HORIZON-MSCA-2022-SE-01-01
(MSCA Staff Exchanges 2022)

This project has received funding from the European Union's HORIZON.1.2 - Marie Skłodowska-Curie Actions (MSCA) programme under grant agreement ID 101131481.

distributed ledger technologies and user-driven automation towards self-SOVEREIGN mobile data access in beyond 5G networks

SOVEREIGN

WP2– System Architecture, Requirements and Data

Deliverable D2.2 "SOVEREIGN system architecture and technical specs"

Editor(s):	Dionysis Xenakis (UOA)
Author(s):	Dionysis Xenakis (UOA), A. Tsiota (UOA), N. Pappas (LIU), S. Chatzinotas (LIU), B. Padeanu (IQB), G. Zachos (IT), A. Karvellas (INC), G. Kalpaktsoglou (FOG), M. Dolapsaki (UOA), O. Orza (IQB), A. Plechi (IQB), A. Sotiropoulos (FOG), A. Apostolou (FOG), I. Kontogiannis (FOG), M. Touloumi (FOG), P. Theodoropoulos (UOA), K. Palias (UOA), I. Vittorakis (UOA), E. Doost (IT), I. Stamoulias (FOG), M. Karavasili (LIU), A. Nikkhah (LIU), M. Salimnejad (LIU)
Dissemination Level:	**Public**

Type:	**R**
Version:	1

Project Profile

Contract Number	101131481
Acronym	SOVEREIGN
Title	distributed ledger technologies and user-driven automation towards self-SOVEREIGN mobile data access in beyond 5G networks
Start Date	January 1st, 2024
Duration	48 Months

Document History

VERSIONS

Version	Date	Author	Remarks
0.1	01/09/2024	Dionysis Xenakis (UOA)	Table of Contents
0.2	1/12/2025	B. Padeanu (IQB)	Self-sovereign Identities Section
0.3	30/12/2025	Alex Karvellas (INC)	Pricing Algorithm
0.4	30/12/2025	Ioannis Stamoulias (FOG)	KPIS and methods for the SOVEREIGN Blockchain
1	31/12/2025	Symeon Chatzinotas (ULU)	SOVEREIGN Scenarios, Merging of partner contributions, Final version ,

Executive Summary

The main objective of WP2 is to i) specify the SOVEREIGN access scenarios and business models for self-sovereign mobile data access over natively disintermediated B5G infrastructures, ii) to conceptualize an end-to-end DLT-backed B5G service architecture that enables B5G stakeholders to advertise available resource pools and service capabilities on-the-fly, allowing intelligent nodes to dynamically discover, negotiate, formalize, deliver, and consume them in view of MPO1, iii) to provide a meticulous study on necessary functional upgrades to the 3GPP 5G system architecture and iv) to develop intelligent strategies enabling online service pricing and function chaining in B5G.

For this specific deliverable, the project aims to amalgamize the outputs of Task 2.2 by specifying APIs and methods for the SOVEREIGN DLT-backed B5G platform, providing technical requirements and tools for the various service modules (pricing, decentralized AAA, anonymity services, user-driven AI) and study relevant 5GPP/3GPP standards. Also, the outputs of Task 2.3 are included in this deliverable, by valorizing recent standards and studies for B5G access to detail the SOVEREIGN service architecture, its architectural entities, context, data, APIs, and methods, as well as providing the online asset pricing framework for DLT-backed B5G networks.

Contents

Table of Tables

1. Introduction, Challenges and Access Scenarios

1.1. The need for self-sovereign mobile data access in B5G networks with NTN

The global connectivity ecosystem is entering a new phase of convergence between terrestrial networks (TNs), including 5G, emerging 6G, Wi-Fi 6/7, and LPWAN technologies, and non-terrestrial networks (NTNs) such as LEO, MEO, and GEO satellites, as well as High-Altitude Platform Stations (HAPS). This convergence is widely recognized as a cornerstone of future “always-on” connectivity across land, sea, and air, particularly for mobile, industrial, IoT, and mission-critical services. However, while network technologies themselves are rapidly evolving, the economic and operational models governing access to connectivity remain largely static and fragmented.

Today’s mobile data access is still dominated by long-term subscriptions, national contracts, and provider-specific billing and authentication systems, which are poorly aligned with modern mobility patterns. Users increasingly move across heterogeneous environments, urban and rural areas, islands, airspace, maritime routes, yet connectivity remains bounded by provider-specific coverage and bilateral roaming agreements. Even when connectivity exists, “blind spots” persist, especially in mountainous, remote, or sparsely populated regions, as well as in aircraft cabins and open seas. Roaming, despite improvements in selected regions such as the European Union, remains bureaucratic, costly, and constrained by inter-operator agreements, requiring advance configuration and offering limited flexibility.

At the same time, the emergence of NTNs introduces new access opportunities but also new discontinuities, as terrestrial and non-terrestrial ecosystems operate as isolated commercial and technical silos, each with proprietary pricing, settlement, and access mechanisms. As a result, there is currently no common, transparent mechanism that allows users, or autonomous devices, to dynamically purchase data access from any available provider, terrestrial or non-terrestrial, in real time, based on availability, quality, and cost, without contracts, SIM provisioning, or manual intervention.

This lack of flexibility has significant social and economic consequences. According to the International Telecommunication Union (ITU), more than 2.6 billion people worldwide still lack reliable or affordable Internet access, highlighting persistent digital inequality despite technological progress. The World Economic Forum (WEF) further emphasizes that future connectivity systems must evolve toward self-sovereign, automated, and AI-assisted economic models in order to support inclusive and sustainable development. Meanwhile, the GSMA Mobile Economy Report 2024 anticipates that 6G architecture will natively integrate NTNs, making seamless cross-domain connectivity technically feasible, but economically unresolved.

In summary, while the physical network infrastructure is converging, economic and billing integration is missing. The prevailing subscription-centric model, designed for static users and single providers, is fundamentally incompatible with the dynamic, short-lived, and location-dependent connectivity demands of Beyond-5G environments.

Hence, mobile data access in pre-6G systems is static and lacks the flexibility required to support on-the-fly service creation on top of the abundant edge resources coexisting under the numerous 5G service domains. Another critical issue relates to how the key 5G stakeholders (including end users) shall position themselves against the complexity, size, and peculiar characteristics of the emerging 5G market. On the one hand, the 5G service providers should comply with the newest regulations and operating standards (e.g., EU General Data Protection Regulation - GDPR), to attain a minimum service coverage over large geographical areas and to serve as anchor points for supporting external OTT services while meeting the QoS, or QoE, requirements set per user. Adding to this, the MNOs have increasingly become responsible for authenticating and charging their subscribers to third-party services, using dedicated core network units (e.g., Authentication Server Function - AUSF) that are potential single points of failure and targets of distributed denial-of-service (DDoS) attacks [1]. 5G service providers with large-scale coverage inevitably offer "canned" (and thus non-personalized) service contracts (e.g., monthly data usage plan at a fixed price) to geographical regions with diverging characteristics (e.g., demographic density, network topology, available technology, user profiles), boosting the risk of increased customer churn and investments of low added value.

On the other hand, fully personalized service consumption dictates that end users share their personal preferences with their service provider(s) in a persistent and typically eponymous fashion (i.e., social media identities, physical network identifiers, service credentials linked to their user identity, and location data) [3]. From a technological viewpoint, those features are necessary for seamless service discovery/advertisement, service negotiation and parameterization, pricing, and online service optimization (including mobility management and QoE provisioning). However, the increased awareness raised by the end users on recent technological advancements enabling service decentralization and enhanced user privacy, e.g., blockchain-backed systems and anonymity services [6], [7], urge the 5G service providers to revisit the way they implement their services. Besides, end users are more skeptical of the numerous ways through which the 5G service providers collect and distribute their subscribers’ data to third-party brokers [8]. Having familiarized with the concepts of sharing/circular economies, additional skepticism has been raised by consumers on whether the mobile data market can be regulated by a handful of nationwide MNOs that dominate the worldwide mobile data market.

It readily follows that end-to-end (e2e) service provisioning, continuity, charging, user authentication and data privacy across the plethora of service domains thriving under the 5G umbrella, urge for the design and wide deployment of fully-distributed mobile data access models (and mechanisms) that will enable flexible creation and user-centric service consumption on top of the numerous *network assets* available at the 5G network edge. Service delivery of this type goes beyond the "modus operandi" of existing systems and standards, necessitating steps forward from the current network-controlled user-assisted service provisioning model dominating the pre-5G service market, i.e., service control by the "home" network. On-the-fly user-driven network-assisted mobile service consumption, which is specifically designed to exploit the superior performance features of 5G and Beyond networks, is the future of mobile data access.

To address this gap, the SOVEREIGN platform proposes a fully decentralized, blockchain-based micro-payment infrastructure that enables real-time, pay-as-you-go mobile data access across heterogeneous terrestrial and non-terrestrial networks. Rather than replacing existing networks or radio technologies, SOVEREIGN operates strictly at the economic and settlement layer, providing a neutral and open marketplace for data access.

The platform will be built on an EVM-compatible blockchain and introduce a special-purpose crypto-economic system that supports fine-grained micro-payments for data consumption. Through smart contracts, the platform enables users, devices, and IoT endpoints to dynamically purchase connectivity from multiple providers, MNOs, MVNOs, satellite operators, or aggregators, without long-term contracts, SIM provisioning, or centralized intermediaries. Payments are calculated automatically based on actual consumption and settled transparently on-chain, while high-frequency interactions are aggregated off-chain to ensure scalability.

SOVEREIGN functions as an open “enabler” infrastructure, allowing providers to publish service offers in real time (e.g., price per MB, latency, coverage, capacity) and compete in a transparent marketplace. Users interact with the system via a mobile wallet or API, which autonomously selects the optimal provider based on cost, quality of service (QoS), and historical performance. Importantly, the platform does not perform authentication or radio access control; these remain under the providers’ domain. Instead, SOVEREIGN focuses on how payments are executed, verified, and finalized across administrative and technological boundaries.

An AI-assisted decision layer enhances the platform by enabling:

- automatic discovery of available terrestrial and non-terrestrial providers,
- dynamic price negotiation without human intervention,
- autonomous execution of payments via smart contracts.

This approach creates an open, competitive, and provider-neutral data market, where reliable providers are economically rewarded, and users retain sovereignty over how, when, and from whom they purchase connectivity. The platform will be delivered as a minimum viable product (MVP) demonstrating the feasibility of AI-driven, tokenized connectivity markets, aligned with prior research on blockchain-based network resource trading and micro-payment architectures.

In this deliverable, we provide an in-depth study on how the blockchain technology can be utilized to turn the today’s evidently under-organized and vastly heterogeneous mobile data network, where different operators, regional network and content providers, user-installed access points and end terminals, share no interest in improving the networking experience of end users belonging outside their subscriber’s whitelist, to a fully decentralized, dynamic and competitive (by consensus) market where different stakeholders have clear incentives to improve mobile data access and content consumption of mobile users falling within their coverage. To this end, we propose and investigate the potential of a new operator-less (in the sense of fixed term contracts signed offline) mobile data access model where the key 5G stakeholders can trade, share and consume mobile edge network assets (access to the internet, spectrum, processing, storage, local content etc.) in a fully decentralized, instantaneous and anonymous fashion. The main vehicle used to achieve this is a specialized crypto-currency platform that we propose, a platform enabling all 5G stakeholders to act both as network asset servers and clients (consumers) by integrating disruptive new blockchain mechanisms that are specifically designed to effectively support the demanding 5G and Beyond mobile data access use scenario.

The proposed crypto-currency platform can be implemented in the form of two specialized smart contracts (SCs) that remain unchanged for the system lifetime and are designed so as to i) democratize the block validation process, by employing Delegated Proof of Stake (DPoS) over the multi-billion nodes constituting the heterogeneous 5G and Beyond mobile data network infrastructure, ii) scale the transactions capacity of the system to multi-millions transactions per second, enabling support of multi-billion mobile data network devices, and iii) safeguard the anonymity of service peers that jointly operate in both the network and blockchain domains. Our design is not specific to any blockchain framework but requires support of smart contracts (SC) by the blockchain infrastructure.

To enable a more comprehensive understanding of the concepts and solutions discussed, we focus on the sharing and trading of mobile video content. We choose to do so, provided that mobile video will account for over 74% of the total mobile data traffic by 2024 [9]. Besides, other network assets, such as Internet/local connectivity, spectrum, processing, and storage, can be utilized in the process of improving the mobile video delivery service. For example, an asset server can deliver the requested mobile video content by simply providing Internet connectivity towards a free-of-charge video content provider (e.g., YouTube). Alternatively, an asset server can mobilize local storage resources to cache mobile video content during off-peak periods, downscale the pre-cached video content using MEC transcoding upon request, and deliver it on demand with minimum backhaul link usage.

Next, we identify and provide specific protocols to address three main practical challenges towards the smooth integration of blockchain-backed service support into the baseline operation of 5G and Beyond mobile data networks: the consensus protocol, the transactions capacity, and the blockchain anonymity challenges. All three challenges are detailed in sections 1.2.1, 1.2.2, and 1.2.3, respectively.

1.2. Challenges for self-sovereign mobile data access in B5G networks with NTN that drive technical requirements

1.2.1. BG5-ready Consensus Protocols and Democratization of the Chain

In the heart of every blockchain system lies the distributed *consensus protocol*, which ensures that all participating nodes share a common view on the transaction history recorded in the public ledger (i.e., the blockchain) [10], [11]. The distributed consensus protocol specifies message passing across the consensus network, local decision making at each node, and the methodology used to append new blocks of transactions to the blockchain (at least 51% should follow the respective blockchain update). Wide acceptance of a blockchain-backed mobile content delivery platform by the key 5G stakeholders, necessitates the implementation of a scalable consensus protocol enabling active engagement of the vast number of 5G service peers (end devices, core network entities, servers, etc.) to the consensus process while taking into consideration their heterogeneous functional capabilities (e.g., limited access to spectrum, processing, storage and energy sources).

The consensus protocol should also incentivize the 5G service peers to engage with both the blockchain maintenance process and the actual 5G service implementation, e.g. by electing block sealers in the blockchain domain and by sharing their available resource pools in the network domain, respectively. Existing consensus protocols typically consider homogeneous capabilities across the consensus network or assume the formation of clusters with stable connectivity to the Internet (e.g., mining pools in BTC), or delegate blockchain maintenance to a privileged set of consensus nodes that are fixed and a priori known. On the other hand, mobile content trading in 5G and Beyond networks dictates the design of forward-thinking incentive engineering mechanisms that encourage active engagement of the myriads 5G service components in both the blockchain and network domains (e.g. by striking a good balance between blockchain and network domain reward mechanisms), but also enforce trusted operation of the key blockchain actors through the deployment of credible yet sustainable penalty mechanisms.

1.2.2. Multi-million Transactions Throughput

Existing crypto-currency platforms, including Bitcoin (BTC) [6] and the SC-enabled ETH platform [7], lack the scalability to carry out the transactions volume required by a *pay-per-chunk* mobile video delivery model. According to official YouTube statistics [12], by Q3 of 2020, video consumers generate billions of views on the platform to watch over one billion (1B) hours of video every day, with 70% of the YouTube watch time coming from mobile devices. Similarly, 100 million hours of video are consumed on Facebook every day, with 96% of users accessing content through their mobile device [13] (an approximate number of 1.59 billion users per day). Calculated on the basis of only a few of the popular social media platforms like YouTube and Facebook, the total number of *video consumption requests per second* in a world-wide scale for 2020 is estimated to reach 100-200K, highlighting the vast *transactions throughput* (i.e. the number of finalized transactions per second) requirements that a blockchain-backed mobile video trading platform has in 5G and Beyond service scenarios.

This number is, in fact, a tight lower bound if we consider that i) the average watch time of mobile video consumers is distributed across multiple content delivery platforms, ii) mobile video consumers typically generate multiple video views with shorter video watch times, and iii) mobile video consumers are required to handover across multiple 5G service peers due to user mobility. The combined impact of these effects stresses the need for a blockchain-backed mobile content delivery platform that is capable of supporting multi-million transactions per second (TPS). In comparison, by Q3 2020, widely-used crypto-currency platforms like BTC and ETH currently process an average of up to 7 [14] and 17 [15] TPS, respectively. Besides, even with the use of traditional payment services, support of multi-million peer-to-peer (P2P) payments in fiat currencies is currently impossible [16], [17].

1.2.3. Blockchain anonymity services

Although the wide public considers that existing crypto-currency platforms enable anonymous payments in a fully decentralized fashion, an increasing body of academic studies and analysis tools have revealed that the identity of blockchain users can be exposed using simple deanonymization attacks [17], [18], [19], [20]. Accordingly, a new market of anonymity-enhancing services has emerged [21], [22], [23], [24], [25], [26], [27], [28], [29]. The so-called *mixing services* typically implement coin mixing (or tumbling) protocols that are specific to the platform for which they have been designed. In most cases, mixing protocols target to the support of *k-anonymity*. Such protocols take as input the same (fixed) number of coins from individual users (a.k.a. *payers*) and redistribute them to a designated set of output addresses (a.k.a. the *payees*' addresses) in a way that makes it difficult for third parties to link the payment to a specific payer-payee pair. Depending on the mixing protocol, different levels of anonymity can be provided. For example, some protocols preserve anonymity from outside observers but enable the mixing server to be inferred on the input/output mixing pairs [22].

Preserving user privacy and anonymity in a blockchain-backed 5G service ecosystem poses unique challenges that have not been previously addressed by the blockchain community. Existing mixing protocols cannot cope with a scenario where the 5G service peers are in physical proximity to deliver/consume mobile data services using both their network-level (MAC, IP, IMEI, CGI, SSID, etc.) and blockchain-level (public addresses) identifiers. Network-level interaction combined with online posting of transaction at the blockchain-level (to initiate, continue, or conclude the 5G service delivery) enables potential adversaries to couple the two types of identifiers, opening the door to successful deanonymization attacks (Figure 1). Accordingly, joint network-level and blockchain-level interaction hinders the effectiveness of fundamental user privacy protection measures, such as the use of fresh blockchain identifiers per new transaction. The *network/blockchain ID coupling problem* is further exacerbated by any event that increases the volume of transactions between the service peers, including handover events due to user mobility or the employment of micro-payments.

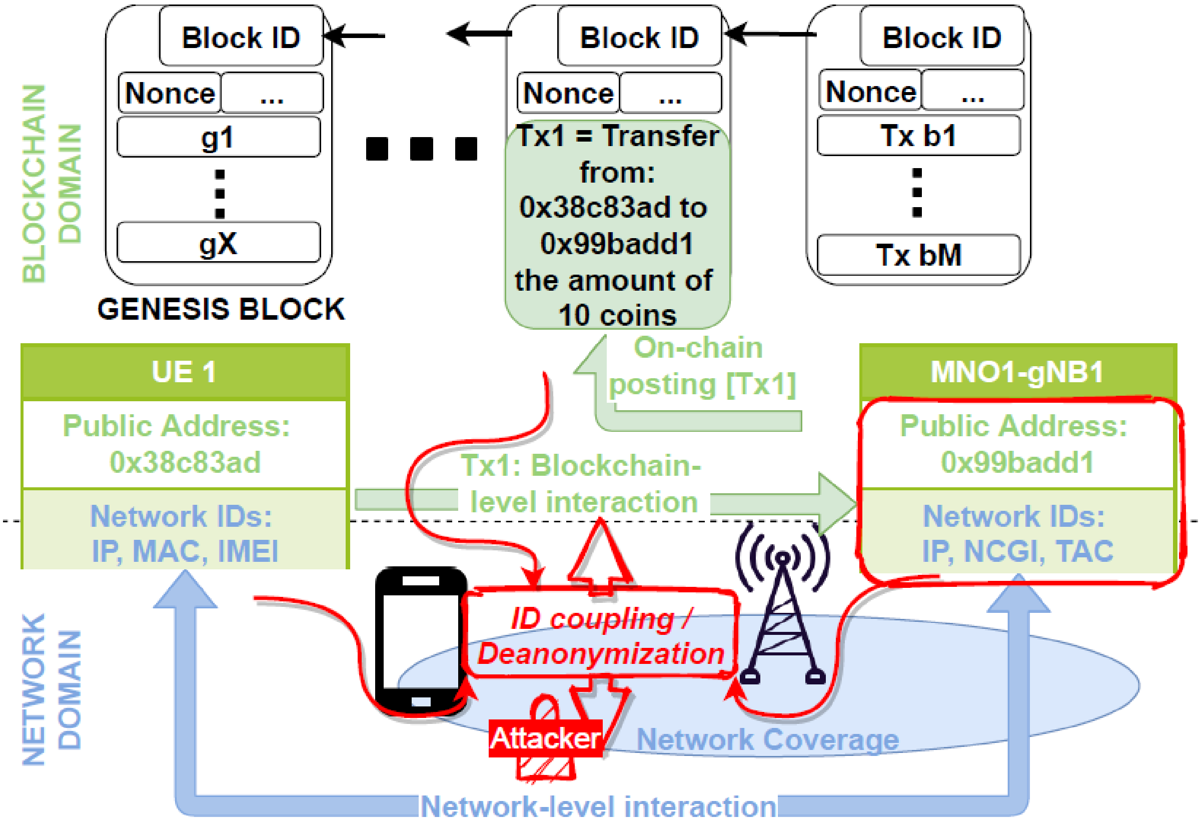

BLOCKCHAIN DOMAIN
Block ID
Nonce
...
g1
gX
GENESIS BLOCK
Block ID
Nonce
...
Tx1 = Transfer from: 0x38c83ad to 0x99badd1 the amount of 10 coins
Block ID
Nonce
...
Tx b1
Tx bM
On-chain posting [Tx1]
UE 1
Public Address: 0x38c83ad
Network IDs: IP, MAC, IMEI
Tx1: Blockchain-level interaction
MNO1-gNB1
Public Address: 0x99badd1
Network IDs: IP, NCGI, TAC
ID coupling / Deanonymization
Attacker
Network Coverage
NETWORK DOMAIN
Network-level interaction

Figure 1: The Network/Blockchain ID coupling problem

ToR networks [18], [30] and MAC/IP spoofing techniques [31], [32], [33], [34] also fall short under the ID coupling problem given that: i) ToR networks are designed for fixed data networks (non-wireless) assuming a stable connection to the Internet and cannot scale in view of a wireless service scenario, and ii) not all mobile data users will be in position to effectively hide their physical network identifiers using such techniques due to their limited resources (e.g. energy, processing, spectrum). Besides, distributed deanonymization attacks can be set up by closely monitoring many 5G service peers across distant geographical areas and tracking their participation to specific mixing services.

1.3. Self-Sovereign mobile data access scenarios in B5G TN/NTN setups

The integration of NTN into B5G systems extends coverage, improves resilience, and introduces new connectivity and compute “asset pools” that can be opportunistically discovered and consumed by mobile users [36]. In SOVEREIGN, self-sovereign mobile data access is interpreted as the ability of users (and machines) to discover, negotiate, authenticate, consume, and pay for network/edge assets on-demand, without relying on fixed-term, operator-controlled contracts and without exposing persistent identifiers that enable long-term tracking. This section outlines representative scenarios where joint TN/NTN deployments highlight the necessity of (i) scalable consensus, (ii) high transaction throughput, and (iii) strong privacy protection under network/blockchain identifier coupling, as discussed in previous sections.

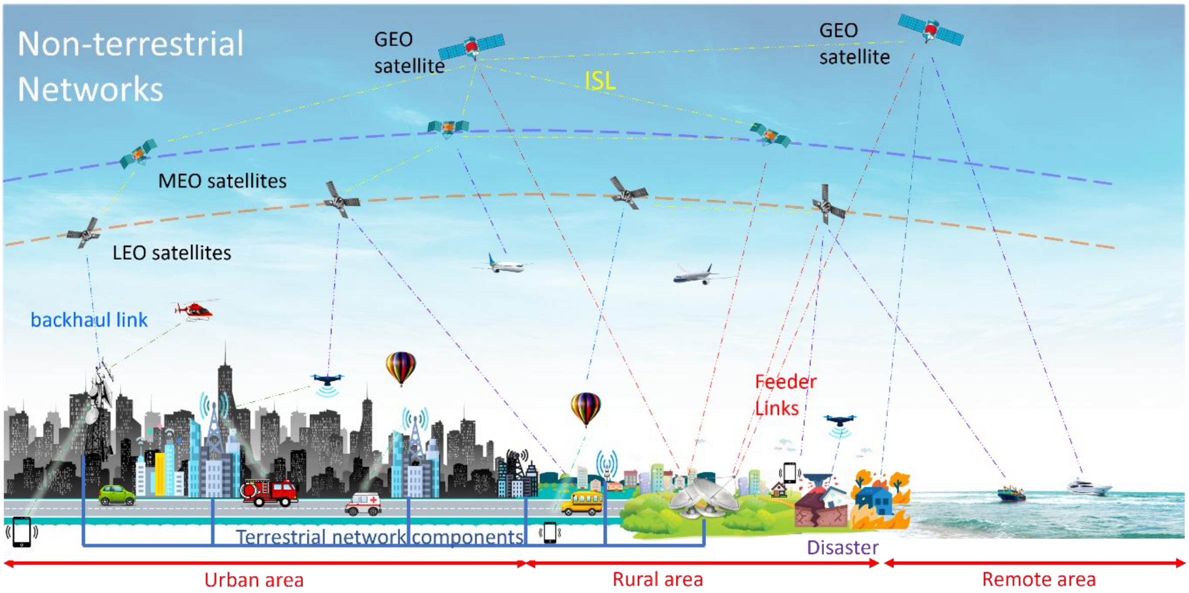


Figure 2. TN / NTN coverage setup

1.3.1. Scenario A — Coverage extension and “instant roaming” via NTN backhaul

In Scenario A, self-sovereign access is enabled when a user located in a rural/remote or underserved area can opportunistically attach to a nearby terrestrial access point (e.g., community Wi-Fi, a private small cell, a pop-up RAN node) whose backhaul is provided through an NTN gateway (e.g., LEO feeder link toward a satellite constellation) [37]. Instead of relying on a pre-negotiated roaming agreement with a “home” operator, the user device discovers one or more reachable “asset servers” advertising available connectivity assets (bandwidth/time quota, QoS class, slice profile, local breakout option) and selects an offer based on price, expected performance, and trust score. Access is granted through a short-lived, privacy-preserving authorization token tied to service entitlements (what the user is allowed to consume) rather than a persistent user identity [38]. Settlement follows a pay-per-use model: the service provider (or local edge broker) records consumption evidence (e.g., delivered bytes/time/QoS class) and triggers micro-settlement using a lightweight transaction workflow that is resilient to NTN intermittency [39]. This scenario is especially important in TN/NTN setups because the NTN gateway naturally becomes an aggregation point; therefore, the architecture must prevent network/blockchain identifier coupling by ensuring that ledger-facing identifiers and radio-facing identifiers cannot be trivially correlated across repeated sessions, while still providing accountability (e.g., dispute resolution, fraud prevention) and regulatory compliance.

From an architectural viewpoint, Scenario A exercises several key WP2 aspects. First, it requires asset discovery and advertisement mechanisms that work even when backhaul is constrained (e.g., cached offers at the edge, periodic broadcast beacons, or opportunistic discovery via local brokers). Second, it needs an access negotiation and authorization procedure that can complete with minimal RTT dependency on the core/ledger (e.g., local authorization with delayed settlement, or pre-authorized credit/deposit models). Third, it requires a metering and evidence collection function at the access node or edge gateway that is verifiable but privacy-preserving (e.g., signed usage receipts, cryptographic commitments, or blinded proofs). Finally, it must support service continuity across intermittent NTN conditions, including buffering and graceful degradation policies (e.g., temporarily throttling, switching to best-effort, or resuming service based on remaining credit) while maintaining a consistent accounting state. These aspects directly map to the requirements you already motivate in earlier sections: scalable trust/consensus participation (to avoid centralized AAA bottlenecks), high transaction throughput (micro-settlement), and strong anonymity under joint TN/NTN interactions.

1.3.2. Scenario B — Mobility-heavy users with TN-NTN service continuity

Scenario B addresses high-mobility users and platforms—such as vehicles, trains, cross-border commuters, and unmanned systems—that traverse heterogeneous coverage areas where terrestrial and non-terrestrial access alternate frequently [40]. In such environments, service continuity cannot rely on traditional home-network anchoring, static roaming agreements, or centralized authentication functions, as these introduce excessive latency, single points of failure, and privacy risks. Within the SOVEREIGN framework, mobility-aware self-sovereign access enables users to dynamically and repeatedly acquire short-lived service entitlements from multiple service peers while preserving privacy and maintaining accountability across handovers [40]. In this scenario, a mobile user initially attaches to a terrestrial access node and obtains a time- or volume-bounded authorization token that encodes service rights (e.g., bitrate class, latency profile, edge service eligibility) independently of any persistent user identity. As the user moves, handover events trigger rapid re-attachment to new TN access points or to an NTN link (e.g., LEO satellite access) when terrestrial coverage degrades. Instead of full re-authentication against a centralized core, the new access node validates the user’s entitlement locally or via a nearby edge broker, enabling make-before-break continuity [41]. Crucially, each handover may generate micro-settlement events or usage receipts, but these are decoupled from immediate ledger finalization, allowing operation under NTN-induced latency and intermittent connectivity.

Architecturally, Scenario B stresses the interaction between mobility management, settlement logic, and privacy preservation. Frequent handovers dramatically increase the volume of control-plane interactions and potential micropayments, exacerbating the transactions throughput challenge highlighted earlier. At the same time, repeated network-level interactions across access nodes increase the risk of correlating radio identifiers with ledger-facing identifiers, intensifying the network/blockchain ID coupling problem. To mitigate this, SOVEREIGN can support ephemeral identifiers, token re-randomization, and edge-assisted settlement aggregation, ensuring that mobility does not translate into long-term traceability. Edge brokers play a key role by caching trust state, aggregating usage evidence across multiple access points, and synchronizing with the distributed ledger opportunistically, thereby masking mobility patterns from the blockchain layer while maintaining verifiable accounting [42].

1.3.3. Scenario C — Emergency, disaster recovery, and infrastructure outage scenarios

Scenario C focuses on emergency and disaster situations—such as natural disasters, large-scale power outages, cyber-attacks, or infrastructure failures—where terrestrial network coverage is partially or fully unavailable and centralized control functions (e.g., core-network, billing platforms, roaming hubs) may be unreachable or overloaded. In these conditions, NTN, including LEO satellite systems and rapidly deployable gateways, play a critical role in restoring baseline connectivity and enabling coordination among first responders, public authorities, and affected users [42]. The SOVEREIGN framework addresses these scenarios by enabling self-sovereign, infrastructure-light access that remains operational even under severe network fragmentation. In this scenario, users and emergency teams dynamically attach to any reachable access point, including portable base stations, ad-hoc Wi-Fi hotspots, vehicle-mounted relays, or temporary small cells, whose backhaul is provided through NTN gateways [43]. Rather than relying on centralized authentication servers or pre-existing subscription relationships, access is granted via pre-authorized or locally issued emergency entitlements that encode service priorities, access duration, and permissible resource usage. These entitlements may be provisioned in advance (e.g., to first responders) or issued opportunistically by trusted local brokers operating in degraded network conditions. Crucially, authorization and policy enforcement must remain functional even when continuous connectivity to the distributed ledger is unavailable, enabling rapid access without sacrificing accountability.

From an architectural standpoint, Scenario C stresses resilience, offline tolerance, and attack resistance. Settlement and accounting functions are intentionally decoupled from real-time service delivery: usage evidence is locally recorded, cryptographically protected, and buffered at the edge until reliable backhaul connectivity is restored. This design mitigates the risk of denial-of-service attacks against centralized AAA entities and ensures continued operation under intermittent NTN links. At the same time, privacy preservation remains essential, as emergency contexts often amplify the sensitivity of location and identity information [43]. The SOVEREIGN architecture therefore enforces identity minimization and unlinkability, preventing adversaries from correlating network-level observations (e.g., access attempts at emergency hotspots) with ledger-level transactions once settlement resumes.

1.3.4. Scenario D — Maritime, aviation, and remote industrial operations

Scenario D targets maritime, aviation, and remote industrial environments, such as ships at sea, aircraft in flight, offshore platforms, remote mining sites, and energy infrastructure, where NTN constitute the primary or exclusive wide-area connectivity solution. In these settings, connectivity is characterized by long propagation delays, constrained and costly backhaul capacity, intermittent contact windows, and limited on-site infrastructure, making traditional centralized authentication, billing, and roaming mechanisms inefficient or infeasible [44], [45]. The SOVEREIGN framework enables self-sovereign mobile data access by allowing users, onboard systems, and machines to dynamically acquire connectivity and computing resources in a contract-free, entitlement-based, and privacy-preserving manner. In this scenario, a vessel, aircraft, or remote site operates an onboard or on-site edge domain, which may include local access points, edge computing resources, and caching capabilities. Users and devices attach to the local access network and obtain service entitlements that specify connectivity quotas, QoS levels, and access to local or remote services. Rather than performing real-time authentication and settlement with a remote core network, authorization is handled locally by an edge broker that enforces policies and usage limits based on pre-funded credits, delegated trust, or mission-specific allowances. When NTN connectivity is available, the edge broker synchronizes with the broader SOVEREIGN ecosystem to refresh trust state, reconcile usage evidence, and obtain updated policies, while remaining fully operational during periods of disconnection.

Architecturally, Scenario D highlights the importance of edge autonomy and backhaul optimization. Given the high cost and limited capacity of NTN links, the SOVEREIGN architecture promotes local breakout, caching, transcoding, aggregation, and semantic processing to minimize unnecessary backhaul usage. For example, popular content or mission-critical data can be cached locally, processed at the edge, or selectively forwarded based on priority and relevance [44]. Settlement with the distributed ledger is asynchronous and batched, ensuring that fine-grained usage accounting does not translate into excessive signaling over satellite links. Privacy preservation remains critical, as repeated interactions via a small number of satellite gateways could otherwise enable long-term tracking of users, vessels, or assets. The architecture therefore enforces identifier rotation, settlement aggregation, and gateway-level obfuscation to prevent network/ledger identifier coupling.

1.3.5. Scenario E — Dense events and temporary hotspots with NTN-assisted offload

Scenario E addresses high-density, short-duration traffic surges generated by large public events such as concerts, sports matches, festivals, exhibitions, or emergency gatherings, where the number of active users and service requests far exceeds the dimensioned capacity of the existing terrestrial infrastructure [46], [47]. In these environments, traffic demand is highly spatiotemporally concentrated, and conventional mobile data access models—based on static subscriptions and pre-planned capacity provisioning—struggle to provide acceptable QoS and QoE. The SOVEREIGN framework enables a self-sovereign access paradigm in which temporary infrastructure, third-party assets, and NTN backhaul can be dynamically mobilized and economically incentivized to absorb demand peaks. In this scenario, event organizers, local authorities, or third-party providers deploy temporary terrestrial access nodes, such as pop-up small cells, private Wi-Fi hotspots, or rapidly deployable RAN units [48]. These access nodes advertise available connectivity and edge resources as tradable assets, which nearby users can discover and consume on-demand. NTN connectivity (e.g., LEO satellite backhaul) is leveraged either to complement congested terrestrial backhaul or to provide independent connectivity where fiber access is unavailable or insufficient. Users acquire short-lived service entitlements—often valid only for minutes or hours—allowing them to dynamically attach to the most suitable access point without prior subscription or operator lock-in. Settlement follows a fine-grained, usage-based model, enabling fair compensation for temporary infrastructure providers and creating economic incentives for rapid capacity deployment.

Architecturally, Scenario E stresses extreme scalability and concurrency. Thousands of users may simultaneously perform access negotiation, authorization, and settlement operations within a confined geographical area, amplifying the multi-million transactions-per-second requirement identified earlier. Moreover, dense physical proximity between users and access nodes significantly increases the risk of network/blockchain identifier coupling, as adversaries could correlate radio-level observations with ledger transactions during repeated interactions. The SOVEREIGN architecture mitigates these risks by employing ephemeral identifiers, short-lived tokens, and settlement aggregation at the edge, ensuring that high-frequency interactions do not translate into long-term traceability. Edge brokers play a critical role by caching offers, performing local authorization, aggregating metering evidence across many users, and synchronizing with the distributed ledger asynchronously to avoid congestion collapse.

1.3.6. Scenario F — IoT/IIoT services with sporadic and delay-tolerant NTN connectivity

Scenario F addresses massive IoT and Industrial IoT (IIoT) deployments operating in remote, sparsely connected, or infrastructure-poor environments, such as environmental monitoring, asset tracking, smart agriculture, maritime sensing, pipeline monitoring, and remote industrial automation [45]. In these settings, devices typically generate small, intermittent data volumes, operate under strict energy and cost constraints, and rely on sporadic NTN connectivity with limited contact windows. Traditional mobile data access models—based on persistent subscriptions, continuous connectivity, and centralized authentication—are ill-suited to these characteristics [49]. The SOVEREIGN framework enables a self-sovereign, lightweight, and delay-tolerant access model tailored to machine-driven communication. In this scenario, large numbers of IoT devices connect to local collectors or gateways (e.g., sensor hubs, industrial controllers, mobile aggregators), which act on their behalf for access negotiation, authorization, and settlement. Devices themselves do not directly interact with the distributed ledger; instead, trust and economic responsibility are delegated to gateways or edge brokers that manage groups of devices. These gateways issue device- or task-scoped service entitlements, defining when, how often, and under which conditions data may be transmitted. Data is buffered, pre-processed, or aggregated locally and forwarded only when NTN connectivity becomes available, reducing signaling overhead and conserving energy. Settlement with the ledger is performed asynchronously and in batches, decoupled from individual device transmissions.

Architecturally, Scenario F emphasizes extreme scalability, delegation, and delay tolerance. The number of devices may reach thousands or millions, making per-device authentication or real-time settlement infeasible. The SOVEREIGN architecture therefore supports hierarchical trust models, where gateways and edge brokers act as accountable intermediaries, enforcing policy, managing credits, and maintaining verifiable usage records. Privacy is preserved by preventing direct exposure of device identifiers to the ledger and by aggregating transactions across many devices, mitigating long-term traceability of individual sensors or assets. NTN gateways further amplify the need for aggregation, as repeated uplinks through the same satellite path could otherwise reveal spatial or operational patterns.

1.3.7. Summary and comparison of SOVEREIGN NTN access scenarios

Table 1 summarizes the key use cases and features of the considered scenarios. Scenarios A and F target sparsely connected environments—rural users and massive IoT deployments, respectively—where NTN primarily complements limited terrestrial coverage. Both favor lightweight, localized authentication (short-lived tokens or local aggregators) and are characterized by small to medium scalability needs, reflecting constrained traffic volumes and intermittent connectivity.

In contrast, Scenarios B and D focus on mobility-centric and mission-critical transport domains such as vehicles, ships, and aircraft. These scenarios require seamless TN–NTN handovers or NTN-dominant backhauling, supported by edge or local brokers to ensure continuity and low latency. Their medium scalability reflects a balance between user density and stringent reliability requirements.

Scenarios C and E represent the most demanding operational extremes. Emergency and disaster scenarios (C) rely on NTN as a lifeline, necessitating robust, pre-authorized or locally issued entitlements that remain functional even under partial infrastructure failure, with scalability ranging from medium to large. Event-driven scenarios (E), on the other hand, prioritize massive scalability and highly dynamic access, addressed through very short-lived, on-demand tokens and NTN-assisted backhaul to absorb sudden traffic surges.

Overall, the comparison highlights how NTN integration strategies and trust models must be tightly aligned with the operational context: from resilience and autonomy in emergencies, to elasticity and scale in mass events, and efficiency and simplicity in remote or sensor-driven deployments.

Table 1: Summary of the scenarios

Scenarios	Use case/ users	Connectivity pattern	Authentication method	Scalability
Scenarios A	Rural/ remote users	NTN backhaul	Short-lived, entitlement-based tokens	Small
Scenarios B	Vehicles/ trains	TN – NTN handovers	Edge broker	Medium
Scenarios C	Emergency/ disaster	NTN as lifeline	Pre-authorized or locally issued emergency entitlements	Medium/ Large
Scenarios D	Ship, aircraft, offshore platform	NTN main backhauling	Local edge broker	Medium
Scenarios E	Events	NTN-assisted backhaul	Very short-lived, on-demand tokens	Massive
Scenarios F	Massive IoT/ sensors	Sporadic NTN via gateways	Local aggregator	Small/Medium

1.3.8. Contract less mechanism in 3GPP NTN

The integration of TN and NTN is the key element in the 6G networks toward achieving global. Started from Release-17, foundation standards were introduced to support NTN-based new radio (NR) transmissions, enabling communication via NTN (LEO, MEO, GEO satellites and HAPs) elements [50]. Within this framework, the concept of **contract-less mechanisms**—procedures that allow user equipment (UE) to access the network services without pre-established roaming agreements—emerges as a natural consequence of the TN-NTN architecture. Although 3GPP has not explicitly specified the “contract-less” mechanism yet, there exist several standardized features which are implicitly supporting smooth transition between TN and NTN segments:

- **Public Land Mobile Network (PLMN) selection flexibility**: Release 17 has provided UEs a feature that may autonomously select PLMNs broadcast by satellites. This enables the UEs to opportunistically access any NTN operator [51]. Release 18 integrates NTN into 5G-Advanced with more robust, multi-layer selection capabilities. While Release 19 enhance the feature with multi-operator and multi-orbit flexibility. The latest Release 20 aims to provide a truly global, service-centric PLMN selection [52].

- **Enhanced mobility and service continuity procedures**: Doppler compensation, timing advance adaptations, and mobility enhancements specified in Release 17 are essential to provide seamless handover between TN and NTN networks [51]. Mobility-aware and QoS-aware continuity with multiple layer integration were proposed in Release 18. In Release 19 supports global and multi-operator continuity via advanced mobility management and multi-orbit handover capability. Finally, the future Release 20 envisages seamless TN-NTN convergence in which AI/ML-based mobility is employed and service-centric continuity management [52].

1.3.9. Metrics and KPIs for self-sovereign mobile data access in TN/NTN scenarios

This section details the technical specifications and reference implementation choice for realizing self-sovereign mobile data access in an integrated TN/NTN B5G setup at ULU. The objective is to enable contract-less, on-the-fly access provisioning with verifiable charging, while preserving user sovereignty over identity, data disclosure, and lifecycle controls. From a system standpoint, ULU adopts a hybrid control and settlement design, where service discovery, negotiation, and runtime access management are executed off-chain to satisfy mobility, latency, and scalability constraints, while the DLT layer is used for accountability without storing fine-grained session telemetry on-chain. On service discovery, the UE issue a discovery request and receives a ranked set of mobile data access offer spanning TN and NTN connectivity, where each offer includes (i) access type and availability (TN gNB / NTN satellite or gateway), (ii) expected QoS/QoE envelope (latency/jitter/throughput), (iii) coverage validity window and mobility constraints (e.g. predicted NTN visibility), (iv) price model and settlement mode (on-chain or relay-assisted micro settlement), and (v) privacy option (pseudonymous credential/attribute proof). Based on the discovered offers, the UE negotiates a short-lived service agreement that fixes RAT selection and fallback triggers (TN->NTN/NTN->TN), outage threshold, and charging parameters, and then performs online access management with progressive fair exchange (e.g. per time slice, traffic quota, or content chunk) and abort-safe semantics to tolerate NTN intermittency and prevent pay-without service or service without pay. A central ULU requirement is explicit data distribution awareness across TN/NTN domains and stakeholders (UE, access nodes, MEC, content servers), acknowledging non-IID behavior, temporal drift, and system heterogeneity; therefore, orchestration relies on privacy-preserving distribution descriptors rather than raw mobility traces or application payloads, with SSI-bound consent policies controlling what is disclosed and at what granularity. Finally, to operationalize the "right to be forgotten" in a distributed learning setting, ULU specifies machine unlearning as a first-class privacy safeguarded, where upon consent revocation or retention expiry, the platform must support removal of the influence of identified user contribution from local or aggregated models without re-exporting raw data, using contribution reference, model versioning, and signed unlearning attestations that can be verified under TN/NTN connectivity without dounded computation and communication overhead. Table 2 presents the main KPI parameters of different scenarios.

Table 2: Summary of main KPIs

Scenarios	User density	Connection latency	Coverage	Area throughput	Per-user throughput
Scenarios A	~ 100 users per km2; 80/20 distribution	>= 2 RTT	> 99%	Hundreds Mpbs	~ 10 Mbps
Scenarios B	~ 1000 users per spot	10 ms	> 99.9%	Up to 1 Gbps per spot area	~ 20 Mbps
Scenarios C	Varies	2 RTT	> 90%	< 100 Mbps per area	~ 3 Mbps
Scenarios D	~ 1000 users per spot	NTN RTT	> 99.9%	Up to 1 Gpbs per spot area	~ 20 Mbps
Scenarios E	> 1000 users per spot area	10 ms	> 99.99%	Up to 1 Gbps per spot area	~ 20 Mbps
Scenarios F	Massive IoT/ sensors	Delay-tolerant	punctual	Low	Very low

2. SOVEREIGN Service Architecture, Domains and Roles

We consider a heterogeneous multi-orbit wireless network (HMOWN) infrastructure that is composed of multiple networking tiers. Network components spanning the network tiers support different RATs and/or have heterogeneous networking capabilities, e.g., utilize different spectrum bands, host diverse processing and storage capacity. Video consumers are considered to be part of the HMOWN infrastructure and utilize a number of RAT interfaces to access the different network tiers. On top of the HMOWN infrastructure, we consider a software architecture that implements the blockchain-enabled SOVEREIGN content trading platform, which consists of the user, control and blockchain domains (figure below).

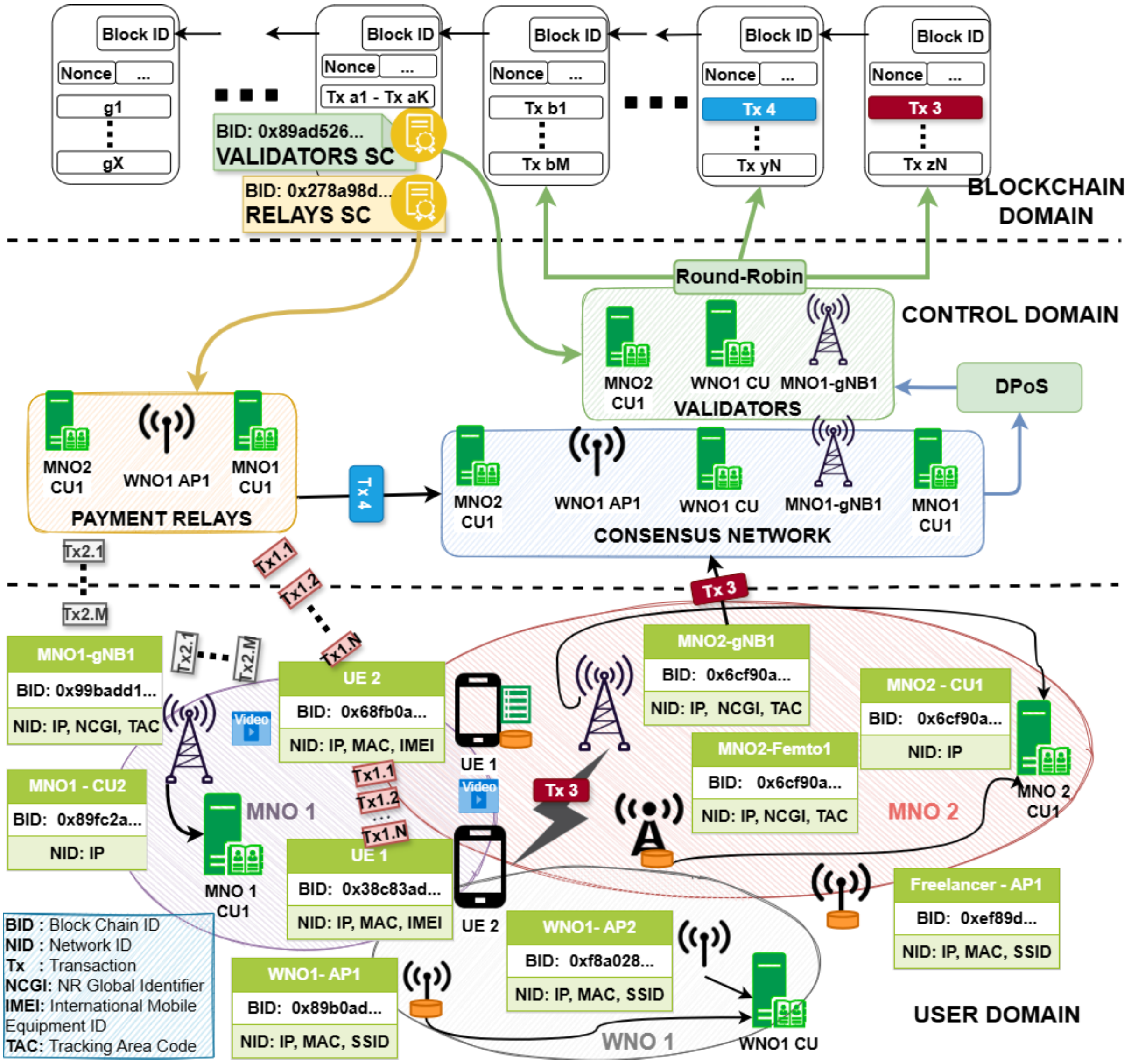


Figure 3: SOVEREIGN Service Architecture and Domains

The *SOVEREIGN user domain* is the place where the actual mobile video content service is delivered, and the issuing of SOVEREIGN payments is performed. Network nodes at the SOVEREIGN user domain run a specialized blockchain-backed software that enables them to dynamically trade available network assets by acting either as network asset clients (consuming network assets and issuing payments), or as network asset servers (sharing their network assets and receiving payments), or as both. Each SOVEREIGN node is the owner of at least one public address (i.e., blockchain ID) that is used to issue/receive payments and interact with the SOVEREIGN control domain (e.g., for consuming payment relay services). All network-level service phases of Figure 4 will be implemented in the SOVEREIGN user domain.

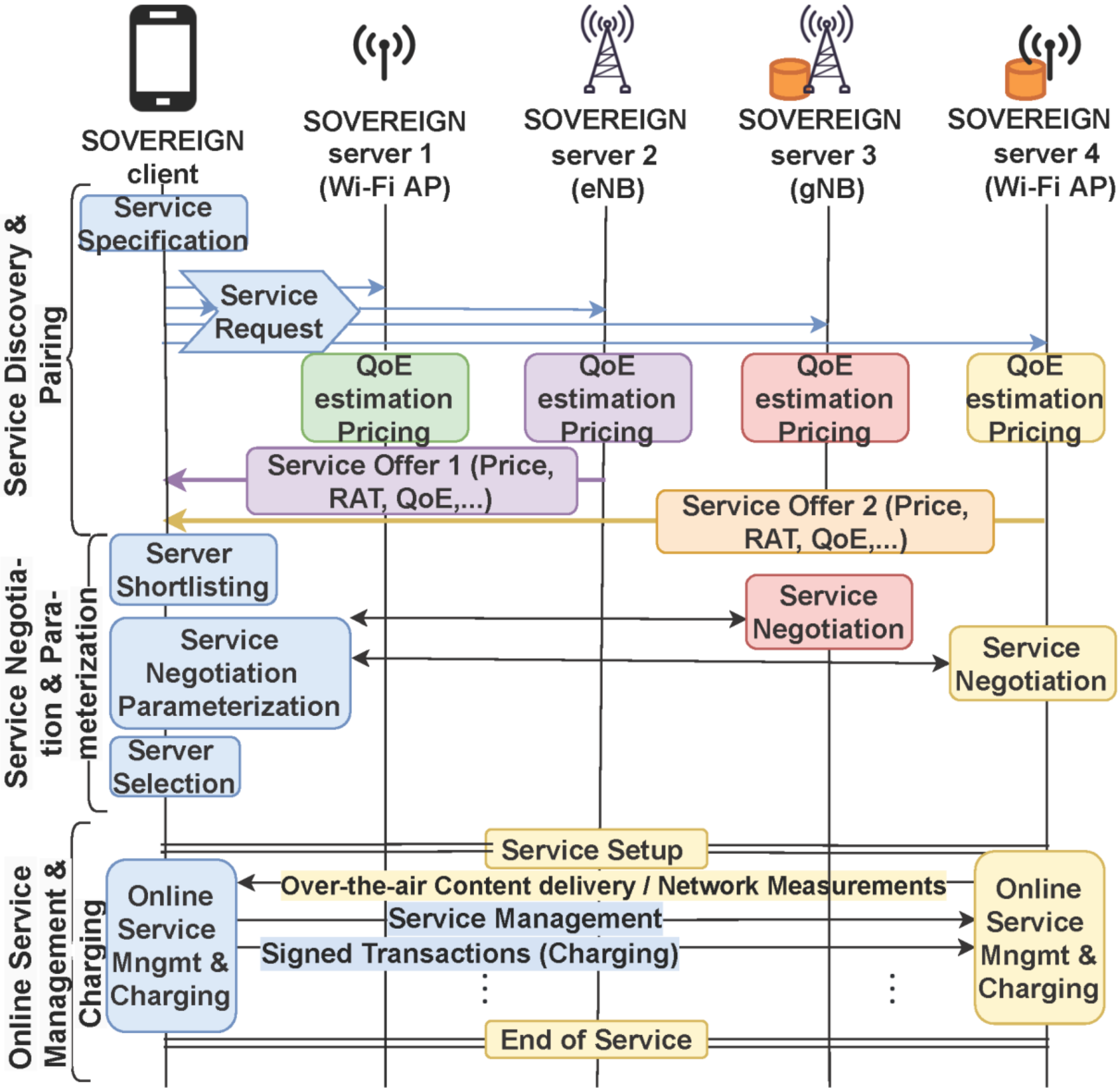

SOVEREIGN client
SOVEREIGN server 1 (Wi-Fi AP)
SOVEREIGN server 2 (eNB)
SOVEREIGN server 3 (gNB)
SOVEREIGN server 4 (Wi-Fi AP)
Service Discovery & Pairing
Service Specification
Service Request
QoE estimation Pricing
QoE estimation Pricing
QoE estimation Pricing
QoE estimation Pricing
Service Offer 1 (Price, RAT, QoE,...)
Service Offer 2 (Price, RAT, QoE,...)
Service Negotia-tion & Para-meterization
Server Shortlisting
Service Negotiation
Service Negotiation Parameterization
Service Negotiation
Server Selection
Online Service Management & Charging
Service Setup
Online Service Mngmt & Charging
Over-the-air Content delivery / Network Measurements
Service Management
Signed Transactions (Charging)
Online Service Mngmt & Charging
End of Service

Figure 4. SOVEREIGN service phases (from the project Deliverable D2.1)

Service control of the SOVEREIGN platform (including charging) is implemented at the *SOVEREIGN control domain*. At this layer, the SOVEREIGN nodes undertake different roles towards distributed consensus in line with their functional capabilities and run specialized software that implements the SOVEREIGN control protocols. For example, all SOVEREIGN nodes are considered capable of acting as *witnesses*, staking coins in order to delegate other SOVEREIGN nodes the role of payment relays, coin mixers, and validators for a prescribed time period. However, not all SOVEREIGN nodes are required to act as full consensus nodes, buffering and propagating new transactions towards block validators, storing and communicating blockchain data to other SOVEREIGN nodes on demand, etc. (section 2.1).

In the SOVEREIGN control domain, validators are the only nodes authorized to append blocks in the SOVEREIGN blockchain, whereas payment relays/coin mixers are the only nodes authorized to aggregate payments/mix coins.

The *SOVEREIGN blockchain domain* hosts the public ledger structure that records the blockchain-level interactions between SOVEREIGN nodes. The SOVEREIGN service control logic is hosted by two specialized SCs: the validators and the relays SCs. The validators SC (VSC) defines all system parameters and functions necessary to implement the SOVEREIGN DPoS protocol for distributed consensus, also implementing sophisticated reward/penalty mechanisms that enforce honest operation of elected validators. The VSC is designed so as to allow different roles and levels of engagement across the SOVEREIGN nodes during distributed consensus, further safeguarding system robustness in the long-term. The relays SC (RSC) defines all parameters and mechanisms necessary to implement credible payment relay and mixing services.

All SOVEREIGN protocols are designed to enable offline delivery of the mobile video service by local servers while enforcing honest operation of the key SOVEREIGN actors (validators, payment relays, mixing servers) in a fully decentralized fashion, through the deployment of sophisticated incentive engineering mechanisms that are implemented using on-chain SCs. This design approach is not only relevant to the problem under scope, i.e. the mobile video delivery service requires physical proximity and offline service consumption, but it also enables minimum interactions with the public ledger; thus, minimum transactions capacity requirements and on-chain costs, ideally only for i) the establishment of payment channels, ii) on-chain dispute resolution between the SOVEREIGN nodes and actors, and iii) for SC-driven rewards/penalties to dishonest SOVEREIGN actors. The common basis of all incentive engineering mechanisms is the requirement to time-lock to the respective SOVEREIGN SCs an amount of funds that is proportional to the risks following from a potential dishonest operation of the key SOVEREIGN actors, an approach that is compatible with the overall PoS-based design of the SOVEREIGN blockchain platform.

The SOVEREIGN platform will incorporates three innovative blockchain-based protocols for Beyond 5G distributed consensus, payment relay and coin mixing, all of which are designed to support different roles and levels of engagement in distributed consensus while preserving anonymity and requiring only a low transactions throughput onto the SOVEREIGN blockchain.

2.1. SOVEREIGN Roles

SOVEREIGN nodes undertake different roles in light of their functional capacity and desired level of engagement with the SOVEREIGN service domains. In this section, we overview these roles and briefly discuss relevant implementation details.

Validators. High-end SOVEREIGN nodes that are authorized to seal blocks in a round-robin fashion for a given time epoch (measured in blocks). The number of validators is a system parameter that can be amended in the long term as soon as the amendment is supported by the majority of validators for a certain number of consecutive epochs. Validators are elected through the SOVEREIGN DPoS consensus protocol that enables any SOVEREIGN node to run as a candidate validator for a target epoch *e*, participating in an action-based scheme during the *validators' election epoch e-1*. During the election epoch, candidate validators should i) lock a minimum guarantee fund to the VSC, which is used to enforce honest operation of validators through VSC-driven penalties, ii) lock a reward fund that will be shared across SOVEREIGN nodes that vote (stake) in favor of the candidate (if it gets elected), and iii) a transaction fee that is paid to the validator on a per sealed block basis. Validators with the highest stakes (including their own guarantee fund) get shortlisted and elected on the basis of available validator seats per epoch. If elected, validators receive transactions fees and preserve their role for a given epoch, assuming that they act honestly. Penalty mechanisms and validator replacement methods are provisioned to safeguard system robustness against inadvertent validator behaviors.

Validator witnesses. SOVEREIGN nodes that stake in favor of candidate validators for a given epoch. V-witnesses are incentivized to actively participate in the DPoS consensus to: i) share the reward fund offered by the candidate validator (in proportion to their v-witness stakes), ii) receive free service from *FoC servers* supporting the candidate validator, and iii) receive priority in the processing of their transactions. SOVEREIGN nodes shall employ their own logic towards the selection of a candidate validator and the amount of funds to be staked per candidate validator.

Free-of-charge servers for validators. Special case of v-witnesses that, instead of staking an arbitrary number of coins in favor of a candidate validator, promise to offer FoC service to v-witnesses of a tagged validator (if elected). FoC servers also time-lock funds in favor of a candidate validator in proportion to the FoC service they promise to offer. Locked funds are used to incur penalties to FoC servers that fail (or refuse) to deliver the promised service to v-witnesses. FoC servers leverage their network-level capabilities to attract more v-witnesses towards blockchain-level consensus.

Payment relays. High-end SOVEREIGN nodes that are authorized to act as payment intermediaries, enabling instant off-chain payments for a given time epoch. The number of payment relays per epoch varies in line with i) the (estimated) transactions capacity of the blockchain and ii) the type of relay licenses requested by other candidates. Candidate relays lock to the RSC a minimum guarantee fund that is calculated based on i) the number of clients that the relay requests to support, ii) the total number of coins that can be attached to the relay (using inbound payment channels), and iii) the transactions throughput that the relay promises to spur into the SOVEREIGN blockchain. Relay licensing follows a similar approach to the validators' election process, enabling SOVEREIGN nodes to stake funds, or FoC service, in favor of a candidate relay. Authorized relays establish payment channels with SOVEREIGN clients and servers only through the RSC. Payment relays also convert off-chain payments to on-chain balance updates only through the RSC.

Honest relays receive transaction fees on a per-off-chain transaction that they process, while they can withdraw their guarantee funds upon the expiration of their license. Dishonest relays receive penalties (on their guarantee fund) according to an RSC-driven mechanism that enables disregarded relay clients to trigger on-chain dispute resolution. This process requires i) disregarded clients to submit signed promises (transactions) of the relay and ii) the (reported) relay to submit proofs of its lawful operation. This is possible mainly due to the employment of a fixed time delay window (measured in blocks) by which the relay promises to submit new off-chain transactions. This parameter is termed as relay delay in the sequel, while it is specified by the payment relay during the licensing epoch and is included in the signed off-chain promises issued by the payment relay. Triggering the RSC for on-chain dispute resolution comes with an on-chain transaction cost that is initially paid by the disregarded relay client but is fully reimbursed (along with other costs relevant to the dispute) by the guarantee fund of the dishonest relay. The license of relays can be revoked under certain conditions and relay replacement mechanisms are also provisioned.

Relay witnesses. SOVEREIGN nodes that stake in favor of candidate payment relays for a given epoch. The selection of a candidate relay and the number of coins staked in favor of candidate relays is left up to the implementation of the SOVEREIGN node. R-witnesses are incentivized to actively participate in the relay election to i) share a reward fund offered by candidate relays (if elected) and ii) receive FoC service by servers supporting the tagged payment relay.

Free-of-charge servers for relays. Special case of r-witnesses that support a tagged candidate payment relay for a given epoch. Similar operation to FoC servers for validators.

Payment relay clients. They are SOVEREIGN nodes that consume a payment relay service to perform instant off-chain payments at a lower transactions cost. Relay clients select one (or more) payment relays with an active license and establish inbound payment channels on the RSC. To this end, they time-lock an arbitrary amount of funds to the RSC and indicate the relay that is authorized to handle its balance.

Mixing servers. Payment relays that also act as mixing servers on the basis of the payment relay license and the RSC mechanisms attached to it, e.g., payment channels established with the SOVEREIGN clients and servers. The SOVEREIGN mixing servers implement hybrid mixing, an approach that enables centralized payment relay servers to deploy the mixing service offline; however, enforcing their honest operation in a fully decentralized fashion using the RSC logic for on-chain dispute resolution with disregarded relay clients. The SOVEREIGN mixing service extends RSA blinding and puzzle solution/solving protocols of Tumblebit [26] to enable instant fair-exchange of mobile video content and on-chain funds in an anonymous fashion.

Mixing clients. Payment relay clients that additionally consume SOVEREIGN mixing services provided by the payment relay. Mixing clients pay an additional fee for the use of mixing services, aiming to employ both instant and anonymous off-chain payments.

Full consensus nodes. High-end SOVEREIGN nodes that are responsible for propagating new transactions through the consensus network, keeping track of blocks issued by validator nodes, storing the full blockchain data, and providing them upon request to other SOVEREIGN nodes. Full consensus nodes are not necessarily part of the network infrastructure, which consumes/delivers mobile video content.

2.2. Service flow and charging example

In Figure 3, the SOVEREIGN user domain is composed of individual service hotspots (e.g., Freelancer AP1), Wi-Fi Network Operators (WNO) (e.g., WNO1 with the Wi-Fi access points AP1 and AP2 attached to the WNO core unit WNO1 CU), 5G MNOs (e.g., MNO1 with gNB1 that is attached to the MNO1 - core unit 1), and user equipment (UEs) (e.g., UE1 and UE2), which support device-to-device (D2D) communications. Some SOVEREIGN nodes utilize local storage resources to employ content caching and instantly deliver mobile video content on demand. The SOVEREIGN control domain is composed of a subset of the user-domain SOVEREIGN nodes, which have been additionally engaged in the roles of payment relays (e.g., MNO1-CU1), validators (e.g., MNO-gNB1), and full consensus nodes (e.g., WNO1-CU). The SOVEREIGN blockchain is maintained by full consensus nodes and updated only by elected validators. The VSC and RSC are deployed in the early blocks of the SOVEREIGN blockchain, enforcing honest operation of validators and payment relays at the SOVEREIGN control domain.

Moving again to the SOVEREIGN user domain, UE2 consumes popular video content from two SOVEREIGN servers: UE1, which uses the 5G base station MNO2-gNB1 to relay the requested content, and MNO1-gNB1, which utilizes its backhaul connectivity to reach the content through the Internet. UE2 is assumed to utilize the payment relay services of MNO1-CU1, enabling instant micro-payments Tx1.1, Tx1.2, ..., Tx1.N with UE1 and Tx2.1, Tx2.2, ..., Tx2.M with MNO1-gNB1. On the contrary, UE1 is assumed to issue a direct on-chain payment Tx3 to MNO2-gNB1 and propagate it to the consensus network directly. However, micro-payments Tx1.1-Tx1.N and Tx2.1-Tx2.M are performed off-chain through the payment relay MNO1-CU1, which subsequently aggregates the respective payments into a single on-chain transaction Tx4. Tx4 indicates as recipient, the public address of the RSC, and, when processed by validators and posted on-chain, it triggers the RSC logic to update the balance of UE2, UE1, and MNO1-gNB1 on-chain accordingly.

2.3. Resource Usage

The energy efficiency of a blockchain-backed system typically comes down to the requirements of the distributed consensus protocol and the size of the consensus network. Popular PoW-based platforms like Bitcoin consume vast amounts of processing and energy resources due to the participation of myriad consensus nodes in the puzzle solution process (mining). Through this process, consensus nodes gain an opportunity to seal new blocks and receive block rewards attached to them. Blockchain-backed mobile data access should be sustainable and energy-efficient, enabling also network nodes to adapt the level of their engagement in distributed consensus to their functional capabilities.

The SOVEREIGN platform is designed to meet the energy-efficiency requirements set for 5G and Beyond mobile data networks by employing a DPoS consensus protocol where a very small set of validators seal blocks in a deterministic (round-robin) fashion. Although validators are elected on an epoch-by-epoch basis by SOVEREIGN nodes, which are provided with clear incentives to do so (section 2.1), our system design does not oblige SOVEREIGN nodes to act as v-witnesses. Even if SOVEREIGN nodes choose to participate in the DPoS process, they will be only required to sign their stakes (votes) by computing a single hash function and broadcasting this short message to the consensus network. Adding to this, SOVEREIGN nodes are not required to be actively engaged with the maintenance of the SOVEREIGN blockchain, by acting as consensus nodes that propagate transactions and keep track of the current blockchain status. Instead, SOVEREIGN nodes can assess the SOVEREIGN blockchain status by querying consensus nodes using special calls, e.g., JSON queries to Open Nodes in ETH. Hence, at minimum, a SOVEREIGN node is only required to i) be holder of a SOVEREIGN public address and operate a simple wallet application to issue/receive payments, ii) be capable of computing/verifying only a few cryptographic signatures per second (e.g. smartphones can compute thousands of hash signatures per second) and iii) query consensus nodes to assess the SOVEREIGN blockchain status. At maximum, a SOVEREIGN node can actively participate in the SOVEREIGN consensus process (e.g., acting as validator, full consensus node, witness, FoC server), or by being a payment relay that aggregates (or mixes) off-chain payments.

It readily follows that the design of the SOVEREIGN platform is fully aligned with the heterogeneous nature of a 5G and Beyond mobile data access, enabling network nodes to match their level of engagement with the system in view of their prospects, operational requirements, and functional capabilities. The employment of DPoS consensus mitigates unnecessary consumption of computation and energy resources, limiting the number of block sealers to the minimum and generating blocks in a deterministic fashion, thus enabling energy-efficient and sustainable maintenance of the SOVEREIGN blockchain in the long term. Besides, the employment of off-chain payments and their aggregation through the SOVEREIGN payment relay service substantially reduces the number of transactions (and messages) propagated across the consensus network, keeping the operational requirements of the platform to the minimum (i.e., size of the consensus network, computation, and energy consumption).

2.4. Implementation aspects

The proposed blockchain-backed payment service aims to revolutionize service charging in 5G and Beyond networks. Different implementations of the proposed payment service can be delivered depending on the architectural and functional capabilities of the mobile data network under scope. In this section, we present some ideas on potential implementation under the service-oriented architecture (SOA) of the Release 16 3GPP 5G System (5GS) [35].

The payment relay service shall run on top of the standard network protocol stack (PHY, MAC, IP, TCP/UDP) and specifically at the application (APP) layer. RAN nodes are not required to implement the SOVEREIGN server software and hold a unique public address; instead, the 5GS core can instantiate a single SOVEREIGN server at the 5GS core network and attach to it many RAN nodes. Alternatively, a separate SOVEREIGN server instance can be instantiated per network slice. The SOVEREIGN payment relay server can be implemented as a traditional HTTP server that allows SOVEREIGN clients and servers to consume its services using RESTful APIs. This approach is in line with the current design of the 5GS core. The SOVEREIGN client software can also be implemented as a simple HTTP client that is bound to a wallet software, enabling blockchain-level interactions and APP-layer session management (including network selection).

In the 5GS, the SOVEREIGN server logic can be integrated as part of the network services, taking into consideration the functionality available by the existing 5G core network functions (NFs). The SOVEREIGN client shall attach to the Access and Mobility Function (AMF) through the RAN nodes. The AMF shall be responsible for negotiating the video delivery - payment time plan with the SOVEREIGN client, granting it access to the 5GS, and implementing connection management for the entire service lifetime. The AMF shall also determine the Session Management Function (SMF) that is best suited to handle the SOVEREIGN client session (user plane traffic), while the SMF shall instantiate and subscribe to a Charging Function (CHF) service that shall implement the SOVEREIGN server software. The CHF service shall be responsible for handling SOVEREIGN client payments (potentially via the payment relay service) and triggering access authorization/session termination to the SMF/AMF accordingly. Context information on the SOVEREIGN service can be stored in the form of “unstructured” data in the 5GS using the Unstructured Data Storage Function (UDSF).

At this point, we clarify that RAN and core network nodes that implement the SOVEREIGN server software are not necessarily full consensus nodes (section 2.1). Instead, they are considered capable of assessing the SOVEREIGN blockchain status through full consensus nodes and issuing/receiving payments as described in section 2.3.

3. Technical requirements for the SOVEREIGN DLT-backed B5G platform

3.1. SOVEREIGN Blockchain Engine

The SOVEREIGN blockchain engine should support a pay-per-chunk or micro-payment–driven service model (e.g., mobile video delivery). Such a model inherently generates extremely high transaction volumes, requiring hundreds of thousands to millions of service-level transactions per second, due to massive user populations, short video segments, user mobility, and frequent handovers. Existing layer-1 blockchain platforms, such as Bitcoin and Ethereum, with single-digit to tens of transactions per second (TPS), are therefore orders of magnitude below these requirements. Simply increasing block size or relying solely on traditional consensus mechanisms would either compromise decentralization or introduce unacceptable latency, bandwidth, and energy costs.

To address these limitations, we adopt a combined architectural approach: (i) Delegated Proof of Stake (DPoS), which drastically reduces block production overhead and increases baseline on-chain throughput, and (ii) off-chain payment aggregation via payment relays, which shifts the vast majority of micro-transactions off-chain while preserving on-chain verifiability, accountability, and dispute resolution. This design allows the blockchain to handle only aggregated balance updates and control operations, rather than every individual service interaction, effectively decoupling service-level transaction volume from blockchain throughput constraints and enabling the system to scale to multi-million TPS at the service layer without violating decentralization or security assumptions.

Considering the requirements of SOVEREIGN, the blockchain platform should be implemented on an EVM-compatible client that supports a Byzantine Fault Tolerant (BFT) Delegated Proof-of-Stake (DPoS) consensus mechanism and enables smart-contract–based governance and management of the on-chain validator set. This capability is necessary to ensure that validators can be elected, updated, and revoked deterministically via contract logic that accounts for: (i) stakes provided by validator candidates, (ii) stakes delegated by witnesses, and (iii) penalties accrued through protocol-defined slashing mechanisms.

The chosen consensus protocol must deliver deterministic finality, low-latency block confirmation, and configurable consensus parameters, including block period, epoch length, and timeout thresholds. Finally, the selected EVM client should be actively maintained and preferably enterprise-grade, while remaining fully compatible with standard Ethereum development and testing tools (e.g., Hardhat, Foundry, Truffle, Remix) and widely adopted web3 libraries, in order to streamline smart-contract development, system integration, and long-term maintainability.

3.2. Partitioning and Functionality of the SOVEREIGN Blockchain (High-level)

SOVEREIGN’s blockchain functionality is partitioned across two core smart contracts: the Validators Smart Contract (VSC) and the Relay Smart Contract (RSC). This separation is motivated by scalability, modularity, and security considerations. The VSC is solely responsible for consensus- and governance-related operations, enabling a lightweight, energy-efficient DPoS mechanism tailored to highly heterogeneous 5G and Beyond network nodes. In contrast, the RSC focuses on transaction scalability through payment relays and off-chain aggregation. By decoupling consensus enforcement from high-frequency payment operations, the system avoids overloading a single contract with conflicting requirements, reduces on-chain execution complexity, and limits the attack surface of critical consensus functions. This modular approach also allows each contract to evolve independently in response to different performance and security demands, while collectively supporting a fully decentralized, high-throughput, and robust mobile data trading ecosystem.

The Validators Smart Contract (VSC) should govern the lifecycle, selection, and operation of validator nodes responsible for block production. Its functionality can be divided into two categories: operations available to validators and operations available to validator supporters. Validator functions include establishing validator candidacy by locking a minimum stake, accepting a validator role, withdrawing validator stakes from previous epochs, voting on adjustable protocol parameters through epoch-based amendment ballots, reporting benign or malicious behavior by other validators (with support for proof-based maliciousness reports), and replacing lower-staked validators following slashing events. The contract also exposes read-only access to epoch validator sets and candidate lists, enabling transparent inspection and auditability of active and historical validator configurations. Supporter functions allow network participants to vote for a validator by staking either as a witness or as a service provider, with service providers optionally committing measurable free-service capacity as part of their support. Supporters may withdraw witness or service-provider stakes from previous epochs, and in the case of witnesses, request payment for backing a winning validator. Finally, the VSC enables all eligible participants to claim pending credits or rewards accrued from prior protocol interactions, ensuring correct economic settlement across epochs.

The Relay Smart Contract (RSC) should govern the lifecycle and operation of relay nodes, which are responsible for routing off-chain transactions and servicing payment channels. Its functionality is organized into three interrelated groups: Relay Elections, Relay Core, and Relay Channels, supported by signature and stake-calculation libraries. The Relay Elections group manages relay candidacy and selection: relays announce themselves as candidates by staking according to a required-stake curve, publishing service parameters (including capacity limits, throughput, delay bounds, and fees), and, once elections close, requesting licensing to become active relays for the subsequent epoch. In parallel, network participants may act as relay supporters by staking and voting either as witnesses or free service providers; they may later withdraw their stakes from past epochs and, for witnesses, claim reward payments if they supported a winning relay. The Relay Core group serves as the canonical on-chain registry of elected relays and their epoch-specific parameters. It enables elections to finalize relay sets and provides the authoritative state required for penalty enforcement by the channels. The contract stores each relay’s fee and capacity commitments, tracks active and revoked status, exposes read-only access to relay configurations, and accepts authorized updates such as stake reductions, capacity decreases, license revocations, and delayed-payment notifications. The Relay Channels group implements the live economic relationships between users, servers, and relays through inbound and outbound payment channels. Clients deposit funds to relays and withdraw once channels are unlocked; servers open or renew channels, reallocate or release outbound funds, and execute signed off-chain channel state updates. The contract enforces relay throughput limits and maintains on-chain dispute resolution mechanisms for delayed payments and over-withdrawals. Accusers may open disputes, relays may respond with Merkle-proof-based evidence, and either party may settle disputes or claim refunds, with stake-backed penalties applied where violations are proven. Collectively, these functional groups ensure that relays are competitively elected, adequately collateralized, transparently parameterized on a per-epoch basis, and economically constrained to service off-chain payments reliably under on-chain accountability and dispute resolution.

3.3. Required Functionality of the Validators Smart Contract

The VSC is responsible for consensus-related operations in the SOVEREIGN blockchain. It incorporates both fixed and adjustable parameters. Fixed parameters are used to safeguard system robustness against inadvertent or malicious behaviors by validators and are hard-coded into the VSC. Adjustable parameters enable flexible operation of the DPoS consensus protocol in view of the current state of the SOVEREIGN blockchain, and a specific amendment procedure is followed to update their values. Table 3 summarizes the key VSC parameters for a tagged epoch *e*.

Table 3. VSC parameters for a tagged epoch e.

Parameter	**Notation**	**Type**
Epoch duration (in blocks)	B_V	Fixed
Election window deadline (in blocks)	T_V	Fixed
Baseline emission rate (coins per block)	R_V	Fixed
Table of disinflation rates (percentages)	D_V	Fixed
Number of consecutive epochs for amendment	C_V	Fixed
Transaction fee paid for direct payments	c_{min}	Fixed
Current No. of validators	$V[e]$	Adjustable
Max No. of validators	V_{max}	Fixed
Min No. of validators	V_{min}	Fixed
Current baseline penalty for offline validators	$P_{Vo}[e]$	Adjustable
Min baseline penalty for offline validators	P_{Vo}^{min}	Fixed
Current baseline penalty for malicious validators	$P_{Vm}[e]$	Adjustable
Min baseline penalty for malicious validators	P_{Vm}^{min}	Fixed
Minimum stake for validators	$M_V[e]$	Adjustable
Minimum stake for v-witnesses	$W_V[e]$	Adjustable
Free-of-charge service tariff per MB	$f_V[e]$	Adjustable

The SOVEREIGN DPoS mechanism operates on an epoch-by-epoch basis, where each *validation epoch* lasts for exactly B_V blocks. A validation epoch represents the time interval during which a given set of validators is authorized to seal blocks. To be elected for a target epoch *e*, the validators participate in an auction-based scheme that starts from the first block of period *e-1* and concludes exactly T_V blocks before the beginning of the target epoch *e*. Parameter R_V defines the number of newly minted coins generated per block in the first epoch. This value is subsequently adapted according to the disinflation table D_V, which specifies a disinflation rate applied on R_V as a function of the epoch number. Sealed blocks specifying a different amount of new coins (validator rewards) from this value are considered invalid. This mechanism gradually reduces the number of new coins generated per block in the long term. R_V, and D_V are fixed parameters, ensuring a predictable supply of new coins in the system and discouraging validators from voting in favor of a higher block reward. Similarly, the minimum transaction fee c_{min} for direct on-chain payments is fixed for the same reason. The value of c_{min} should be tuned so as to enforce the use of payment relay services, allowing the SOVEREIGN blockchain to scale with the transactions generated by the myriads of SOVEREIGN service peers. C_V is a fixed parameter defining the number of consecutive blocks that are necessary to amend an adjustable VSC parameter (amendment procedure described below).

The number of validators $V[e]$ plays a key role in the performance of the SOVEREIGN blockchain. A low number of validators increases the risk for block sealing failures due to inadvertent and malicious behaviors of the validators but enables the system to attain a higher transactions capacity. A large number of validators can safeguard system robustness against failures and dishonest behaviors by the validators, but also reduce the incentives offered to validators towards block sealing (i.e. rewards decrease proportionally). The VSC enables the validators to amend the value of $V[e]$, keeping it within specific VSC-defined limits (i.e. $[V_{min}, V_{max}]$).

The VSC logic distinguishes between the two scenarios where a validator i) fails to deliver a sealed block on time, e.g. service outage, or ii) seals an invalid block. A lower penalty $P_{Vo}[e]$ is employed in the first scenario, to discourage validators that lack the required functional capacity to perform block sealing, whereas a higher penalty $P_{Vm}[e]$ is employed for the second scenario, to quickly revoke the license of dishonest validators and exclude them from block sealing. Both parameters can be amended by the validators, but they should be above the prescribed VSC-defined thresholds P_{Vo}^{min} and P_{Vm}^{min}, respectively.

Active participation in the DPoS mechanism, either by setting candidacy as a validator, or by acting as a v-witness, requires on-chain locking of funds to the VSC. The VSC enforces a minimum stake for both candidate validators and v-witnesses, denoted by $M_V[e]$ and $W_V[e]$, respectively, aiming to discourage nodes from acting dishonestly. Parameter $f_V[e]$ specifies the amount of coins that a FoC server should lock onto the VSC for a given FoC service promise: by locking X coins, the FoC server promises $X/f_V[e]$ MBs per v-witness if the respective candidate validator gets elected.

Beyond parameter management, the VSC exposes a comprehensive set of read (view) and write (state-changing) functions that implement its operational logic. Read functions modify on-chain state and do not consume gas, while write functions update contract state and therefore incur transaction costs.

The read (view) functions include:

1. Function “getCandidates(uint256 epoch)”, which returns the list of validator candidate addresses for a given epoch.
2. Function “getValidatorFreeServiceProviders(uint256 epoch, address candidate)”, which returns the service-provider addresses backing a given candidate for a given epoch.
3. Function “getValidatorWitnesses(uint256 epoch, address candidate)”, which returns the witness addresses for a given candidate for a given epoch.
4. Function “getValidatorsByEpoch(uint256 epoch)”, which returns the shortlisted validator set (winners by total funds) for a given epoch.
5. Function “getActiveValidatorsByEpoch(uint256 epoch)”, which returns the validators that accepted their role as validators for a given epoch.
6. Function “getValidators()”, which returns the currently operational validator set, the validators that produce blocks.
7. Function “getValidatorsNumber(uint256 epoch)”, which returns the number of shortlisted

validators (winners by total funds) for a given epoch.

8. Function “calculateReward(uint256 issuanceBlock, uint256 lastClaimedIssuanceBlock)”, which computes total block rewards from the block after lastClaimedIssuanceBlock up to and including issuanceBlock, based on the current epoch’s disinflation rate.
9. Function “getDisinflationRate(uint256 epoch)”, which returns the disinflation rate based on epoch, 100 for epoch ≤ 100, 37 for 101…365, and 0 for > 365.
10. Function “getCurrentBlockNumber()”, which returns the current block number.
11. Function “getCurrentEpoch()”, which returns the current epoch, derived from the current block number and the number of blocks per epoch.
12. Function “getEpochByBlock(uint256 requestedBlock)”, which returns the epoch that the given block belongs to.
13. Function “getTargetEpoch()”, which returns the next epoch (current + 1).
14. Function “getCurrentEpochEnd()”, which returns the block number at which the current epoch ends.
15. Function “getCurrentValidatorsElectionEnd()”, which returns the block number at which the current validators' election window closes.

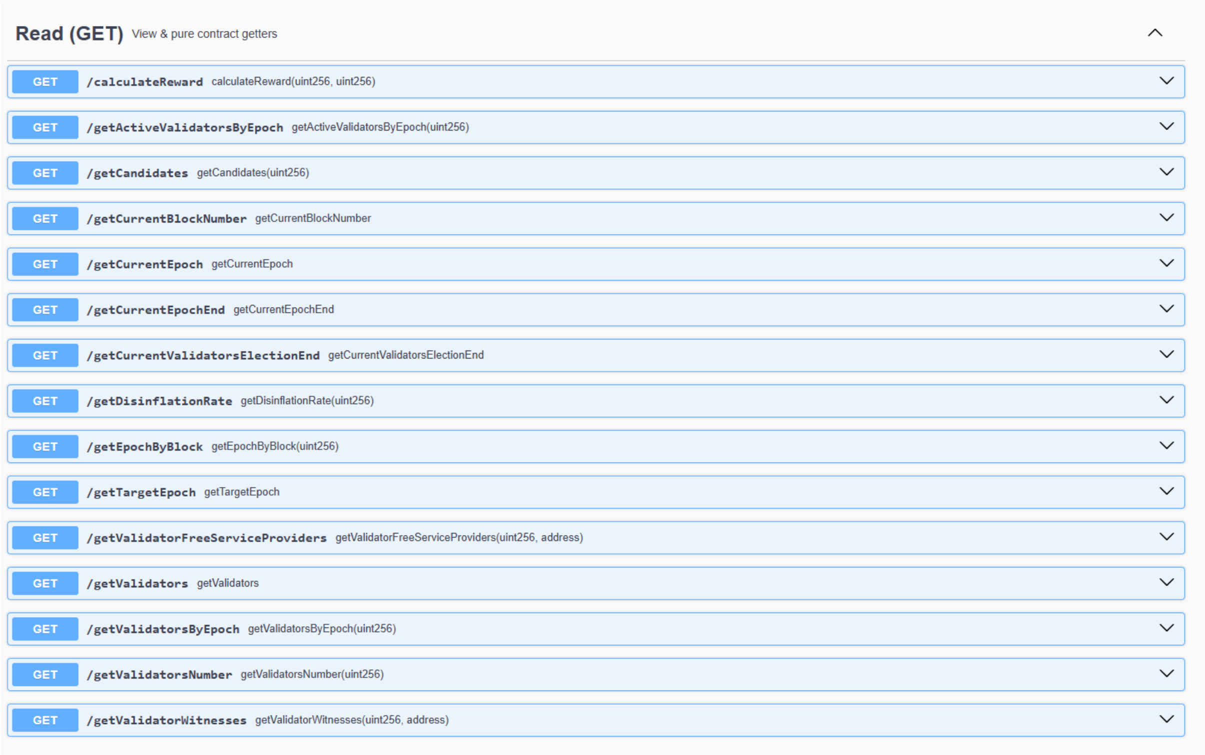
The write (state-changing) functions include:

1. Function “validatorAsCandidate(uint256 stakingFunds, uint256 witnessesFunds, string name)“, which registers the caller node as a next-epoch validator candidate by staking funds for the epoch (stakingFunds) and promising funds to their supporters (witnessesFunds). The function also records a validator's name and attempts to place/replace them in the next-epoch shortlist by total funds.
2. Function “acceptValidator()”, which provides a way for a validator that is among the winning candidates for the current epoch to accept the role and become an active and operational validator.
3. Function “validatorWithdrawRequest(uint256 epoch)”, which provides a way for a validator to withdraw remaining validator stake for a given past epoch. If the validator has no stake in the current epoch, this action will also remove them from the operational validator list.
4. Function “replaceValidator(address validator)”, which provides a way for a validator candidate who has more total funds this epoch (due to penalty changes) to replace the given current validator in the shortlist. If the replacement is successful, the candidate should also call acceptValidator() to become an active and operational validator.
5. Function “voteValidatorAsWitness(address validator)”, which registers a caller node as a witness for a next-epoch candidate by sending at least the witness minimum stake. Also, it increases the candidate’s total staking funds and may push them into the shortlist.
6. Function “voteValidatorAsServiceProvider(address validator, uint256 freeContentInMb)”, which registers a caller node as a Service Provider, who stakes exactly freeContentInMb * v_pricePerMb (the price per Mb is a contract variable that can be adjusted by the validators) funds to back a next-epoch validator candidate with free-content Mb. Also, it increases the candidate’s total stake funds and may push them into the shortlist.
7. Function “vWitnessPaymentRequest(uint256 epoch, address validator)”, which provides a way for a witness to claim their reward share for the given epoch and validator, proportional

to their witness stake among unpaid witnesses.

8. Function "vWitnessWithdrawRequest(uint256 epoch, address validator)", which provides a way for a witness to withdraw their remaining witness stake for a past epoch and validator.
9. Function "vFreeServiceProviderWithdrawRequest(uint256 epoch, address validator)", which provides a way for a service provider to withdraw unspent Service Provider funds for a past epoch and validator.
10. Function "voteAmendment(uint256 parameter, uint256 change)", which provides a way for a current validator to vote for increasing or decreasing one of the adjustable chain parameters (see Table 3). The contract counts votes per epoch. When votes > 50% in an epoch, it updates a streak counter. If there are enough consecutive epoch majorities, the parameter is adjusted.
11. Function "reportBenign(address validator, uint256 blockNumber)", which provides a way for a current validator to report another for a recent benign misbehavior for a specific block. Once more than or equal to 50% report the validator, a penalty is applied, and the funds are redistributed among the other validators. The reported validator is removed if he can't cover the penalty.
12. Function "reportMalicious(address validator, uint256 blockNumber, bytes proof)", which provides a way for a current validator to report another for a recent malicious misbehavior for a specific block, providing proof. Once more than or equal to 50% report the validator, a penalty is applied, and the funds are redistributed among the other validators. The reported validator is removed if he can't cover the penalty.

Finally, a UI API snapshot of the VSC functions is provided using Swagger, an open ecosystem of tools for designing, building, and documenting RESTful APIs. Through this interface, users can invoke contract functions and query blockchain state. Figure 5 presents the read (GET) functions, which do not require gas, while Figure 6 presents the write (POST) functions, which require gas and a sufficient account balance for execution.



Read (GET) View & pure contract getters

GET /calculateReward calculateReward(uint256, uint256)

GET /getActiveValidatorsByEpoch getActiveValidatorsByEpoch(uint256)

GET /getCandidates getCandidates(uint256)

GET /getCurrentBlockNumber getCurrentBlockNumber

GET /getCurrentEpoch getCurrentEpoch

GET /getCurrentEpochEnd getCurrentEpochEnd

GET /getCurrentValidatorsElectionEnd getCurrentValidatorsElectionEnd

GET /getDisinflationRate getDisinflationRate(uint256)

GET /getEpochByBlock getEpochByBlock(uint256)

GET /getTargetEpoch getTargetEpoch

GET /getValidatorFreeServiceProviders getValidatorFreeServiceProviders(uint256, address)

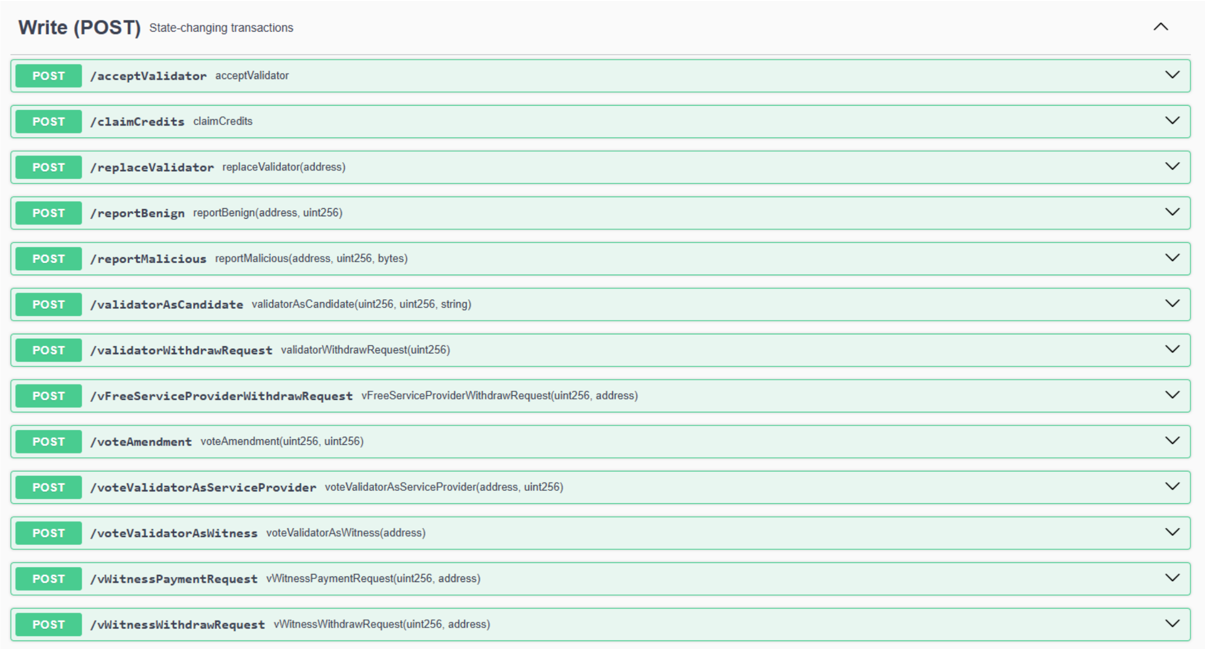
GET /getValidators getValidators

GET /getValidatorsByEpoch getValidatorsByEpoch(uint256)

GET /getValidatorsNumber getValidatorsNumber(uint256)

GET /getValidatorWitnesses getValidatorWitnesses(uint256, address)

Figure 5. Read functions of the Validators Smart Contract.



Write (POST) State-changing transactions

POST /acceptValidator acceptValidator

POST /claimCredits claimCredits

POST /replaceValidator replaceValidator(address)

POST /reportBenign reportBenign(address, uint256)

POST /reportMalicious reportMalicious(address, uint256, bytes)

POST /validatorAsCandidate validatorAsCandidate(uint256, uint256, string)

POST /validatorWithdrawRequest validatorWithdrawRequest(uint256)

POST /vFreeServiceProviderWithdrawRequest vFreeServiceProviderWithdrawRequest(uint256, address)

POST /voteAmendment voteAmendment(uint256, uint256)

POST /voteValidatorAsServiceProvider voteValidatorAsServiceProvider(address, uint256)

POST /voteValidatorAsWitness voteValidatorAsWitness(address)

POST /vWitnessPaymentRequest vWitnessPaymentRequest(uint256, address)

POST /vWitnessWithdrawRequest vWitnessWithdrawRequest(uint256, address)

Figure 6. Write functions of the Validators Smart Contract.

3.4. Lifecycle of the VSC and Execution Flow

The lifecycle of the Validators Smart Contract (VSC) is executed on an epoch-by-epoch basis through a sequence of on-chain function calls that govern validator candidacy, election, activation, block production, reward distribution, governance, and penalization. Prior to the start of a target validation epoch e, candidate validators register their candidacy by invoking the *validatorAsCandidate(...)* function, locking the required validator stake and committing funds for validator supporters. During this pre-epoch phase, network participants may support validator candidates by staking as v-witnesses or free-of-charge service providers through the *voteValidatorAsWitness(...)* and *voteValidatorAsServiceProvider(...)* functions, thereby increasing the candidate's total backing and influencing shortlist ranking.

Once the validator election window closes T_V blocks before epoch e, the VSC deterministically shortlists validator candidates based on total staked funds. Shortlisted candidates must explicitly accept their role by calling *acceptValidator()*, after which they become active validators authorized to seal blocks. The active validator set for each epoch can be inspected using the read functions *getValidatorsByEpoch(...)*, *getActiveValidatorsByEpoch(...)*, and *getValidators()*, ensuring transparency and auditability of the consensus configuration.

During an active validation epoch, validators participate in block production and receive block rewards. Validators may also engage in on-chain governance by voting on adjustable VSC parameters using the *voteAmendment(...)* function. The amendment logic tracks majority support across consecutive epochs and applies parameter changes only after the required threshold C_V is satisfied, thereby preventing abrupt or malicious protocol changes.

The VSC continuously enforces validator accountability by distinguishing between benign and malicious misbehavior. Validators may report missed or delayed block production events by invoking *reportBenign(...)*, or report invalid block sealing by invoking *reportMalicious(...)* and submitting cryptographic proof. Upon reaching the required reporting threshold, the VSC applies penalties $P_{Vo}[e]$ or $P_{Vm}[e]$, redistributes penalized funds among honest validators, and, if necessary, removes validators who can no longer cover the penalty. In cases where stake changes alter validator ranking, higher-backed candidates may replace penalized validators through the *replaceValidator(...)* function, followed by *acceptValidator()* to activate the replacement.

At the conclusion of an epoch, validators and their supporters may withdraw eligible funds from past epochs. Validators reclaim unused or remaining stake using *validatorWithdrawRequest(...)*, while v-witnesses and free service providers withdraw their stakes or claim rewards using *vWitnessWithdrawRequest(...)*, *vFreeServiceProviderWithdrawRequest(...)*, and *vWitnessPaymentRequest(...)*. All participants may query epoch boundaries and reward eligibility through functions such as *getCurrentEpoch()*, *getCurrentEpochEnd()*, and *getEpochByBlock(...)*.

Through this sequence of function invocations, the VSC implements a complete validator lifecycle that combines deterministic validator selection, stake-backed accountability, on-chain governance, and transparent reward distribution. This execution flow ensures that block production remains decentralized, economically secure, and adaptable to the evolving operational requirements of the SOVEREIGN blockchain.

3.5. Required Functionality of the Payment Relays Smart Contract

The RSC focuses on transaction scalability through licensed payment relays and off-chain payment aggregation. Table 4 summarizes the main RSC parameters for a tagged epoch e. Most parameters remain fixed, with the exception of the adjustable RSC parameter $TC_R[e]$, which is used to estimate the current transactions capacity of the blockchain system. Similar to the VSC logic, payment relays are assumed to participate in an auction-based selection process to receive a payment relay license for a target epoch e. The duration of each relay epoch is fixed and equal to B_R blocks. The relay licensing epoch should conclude T_R blocks before the beginning of the target epoch. To support payment promises issued close to the end of an epoch, payment relays are enabled to withdraw their funds only after the end of epoch e plus G_R blocks.

When an on-chain dispute resolution is triggered, $D_R(<G_R)$ is used as time window enabling the reported payment relay to submit signed proofs of its honest operation. D_{max} defines the maximum delay window (in blocks) within which a payment relay can post the outcome of an off-chain transaction on-chain. F_{max} defines the maximum transactions fee that a payment relay can claim per off-chain payment it processes. M_R and W_R define a minimum stake for candidate relays and r-witnesses, respectively. p_R is used as the basis of calculating exponentially increasing penalties to dishonest payment relays, whereas f_R specifies the coins per MB ratio that FoC servers should timelock (similar to the VSC logic). k_R specifies the interval of blocks within which the RSC measures the transactions posted by a tagged payment relay server, enabling evaluation of the mean transactions throughput per relay. $TC_R[e]$ is adapted by the RSC logic on an epoch-by-epoch basis and is used to conclude on the set of elected payment relays.

Table 4. RSC parameters for a tagged epoch e.

Parameter	Notation	Type
Relay epoch duration (in blocks)	B_R	Fixed
Election window deadline (in blocks)	T_R	Fixed
Relay withdrawal guard interval (in blocks)	G_R	Fixed
Dispute resolution time window (in blocks)	D_R	Fixed
Max relay delay threshold (in blocks)	D_{max}	Fixed
Max transactions fee for off-chain payments	F_{max}	Fixed
Minimum stake for relay	M_R	Fixed
Minimum stake for r-witnesses	W_R	Fixed
Baseline relay penalty (in coins)	p_R	Fixed
Free-of-charge service tariff per MB	f_R	Fixed
Relay monitoring period (in blocks)	k_R	Fixed
Current Transactions Capacity	$TC_R[e]$	Adjustable
Relay license tariff table for max users	T_{users}	Fixed
Relay license tariff table for max coins	T_{coins}	Fixed
Relay license tariff table for max throughput	$T_{throughput}$	Fixed

The RSC also includes three tariff tables used to calculate the minimum guarantee fund that the candidate payment relays should lock onto the RSC for relay penalty purposes. T_{users} adapts the required guarantee fund in line with the maximum number of SOVEREIGN clients that the payment relay requests to serve. T_{coins} adapts the required guarantee fund in line with the maximum amount of coins that the payment relay requests to handle. $T_{throughput}$ adapts the required guarantee fund in line with the transactions throughput that the payment relay requests to spur into the SOVEREIGN blockchain.

Beyond these parameters, the RSC comprises a set of read (view) and write (state-changing) functions implementing relay elections, relay registration, channel management, throughput enforcement, and dispute resolution. Read functions do not modify on-chain state and do not require funds for execution, while write functions update contract state and therefore consume gas.

The read (view) functions include:

1. Function “getCurrentRelayersElectionEnd()”, which returns the block number at which the current relays' election window closes.
2. Function “getCandidates(uint256 epoch)”, which returns the array of candidate relay addresses registered for a given epoch.
3. Function “getRelayerFreeServiceProviders(uint256 epoch, address relayer)”, which returns the list of free service providers who supported a given relay in a specific epoch.
4. Function “getRelayerWitnesses(uint256 epoch, address relayer)”, which returns the list of witnesses who supported a given relay in a specific epoch.
5. Function “getRelayerGasThroughput(uint256 epoch, address relayer)”, which returns the relay’s current throughput counters, last sub-epoch block, max allowed throughput, and current sub-epoch number.

The write (state-changing) functions include:

1. Function “relayerAsCandidate(uint256 stakingFunds, uint256 witnessesFunds, string name, string domain, uint256 maxUsers, uint256 maxCoins, uint256 maxTxThroughput, uint256 offchainTxDelay, uint256 fee)”, which registers the caller node as a next-epoch relay candidate by submitting parameters and locking stake and initial witness-reward funds. It also enforces on-chain minimum stake.
2. Function “voteRelayerAsWitness(address relayer)”, which lets a witness support a candidate relay for the next epoch by staking funds, increasing the relay’s total stake and witness totals, and registering the witness if first-time.
3. Function “voteRelayerAsServiceProvider(address relayer, uint256 freeContentInMb)”, which lets a service provider support a candidate relay by staking funds equal to the value of the free content they pledge (priced per MB), updating the relay’s total stake, and registering the provider if first-time.
4. Function “rWitnessPaymentRequest(uint256 epoch, address relayer)”, which allows a witness to claim their pro-rata reward from the relay’s witness-reward budget after or during the specified epoch, based on the witness’s stake relative to all unclaimed witness stakes.
5. Function “rWitnessWithdrawRequest(uint256 epoch, address relayer)”, which lets a witness withdraw their original witness stake for a past epoch, zeroing their recorded stake after payout.
6. Function “rFreeServiceProviderWithdrawRequest(uint256 epoch, address relayer)”, which

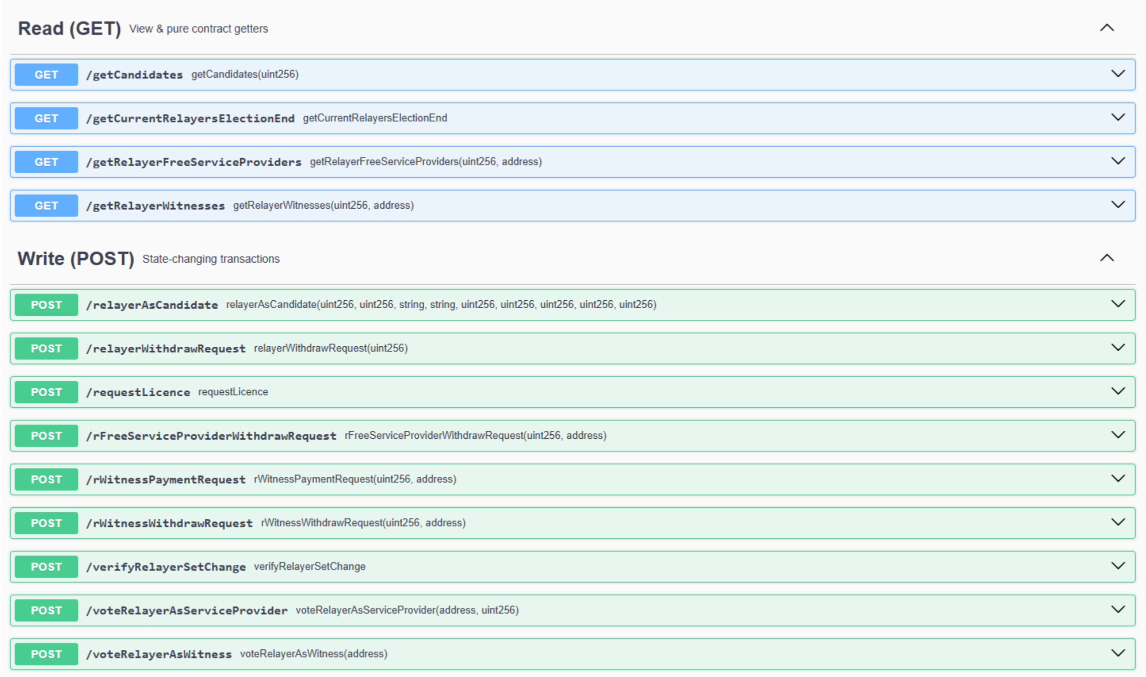
lets a free service provider withdraw their contributed funds for a past epoch, clearing their recorded contribution after payout.

7. Function “requestLicence()”, which lets a relay candidate formally request to be placed on the shortlist, which inserts them into a stake-sorted linked list used for ratification, once the election has finished.
8. Function “verifyRelayerSetChange()”, which finalizes the relay set for the current epoch: it adjusts global transaction capacity based on the last epoch’s delayed payments, then walks the stake-sorted shortlist and elects as many relays as fit into remaining capacity and finally marks the change as ratified.
9. Function “relayerWithdrawRequest(uint256 epoch)”, which allows a relay to withdraw their own stake from a past epoch, clearing the value afterward.
10. Function “depositToRelayer(address relayerId, uint256 /*lockUntilBlock*/)”, which lets the user deposit funds into a relay’s inbound channel for the current epoch, ensuring the relay is active, the deposit is large enough, and the relay’s inbound user/coin limits are respected, then locks the funds until epoch end and updates totals.
11. Function “withdrawFundsFromRelayer(uint256 epoch, address relayerId)”, which allows a user to withdraw their inbound deposit from a relay once the lock period for that epoch is over, zeroing the stored balance and paying funds out.
12. Function “depositToServer(address relayerId, uint256 lockUntilBlock)”, which lets an active relay deposit funds into an outbound channel toward a server for the current epoch, requiring enough stake, a valid future lock that doesn’t exceed epoch end, no lock-shortening, and ensuring outbound funds never exceed inbound funds.
13. Function “withdrawFundsFromServer(uint256 epoch, address relayerId, uint256 amount)”, which allows a relay to withdraw part of its outbound deposit to a server after the lock expires, and updates outbound totals accordingly.
14. Function “reallocateServerFunds(address serverIdOld, address serverIdNew, uint256 amount, uint256 lockUntilBlock)”, which moves outbound funds from one server channel to another during the current epoch, ensuring the old channel is unlocked, balances are sufficient, the relay is still staked, the moved amount is meaningful, and the new channel’s lock is valid and not shortened.
15. Function “releaseServerFunds(address[] serverIds, uint256[] amounts, bytes32[] merkleRoots)”, which lets Relay batch-releases outbound funds to servers by validating inputs, checking total and per-server balances and expiry, recording per-server update roots and amounts, then enforcing gas throughput and reducing outbound totals.
16. Function “updateClientChannel(address[] clientIds, uint256[] amounts, uint256[] amountOfTxs, bytes32[] merkleRoots)”, which lets relay batch-updates inbound client channels by charging each client an amount plus fees, ensuring inbound totals remain solvent relative to outbound totals, ensuring each client has enough balance, recording update roots, enforcing throughput, then paying the relay the collected funds.
17. Function “serverWithdraw()”, which lets a server withdraw all funds that relays have released to it and accumulated in its tab, then clears the tab.
18. Function “reportDelayedPayment(address signer, address relayer, bytes32 h, uint8 v, bytes32 r, bytes32 s, bytes32 rh, uint8 rv, bytes32 rr, bytes32 rs, bytes32 nonce, uint256 fee, uint256

txUntilBlock, address beneficiary, uint256 amount)”, which lets a server open a delayed-payment dispute by verifying both the user’s and relay’s off-chain signatures, confirming the payment deadline passed, and ensuring this transaction hasn’t been disputed before, then records the dispute state.

19. Function “relayRespondDispute_verifyUserPayloads(address[] calldata clients, bytes32[] calldata h, uint8[] calldata v, bytes32[] calldata r, bytes32[] calldata s, bytes32[] calldata nonce, uint256[] calldata fee, address[] calldata beneficiary, uint256[] calldata amount, uint8 disputeType)”, which lets a Relay verifies a set of user payment payloads by checking their signatures, aggregating their amounts and fees, optionally ensuring no transaction is being double-spent in client disputes, then stores the verified totals under the user Merkle root and returns it.
20. Function “relayRespondDispute_verifyRelayPayloads(bytes32 userPayloadMerkleHash, bytes32[] calldata h, bytes32[] calldata rh, uint8[] calldata rv, bytes32[] calldata rr, bytes32[] calldata rs, uint256[] calldata txUntilBlock)”, which lets a Relay prove its own relay-signed payloads correspond to a previously verified user payload set, verifies each relay signature, then stores and returns the relay Merkle root.
21. Function “relayRespondDelayedPayment_settleDispute(address accuser, bytes32 transactionHash, bytes32 merkleHash, uint128 index, bytes32[] h, bytes32[] rh)”, which lets, after both payload verifications, relay settle a delayed-payment dispute for a specific tx index, applying penalties or closing the dispute.
22. Function “serverRefund(uint256 targetEpoch, bytes32 rh)”, which lets Server refund path after the dispute window if the relay didn’t settle, requiring relay existence in that epoch, still-guarded stake, a valid open dispute, period expiry, non-zero claim, and that the relay hasn’t already settled, then applies penalties and closes.
23. Function “reportOverwithdraw(uint256 targetEpoch, address relayerId, uint256 updateId, bytes32 merkleHash)”, which lets Client open an over-withdrawal dispute against a relay’s client update, only if the relay existed in that epoch, the opening window is still valid, and the provided Merkle root matches the stored update root.
24. Function “respondOverwithdraw_settleDispute(address payable accuser, bytes32 merkleHash, uint128 index, bytes32[] h)”, which lets Relay settle a client over-withdrawal dispute inside the window by proving the disputed root, proving a verified user payload root, matching the stored update index, penalizing if the update overcharged compared to true payload totals, then marking transactions spent and settling the root.
25. Function “clientRefund(uint256 targetEpoch, bytes32 merkleHash)”, which lets Client refund path after dispute expiry if the relay didn’t settle, requiring relay existence, stake still guarded, a valid open dispute, period expiry, non-zero amount, and no prior settlement, then penalizes and closes.

Finally, we provide a UI API snapshot for the RSC functions using Swagger.



Read (GET) View & pure contract getters

GET /getCandidates getCandidates(uint256)

GET /getCurrentRelayersElectionEnd getCurrentRelayersElectionEnd

GET /getRelayerFreeServiceProviders getRelayerFreeServiceProviders(uint256, address)

GET /getRelayerWitnesses getRelayerWitnesses(uint256, address)

Write (POST) State-changing transactions

POST /relayerAsCandidate relayerAsCandidate(uint256, uint256, string, string, uint256, uint256, uint256, uint256, uint256)

POST /relayerWithdrawRequest relayerWithdrawRequest(uint256)

POST /requestLicence requestLicence

POST /rFreeServiceProviderWithdrawRequest rFreeServiceProviderWithdrawRequest(uint256, address)

POST /rWitnessPaymentRequest rWitnessPaymentRequest(uint256, address)

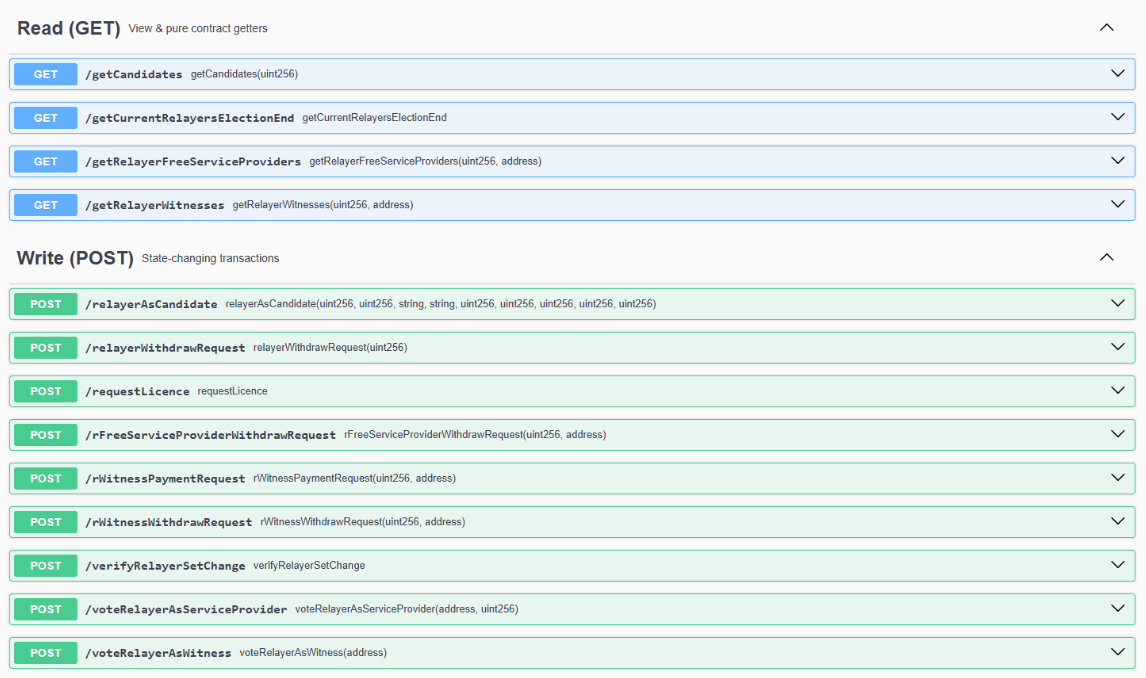
POST /rWitnessWithdrawRequest rWitnessWithdrawRequest(uint256, address)

POST /verifyRelayerSetChange verifyRelayerSetChange

POST /voteRelayerAsServiceProvider voteRelayerAsServiceProvider(address, uint256)

POST /voteRelayerAsWitness voteRelayerAsWitness(address)

Figure 7 presents the read (GET) and write (POST) functions related to the election processes, while Figure 8 presents the read (GET) and write (POST) functions related to the payment-channel processes of the RSC.



Read (GET) View & pure contract getters

GET /getCandidates getCandidates(uint256)

GET /getCurrentRelayersElectionEnd getCurrentRelayersElectionEnd

GET /getRelayerFreeServiceProviders getRelayerFreeServiceProviders(uint256, address)

GET /getRelayerWitnesses getRelayerWitnesses(uint256, address)

Write (POST) State-changing transactions

POST /relayerAsCandidate relayerAsCandidate(uint256, uint256, string, string, uint256, uint256, uint256, uint256, uint256)

POST /relayerWithdrawRequest relayerWithdrawRequest(uint256)

POST /requestLicence requestLicence

POST /rFreeServiceProviderWithdrawRequest rFreeServiceProviderWithdrawRequest(uint256, address)

POST /rWitnessPaymentRequest rWitnessPaymentRequest(uint256, address)

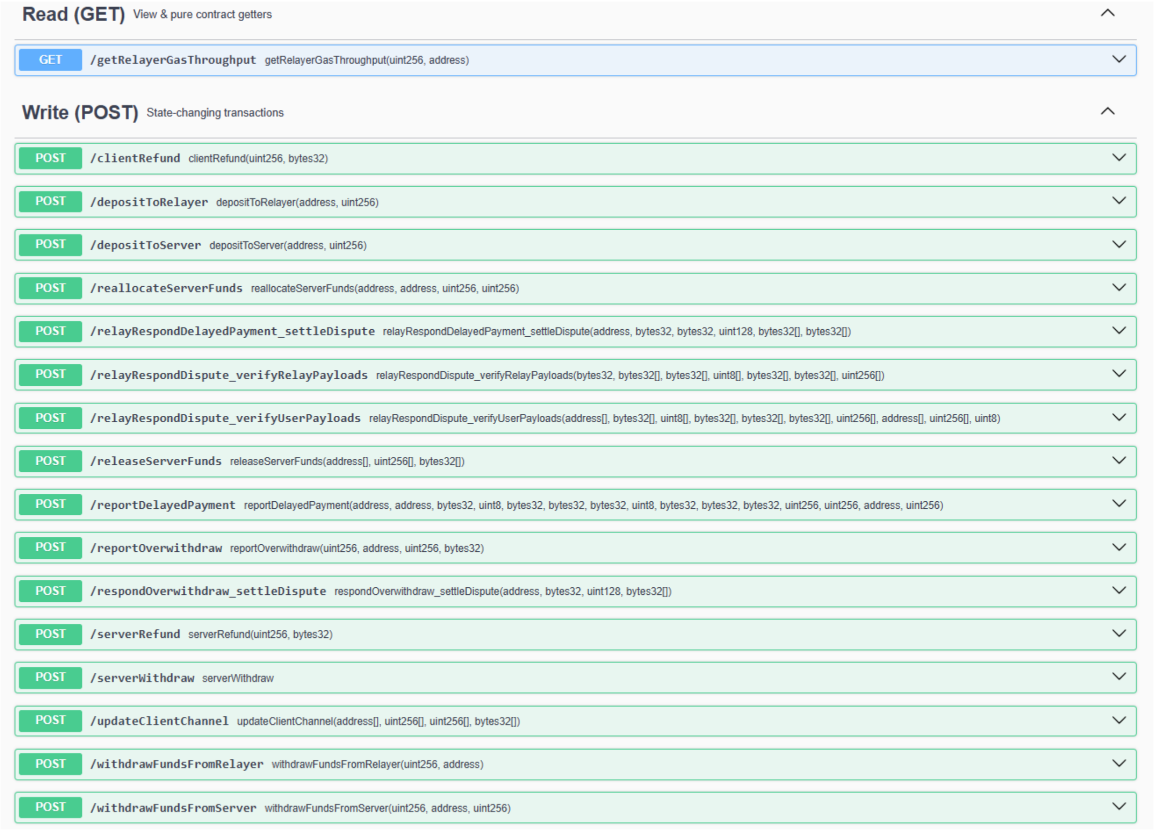
POST /rWitnessWithdrawRequest rWitnessWithdrawRequest(uint256, address)

POST /verifyRelayerSetChange verifyRelayerSetChange

POST /voteRelayerAsServiceProvider voteRelayerAsServiceProvider(address, uint256)

POST /voteRelayerAsWitness voteRelayerAsWitness(address)

Figure 7. Read and Write functions of the Relay Smart Contract for the Election processes.



Read (GET) View & pure contract getters

GET /getRelayerGasThroughput getRelayerGasThroughput(uint256, address)

Write (POST) State-changing transactions

POST /clientRefund clientRefund(uint256, bytes32)

POST /depositToRelayer depositToRelayer(address, uint256)

POST /depositToServer depositToServer(address, uint256)

POST /reallocateServerFunds reallocateServerFunds(address, address, uint256, uint256)

POST /relayRespondDelayedPayment_settleDispute relayRespondDelayedPayment_settleDispute(address, bytes32, bytes32, uint128, bytes32[], bytes32[])

POST /relayRespondDispute_verifyRelayPayloads relayRespondDispute_verifyRelayPayloads(bytes32, bytes32[], bytes32[], uint8[], bytes32[], bytes32[], uint256[])

POST /relayRespondDispute_verifyUserPayloads relayRespondDispute_verifyUserPayloads(address[], bytes32[], uint8[], bytes32[], bytes32[], bytes32[], uint256[], address[], uint256[], uint8)

POST /releaseServerFunds releaseServerFunds(address[], uint256[], bytes32[])

POST /reportDelayedPayment reportDelayedPayment(address, address, bytes32, uint8, bytes32, bytes32, bytes32, uint8, bytes32, bytes32, bytes32, uint256, uint256, address, uint256)

POST /reportOverwithdraw reportOverwithdraw(uint256, address, uint256, bytes32)

POST /respondOverwithdraw_settleDispute respondOverwithdraw_settleDispute(address, bytes32, uint128, bytes32[])

POST /serverRefund serverRefund(uint256, bytes32)

POST /serverWithdraw serverWithdraw

POST /updateClientChannel updateClientChannel(address[], uint256[], uint256[], bytes32[])

POST /withdrawFundsFromRelayer withdrawFundsFromRelayer(uint256, address)

POST /withdrawFundsFromServer withdrawFundsFromServer(uint256, address, uint256)

Figure 8. Read and Write functions of the Relay Smart Contract for the payment Channels processes.

3.6. Lifecycle of the RSC and Execution Flow

The lifecycle of the Relay Smart Contract (RSC) is executed on an epoch-by-epoch basis through a well-defined sequence of on-chain function calls that govern relay election, channel operation, throughput monitoring, dispute resolution, and fund withdrawal. During the relay election phase preceding a target epoch *e*, candidate relays register their candidacy by invoking the *relayerAsCandidate(...)* function, submitting their service parameters (maximum users, coins, throughput, relay delay, and fee), and locking the required guarantee fund as determined by the tariff tables T_{users}, T_{coins}, and $T_{throughput}$. In parallel, network participants support relay candidates by staking as r-witnesses or free service providers using the *voteRelayerAsWitness(...)* and *voteRelayerAsServiceProvider(...)* functions. Once the election window closes T_R blocks before epoch *e*, relay candidates request formal licensing through *requestLicence()*, and the RSC finalizes the relay set by invoking *verifyRelayerSetChange()*, which deterministically selects licensed relays whose aggregate capacity fits within the system transaction capacity $TC_R[e]$.

During the active relay epoch, licensed relays establish client-to-relay (inbound) payment channels through user calls to *depositToRelayer(...)*, while relays establish relay-to-server (outbound) payment channels using *depositToServer(...)*. Off-chain payments are executed between clients, relays, and servers using signed payloads, and relays periodically aggregate and post balance updates on-chain via *updateClientChannel(...)* and *releaseServerFunds(...)*. Throughout this phase, the RSC enforces solvency constraints and throughput limits by monitoring transaction counters over sliding windows of k_R blocks, exposed through *getRelayerGasThroughput(...)*. Clients and relays may reallocate or withdraw unlocked funds using *reallocateServerFunds(...)*, *withdrawFundsFromServer(...)*, and *withdrawFundsFromRelayer(...)*, subject to epoch and lock constraints.

If abnormal behavior occurs, such as delayed payments or over-withdrawals, the RSC enables on-chain dispute resolution. Disregarded servers initiate delayed-payment disputes using *reportDelayedPayment(...)*, while clients initiate over-withdrawal disputes using *reportOverwithdraw(...)*. In response, relays must submit signed proofs of honest operation within D_R blocks by invoking *relayRespondDispute_verifyUserPayloads(...)* and *relayRespondDispute_verifyRelayPayloads(...)*, followed by settlement through *relayRespondDelayedPayment_settleDispute(...)* or *respondOverwithdraw_settleDispute(...)*. Failure to respond within the dispute window allows affected parties to reclaim funds through *serverRefund(...)* or *clientRefund(...)*, while the RSC applies penalties to the relay’s guarantee fund and updates relay capacity and delay counters accordingly.

At the end of the epoch, and after the expiration of the guard interval G_R, r-witnesses and service providers withdraw their stakes using *rWitnessWithdrawRequest(...)* and *rFreeServiceProviderWithdrawRequest(...)*, while witnesses of elected relays may claim rewards through *rWitnessPaymentRequest(...)*. Honest relays may finally withdraw their remaining guarantee funds using *relayerWithdrawRequest(...)*, completing the relay lifecycle for the epoch. Through this sequence of function calls, the RSC enforces a complete operational lifecycle that combines scalable off-chain payments with on-chain accountability, deterministic settlement, and economically enforced honest behavior.

3.7. Transactions Throughput Requirements

A critical performance dimension of blockchain-backed mobile service provisioning is the relationship between the achievable transactions per second (TPS) and the number of active users. In Beyond-5G environments, this relationship becomes decisive, as mobile services such as video streaming naturally induce fine-grained, high-frequency charging events, especially under pay-per-chunk or micro-payment–based service models. In this subsection, we analyze the scalability limits of different architectural choices and quantify what is practically achievable under realistic operational assumptions.

We begin by considering a baseline service model in which users consume mobile video content using adaptive streaming, generating one payment per video chunk. With chunk durations on the order of 1–2 seconds, and additional payment events triggered by user mobility and network handovers, each active user produces multiple payment events per minute. Under such conditions, the aggregate transaction demand grows rapidly and approximately linearly with the number of users.

In the first scenario, all service payments are executed directly on-chain. In this case, the required TPS scales linearly with the number of active users. Even for moderate user populations (on the order of $10^4 - 10^5$ users), the resulting TPS demand already exceeds the capabilities of contemporary smart-contract-enabled blockchains. For example, Bitcoin (not smart-contract-enabled) processes approximately 3–7 TPS on its base layer, Ethereum Layer-1 supports roughly 15–238 TPS, and modern high-throughput blockchains, such as Solana, report average sustained throughputs in the range of 700–1,500 TPS. Scaling this model to 10^6 users would require hundreds of thousands to millions of TPS, rendering direct on-chain micro-payments fundamentally infeasible. This scenario clearly demonstrates that naive blockchain-based charging models cannot support the massive mobile service provisioning.

A second scenario considers improving the baseline by adopting a more efficient layer-1 consensus mechanism, such as Delegated Proof of Stake (DPoS). While DPoS significantly reduces block production overhead and increases achievable TPS compared to Proof-of-Work systems, the fundamental scaling behavior remains unchanged: TPS still grows linearly with the number of users if each service interaction results in an on-chain transaction. Consequently, even with optimistic DPoS configurations, the system cannot sustain large user populations combined with fine-grained charging and high mobility. Even when considering layer-2 technology (e.g., sequencer layers or rollups), throughput may increase into the low thousands, but at the cost of increased hardware requirements and reduced decentralization.

The third scenario, which reflects the full SOVEREIGN design, introduces payment relays that aggregate off-chain micro-payments. In this architecture, users issue frequent off-chain payment promises to licensed payment relays, while relays periodically post aggregated balance updates on-chain. As a result, the blockchain no longer processes individual service payments but only aggregated settlement transactions and control operations. Crucially, this decouples blockchain-level TPS from the number of end users. Instead, TPS becomes a function of the number of active relays, their settlement frequency, and the epoch configuration of the blockchain.

Quantitative evaluation shows that, under the SOVEREIGN model, millions of users can be supported while maintaining on-chain TPS in the order of hundreds to low thousands. The system scales horizontally by adjusting relay capacity, increasing the number of relays, or tuning aggregation windows, without violating blockchain throughput constraints. Notably, while the service-level transaction volume increases with user count, the blockchain-level transaction volume remains bounded, effectively flattening the TPS-versus-users curve.

Table 5 presents indicative system-level specifications derived from the required TPS, considering the large number of users expected to utilize the SOVEREIGN platform. The values are not prescriptive, but rather illustrate feasible operating points enabled by the proposed DPoS consensus and payment relay architecture.

Table 5. Indicative System-Level Specifications for the SOVEREIGN Blockchain Platform

Specification	Target / Example Value	Rationale
Active users supported	$\sim 1M - 10M$	Target scale for 5G/Beyond-5G ecosystems

Service-level payment rate per user	0.5–1 payments/s	Reflects pay-per-chunk video delivery (1–2 s segments)
Aggregate service-level transactions	$\sim 500k - 10M$ tx/s	Illustrates the infeasibility of direct on-chain charging
On-chain TPS (sustained)	$10^2 - 10^3$ TPS	Achievable with DPoS and bounded settlement traffic
On-chain TPS (peak)	$\leq 2\ x\ 10^3$ TPS	Covers bursts due to relay settlement and disputes
Number of active validators	5 - 25	Balances throughput and fault tolerance under DPoS
Number of payment relays	50 - 500	Enables horizontal scaling and geographical distribution
Users per relay	$10^3 - 10^5$	Conservative to optimistic aggregation assumptions
Relay settlement interval	10 – 43k blocks	Trades off latency vs. on-chain load
Blockchain role	Settlement & control only	Blockchain is not used for per-service micro-payments

In Table 5, we explore both conservative and optimistic relay configurations. Conservative configurations assume frequent on-chain updates and a limited number of users per relay to maximize security margins, while optimistic configurations allow larger aggregation windows and higher relay utilization, assuming a certain level of honesty. Even under conservative assumptions, the proposed architecture supports orders of magnitude more users than direct on-chain models. Under optimistic yet realistic assumptions, the system can accommodate multi-million user populations with a manageable blockchain load.

Overall, this analysis demonstrates that scalable blockchain-backed mobile service provisioning is only achievable when the blockchain is treated as a settlement and control plane, rather than a per-service transaction processor. The combination of DPoS consensus and off-chain payment aggregation will enable the SOVEREIGN platform to meet the extreme throughput requirements of Beyond-5G environments while preserving decentralization, security, and economic accountability.

4. Decentralized AAA: SOVEREIGN SSI Architecture components

The SOVEREIGN SSI architecture should comprise three primary entities which are the *Issuer, the Holder, and the Verifier*. The Issuer and the Verifier interact with the SOVEREIGN Blockchain / Trust Registry and communicate through secure channels, facilitated by a framework such as Hyperledger Aries.

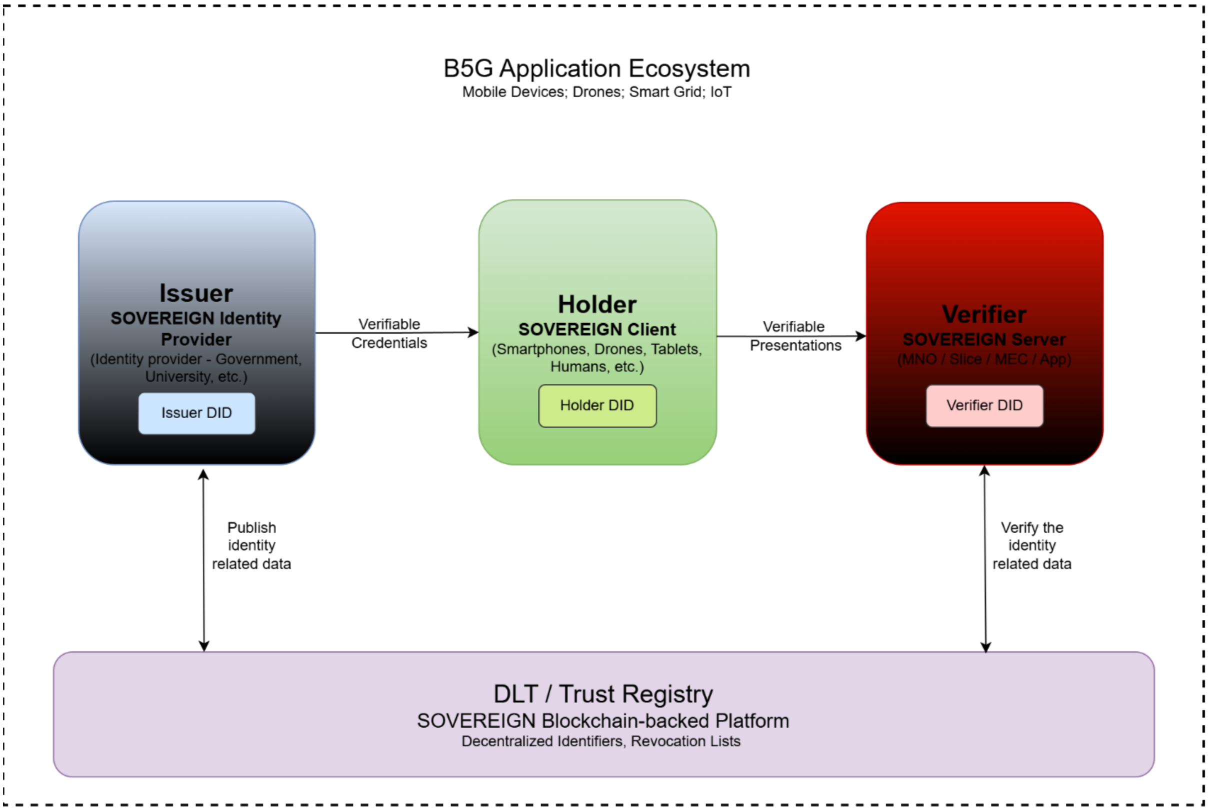

B5G Application Ecosystem
Mobile Devices; Drones; Smart Grid; IoT
Issuer
SOVEREIGN Identity Provider
(Identity provider - Government, University, etc.)
Issuer DID
Verifiable Credentials
Holder
SOVEREIGN Client
(Smartphones, Drones, Tablets, Humans, etc.)
Holder DID
Verifiable Presentations
Verifier
SOVEREIGN Server
(MNO / Slice / MEC / App)
Verifier DID
Publish identity related data
Verify the identity related data
DLT / Trust Registry
SOVEREIGN Blockchain-backed Platform
Decentralized Identifiers, Revocation Lists

Figure 9: SOVEREIGN SSI architecture (high-level)

Issuer: is an identity provider, such as a government, university, or other organisation, that is responsible for creating and issuing Verifiable Credentials (VCs). In the context of SOVEREIGN, an Issuer could be an entity providing credentials about a device, user, or service within the B5G ecosystem.

- The Issuer contains an Issuer DID (Decentralised Identifier). This DID is a unique, blockchain-based identifier that is created and controlled by the Issuer itself, rather than being assigned or imposed upon them by an external central authority. This means that even when a government or university acts as an Issuer of Verifiable Credentials (VCs), their own Issuer DID is a self-generated and managed identifier, aligning with the core SSI principle of identity being controlled by the individual or entity it represents. The role of such an Issuer is to cryptographically sign and distribute VCs about other entities (the Holders), not to issue the corresponding DIDs.

- Key Functions:

- The Issuer verifies real-world attributes of a subject (Holder) and then cryptographically signs claims about that subject to create a Verifiable Credential (VC).
- DID Publication: The Issuer publishes its own DID document (containing its public keys and service endpoints) to the DLT/Trust Registry. This allows others to verify the Issuer's identity and locate its VCs.

- Communication:
  - Sends VC to Holder: The Issuer issues and sends the Verifiable Credential directly to the Holder.
  - Publishes Data to DLT/Trust Registry: The Issuer publishes its DID and potentially credential schemas or revocation information to the DLT.

Holder: is the individual or entity (e.g., smartphone, drone, tablet, human) who receives, stores, and presents credentials. In SOVEREIGN, intelligent endpoints in B5G networks are empowered to be Holders, gaining full control of their identities and data.

- Holder DID: The Holder generates and controls its own unique, blockchain-based Decentralised Identifier. This DID is used for secure communication and authentication.
- SSI Wallet: Operates as a digital safe within the Holder's application (SSI Controller) that securely stores and organises digital credentials and identity information. It uses cryptography to ensure only the owner can access and manage these details and enables selective disclosure.
- SSI Controller: Manages and controls the Holder's DID and holds VCs issued by trusted issuers. It represents the application logic that directs the SSI Agent.
- SSI Agent: A software component deployed on the SSI Controller that is integral to managing VCs, including issuing, storing, verifying, and presenting them securely. It acts as an intermediary between the Holder's device and Verifiers, handles messaging (via DIDComm), credential exchange, and proof requests, and adheres to W3C standards for DIDs and VCs.
- Key Functions:
  - DID Management: Controls its unique, blockchain-based DID.
  - VC Storage: Securely stores Verifiable Credentials received from Issuers in its SSI Wallet.
  - Verifiable Presentation (VP) Generation: When requested by a Verifier, the Holder selects relevant VCs and creates a VP, which is a bundle of one or more VCs (or a derived proof) that the Holder chooses to present. This presentation is digitally signed by the Holder to prove control of the DID, and can use selective disclosure to reveal only strictly required data.

- Communication:
    - The Holder receives the Verifiable Credential from the Issuer.
    - The Holder requests a service from a service provider. In some cases, the Service Provider can be an SSI Verifier.
    - The service provider (Verifier) requests the Verifiable Presentation from the Holder.
    - The Holder sends the Verifiable Presentation (proof) to the Verifier.

Verifier: is the party (in some cases Mobile Network Operator, Slice orchestrator, Multi-access Edge Computing service, or Application) that requests proof from a Holder and verifies the authenticity and validity of the credential. After successful verification, the verifier provides a service or requests service providers to provide to the holder the requested service(s).

- The Verifier contains a Verifier DID.
- Key Functions:
    - Proof Requesting: Requests a verifiable presentation from the Holder, specifying the required credentials or attributes.
    - Verification Engine: It checks the Issuer's digital signature, the Holder's proof of control over their DID, and the credential's validity and revocation status (by querying the DLT).
    - Decision-Making: Based on the verification outcome (verified/not-verified), the Verifier decides whether to grant access to the subject (e.g., user, device, service).
- Communication:
    - The Verifier receives the Verifiable Presentation from the Holder.
    - The Verifier interacts with the DLT/Trust Registry to retrieve the Issuer's public key (via its DID document) and check the credential's revocation status.
    - Informs B5G Application Ecosystem: The Verifier's decision (verified/not-verified) is then used by the various B5G applications (V2X, XR/Metaverse, Drones, Smart Grid, IoT) to grant access to network slices, Ultra-Reliable Low Latency Communications (URLLC) services, or other resources.

B5G Application Ecosystem: This represents the various B5G verticals and services that will utilise the verified identities provided by the SSI architecture.

- Examples: Mobile Devices, Drones, Smart Grid, IoT. SOVEREIGN envisions this to include V2X, XR/Metaverse, Drones, Smart Grid, and IoT applications.

- Role: Within the B5G Application Ecosystem, Verifiers (who may be service providers) leverage the "verified / not-verified" result obtained from the verification process to grant or deny the Holder access to B5G services or resources. This enables functionalities such as access to network slices, Ultra-Reliable Low-Latency Communication (URLLC) services, or other B5G resources and services. The ecosystem, encompassing diverse verticals such as Mobile Devices, Drones, Smart Grid, IoT, V2X, and XR/Metaverse applications, represents the various contexts and services to which the Verifier's access decision is applied based on the validated identity.

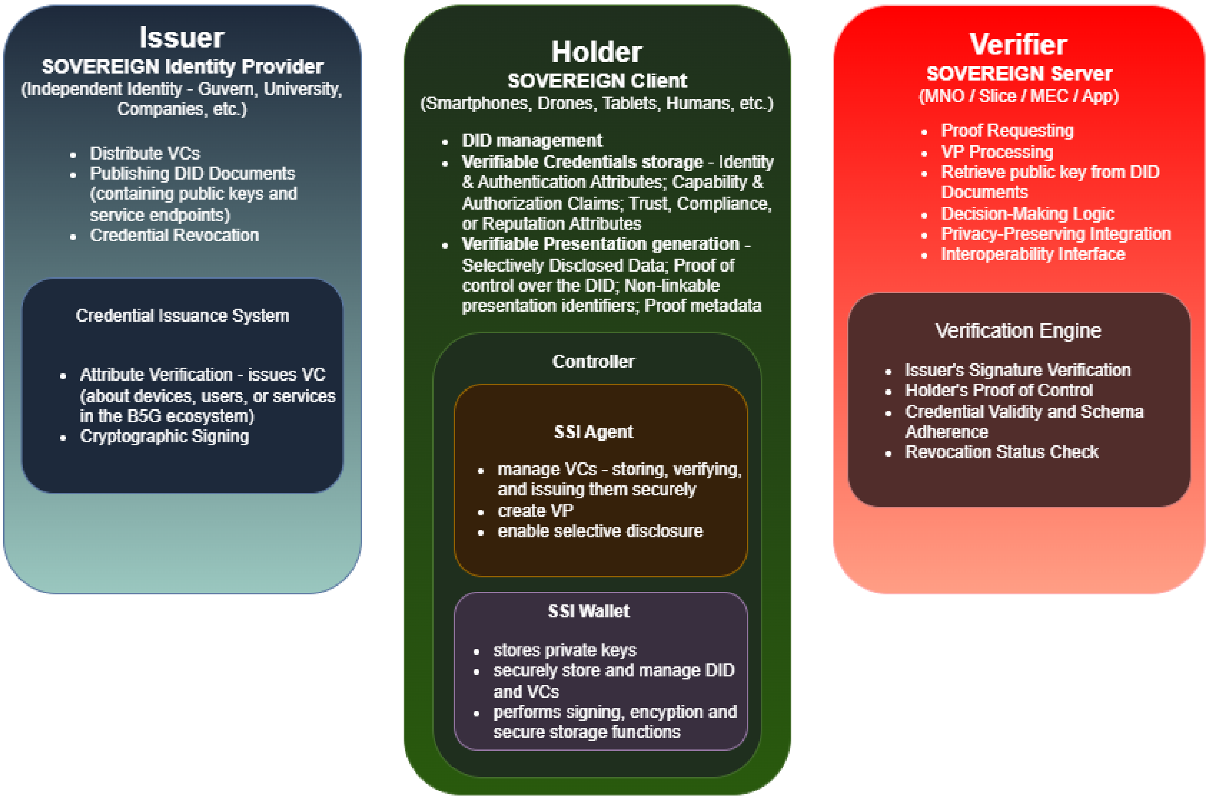

Issuer
SOVEREIGN Identity Provider
(Independent Identity - Guvern, University, Companies, etc.)
Distribute VCs
Publishing DID Documents (containing public keys and service endpoints)
Credential Revocation
Credential Issuance System
Attribute Verification - issues VC (about devices, users, or services in the B5G ecosystem)
Cryptographic Signing
Holder
SOVEREIGN Client
(Smartphones, Drones, Tablets, Humans, etc.)
DID management
Verifiable Credentials storage - Identity & Authentication Attributes; Capability & Authorization Claims; Trust, Compliance, or Reputation Attributes
Verifiable Presentation generation - Selectively Disclosed Data; Proof of control over the DID; Non-linkable presentation identifiers; Proof metadata
Controller
SSI Agent
manage VCs - storing, verifying, and issuing them securely
create VP
enable selective disclosure
SSI Wallet
stores private keys
securely store and manage DID and VCs
performs signing, encyption and secure storage functions
Verifier
SOVEREIGN Server
(MNO / Slice / MEC / App)
Proof Requesting
VP Processing
Retrieve public key from DID Documents
Decision-Making Logic
Privacy-Preserving Integration
Interoperability Interface
Verification Engine
Issuer's Signature Verification
Holder's Proof of Control
Credential Validity and Schema Adherence
Revocation Status Check

Figure 10: SOVEREIGN SSI Entities’ components and their roles

5. Online asset pricing for DLT-backed B5G networks

5.1. Introduction

The rapid and constant expansion of mobile broadband traffic keeps exerting strain on current cellular network infrastructures. As reported in Cisco’s Annual Internet Report, worldwide mobile data traffic is projected to increase at a compound annual growth rate surpassing 20% fuelled by bandwidth-intensive applications, like video streaming, cloud computing, and interactive real-time services [53]. This expansion increases capacity requirements not just in crowded urban centers, but also in rural and sparsely populated areas, where installing conventional macro-cell facilities continues to be financially difficult. Consequently, mobile network operators (MNOs) need to manage capital expenditure (CAPEX) for network growth alongside rising operational expenditure (OPEX) while users concurrently demand widespread, superior, and affordable connectivity [53], [54].

This conflict reveals a core inefficiency within the mobile access framework. Networks are generally designed to accommodate peak usage, leaving vast regions either underserved or completely without service since the expense of deploying and sustaining base stations is not warranted by the sparse local demand [54]. Consequently, coverage voids continue to exist even though the network is overprovisioned on a level. Conventional rate or broad usage-based pricing schemes worsen this issue since they do not account for spatial, temporal, and service-specific differences in resource availability and delivery expenses [55], [56].

Recent developments in radio access network (RAN) designs, network virtualization, and customizable scheduling in 5G frameworks offer a chance to reconsider this model. The 3GPP 5G New Radio (NR) framework presents Quality of Service (QoS) differentiation, flow-specific scheduling, and service-conscious resource distribution, allowing diverse service provision within a unified radio access setup [57], [58]. These technological features enable decentralized network operators, like local enterprises, community networks, or individual users, to install lightweight small cells in areas where large MNOs cannot cost-effectively grow or do not wish to do so. These micro-providers can offer coverage and capacity at much reduced expenses, effectively supplementing conventional macro-cell deployments.

Nonetheless, with these technological improvements, the financial systems necessary to back decentralized mobile access are still insufficiently developed. Current mobile network pricing techniques mainly depend on fixed subscription models or broad congestion-based modifications that function over periods and disregard real-time radio and backhaul states [55], [59]. Earlier studies on pricing and congestion-sensitive billing have shown that pricing can serve as a powerful tool to regulate demand, enhance utilization, and boost revenue consistency [60], [61], [62]. However, the majority of these models overlook the radio access network and exclude real-time parameters, like individual User Equipment (UE) signal strength, physical resource block (PRB) usage, or scheduler activity. As a result, they are unable to leverage the potential of contemporary 5G systems.

In this study, we introduce a dynamic pricing structure for next-generation mobile networks that specifically tackles this issue. Rather than depending on fixed subscription fees, the suggested approach allows end users to flexibly choose from various accessible cell towers and providers according to both performance and cost, while the cell towers broadcast prices that represent their existing resource usage, radio environment, backhaul limitations, and past engagements with users. By integrating real-time indicators, such as reference signal received power (RSRP), spectral efficiency, PRB availability, cell tower load, individual UE service records, and mobility trends, the pricing system establishes a market-style interaction between users and access points, tightly linking motivations with actual network states.

From the user’s viewpoint, this method will become more economically viable than the fixed contracts currently in use, as they would only be paying for data that they use. Furthermore, it brings about price rivalry at the access stage, enabling users to choose the provider offering the price that meets their QoS needs. Individuals in covered locations can enjoy reduced prices owing to competition and surplus capacity, whereas those in distant or poorly served areas obtain affordable connectivity via micro-providers whose operating expenses are significantly less than those of conventional MNO infrastructures. This kind of varied pricing matches user charges to the usage of resources and to the level of quality the service provided, a concept extensively supported in network economics research [60], [61].

From the viewpoint of the network operator, dynamic pricing allows for usage of diverse infrastructure by shifting specialized or local demand to third-party or community-run cell towers. Furthermore, it saves them the expense of installing expensive cell towers in remote low-density regions, as they (the operators) can depend on decentralized providers while keeping financial control through pricing strategies. Pricing that considers utilization and backhaul capacity also helps avoid congestion and quality decline by deterring demand when resources are limited [62], [63].

Collectively, this framework supports the emergence of a hybrid decentralized mobile ecosystem in which large MNOs and small-scale providers coexist and compete. Economic incentives embedded in the pricing model naturally balance supply and demand, reduce unnecessary infrastructure investment, and enable personalized connectivity tailored to each user’s radio environment, mobility, and service requirements. This aligns with broader visions of flexible, service-aware, and economically efficient 5G and beyond-5G networks [57], [58].

5.2. Previous research

The evolution of mobile networks from circuit-switched voice systems to today’s data-intensive 5G architectures has significantly influenced the development and evolution of network pricing models. Effective pricing must strike a balance between the demands and needs of its users, with the fluctuating and limited capacity available to the network. Research on pricing for communication networks spans several decades and ranges from classical congestion pricing to modern market-based mechanisms for multi-operator 5G slicing environments. This section will review the main lines of work most relevant to UE-centric, cross-layer dynamic pricing in 5G NR systems, ultimately highlighting the gap addressed by our packet-level, scheduler-integrated, dynamic pricing schemes.

5.2.1. Congestion-Based and Resource-Aware Pricing

A foundational line of research views pricing as a control mechanism that aligns user demand with network resources. Early formulations generated “congestion prices” from network optimization, demonstrating that locally computed prices can drive distributed users towards efficient equilibrium. Heikkinen [64] proposed a linear congestion pricing with distributed resource control for wireless networks that integrates pricing with distributed resource control and learning-based adaptation. Such work sets the foundation that prices can indicate real-time scarcity and help maintain network performance [64].

In broadband environments, congestion pricing has been widely explored as a method to prevent excessive resource use and to implement varied service categories. Abu Ali proposed a congestion-driven pricing and resource allocation model for broadband wireless systems using pricing elements to influence demand and alleviate congestion [62]. Related research also integrates Call Admission Control (CAC) alongside the dynamic pricing to prevent network overload and improve utilization under explicit Quality of Service (QoS) constraints [65]. While these methods offer intuition that prices ought to rise during network congestion, their frameworks generally operate at an abstract level and typically omit detailed RAN scheduling behaviour, Physical Resource Block (PRB) allocation, or individual UE radio metrics as produced by NR Medium Access Control (MAC) schedulers [62].

A complementary area of interest focuses on “Smart Data Pricing” (SDP), where economic incentives are explicitly used to shift demand and mitigate congestion. Sen et al. [66] surveyed mechanisms such as time-dependent pricing and offloading incentives, highlighting how economic tools can manage network load while balancing operator revenue and user welfare. These models are conceptually aligned with dynamic, utilization-aware charging, but they are usually evaluated at aggregate traffic levels (e.g., time-of-day demand) rather than per-slot, per-UE RAN resource allocation [66].

5.2.2. Dynamic Pricing of Mobile Data Plans and Demand Management

Another area of research that has been gaining traction focuses on mobile pricing at the level of subscription plans and operator tariff design. Ma et al. [67] developed an optimization-based model for dynamically controlling the availability and pricing of mobile data plans under congestion. They converted the resulting decision-making process into manageable optimization forms, enabling the evaluation of revenue and congestion trade-offs. Subsequent studies on dynamic plan control [55] similarly treats pricing as an instrument to manage demand, emphasizing operator-side control over plan offerings to mitigate congestion. These contributions matter as they link pricing choices to congestion effects and revenue [55], [67], but they typically remain above the RAN layer: congestion is modelled via aggregate capacity constraints and does not directly integrate NR scheduler decisions, PRB utilization, or radio quality variability across UEs.

Further studies examine dynamic pricing to maximize revenue under uncertainty and evolving demand, including sequential pricing formulations and long-term revenue objectives [68]. Such formulations reinforce the value of dynamic pricing and optimization, yet they commonly assume a monopoly operator or exclude explicit multi-operator access selection with radio-aware user association.

5.2.3. Competitive and Multi-Operator Pricing with Access Selection

When multiple providers compete or collaborate, pricing interacts tightly with user association (i.e., which access point the user selects). Zhang et al. [61] proposed a hybrid pricing framework for mobile collaborative Internet access, modelling incentives and market equilibria when users can access connectivity through multiple mechanisms and providers. This body of work demonstrates that pricing can be used to create “market-like” outcomes in mobile access ecosystems, particularly when users have alternatives and can respond to price differences.

Specifically, in scenarios with multiple operator wireless settings, a Stackelberg game framework has been applied to combine pricing and access selection, where operators set prices and a user (or home operator) selects among available networks to maximize profit or QoS-based satisfaction [69]. These models are structurally close to our UE-centric approach: the user decision is responsive to both price and quality [69]. Nonetheless, numerous studies depend on stylized QoS models rather than explicit NR RAN measurements (e.g., Reference Signal Received Power (RSRP) / Signal-to-Interference-plus-Noise Ratio (SINR), spectral efficiency, PRB scheduling), which limits their ability to capture short-timescale dynamics and scheduler-driven performance variations.

5.2.4. 5G Specific pricing: Slicing, Multi-Tenancy, and Cost Models

Within the 5G network, pricing is increasingly linked with slicing, virtualization costs, and service differentiation. Malolli [56] discusses pricing strategies and emerging business models for 5G data communication, emphasizing the complexity introduced by heterogeneous services and deployment models. Flamini and Naldi [63] analyzed optical pricing within a rented 5G infrastructure scenario with what they call “sticky” customers, capturing economic frictions that arise when users do not instantly switch providers in response to prices. These perspectives are highly relevant to decentralized and hybrid ecosystems, where micro-providers and MNOs may coexist, and user switching costs may influence market dynamics [63].

For network slicing and multi-tenancy, pricing models frequently adopt game-theoretic or market-based mechanisms to allocate resources among tenants and to determine slice prices. Anantha Kumar et al. [70] evaluated pricing strategies for 5G multi-tenancy (including rural non-public network scenarios) and compared cooperative games and bargaining-based mechanisms, highlighting how pricing choices affect stakeholder incentives and investment. Cost modelling of slice provisioning is also critical. Walia et al. [71] proposed a virtualization infrastructure cost model for 5G slice provisioning, linking slice demands to Total Cost of Ownership (TCO) and per-slice cost breakdowns. These studies encourage incorporating technology and cost factors (e.g., backhaul and infrastructure costs) into pricing decisions, but they generally focus on slice-level economics rather than UE-level, per-slot pricing tied to PRB scheduling and instantaneous radio conditions.

5.2.5. Synthesis and Gaps Addressed by Current Work

Throughout the literature, there are multiple recurring themes that become apparent:

- Pricing as control: Congestion-based and smart data pricing shows that prices can effectively shape demand and improve network efficiency [62], [64], [66].
- Pricing under congestion: plan-level dynamic pricing explicitly studies the revenue-congestion trade-offs [55], [67].
- Competition and user choice: multi-operator models emphasize that pricing interacts with access selection and market equilibrium [61], [69].
- Economic complexity of 5G: The introduction of slicing, virtualization, and infrastructure sharing brings cost and incentive frameworks that pricing strategies need to accommodate [63], [70], [71].

However, a practical gap persists between high-level economic formulations and the operational reality of 5G NR RANs: most prior models do not jointly incorporate (1) per-UE radio quality (e.g., RSRP/SINR, spectral efficiency), (2) fine-grained MAC scheduling outcomes, and (3) PRB-level utilization as the direct scarcity signal that drives service performance. This motivates the UE-centric, cross-layer dynamic pricing approach evaluated in the current research, which connects pricing to measurable RAN and backhaul indicators at short time scales and under explicit multi-provider competition.

5.3. System Model

This section describes the model used to evaluate the proposed UE-centric dynamic pricing framework. The model is designed for a decentralized 5G Radio Access Network (RAN) where multiple independent access providers compete for user traffic based on real-time network conditions.

5.3.1. Entities and roles

The system contains three primary entities:

- **User Equipment (UE)**: Mobile agents that generate service requests (e.g., voice, video, or URLLC). Each UE monitors the radio environment and triggers the pricing mechanism by broadcasting service requirements to reachable access points.
- **Access Providers (gNBs)**: Independent base stations or Virtual Mobile Network Operators (VMNOs) that manage local radio and backhaul resources. They operate autonomously without a centralized controller or inter-operator coordination.
- **Pricing and Selection Logic**: A functional interface where gNBs independently compute price offers based on local telemetry, and UEs evaluate these offers to select the provider they wish to use.

5.3.2. Network topology and competition model.

We model a heterogeneous deployment where multiple access providers have overlapping coverage areas. This creates a local market where a single UE can reach *N* candidate gNBs simultaneously. Competition is modelled as a non-cooperative game: each provider aims to optimize its own objectives (e.g., revenue or load balancing) by advertising a price *P* for a specific service request without knowledge of its competitor's internal states or pricing strategies.

5.3.3. Information Flow and Measurement Sources

The pricing framework is driven by real-time data extracted from the simulation's RAN and transport layers. To ensure practical feasibility, the model only uses information locally observable by the provider or historical data from past interactions. The data inputs, their origin within the system, and their purpose in the pricing logic are outlined in Table 6.

5.3.4. Interaction Sequence

The operational flow follows a two-stage transactional process:

1. **Service Request Phase**: A UE broadcasts a request containing its service type (latency, throughput requirements) and current radio context (RSRP/SINR) to all reachable providers.

2. **Pricing Phase**: Each provider processes the UE’s radio metrics against its own instantaneous load (PRB utilization) and backhaul state. The provider then advertises a price per unit of data (e.g., per Mbit).

The final selection decision is made by the UE based on its internal utility function (balancing cost vs. quality), which serves as the trigger for resource allocation in the simulation. Note that the UE decision is beyond the scope of the current research, as it focuses on the pricing dynamics of the network.

5.3.5. Scope and Assumptions

In order to focus on the pricing mechanism’s performance, the following constraints are applied:

- **Decentralization**: No global controller manages the network; all decisions are local to the gNB.

- **Mobility**: UEs follow arbitrary mobility patterns, which induces temporal variations in radio quality and local congestion, testing the model’s responsiveness.

- **Monetary interpretation**: Prices are expressed in explicit monetary units to allow for direct comparison between different types of infrastructure (e.g., expensive rural micro-cells vs. high-capacity urban macro-cells).

Furthermore, the decision of each UE as to which gNB they select is beyond the scope of the current deliverable. Instead, it focuses solely on the pricing mechanism at work and how to adapt that to the needs of each access provider.

Table 6 Pricing input categories.

Pricing input	**Information owner**	**Measurement source**	**Time scale**	**Role in pricing**
Radio utilization	Access provider	Scheduling and resource allocation state	Short-term	Congestion-aware pricing

Radio quality (spectral efficiency)	UE / Access provider	Link adaptation (CQI/MCS outcomes)	Short-term	Radio cost per delivered bit
Backhaul characteristics	Access provider	Infrastructure configuration	Long-term	Infrastructure-aware base price
QoE / reputation history	Access provider	Past service outcomes	Long-term	Pricing stability and trust
Revenue history	Access provider	Accounting records	Long-term	Fairness and sustainability

5.4. Proposed Pricing Methodology

This section presents the proposed pricing methodology for the UE-centric dynamic access framework. The objective of the pricing mechanism is to translate instantaneous network conditions, competition, and historical performance into prices that balance efficiency, user experience, and operator sustainability. Unlike traditional plan-based or coarse-grained pricing schemes, the proposed approach operates at fine time granularity and explicitly incorporates radio-level measurements and competitive dynamics to provide a pay-as-you-use price.

The pricing model is designed as a modular multiplicative factor model. Each factor captures a distinct dimension of cost, scarcity, or market behaviour (e.g., congestion, radio efficiency, crowding, loyalty), and can be tuned independently without changing the overall structure. This improves interpretability and supports incremental extensions as the system model evolves.

5.4.1. Pricing components and factor definitions

We consider a UE, u, requesting service class s from provider o at time t. The provider advertises a price $P_{o,u,s}(t)$ constructed from measurable signals. The user will then have an overview of the available prices they have to select from and make a decision based on their needs. A summary of all the pricing components and their relevant information can be found in Table 7.

Base price

A provider will define a **base price** P_0 (e.g., deliver Mbit, per unit time, or per session). Instead of assuming a uniform base price across all access points, the proposed framework incorporates infrastructure costs directly into the base price. This reflects the fact that different access points may rely on different backhaul technologies with substantially different costs to the operator. If we let $P_0(o)$ denote the base price of a specific operator, this is defined as:

$$P_0(o) = P_{ref} \cdot c_{bh}(tech(o))$$

where:

- P_{ref} is a global reference per pricing unit (e.g., per Mbit),
- $tech(o)$ denotes the backhaul technology used by operator o,
- $c_{bh}(\cdot)$ is a cost multiplier tied to the backhaul technology.

Typical values of c_{bh} reflect relative infrastructure cost:

- Fiber backhaul: $c_{bh} \approx 1.0$
- Fixed wireless backhaul: $c_{bh} > 1.0$
- Satellite backhaul: $c_{bh} \gg 1.0$

The selected prices are grounded in real-world costs for the provider. A fiber backhaul is the ideal form of backhaul and is therefore priced between the other two. Fixed wireless backhaul will be slightly more expensive as it has higher maintenance and power costs. Finally, satellite backhauls will be the highest priced, based on the cost to operate the satellite that provides the backhaul data. By incorporating the backhaul directly into the starting price, all the subsequent pricing components will operate relative to the true baseline, better reflecting real-world deployment.

Service differentiation

Service differentiation is captured using a **service weight** w_s, reflecting service priority and expected resource intensity. The service weights are defined relative to a reference service weight w_{ref}, corresponding to a standard best-effort or enhanced mobile broadband (eMBB) service. The relative ordering of service weights is defined as:

$w_{voice} < w_{ref} \leq w_{video} < w_{URLLC},$

where w_{voice} represents conversational voice traffic, w_{video} represents video streaming services, and w_{URLLC} represents ultra-reliable low-latency services.

The selected values are based on real-world data traffic patterns and service characteristics. Voice services are given a slight discount relative to other services, reflecting their role as a baseline utility with low bandwidth requirements but high societal importance. Industry reports consistently show that voice traffic accounts for only a small fraction of total mobile data volume, yet remains critical for basic and emergency communication services [54].

Video streaming represents the dominant share of mobile data consumption, accounting for more than 70% of global mobile traffic, and continues to rise [54]. A weight of $w_s = 1.0$ represents standard video delivery, while an increased value (e.g., $w_s = 1.2$) represents more premium video (e.g., 4K/8K) or during heavily congested periods.

Low-latency services are assigned the highest weights, reflecting the stringent performance requirements and higher resource cost. Ultra-reliable Low-latency communications (URLLC) target specific and mission-critical applications such as industrial automation, remote control, and interactive real-time systems, which require prioritized scheduling, tight latency bounds, and high reliability [72]. These requirements often lead to increased resource reservation and reduced scheduling flexibility, justifying a higher price for the delivered service.

By differentiating prices across services in this manner, the remaining pricing components can focus on capturing real-time network conditions and competition effects without conflating them with long-term service characteristics.

Radio resource utilization

Radio resource utilization is a fundamental driver of pricing in wireless networks. In the proposed framework, utilization reflects the fraction of PRBs consumed on a given gNB and bandwidth part. As utilization increases, the marginal cost of serving additional traffic rises due to increased contention, reduced scheduling flexibility, and higher risk of QoS degradation.

Let $RB_{tot}(o,t)$ denote the total number of PRBs available at provider o for the relevant bandwidth part, and $RB_{used}(o,t)$ the number of RBs used in the same interval, the utilization is:

$$u(o,t) = \frac{RB_{used}(o,t)}{RB_{tot}(o,t)}, \qquad U(o,t) = 1 = u(o,t)$$

where $U(o,t)$ is the unused fraction. We then turn this into an RB scarcity factor that increases with utilization:

$f_{RB}(o,t) \ = \ 1 \ + \ \gamma u(o,t) \ \ ,$

Where $\gamma \ > \ 0$ controls the congestion markup. To impose a stronger penalty near saturation, a convex variant can be used:

$f_{RB}(o,t) \ = \ 1 \ + \ \gamma u(o,t)^p \ , p \ > \ 1$

This means when many RBs are used, $u \approx 0$ and price pressure from scarcity is low; when resources are saturated, $u \to 1$, the scarcity factor increases.

Radio quality (radio cost factor)

Radio conditions directly determine the amount of resources required to deliver a given volume of data. Even at the same offered traffic, a UE with poor radio conditions will consume more resources per delivered bit of the provider. This relationship is captured through the **Modulation and Coding Scheme (MCS)** selected by the scheduler, which is in turn driven by CQI reports and service requirements (QCI/5QI). In the proposed framework, radio quality is expressed via the **effective spectral efficiency** of each UE:

$\eta_u(o,t) \qquad\qquad\qquad [bits/s/Hz]$

This value is obtained from the MCS selected by the MAC scheduler and reflects: modulation order, coding rate, and service-specific constraints (vis QCI/5QI). To then translate this radio efficiency to pricing, we define a radio cost factor:

$f_{radio}(o,u,t) = (\frac{\eta_{ref}}{\eta_u(o,t)})^\beta,$

Where:

- η_{ref} is the reference spectral efficiency (e.g., a mid-range MCS)
- $\beta \ \in \ [0.5, 1.0]$ controls the sensitivity to poor radio conditions

This gives our output the following properties:

- UEs with **high** spectral efficiency ($\eta_u > \eta_{ref}$) incur a **lower** price per bit, as they consume fewer PRBs,
- UEs with **low** spectral efficiency ($\eta_u < \eta_{ref}$) incur a **higher** price, reflecting their higher radio resource cost.

By doing this, the pricing responds directly to scheduler decisions rather than abstract signal strength metrics. This also aligns the pricing decisions of the model with the operational reality of NR MAC scheduling and avoids arbitrary signal-to-price mappings.

Competition and competitive pressure

On the user side, the presence of competing access points fundamentally affects price feasibility. When multiple gNBs are available to a UE, demand becomes more elastic, and prices must reflect competitive pressure. On the other hand, in scenarios with limited alternatives, such as rural or isolated deployments, higher prices may be necessary to reflect operating costs and scarcity. Competition can be incorporated as an additional factor:

$$f_{comp}(u,t) = \frac{1}{1 \; + \; \delta \cdot (N_{comp}(u,t) - 1)}$$

where $N_{comp}(u,t)$ is the number of competing providers visible to UE u, and $\delta > 0$ controls sensitivity. Larger N_{comp} reduces feasible price through competitive pressure.

History: loyalty and value factors

To represent repeated interactions and long-term economics between UEs and gNBs we incorporate a history-based modifier. If we let $K_u(o)$ be the number of prior sessions/flows that a UE u has received from provider o, with scale K_{ref} a loyalty discount becomes:

$$f_{loyalty}(o,u) = \frac{1}{1 \; + \; a_k \frac{K_u(o)}{K_{ref}}}$$

where $a_k > 0$ controls the strength of the discount that the user will receive. If we then let $R_u(o)$ denote the cumulative revenue earned from UE u by provider o, and R_{ref} represent the typical or median revenue per UE, we can then use the reference factor to normalize the cumulative revenue and then apply either a discount for the user with a high previous revenue at this provider:

$$f_{value}(o,u) = \frac{1}{1 \; + \; a_R \frac{R_u(o)}{R_{ref}}}$$

Or it can be used to apply a surcharge on repeating customers, if the provider, for example, is trying to attract new users:

$$f_{value}(o,u) = 1 + a_R \frac{R_u(o)}{R_{ref}}$$

5.4.2. Final utility and price function

Raw price (per UE, per service, per time interval)

The provider computes a raw price by multiplying the component factors mentioned in the previous section:

$$P_{o,u,s}^{raw}(t) \;=\; P_0(o) \cdot w_s \cdot f_{RB}(o,t) \cdot f_{radio}(o,u,t) \cdot f_{loyalty}(o,u) \cdot f_{value}(o,u)$$

The form of the function has been left intentionally modular. Each parameter $(\gamma, \beta, a_k, a_R, etc.)$ can be seen as a tunable knob which controls the sensitivity to scarcity, radio efficiency, history effects, and so on. In the numerical evaluation, these parameters are tuned to explore stability and responsiveness tradeoffs (e.g., avoiding overly spiky prices while still responding to congestion).

Coupling to other users (“price vs. price of other users”).

To incorporate coupling between individual prices and the broader cell price level, we compute the average raw price among all active UEs in provider *o*’s cell at time *t*:

$$\underline{P}_o^{raw}(t) = \frac{1}{|u_o(t)|} \sum_{j \in u_o(t)} p_{o,j}^{raw}(t)$$

and define the final price as:

$$P_{o,u,s}(t) = (1-\lambda)\, P_{o,u,s}^{raw}(t) + \lambda\, \underline{P}_o^{raw}(t)$$

where $\lambda \in [0,1]$ controls the coupling strength:

- $\lambda = 0$: fully individualized pricing
- $\lambda > 0$: a partial anchoring to the cell average (a type of “follow the crowd” effect for the UEs).

This implementation was made to allow the user’s price to respond to the overall market conditions without removing user-specific cost reflectiveness.

Table 7 Pricing components

Pricing input	**Description**	**Observed / Defined at**	**Range / Form**	**Role in Pricing**
P_{ref}	Global reference price per pricing unit	Operator policy	> 0	Sets absolute price scale

	(e.g., per Mbit)			
$tech(o)$	Backhaul technology of operator (o) (fiber, wireless, satellite)	Configuration	Categorical	Infrastructure cost class
$c_{bh}(tech)$	Backhaul cost multiplier	Derived	≥ 1	Captures marginal transport cost
$\boldsymbol{P_0(o)}$	Operator-specific base price $P_{ref} \cdot c_{bh}$	Derived	> 0	Infrastructure-aware baseline price
w_s	Service-type weight for service (s) (voice, video, low-latency)	Operator policy	Typically ([0.8,1.3])	Differentiates services
$RB_{tot}(o,t)$	Total PRBs available at provider (o)	MAC layer	Integer	Radio capacity
$RB_{used}(o,t)$	PRBs used in current interval	MAC layer	Integer	Instantaneous load
$u(o,t)$	Radio utilization fraction RB_{used}/RB_{tot}	Derived	([0,1])	Congestion indicator
$f_{RB}(o,t)$	RB scarcity factor $1 + \gamma u or$ $1 + \gamma u^p$	Derived	≥ 1	Raises price under congestion

$CQI(u,t)$	Channel Quality Indicator reported by UE (u)	PHY layer	Discrete	Scheduler input
$MCS(u,t)$	Modulation and Coding Scheme selected for UE (u)	MAC scheduler	Discrete	Determines spectral efficiency
$\eta_u(o,t)$	Effective spectral efficiency from MCS (bits/s/Hz)	Derived	(>0)	True radio cost per bit
η_{ref}	Reference spectral efficiency	Fixed constant	e.g., 3 bps/Hz	Normalization
$f_{radio}(o,u,t)$	Radio cost factor $(\eta_{ref}/\eta_u)^\beta$	Derived	(>0)	Penalizes inefficient links
$N_{comp}(u,t)$	Competing providers visible to UE (u)	UE context	Integer ≥ 1	Competitive pressure
$f_{comp}(u,t)$	Competition factor (optional)	Derived	$(0,1]$	Lowers prices under competition
$K_u(o)$	Past sessions UE (u) had with operator (o)	History state	Integer	Loyalty signal
K_{ref}	Loyalty	Fixed	Integer	Scaling

	normalization constant			
$f_{loyalty}(o, u)$	Loyalty discount factor	Derived	$(0,1]$	Rewards repeat users
$R_u(o)$	Cumulative revenue from UE (u)	Accounting	≥ 0	Value signal
R_{ref}	Revenue normalization constant	Fixed	> 0	Scaling
$f_{value}(o, u)$	Value factor (discount or surcharge)	Derived	> 0	Monetization vsretention
λ	Price coupling coefficient	Operator policy	$[0, 1]$	Herd / market anchoring
$\underline{p}_o^{raw}(t)$	Average raw price in cell of operator (o)	Derived	> 0	Market reference
$\underline{p}_{o,u,s}^{raw}(t)$	UE-specific raw price	Computed	> 0	Cost-reflective price
$P_{o,u,s}(t)$	Final advertised price	Computed	> 0	Delivered UE price

5.4.3. Qualitative performance expectations

Before presenting numerical results, several qualitative behaviours are expected from the proposed pricing model.

When radio utilization is low and backhaul resources are not under constraint, prices decrease, thereby encouraging traffic consumption and attracting users from neighbouring gNBs. On the other hand, under high utilization or constrained backhaul conditions, the price will increase, which in turn will discourage excess demand and promote load redistribution among competition.

The reputation component is expected to penalize persistent overload, which leads to QoS violations, incentivizing operators to preserve headroom rather than maximize short-term throughput. Compared to the current static pricing models, the proposed approach is expected to both improve QoE stability for users and reduce performance issues due to congestion, thereby providing more stable revenue under dynamic conditions [55], [61], [67].

5.5. Numerical Evaluation and Simulation Study

This section evaluates the proposed UE-centric dynamic pricing framework through simulation. All experiments are conducted using a detailed 5G NR system model implemented in ns-3 with the 5G-LENA module, enabling fine-grained observation of radio access behaviour, scheduling outcomes, and pricing dynamics under controlled conditions [73], [74], [75].

5.5.1. Simulation setup

Simulation platform

The simulation is implemented in **ns-3**, using the **5G-LENA NR module**, which provides a standards-compliant implementation of the 5G NR protocol stack, including PHY, MAC, RLC, PDCP, and RRC layers [73], [74], [75]. The simulation models downlink traffic only and operates in Frequency Range 1 (FR1).

Network topology

A multi-cell deployment is considered, consisting of multiple gNBs arranged in a grid topology using the GridScenarioHelper provided by 5G-LENA [74]. The number of gNBs and inter-site distance are configurable parameters, allowing exploration of scenarios ranging from sparse deployments to dense competitive environments.

Each gNB operates a single sector and serves multiple UEs concurrently. UEs are distributed across the coverage area and may observe multiple gNBs simultaneously, enabling competitive access selection.

Radio and channel configuration

The radio access network follows the 3GPP NR architecture as defined in TS 38.300 [57]. A single component carrier and bandwidth part (BWP) is used in the baseline configuration, with extensibility toward multi-BWP setups.

Radio propagation is modelled using the **3GPP TR 38.901 Urban Micro (UMi) channel model**, as implemented in the ns-3 NR module [76]. Ideal beamforming is assumed, and shadow fading is disabled to isolate pricing and scheduling effects.

Mobility model

UE mobility is modelled using the **Random Waypoint Mobility Model**, with bounded movement and randomized speed and pause times [77]. Mobility induces time-varying radio conditions, handovers, and changes in resource demand, which are central to evaluating dynamic pricing behaviour.

Alternative mobility patterns (e.g., hotspot-based or trace-driven mobility) are left for future work but can be incorporated without modifying the pricing framework.

Traffic and service configuration

Each UE generates downlink traffic corresponding to one of three service classes:

- **Low-latency traffic**, representing interactive or gaming-like applications,
- **Voice traffic**, modelled as conversational flows,
- **Video traffic**, represented by constant bitrate streams.

Service class information is attached to UEs via metadata tags and mapped to standardized 3GPP QoS identifiers (QCIs/5QIs). Traffic parameters (packet size and rate) are configurable per service class.

Scheduling and resource allocation

Downlink scheduling is performed by a **proportional-fair (PF) NR MAC scheduler**, which allocates PRBs dynamically based on channel conditions and fairness objectives [74], [78]. Crucially, the simulation records **actual MAC-layer scheduling outcomes**, including PRB allocation and achieved throughput, at short time intervals, rather than relying on simulation-wide averages.

Pricing configurations

To benchmark the simulation, we compare the following pricings: The following pricing configurations are evaluated:

1. **Static baseline pricing**
   A fixed price independent of utilization, competition, or radio conditions. Optionally parameterized by the backhaul type of the carrier.
2. **Utilization-aware pricing**
   Pricing depends solely on instantaneous radio resource utilization.
3. **The proposed model**
   Pricing incorporates utilization, competition, backhaul abstraction, reputation, and revenue-related components as defined in Section 5.4.

Each configuration uses identical network and traffic settings to ensure fair comparison.

Table 8: Pricing coefficients

Parameter	**Description**	**Typical value(s)**	**Role in pricing behavior**
$\boldsymbol{P_0}$	Base price per service unit	1.0 (normalized)	Sets global price scale
$\boldsymbol{w_s}$	Service weight	Voice: 0.9Video: 1.1Low-latency: 1.3Best-effort: 1.0	Differentiates services by priority and resource intensity
γ	RB congestion coefficient	1.0 – 3.0	Controls price increase under high radio utilization
$\boldsymbol{p}$	Utilization exponent	1 – 2	Shapes sensitivity near saturation (convexity)
β	Radio efficiency sensitivity	0.5 – 1.0	Penalizes inefficient radio links
η_{ref}	Reference spectral efficiency	3 bps/Hz	Normalization point for radio cost

θ	Service congestion coefficient	0.3 – 0.7	Price ramp for crowded services
α_k	Loyalty discount strength	0.3 – 0.6	Rewards repeat users
$\boldsymbol{K_{ref}}$	Loyalty normalization	10 sessions	Scale for loyalty effect
α_R	Revenue sensitivity	0.1 – 0.3	Controls value-based discount or surcharge
$\boldsymbol{R_{ref}}$	Revenue normalization	Median UE revenue	Stabilizes revenue-based pricing
λ	Price coupling coefficient	0 – 0.3	Strength of herd / market coupling
$\boldsymbol{p_{min}}$	Minimum price	Scenario-defined	Prevents underpricing
$\boldsymbol{p_{max}}$	Maximum price	Scenario-defined	Prevents excessive pricing

5.5.2. Performance metrics

To evaluate the impact of dynamic pricing, both **network-centric** and **user-centric** metrics are collected.

- **Network-centric metrics**

- **Radio utilization**: fraction of PRBs used per gNB and BWP over time.
- **Throughput**: aggregate and per-gNB achieved throughput.
- **Load distribution**: variance of utilization across gNBs, indicating load balancing effects.

- **Revenue**: total and per-gNB accumulated revenue over the simulation duration.

- **User-centric metrics**

- **Achieved throughput per UE**, by service class.
- **Packet delay and packet loss**, where applicable.
- **Price paid per bit or per session**, averaged per UE.
- **User engagement**, defined as the fraction of offered traffic successfully served.
- **Competition-related metrics**
- **Access selection dynamics**: frequency of UE switching between gNBs.
- **Market share**: proportion of UEs attached to each gNB over time.
- **Price dispersion**: variance of advertised prices across competing gNBs.

These metrics enable assessment of both efficiency and fairness, as well as the economic stability of the system.

5.6. Conclusion and Future Works

This deliverable set out to investigate a more UE-centric dynamic pricing framework for the next generation of mobile networks. Motivated by the limitations of static or plan-based pricing models, the current model was designed to align the economic incentives of MNOs with real-time radio access conditions and competitive dynamics. By coupling pricing decisions directly to measurable network states such as radio resource utilization, link quality, service type, and access competition, the current model aims to break away from the traditional models focused mainly on the MNOs.

To validate the proposed approach, we evaluate the framework through a 5G NR compliant simulation instantiation (section 5). Contrary to previous works in the field that rely on aggregate or long-term abstractions of data, the proposed framework operates at fine time granularity and incorporates scheduling outcomes and utilization to inform pricing decisions. This coupling of the physical layer’s behaviour and pricing enables the price to reflect instantaneous conditions of the network, thereby supporting more efficient and responsive access selection.

Through the simulation, the study demonstrated that dynamic pricing can improve load distribution across competing gNBs, mitigate congestion during busy periods, and provide more stable overall performance across the different service classes. Compared to static pricing models, the proposed approach offers improved adaptability to fluctuations caused by the mobility of users and increases competition. At the same time, the limitations set by the model maintain a stable operator revenue, as well as a steady user quality experience.

Beyond the results provided by the simulation, the framework provides a flexible foundation to explore decentralized and hybrid mobile ecosystems, wherein traditional MNOs coexist and compete with smaller community-operated access points. By explicitly modelling competition, backhaul differences, and decision-making of users, the current work bridges the gap between economic pricing models and realistic 5G NR simulations.

The current research opens the door for several future works to explore and enhance the framework.

Firstly, the current pricing mechanism assumes perfect and instantaneous knowledge of the network state. Future work can investigate the impact of delay, noise, or incomplete information on pricing stability and user behaviour, as well as predictive pricing approaches that anticipate congestion based on traffic forecasts.

Second, while backhaul characteristics are incorporated as pricing inputs, future simulations may explicitly force backhaul capacity and latency restraints. This will enable the investigation of congestion-aware pricing that considers both the RAN and the transport bottlenecks. This becomes particularly relevant for wireless and satellite backhaul scenarios, which are common in rural and community deployments.

Third, with the rise of AI and its popularity, the pricing model can be extended to “learn”. Reinforcement learning could be used to adapt the utility weights dynamically, thereby enabling operators to balance revenue, fairness, and QoE objectives in competitive conditions.

Fourth, the framework can be generalized to support multi-BWP and network slicing scenarios, something that will grow in 5G infrastructure in the future. In this scenario, different service classes are allocated distinct radio resources and priced accordingly. This would allow the research of slice-aware pricing and admission control in 5G networks with multiple tenants.

Finally, future work may incorporate user churn, session dynamics, and contractual constraints to study market equilibria and strategic behaviour in the long term. Integrating regulatory considerations and real-world billing constraints would further increase the applicability of the current framework to real-world operational deployments.

6. References

[1] A. Tsiota, D. Xenakis, N. Passas and L. Merakos, "On Jamming and Black Hole Attacks in Heterogeneous Wireless Networks," in IEEE Transactions on Vehicular Technology, vol. 68, no. 11, pp. 10761-10774, Nov. 2019.

[2] A. Tsiota, **D. Xenakis**, N. Passas and L. Merakos, “Multi-Tier HetNets With Random DDoS Attacks: Service Probability and User Load Analysis”, *IEEE Transactions on Information Forensics and Security*, vol. 20, pp. 6190-6204, 2025. 10.1109/TIFS.2025.3575599.

[3] **D. Xenakis**, “To DASH, or Not to DASH? Optimal Video Bitrate Selection and Edge Network Caching in MEC-Empowered Slice-Enabled Networks,” *IEEE Transactions on Vehicular Technology*, vol. 73, no. 4, pp. 5556-5571, April 2024, 10.1109/TVT.2023.3329662.

[4] E. Liotou, **D. Xenakis**, V. Georgara, G. Kourouniotis, L. Merakos, “Cache-Enabled Adaptive Video Streaming: A QoE-Based Evaluation Study”, Future Internet 15, 221, 2023. https://doi.org/10.3390/fi15070221.

[5] **D. Xenakis**, C. Koulis, A. Tsiota, N. Passas, C. Xenakis, «Contract-less Mobile Data Access Beyond 5G: Fully-decentralized, high-throughput and anonymous asset trading over the Blockchain», *IEEE Access*, vol. 9, pp. 73963-74016, 2021, https://doi.rorg/10.1109/ACCESS.2021.3079625

[6] S. Nakamoto. (May 2008). Bitcoin: A Peer-to-Peer Electronic Cash System. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[7] V. Buterin, “A next-generation smart contract and decentralized application platform”, White Paper, 2014. [Online] https://cryptorating.eu/whitepapers/Ethereum/Ethereum_white_paper.pdf.

[8] N. Laoutaris, “Why Online Services Should Pay You for Your Data? The Arguments for a Human-Centric Data Economy,” IEEE Internet Computing, Vol. 23, No. 5, Dec. 2019.

[9] Ericsson Mobility Report, November Full report, Nov. 2025.

[10] Y. Xiao, N. Zhang, W. Lou, and Y. T. Hou, "A Survey of Distributed Consensus Protocols for Blockchain Networks", IEEE Communications Surveys & Tutorials, vol. 22, no. 2, pp. 1432-1465, Q2 2020.

[11] W. Wang, D. T. Hoang, P. Hu, Z. Xiong, D. Niyato, P. Wang, Y. Wen, D. In Kim, "A Survey on Consensus Mechanisms and Mining Strategy Management in Blockchain Networks," IEEE Access, vol. 7, pp. 22328-22370, 2019.

[12] YouTube, official YouTube stats report. [Online] https://www.youtube.com/intl/en-GB/about/press/.

[13] Omnicore Agency, "Facebook by the Numbers: Stats, Demographics and Fun Facts", Online Article, Updated Jan. 6, 2021. [ONLINE] https://www.omnicoreagency.com/facebook-statistics/

[14] Blockchain.com statistics on Bitcoin [Accessed online on 21 Sept 2020], https://www.blockchain.com/charts/transactions-per-second

[15] Blockchair.com statistics on Ethereum [Accessed online on 21 Sept 2020], "https://blockchair.com/ethereum/charts/transactions-per-second|.

[16] “Visa fact sheet,” VISA Inc. [Online]. https://usa.visa.com/dam/VCOM/download/corporate/media/ visanet-technology/aboutvisafactsheet.pdf

[17] S. Meiklejohn, M. Pomarole, G. Jordan, K. Levchenko, G. M. Voelker, S. Savage, and D. McCoy, "A fistful of bitcoins: Characterizing payments among men with no names", Proceedings of the 2013 conference on Internet measurement conference, Barcelona - Spain, Oct. 2013.

[18] F. Tschorsch and B. Scheuermann, "Bitcoin and Beyond: A Technical Survey on Decentralized Digital Currencies", IEEE Communications Surveys & Tutorials, vol. 18, no. 3, pp. 2084-2123, Q3 2016.

[19] W. Chan and A. Olmsted "Ethereum transaction graph analysis", 12th International Conference for Internet Technology and Secured Transactions (ICITST), Cambridge, pp. 498-500, IEEE, Dec. 2017.

[20] F. Beres, I. A. Seres, A. A. Benczur, M. Quintyne-Collins, "Blockchain is Watching You: Profiling and Deanonymizing Ethereum Users", arXiv preprint, submitted May 2020, arXiv:2005.14051 [cs.CR].

[21] J. Bonneau, A. Narayanan, A. Miller, J. Clark, J. A. Kroll, and E. W. Felten, "Mixcoin: Anonymity for bitcoin with accountable mixes", International Conference on Financial Cryptography and Data Security, pp 486-504, vol. 8437, Lecture Notes in Computer Science, Springer, Nov. 2014.

[22] L. Valenta, B. Rowan, "Blindcoin: Blinded, accountable mixes for bitcoin", International Conference on Financial Cryptography and Data Security", pp 112-126, vol. 8976, Lecture Notes in CS, Springer, Sept. 2015.

[23] Greg Maxwell, "Coinjoin: Bitcoin privacy for the real world", post on Bitcoin forum, 2013. [Online] https://bitcointalk.org/?topic=279249

[24] T. Ruffing, P. Moreno-Sanchez, and A. Kate, "Coinshuffle: Practical decentralized coin mixing for bitcoin", European Symposium on Research in Computer Security, pp 345-364, vol. 8713, Lecture Notes in Computer Science, Springer, Sept. 2014.

[25] G. Bissias, A. Pinar Ozisik, B. N Levine, and M. Liberatore, "Sybil-resistant mixing for bitcoin", Proceedings of the 13th Workshop on Privacy in the Electronic Society, pp. 149-158. ACM, Nov.2014.

[26] E. Heilman, L. Alshenibr, F. Baldimtsi, A. Scafuro, and S Goldberg, "Tumblebit: An untrusted bitcoin-compatible anonymous payment hub", Netw. and Distr. Syst. Sec. Symp. (NDSS), San Diego - USA, Feb, 2017.

[27] S. Meiklejohn and R. Mercer, "Mobius: Trustless tumbling for transaction privacy", Proceedings on Privacy Enhancing Technologies (PETS), Barcelona - Spain, June 2018.

[28] barryWhiteHat, "Miximus", Github documentation post, 2018. [Online] https://github.com/barryWhiteHat/miximus.

[29] I. A. Seres, D. A. Nagy, P. Burcsi, C. Buckland, "MixEth: efficient, trustless coin mixing service for Ethereum", Int. Conf. on Blockchain Econ., Sec. and Protoc. (Tokenomics), Paris - France, May 2019.

[30] R. Dingledine, N. Mathewson, P. Syverson, "Tor: The Second-Generation Onion Router", Proceed. of the 13th USENIX Sec. Symp., 2004.

[31] E. Letsoalo and S. Ojo, "Survey of Media Access Control address spoofing attacks detection and prevention techniques in wireless networks," 2016 IST-Africa Week Conference, pp. 1-10, Durban - South Africa, 2016.

[32] N. E. Hastings and P. A. McLean, "TCP/IP spoofing fundamentals," Proceedings of the 1996 IEEE Fifteenth Annual Intern. Phoenix Con. on Comp. and Commun., Scottsdale, AZ, USA, 1996, pp. 218-22.

[33] O. A. Osanaiye, "Short Paper: IP spoofing detection for preventing DDoS attack in Cloud Computing," 2015 18th Intern. Conf. on Intelligence in Next Generation Networks, pp. 139-141, Paris - France, 2015.

[34] Y. Chang, K. Yoon, and D. Park, "A Study on the IP Spoofing Attack through Proxy Server and Defense Thereof," 2013 Intern. Conf. on Inform. Science and Applications. (ICISA), Suwon - South Korea, 2013.

[35] 3GPP TS 23.501, “System architecture for the 5G System (5GS)”, V16.8.0, Rel.16, March 2021.

[36] M. M. Azari, S. Solanki, S. Chatzinotas, O. Kodheli, H. Sallouha, A. Colpaert, J. F. M. Montoya, S. Pollin, A. Haqiqatnejad, A. Mostaani, E. Lagunas, and B. Ottersten, "Evolution of Non-Terrestrial Networks From 5G to 6G: A Survey," in IEEE Communications Surveys & Tutorials, vol. 24, no. 4, pp. 2633-2672, 2022.

[37] G. M. Capez, S. Henn, J. A. Fraire, and R. Garello, "Sparse satellite constellation design for global and regional direct-to-satellite IoT services", in IEEE Transactions on Aerospace and Electronic Systems, vol. 58, no. 5, pp. 3786-3801, 2022.

[38] J. Parkin and M. Tripunitara, "Countering Subscription Concealed Identifier (SUCI)-Catchers in Cellular Communications", in International Conference on Information Systems Security, Springer, pp. 107-126, 2024.

[39] A. Saputhanthri, C. De Alwis, and M. Liyanage, "Survey on blockchain-based IoT payment and marketplaces", in IEEE Access, vol. 10, pp. 103411-103437, 2022.

[40] A. A. R. Alsaeedy and E. K. P. Chong, "A survey of mobility management in non-terrestrial 5G networks: Power constraints and signaling cost", in IEEE Access, vol. 12, pp. 107529-107551, 2024.

[41] M. Almekhlafi, A. Lesage-Landry, and G. K. Kurt, "Access Inequality in LEO Satellite Networks: A Case Study of High-Latitude Coverage in Northern Québec", in IEEE Open Journal of Vehicular Technology, pp. 1-17, 2025.

[42] H. B. Tsegaye, "Towards Resilient and Secure Beyond-5G Non-Terrestrial Networks (B5G-NTNs): An End-to-End Cloud-Native Framework", Doctoral Thrsis, Università degli Studi di Trento, 2024.

[43] B. Karaman, I. Basturk, S. Taskin, E. Zeydan, F. Kara, E. A. Beyazit, M. Camelo, E. Bjornson, and H. Yanikomeroglu, "Solutions for Sustainable and Resilient Communication Infrastructure in Disaster Relief and Management Scenarios," in IEEE Communications Surveys & Tutorials, 2025.

[44] A. Giannopoulos, P. Gkonis, A. Kalafatelis, N. Nomikos, S. Spantideas, P. Trakadas, and T. Syriopoulos, "From 6G to SeaX-G: Integrated 6G TN/NTN for AI-Assisted Maritime Communications—Architecture, Enablers, and Optimization Problems", in Journal of Marine Science and Engineering, MDPI, vol. 13, no. 6, p. 1103, 2025.

[45] C. T. Nguyen, Y. M. Saputra, N. Van Huynh, T. N. Nguyen, D. T. Hoang, D. N Nguyen, V.-Q. Pham, M. Voznak, S. Chatzinotas, and D.-H. Tran, "Emerging technologies for 6G non-terrestrial-networks: From academia to industrial applications", in IEEE Open Journal of the Communications Society, vol. 5 pp. 3852-3885, 2024.

[46] G. Giambene, E. O. Addo, Q. Chen, and S. Kota, "Design and Analysis of Low-Power IoT in Remote Areas With NTN Opportunistic Connectivity", in IEEE Transactions on Aerospace and Electronic Systems, 2024.

[47] O. Chukhno, N. Chukhno, A. Ometov, S. Pizzi, G. Araniti, and A. Molinaro, "Application-Driven Offloading of XR Mission Critical via Integrated TN/NTN", in IEEE Network, 2025.

[48] M. A. Mohsin, M. Umer, A. Umar, H. Abou-Zeid, and S. A. Hassan, "Computation offloading strategies in integrated terrestrial and nonterrestrial networks", in Non-Terrestrial Networks, Elsevier, pp. 217-233, 2026.

[49] A. Kak, V.-Q. Pham, H.T. Thieu, and N. Choi, "Role of Software-Defined Networking in Machine-Type Communications and Satellite Connectivity", in Integration of MTC and Satellites for IoT toward 6G Era, Wiley Online Library, pp. 215-244, 2024.

[50] 3GPP TS 38.811, “Study on New Radio (NR) to support non-terrestrial networks,” v15.4.0, 2020.

[51] 3GPP TS 23.501, “System architecture for the 5G System (5GS),” Release 17.

[52] 3GPP. (2025). Release 20 - 3GPP [Website]. 3rd Generation Partnership Project. https://www.3gpp.org/specifications-technologies/releases/release-20

[53] Cisco. 2020. Cisco Annual Internet Report (2018–2023): White Paper. Cisco Systems. https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.pdf

[54] Ericsson. 2023. Ericsson Mobility Report. Ericsson AB.

[55] X. Ma, M. Naldi, and F. Zambonelli. 2020. Dynamic plan control: An effective tool to manage demand considering mobile Internet network congestion. Applied Sciences 11, 1 (2020), 91.

[56] I. Malolli. 2017. Pricing policies and new business models for data communication over 5G networks. Economy & Business Journal 11, 1 (2017), 398–407.

[57] 3GPP. 2023. TS 38.300: NR and NG-RAN Overall Description. 3rd Generation Partnership Project.

[58] 3GPP. 2023. TS 23.501: System Architecture for the 5G System (5GS). 3rd Generation Partnership Project.

[59] K. Dorgham. 2016. A novel dynamic pricing model for mobile calls. In Telecommunication Economics. Springer, 215–232.

[60] B. Jukic. 2004. Congestion-based resource sharing in multi-service networks. Decision Support Systems 38, 3 (2004), 407–426.

[61] M. Zhang, L. Gao, and J. Huang. 2019. Hybrid pricing for mobile collaborative Internet access. IEEE/ACM Transactions on Networking 27, 3 (2019), 1019–1032.

[62] N. AbuAli. 2010. Congestion-based pricing resource management in broadband wireless networks. Ph.D. Dissertation. Queen’s University.

[63] M. Flamini and M. Naldi. 2023. Optimal pricing in a rented 5G infrastructure scenario with sticky customers. Future Internet 15, 2 (2023), 82.

[64] T. M. Heikkinen. 2002. On congestion pricing in a wireless network. Wireless Networks 8, 4 (2002), 347–354. https://link.springer.com/article/10.1023/A:1015578321066

[65] Congestion-prevention in broadband wireless access systems using call admission control-based dynamic pricing. (PDF entry) https://scispace.com/pdf/congestion-prevention-in-broadband-wireless-access-systems-2w001ldnls.pdf

[66] S. Sen, C. Joe-Wong, S. Ha, and M. Chiang. 2013. Smart data pricing: Using economics to manage network congestion. Communications of the ACM 56, 12 (2013), 86–93. https://www.andrew.cmu.edu/user/cjoewong/SDP_CACM.pdf

[67] X. Ma, M. Naldi, and F. Zambonelli. 2017. Optimal dynamic pricing of mobile data plans in wireless communications. Omega 66 (2017), 91–105. https://www.sciencedirect.com/science/article/abs/pii/S0305048316000190

[68] Z. Xiong, Y. Zhang, D. Niyato, P. Wang, and A. Leshem. 2018. Dynamic pricing for revenue maximization in mobile social data market. arXiv:1808.04039. https://arxiv.org/pdf/1808.04039

[69] Access selection and joint pricing in multi-operator wireless networks: A Stackelberg game. (ResearchGate entry) https://www.researchgate.net/publication/279261459_Access_Selection_and_Joint_Pricing_in_Multi-Operator_Wireless_Networks_A_Stackelberg_Game

[70] S. K. Anantha Kumar, P. M. Santos, and others. 2023. Pricing models for 5G multi-tenancy using game theory: Case study for rural areas. IEEE Communications Magazine (2023). https://strathprints.strath.ac.uk/85725/1/Kumar_etal_IEEE_CM_2023_Pricing_models_for_5G_multi_tenancy.pdf

[71] J. S. Walia, E. Lopez-Aguilera, and others. 2021. A virtualization infrastructure cost model for 5G network slice provisioning. Future Internet 10, 3 (2021), 51. https://www.mdpi.com/2224-2708/10/3/51

[72] ITU-R. IMT-2020 Minimum Performance Requirements (M.2410-0). https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-M.2410-2017-PDF-E.pdf

[73] ns-3 Consortium. 2024. ns-3 Network Simulator. https://www.nsnam.org

[74] M. Polese, M. Giordani, T. Zugno, M. Zorzi, and M. Zorzi. 2019. End-to-End Simulation of 5G mmWave Networks. IEEE Communications Surveys & Tutorials 20, 3 (2019), 2237–2263. https://ieeexplore.ieee.org/document/8412483

[75] CTTC. 2024. 5G-LENA: 5G NR Module for ns-3. https://5g-lena.cttc.es/

[76] 3GPP. 2023. TR 38.901: Study on Channel Model for Frequencies from 0.5 to 100 GHz. 3rd Generation Partnership Project.

[77] D. Camp, J. Boleng, and V. Davies. 2002. A survey of mobility models for ad hoc network research. Wireless Communications and Mobile Computing 2, 5 (2002), 483–502.

[78] M. Koutlia, B. Bojovic, Z. Ali, and S. Lagen. 2022. Calibration of the 5G-LENA system level simulator in 3GPP reference scenarios. Simulation Modelling Practice and Theory 119 (September 2022), Article 102580, https://doi.org/10.1016/j.simpat.2022.102580