SNIA DEVELOPER CONFERENCE
SDC 25
By Developers FOR Developers

Hyatt Regency Santa Clara, CA
September 15-17, 2025

Object Store Federation

Enabling Global Namespace And Scalability For Object Storage

Sadhana Kannan, Nutanix
Mayur Sadavarte, Nutanix

Agenda

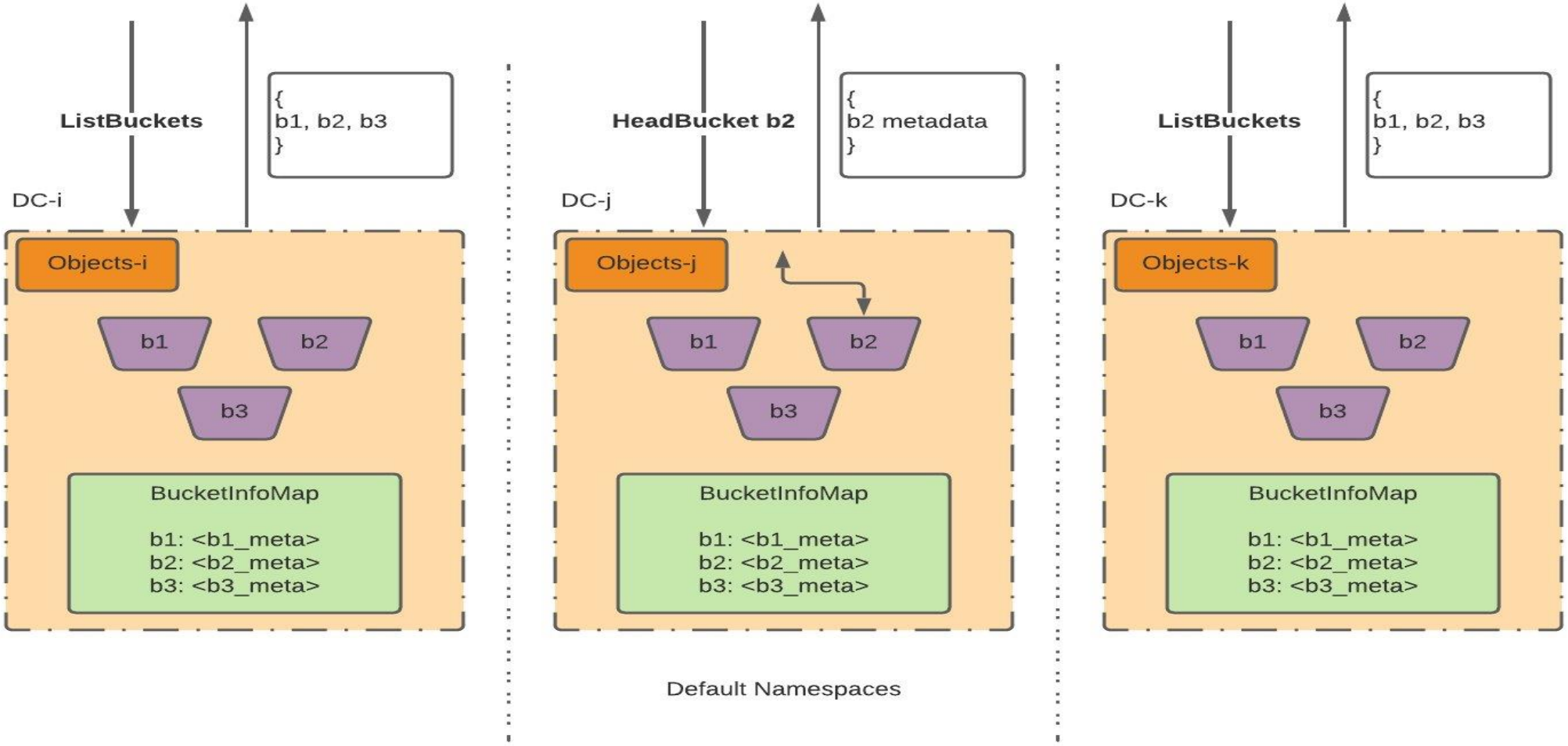
- Multiple Object Store Deployments
- Federation & Global Namespace
- Federation Architecture
- Federation Management
- Virtual Object Store

Multiple object store deployments

Enterprises have object store deployments across on-premise data centers, edge sites, public clouds. Reasons include:

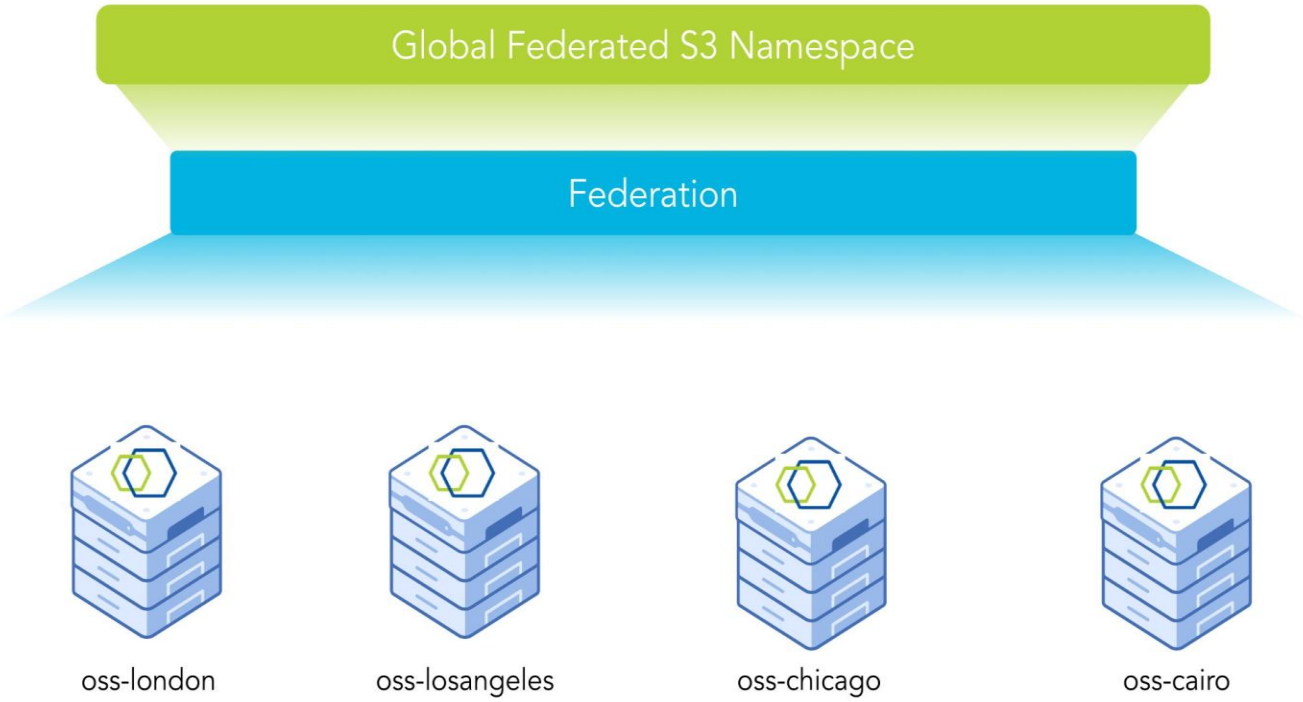
- Workload Optimization
  - High throughput data-heavy analytics, AI/ML workloads
  - Long term archival and backup
  - Cloud native applications
- Data Compliance & Governance
- Scalability
- Resilience And Availability

Default namespaces: Silos



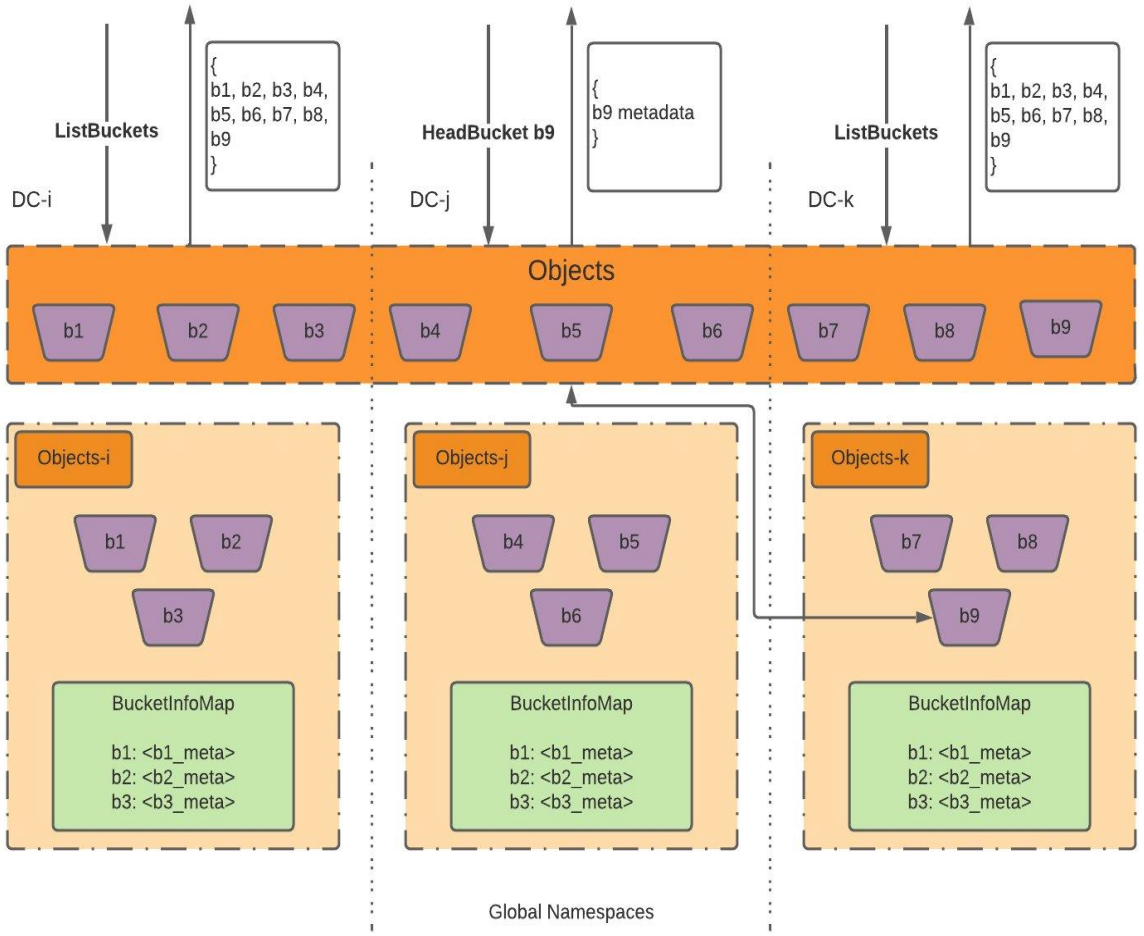
Introducing Federation

- Logical grouping of object stores
- Provides a unified Object Store global namespace
- Adds resources from existing object stores to boost capacity and performance
- Simplifies data access in geo-distributed environments
- Supports extreme scalability (hundreds of PBs)
- Lays the foundation for advanced features like caching, geo-resiliency



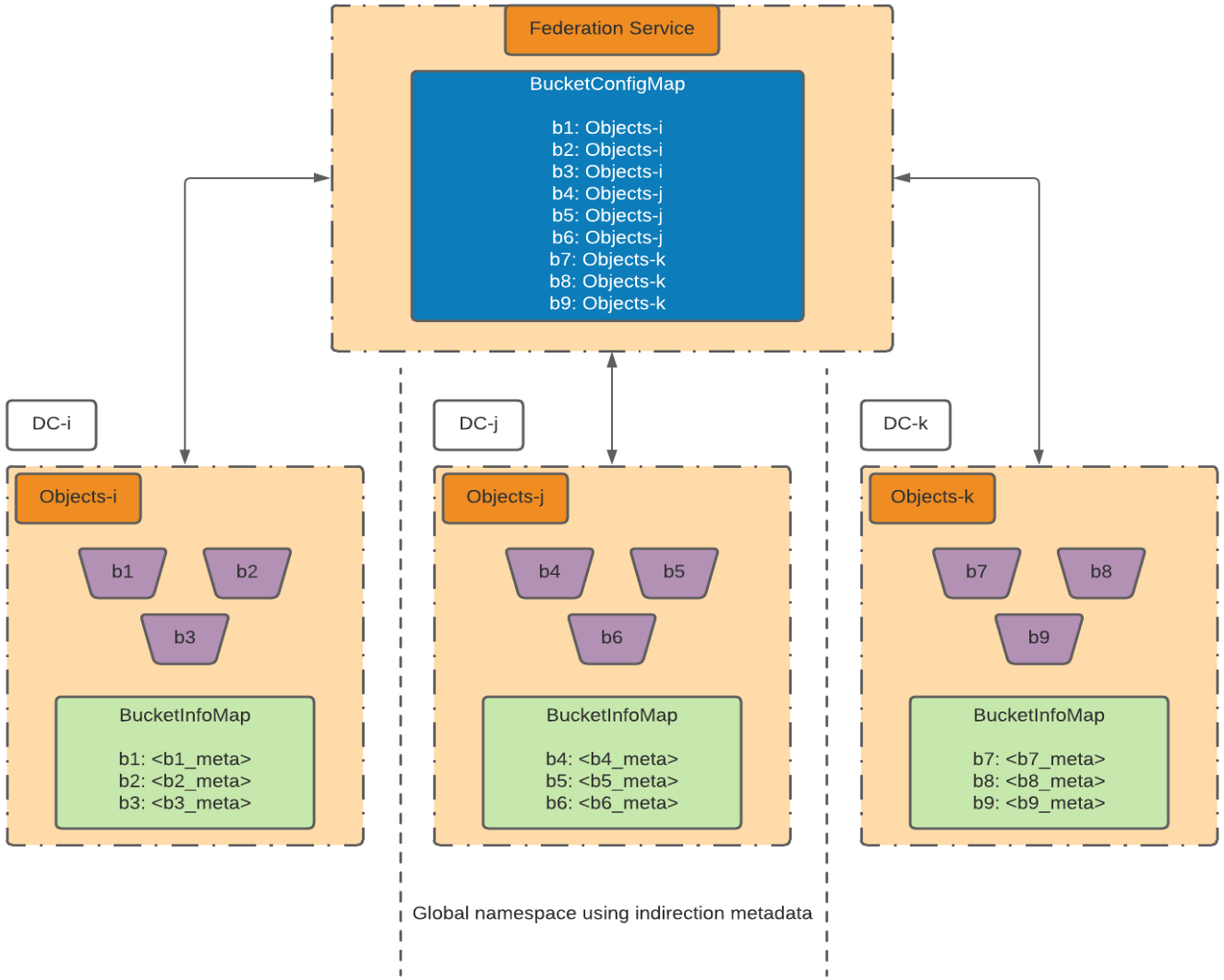
Global Namespace

- Object store namespace spans across multiple Objects instances in Federation.
- Same namespace presented to clients irrespective of service access point.
- Buckets and objects accessible from anywhere within the Federation.

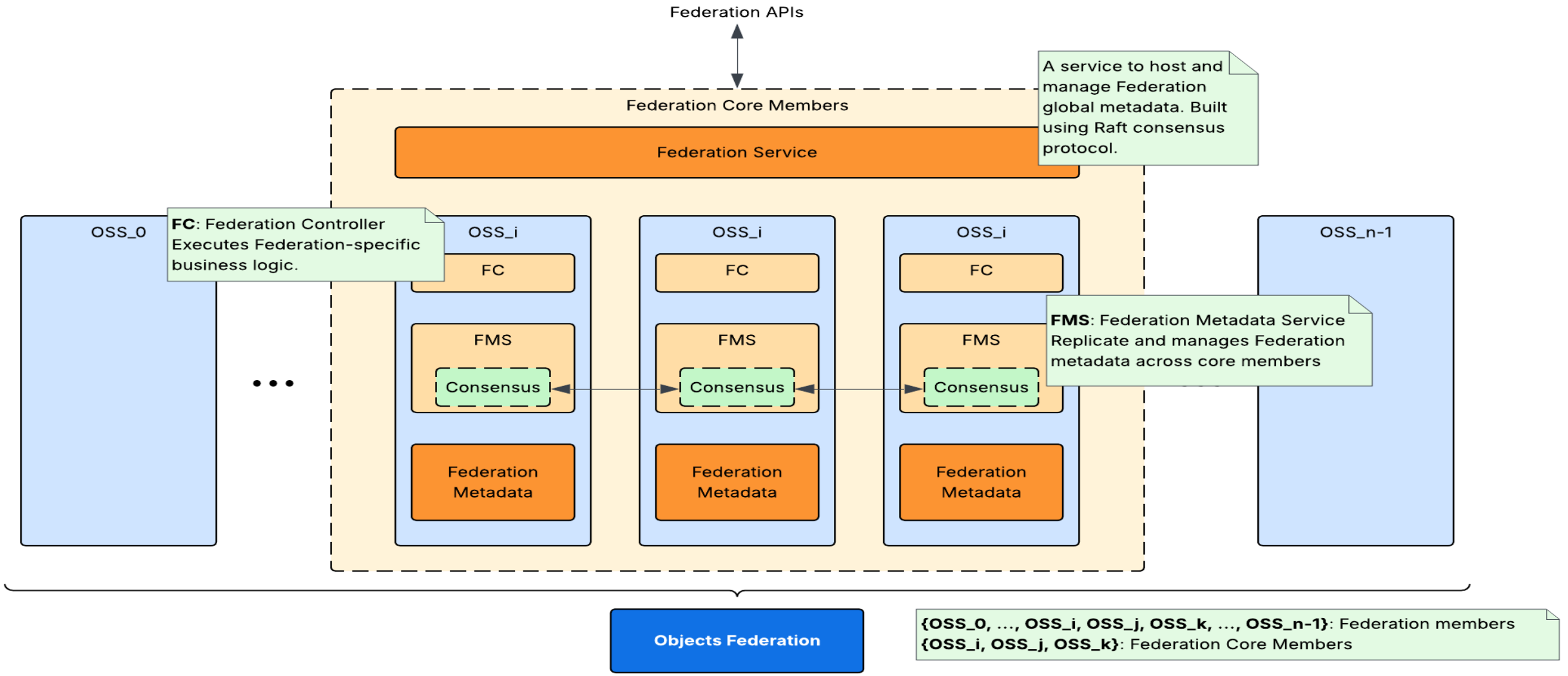


Global Metadata

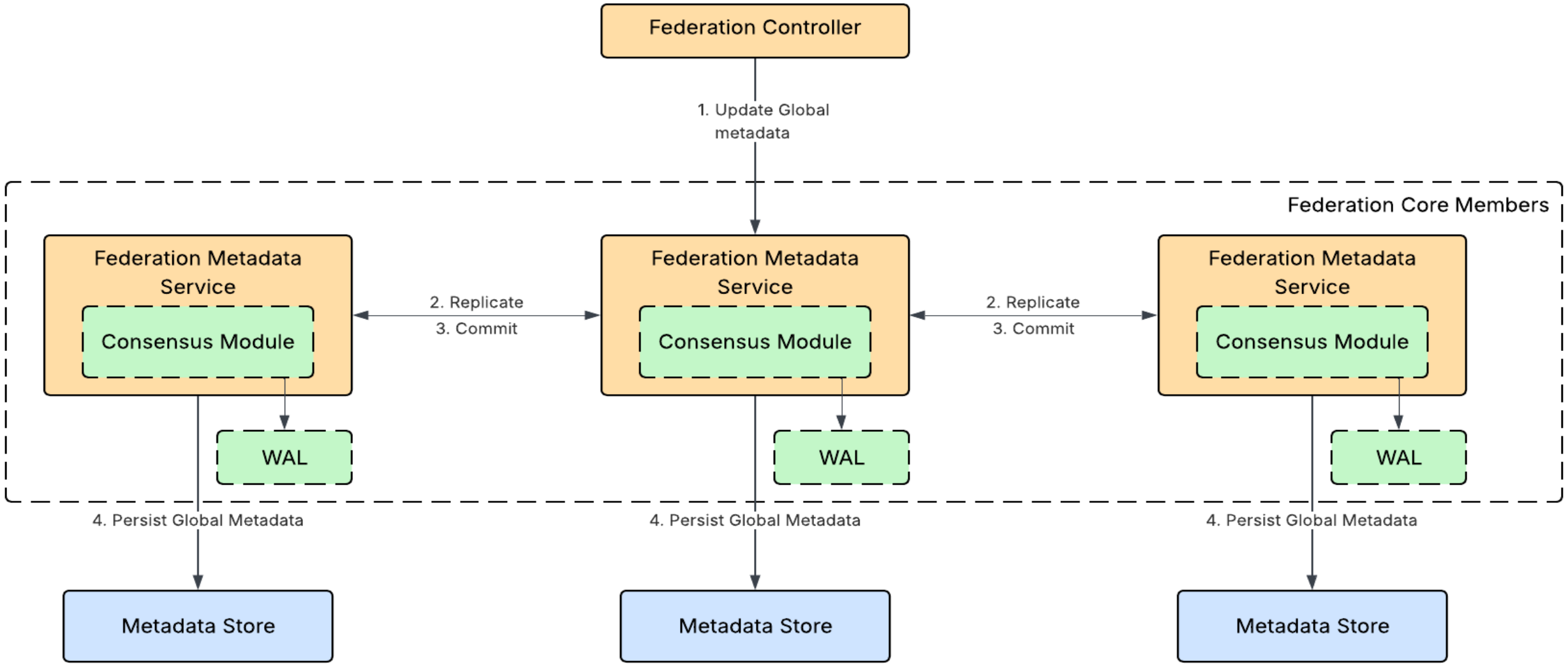
- Ability to list buckets from all Objects instances within the Federation
- Ability to route a request (S3 API) to appropriate Objects instance that's hosting the bucket
- Highly Available and Strongly Consistent



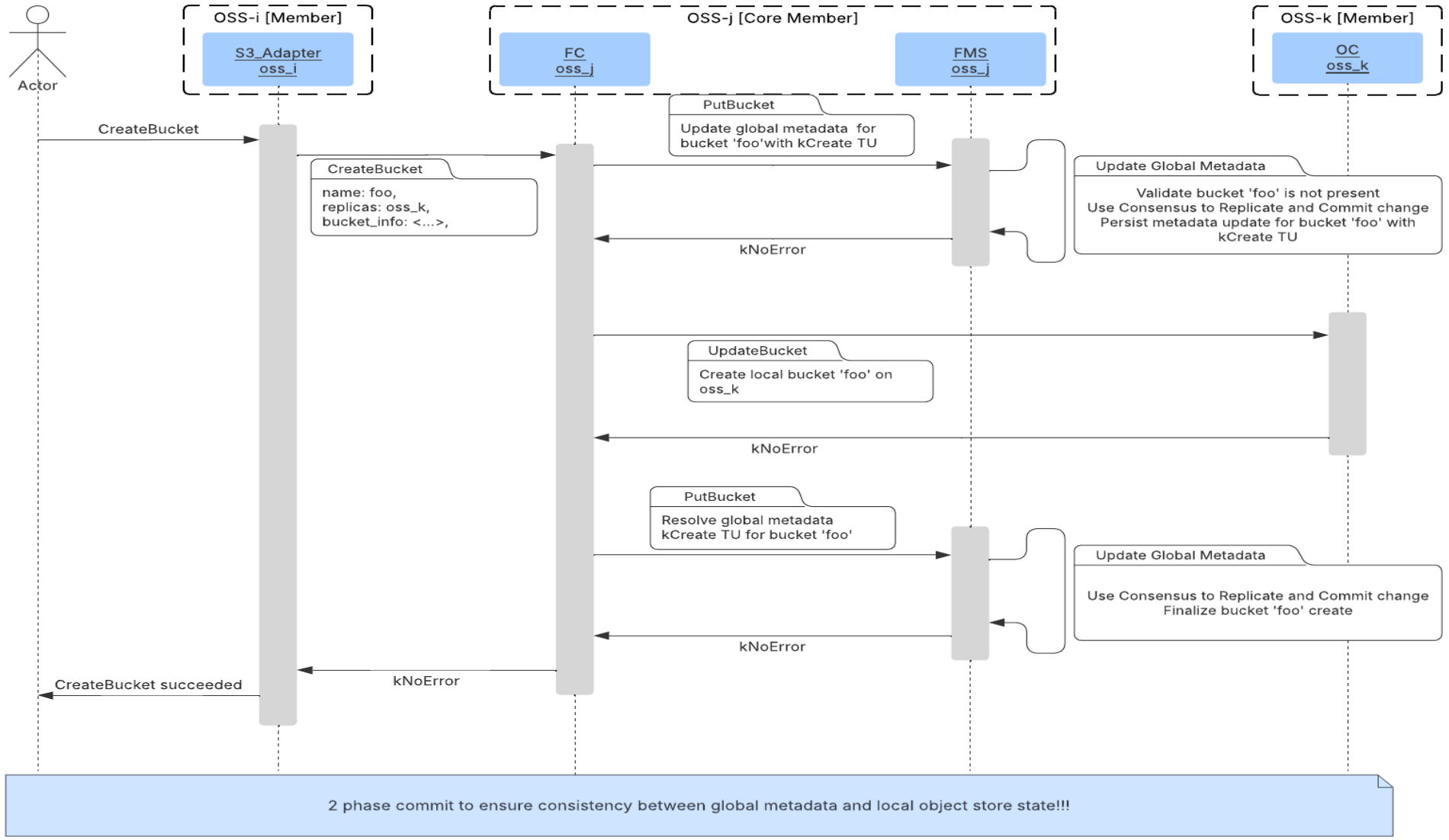
Federation Architecture



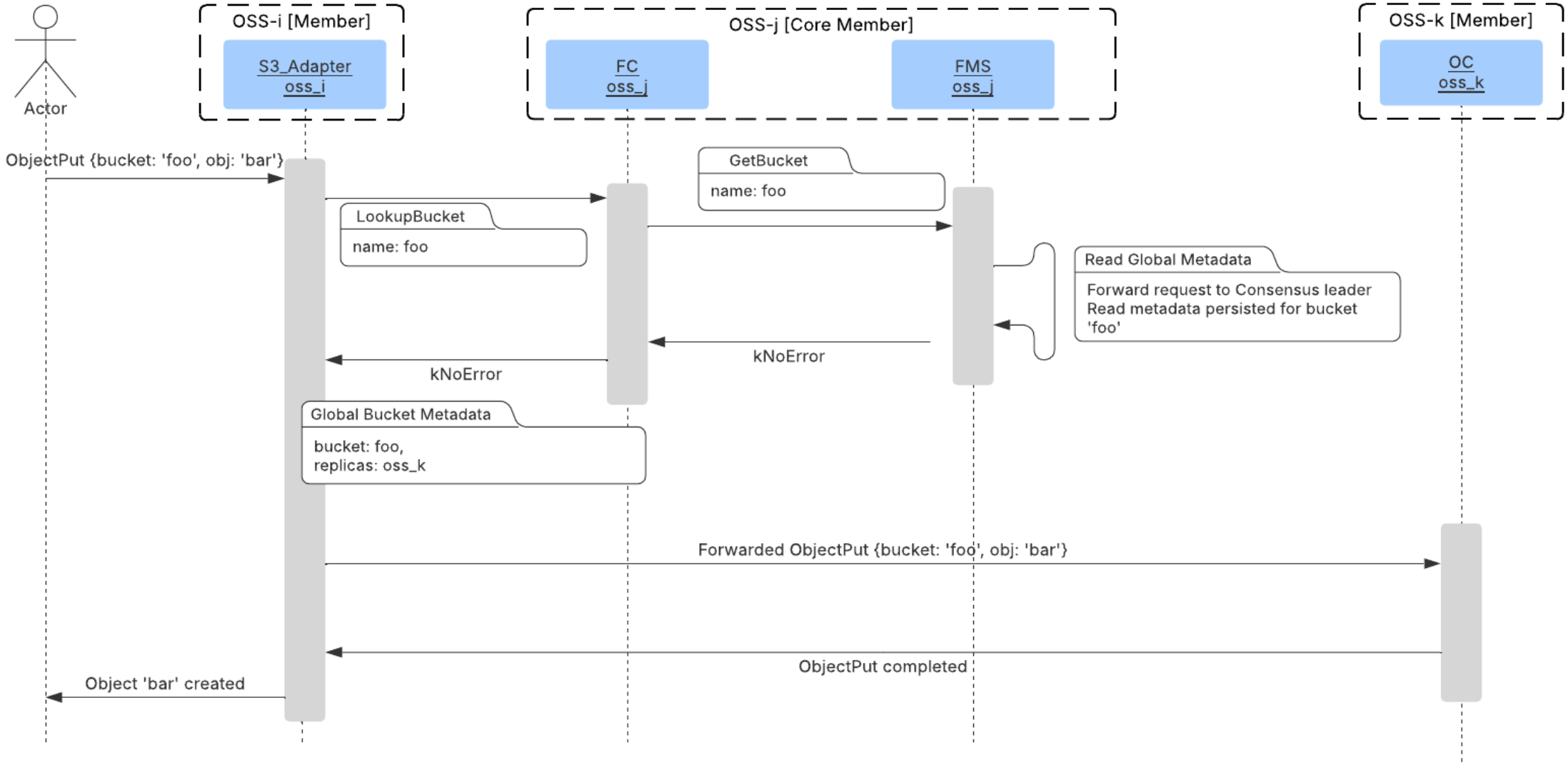
Global Metadata Updates



Federation Bucket Workflows – Create Bucket



Federation Data Access – Object Put



Federation Management

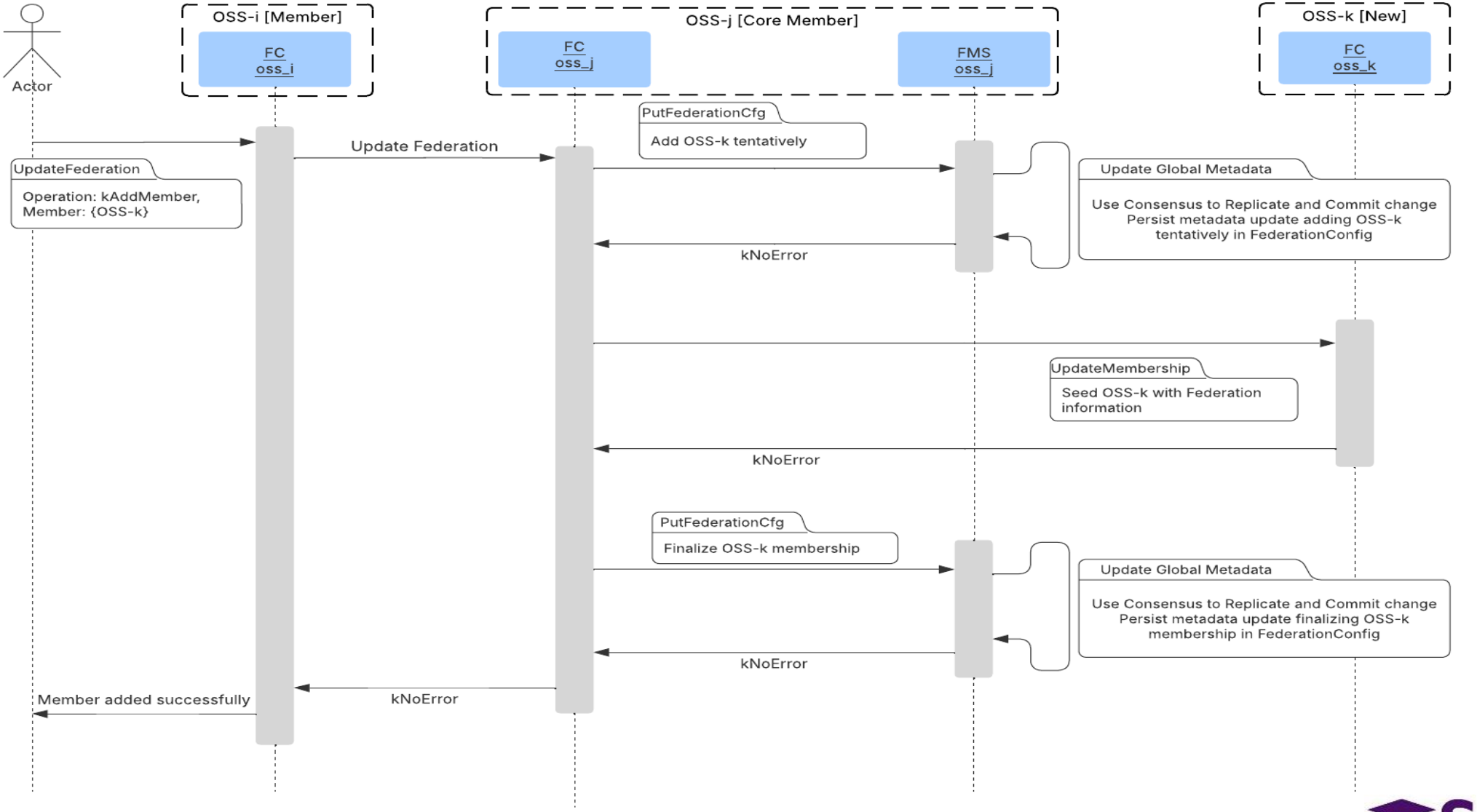
- Federation is a dynamic construct that needs to adapt to Customer workload needs
- Self Service API driven Federation Lifecycle Management
- Simple, seamless, and invisible to clients
- Supports creation and scaling of Federations on demand
- Ability to seed Federation context on new members
- Ability to propagate Federation membership changes to all members

Federation Config

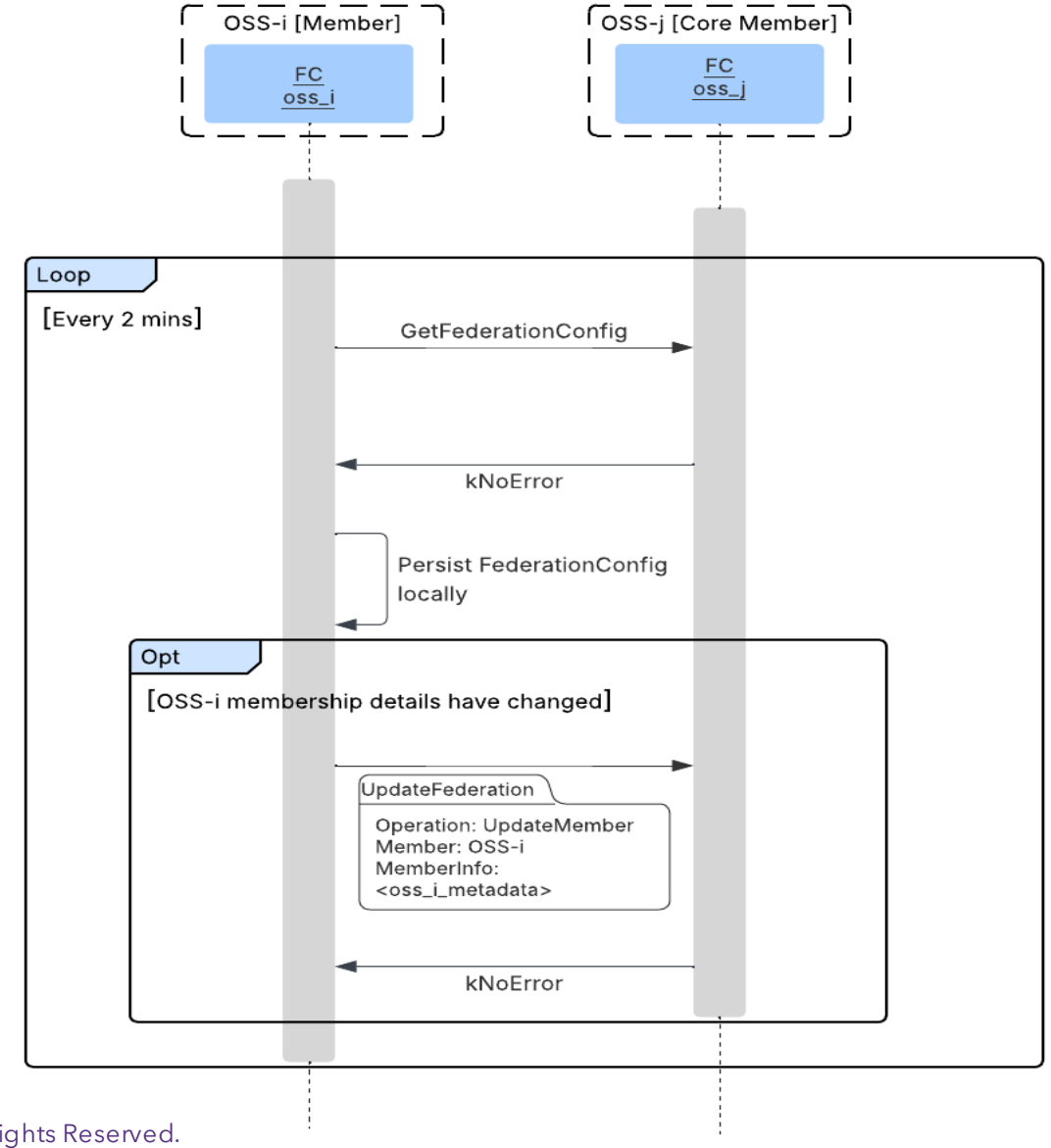
- Stores Federation details, including core and non-core members
- Initially created during Federation creation
- Replicated consistently across core members, along with Federation bucket metadata
- FS provides APIs to retrieve and modify Federation metadata

```
FederationConfig: {
   name: 'Dummy Federation',
   fqdn: 'dummy_fed.ntnx.com',
   core_members : [
      {
         name: 'oss_i',
         fqdn: 'oss_i.ntnx.com',
         <oss_i_metadata>
      },
      {
         name: 'oss_j',
         fqdn: 'oss_j.ntnx.com',
         <oss_j_metadata>
      },
      {
         name: 'oss_k',
         fqdn: 'oss_k.ntnx.com',
         <oss_k_metadata>
      }
   ],
   members: [
      {
         name: 'oss_0',
         fqdn: 'oss_0.ntnx.com',
         <oss_0_metadata>
      },
      .....
      {
         name: 'oss_n-1',
         fqdn: 'oss_n-1.ntnx.com',
         <oss_n-1_metadata>
      }
   ]
}
```


Federation Management Workflow – Add Member

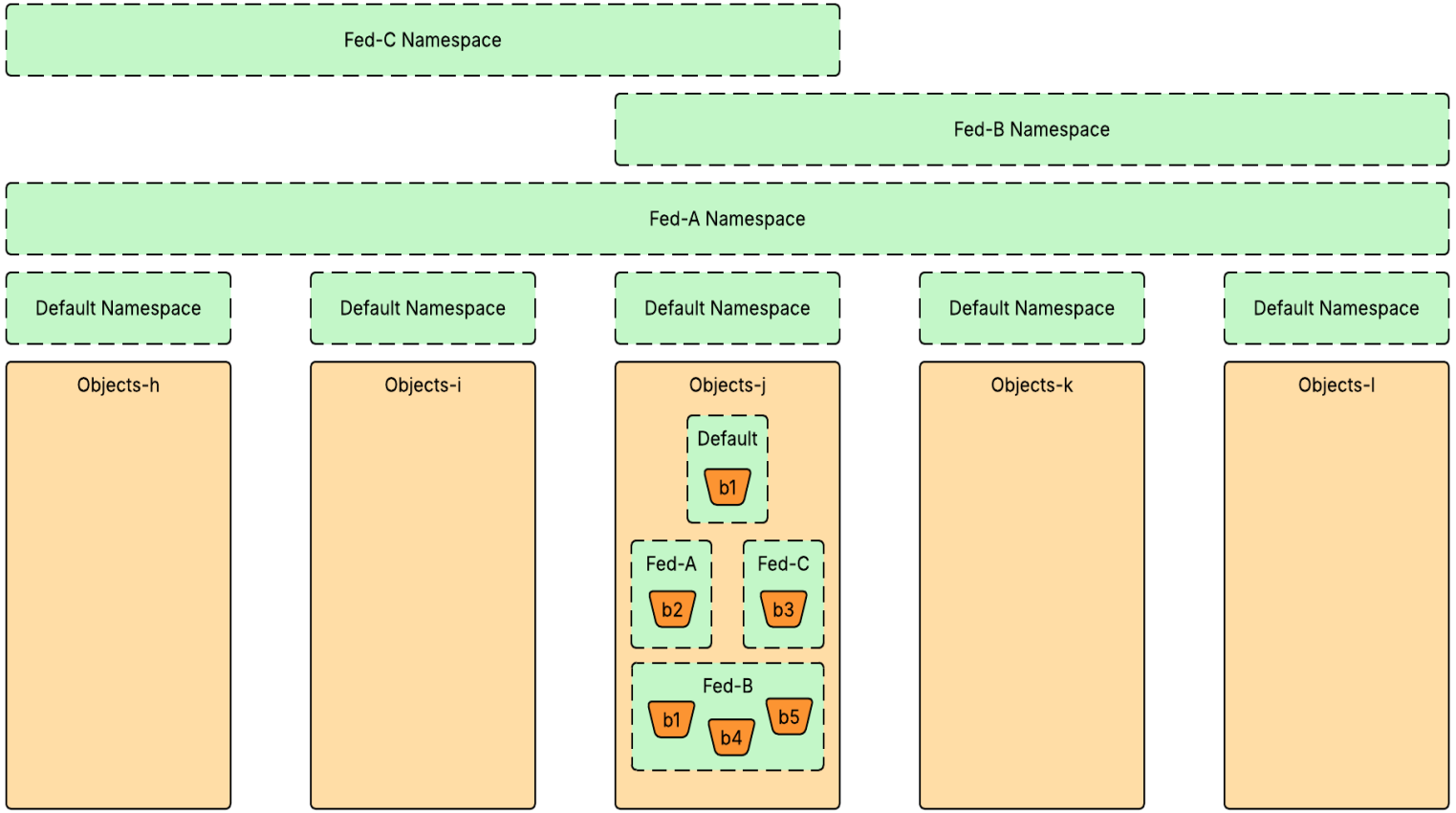


Federation Management Workflow – Sync Member



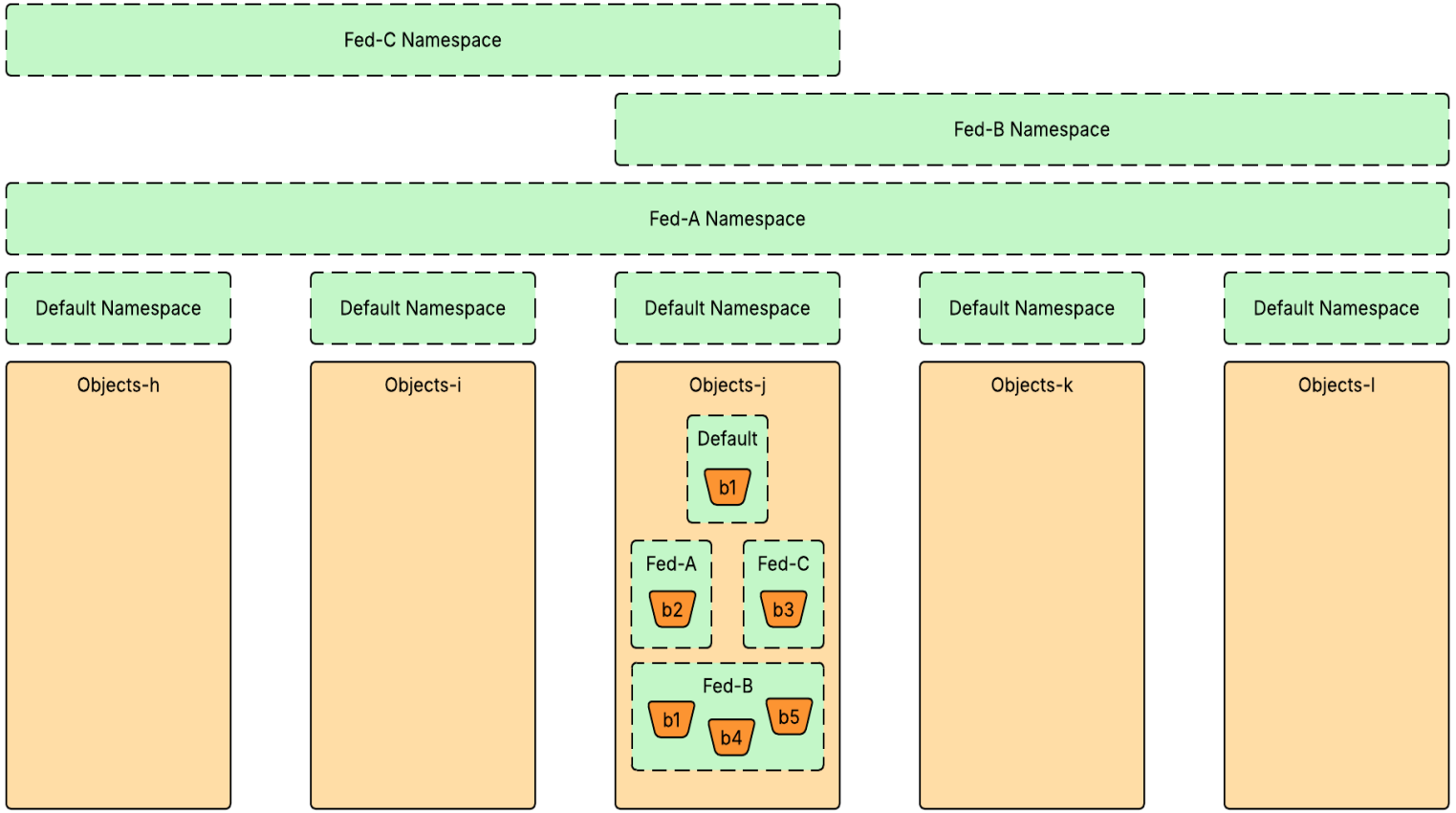
Virtualizing Object Store

- Each object store has its own default local S3 namespace
- An object store can be a part of multiple federated namespaces
- A namespace is a virtual construct!



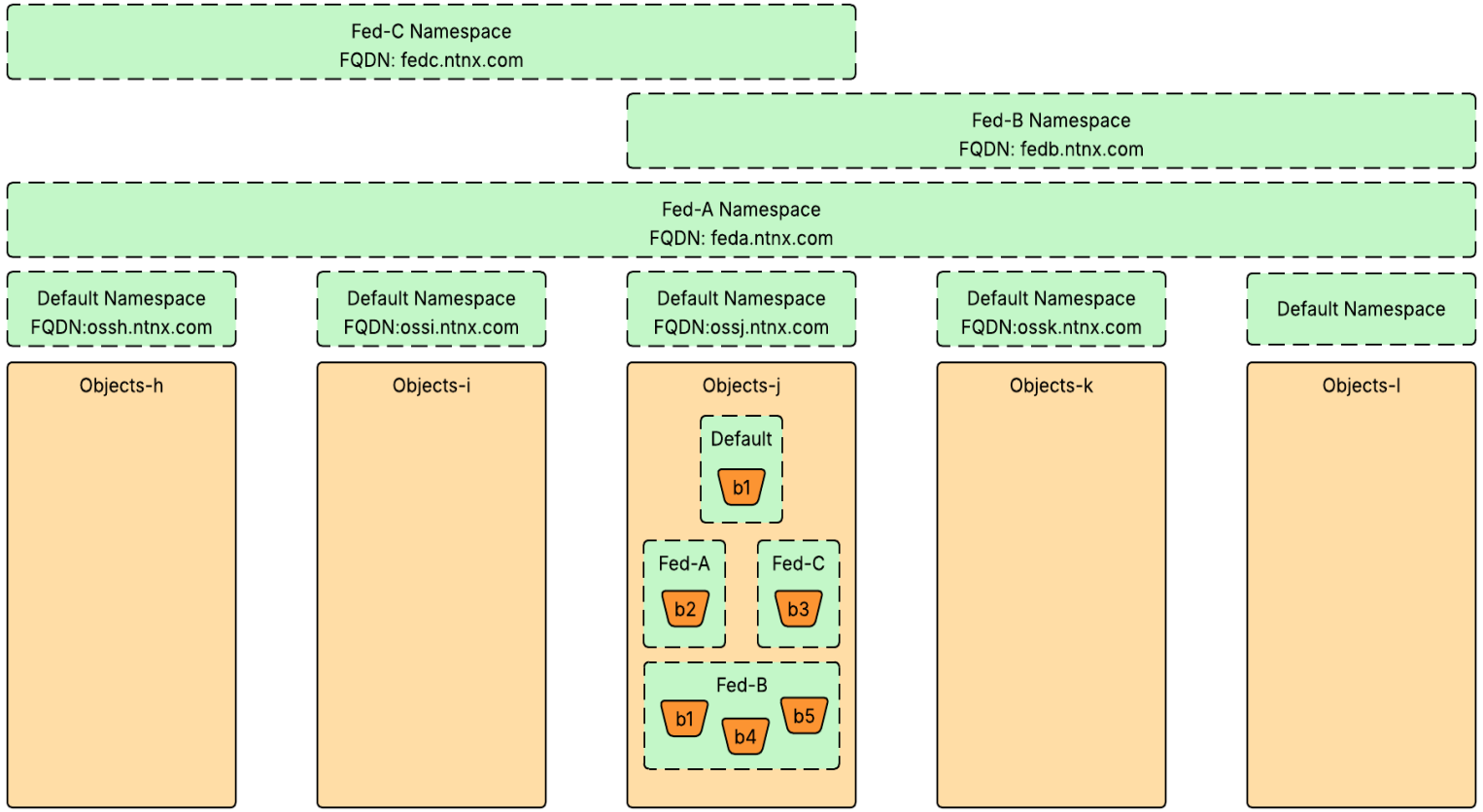
Buckets And Namespace

- Buckets from multiple namespaces can reside in the same object store
- A bucket is uniquely identified by Namespace + Bucket Name

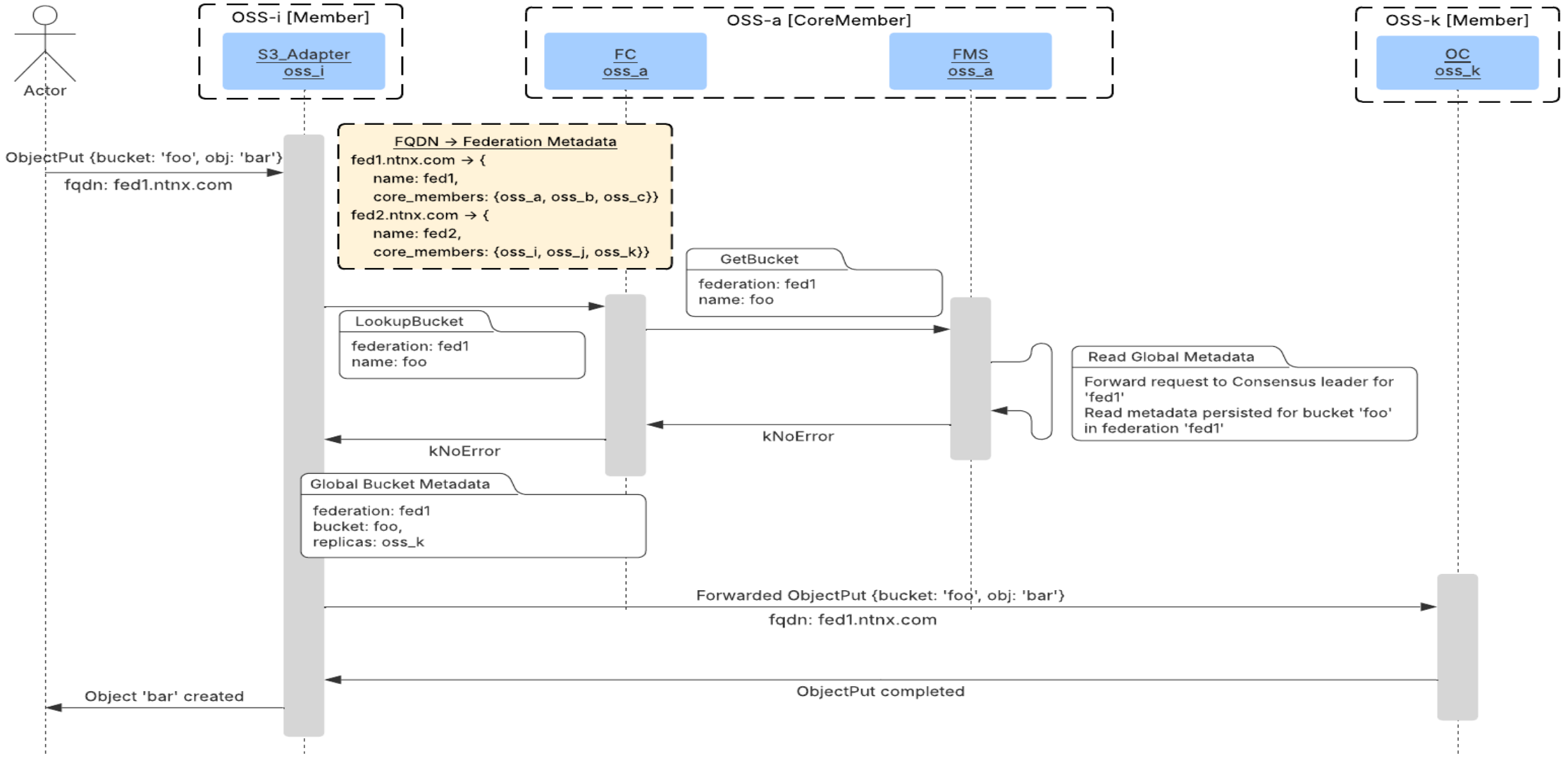


Federation FQDN

- Unique FQDN identifying the Federation
- Provided during Federation creation
- Only requests to this FQDN are treated as Federated requests
- Can be resolved to any of the members public IP addresses

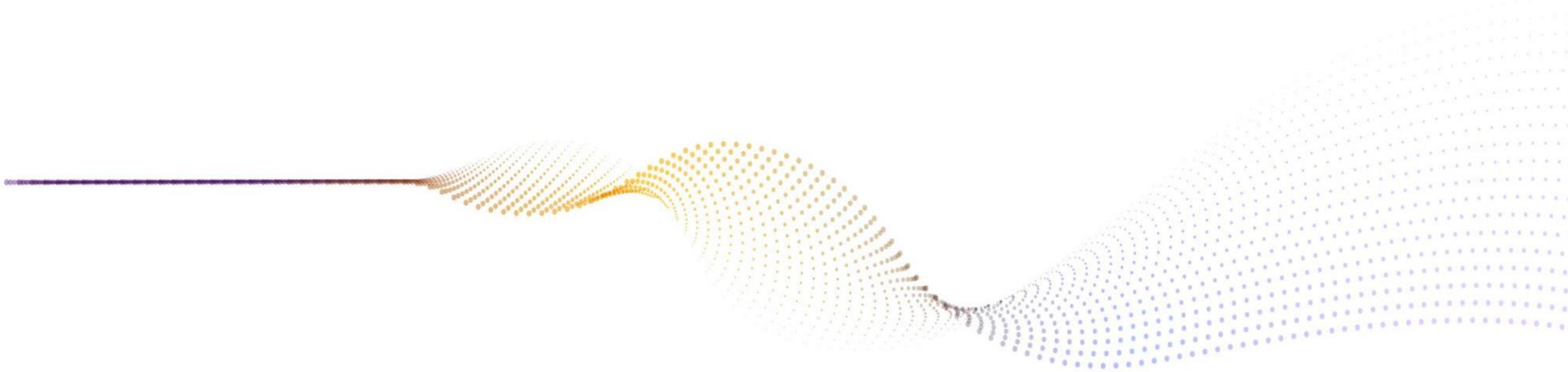


Federated Data Access – Object Put



Key Takeaways

- Leverage proven distributed systems principles to ensure reliability
- Contain complexity within a few services
  - Limit blast radius and isolate impact
  - Avoid major restructuring of existing code or multiple touchpoints
- Develop generic frameworks to maximize reuse
- Build abstractions to support extensible design
- Provide seamless lifecycle management for faster adoption



Thank you for attending!

Please remember to rate this session. You get access the presentations at
http://sniadeveloper.org/conference

