telecom network products at Huawei Technologies India private Ltd., Bengaluru. Since 2014, he has been working with Samsung Electronics, South Korea, as a Principal Engineer and plays Security System Architect role. His research interests include 5G networking, information-centric networking, network resource management, and cyber security.

PRANAY DHONDI was born in Armoor, Nizamabad, Telangana, India, in 1995. He received the B.Tech. degree in computer science and engineering (CSE) from the Indian Institute of Technology Bombay (IIT Bombay), India. He has worked with the Samsung Electronics Research and Development Institute India, Bangalore (SRIB), as a Software Engineer. His research interests include network management systems (NMS), cloud, and micro-service design.

PANKAJ BHIMRAO THORAT received the master’s degree in computer science and engineering from the Indian Institute of Technology Roorkee, India, in June 2011, and the Ph.D. degree in computer engineering from Sungkyunkwan University, South Korea, in March 2018. Since 2011, he has been with Samsung R&D Institute. His research interests include 5G networking, information-centric networking, network resource management, and cyber security.

RAJESH CHALLA (Member, IEEE) received the Ph.D. degree in computer science and engineering from Sungkyunkwan University, South Korea, in 2019. He is currently an Architecture at the Networks Research and Development Division, Samsung Electronics Research and Development Institute India, Bangalore (SRIB). Since 2003, he has been working with SRIB on mobile commercialization activities and conducting research in next-generation mobile networks. His areas of expertise include IP multimedia subsystem (IMS), android radio interface layer (RIL), Linux device driver development, data plane and NAS layer of LTE, and proprietary Samsung handset platform (SHP) solution. He is a CORD Ambassador at the Open Networking Foundation (ONF) Ambassadors’ Program. He has been a contributor in ONOS/CORD projects. He is also representing Samsung Electronics at CNCF TUG Group. He is focusing on 5G cloud-native network design and orchestration at SRIB. His research interests include 5G network slicing and orchestration, predictive and prescriptive service chaining (SFC), and software-defined networking (SDN).