Triple Space Architecture

Martin Murth
Institute of Computer Languages, Vienna University of Technology
http://www.complang.tuwien.ac.at

e Outl

- High-level goals
- Requirements from use cases
- Triple Space architecture
  - Conceptual Triple Space architecture
  - Logical kernel architecture
  - Integration of kernel components with XVSM
  - Outlook on Web service integration and TS security
  - Physical kernel architecture
  - Towards a distributed architecture

High-level Goals

- **Persistent publication** of semantic data
- Retrieval by **semantic matching** in a **time-** and **space-independent** manner
- **Mediation** of data between heterogeneous services
- **Coordination** of concurrent access situations
- Appropriate **security and trust** mechanisms
- Use of **Web service protocol stack** and **Semantic Web** technologies
- **Industrial scenarios** with requirements of scalability, robustness, and flexibility

Requirements from Use Cases

- Size of the space
  - Authorities: 10^4
  - Registered users: 10^6
  - Data: 10^12 triples
  - …
- Reliable infrastructure
  - Mechanism to ensure data consistency during concurrent data access
  - Fault tolerance
  - ...
- Data interoperability
  - Mediation at schema level
  - Mediation at instance level

Requirements from Use Cases (2)

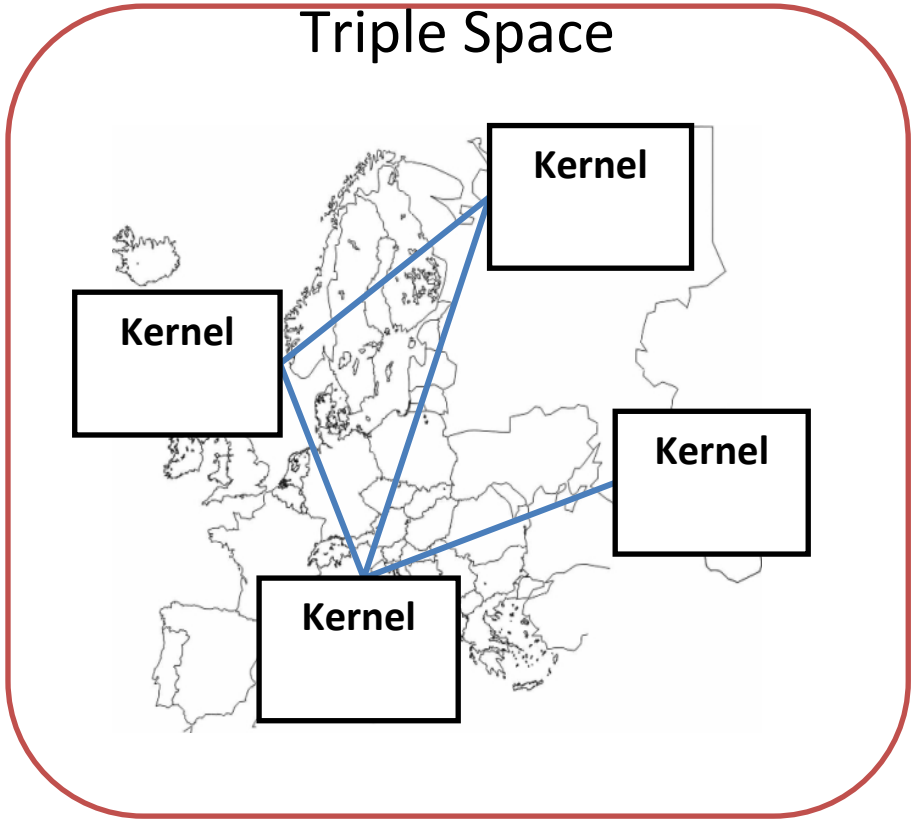
- Coordination and communication mechanisms
  - Synchronisation of concurrent access to data
  - Subscription to changes of the data
  - Compatibility with common interaction mechanisms (e.g., message passing, shared memory, RPC)
- Security
  - Users with multiple roles, access rights
  - Authority hierarchies, no central authority
  - Access logs
  - …

Consequently, we need to achieve:

- **Highly scalable distribution architecture**
- **High performance secure kernel architecture**

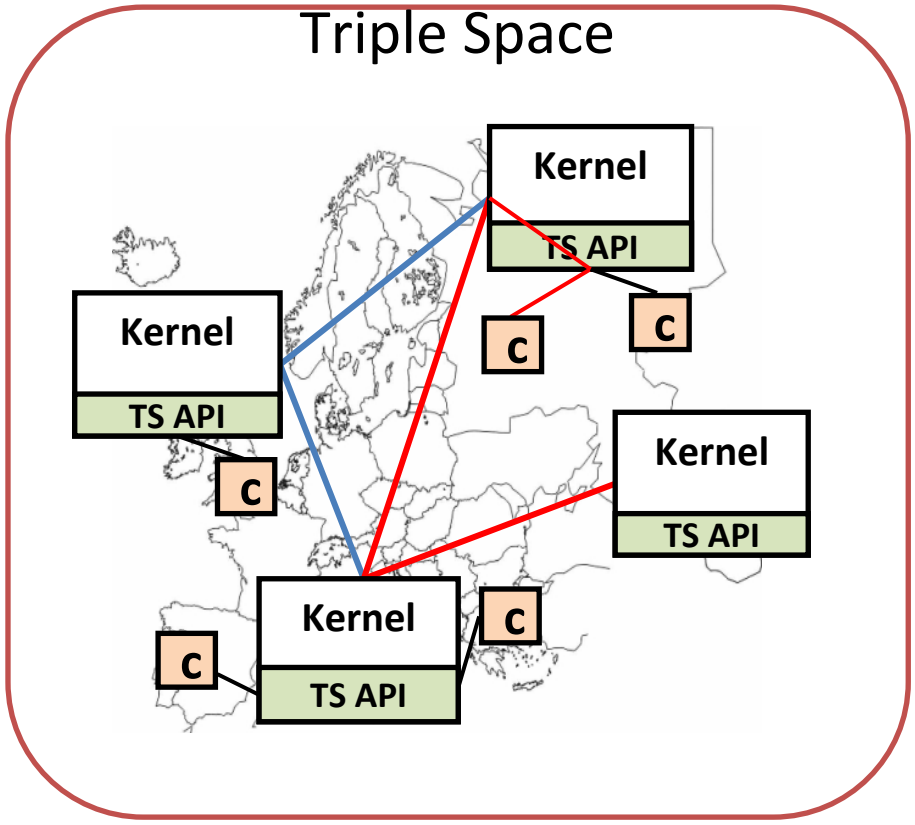
High Level View of Triple Space

- Distributed Infrastructure
  - Without a central, single point of control
  - Consisting of multiple kernels



High-level View of Triple Space

- Distributed infrastructure
  - Without a central, single point of control
  - Consisting of multiple kernels
- Kernel
  - Exposes the Triple Space API
  - Manages a subset of Triple Space data
  - Distributes requests
  - Is under a single authority
- All data is accessed transparently through an arbitrary TS API / kernel

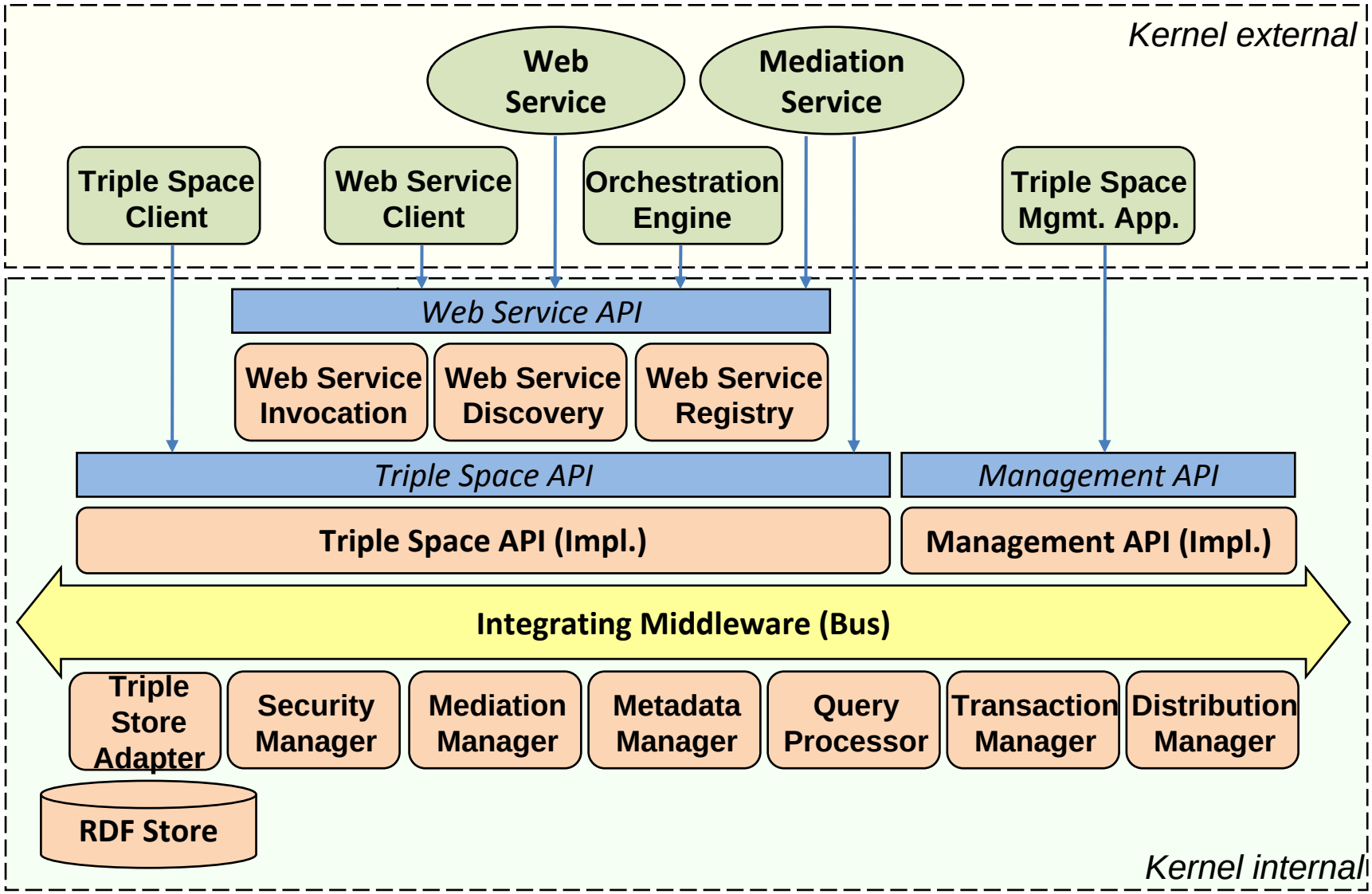


Sixth Framework Programme Priority 2
Information Society Technologies (IST)
Specific Targeted Research Project

Consequences …

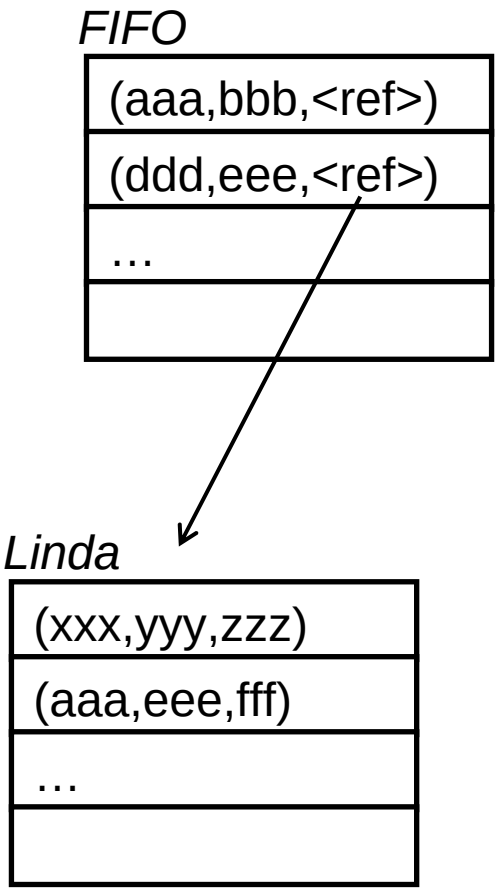
- The TS API embodies the paradigm of a global space.
  - Client connects to ANY kernel.
  - Client does not need to know other kernel addresses.
  - Kernel takes over lookup and routing of requests to other kernels.
- The number of kernels and the presence of kernels are undefined at a specific point in time.
  - There are no guarantees regarding kernel availability.
  - Kernels can leave or join at any point in time, kernels can crash, etc.
  - In order to guarantee availability of data, data replication techniques will have to be employed.

Logical Kernel Architecture

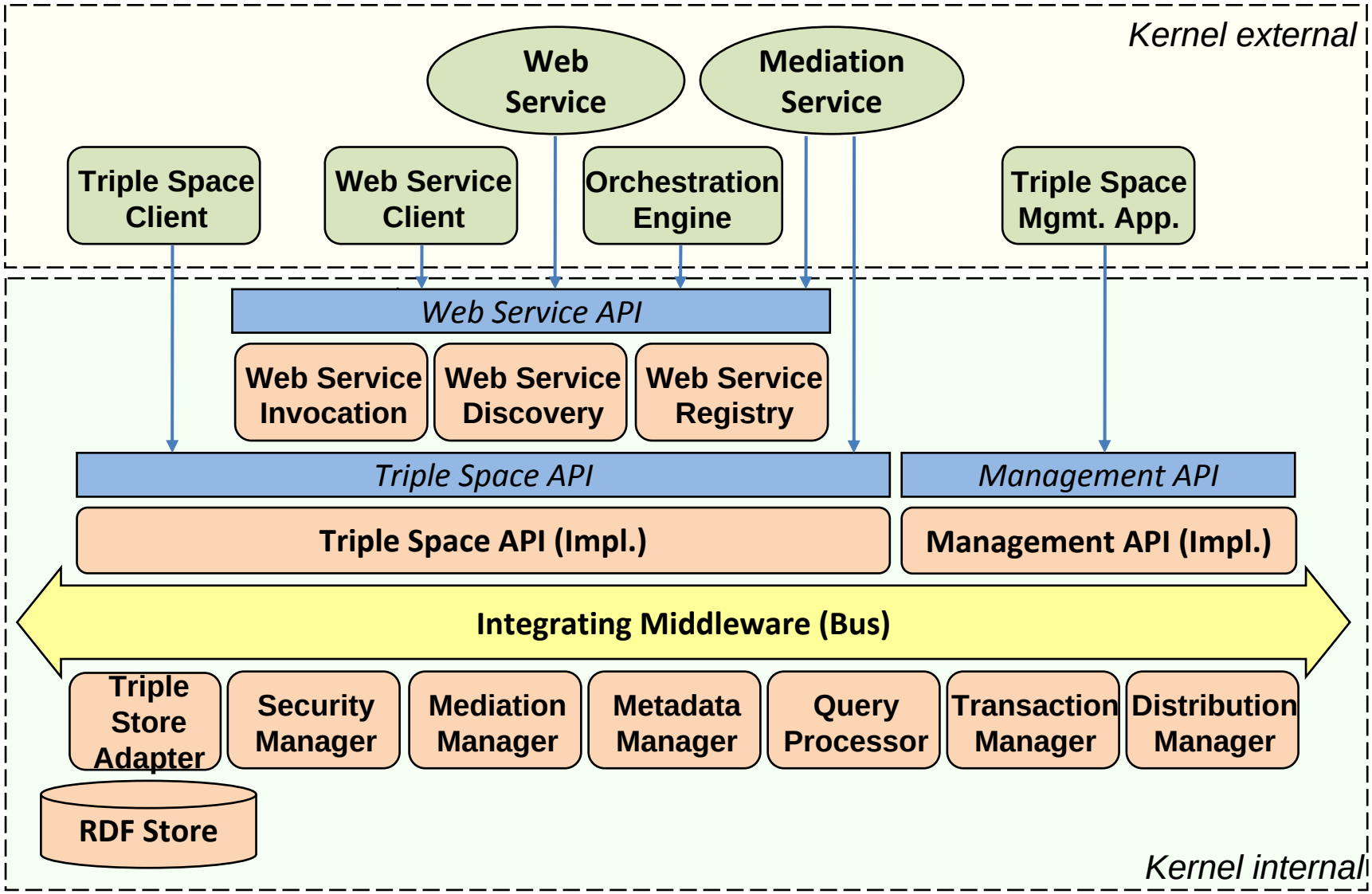


- We implement the space with a space
- **XVSM**

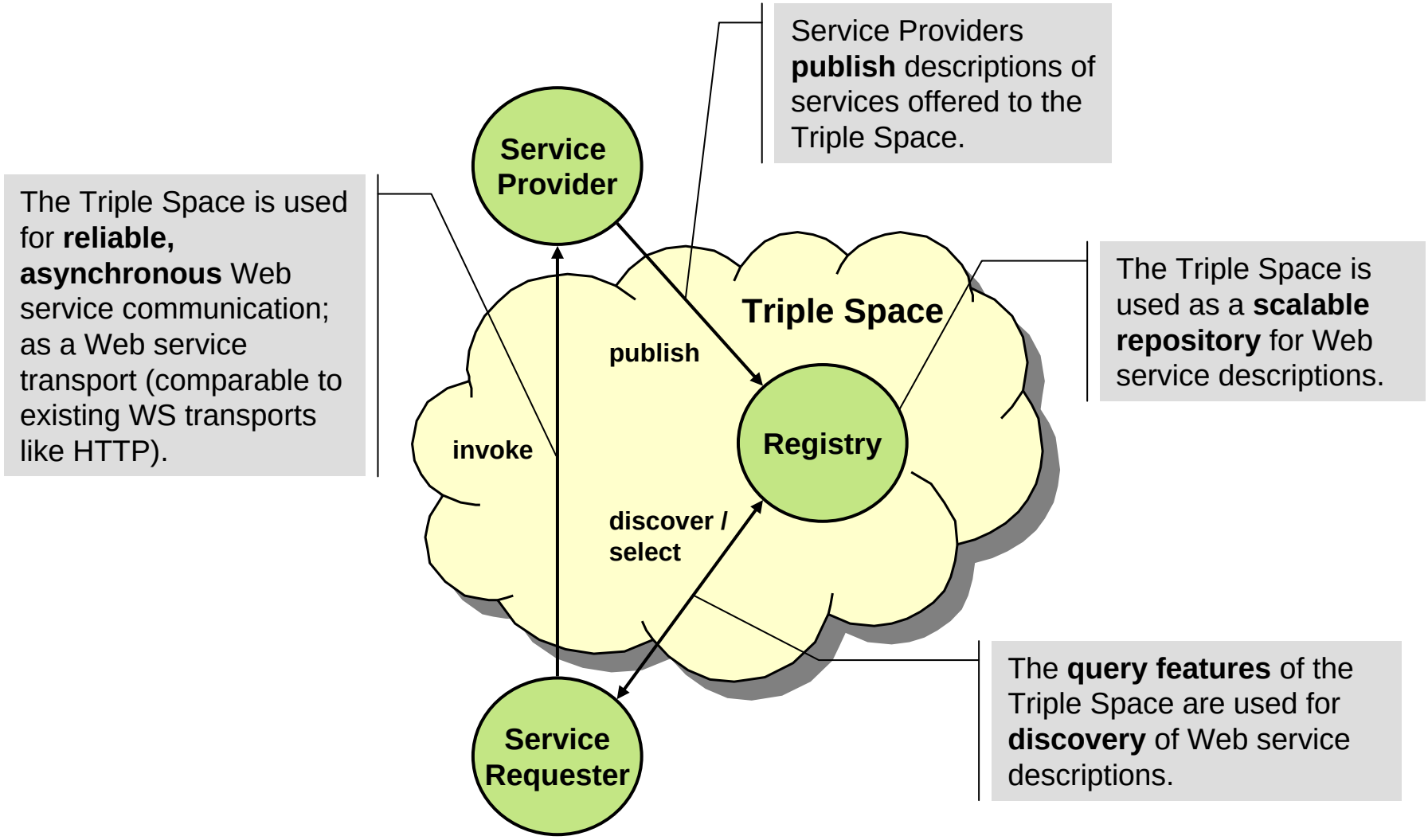
  e**X**tensible **V**irtual **S**hared **M**emory
- Structured sets of containers
- API
  - Container: *create, publish, destroy*, etc.
  - Data: *read, take, write, shift, notify*, etc.
- Bounded buffer model
- Support for multiple coordination types (Linda, FIFO, Map, Stack, …)
- Access through
  - XML based protocol
  - Embedded process



Logical Kernel Architecture

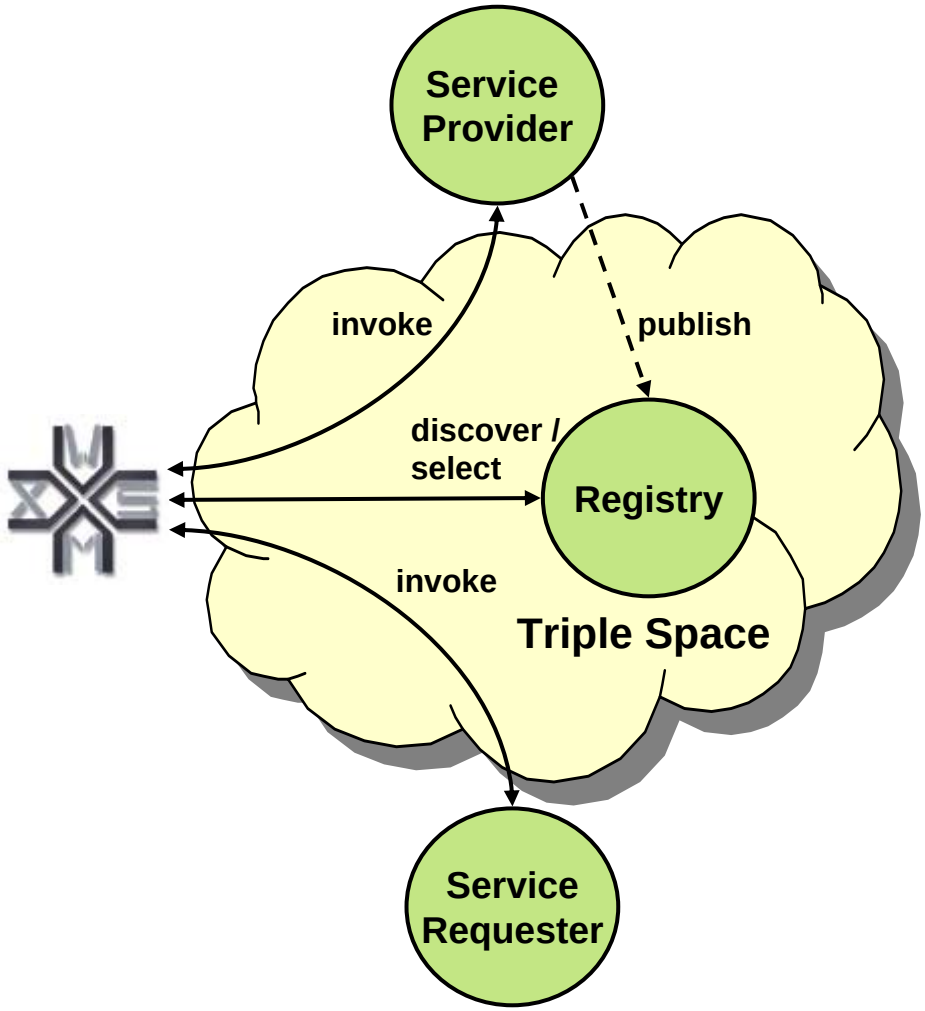


Triple Space and Web Services

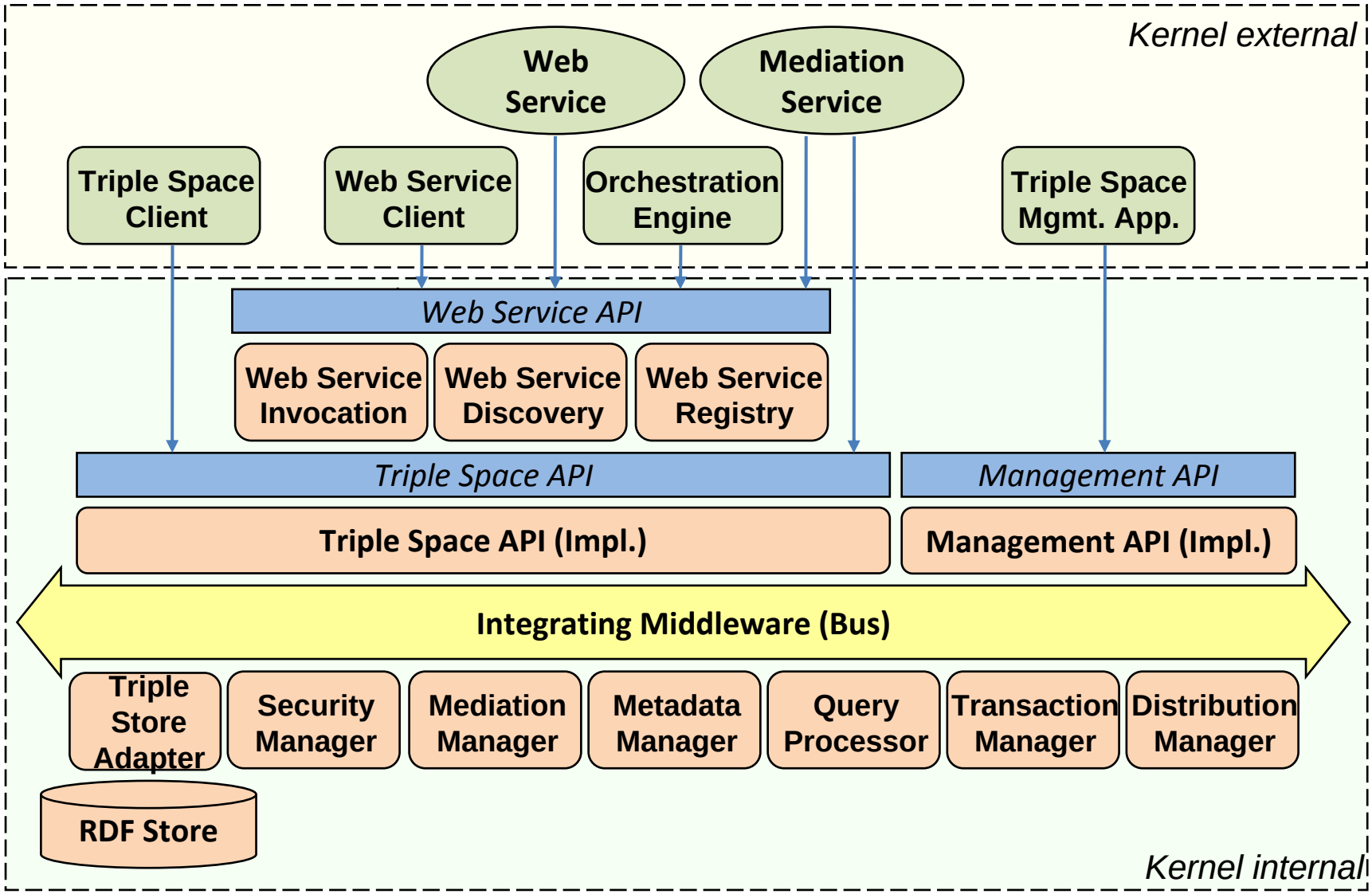


Integration with Semantic Web services

- Semantic Web services (SWS) enhance Web services with semantic descriptions, enabling e.g.
  - Semantic service discovery and selection
  - Mediation
- Web Service Execution Environment (WSMX)
  - Middleware for SWS interaction
- Aspects of Integration of WSMX and TripCom
  - Internal and external WSMX communication
  - Resource management

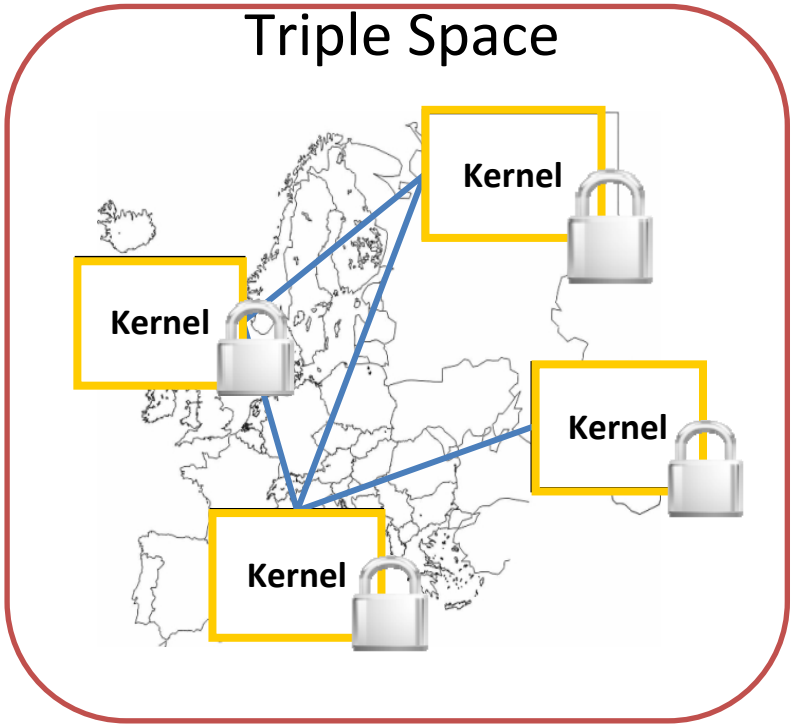


Logical Kernel Architecture

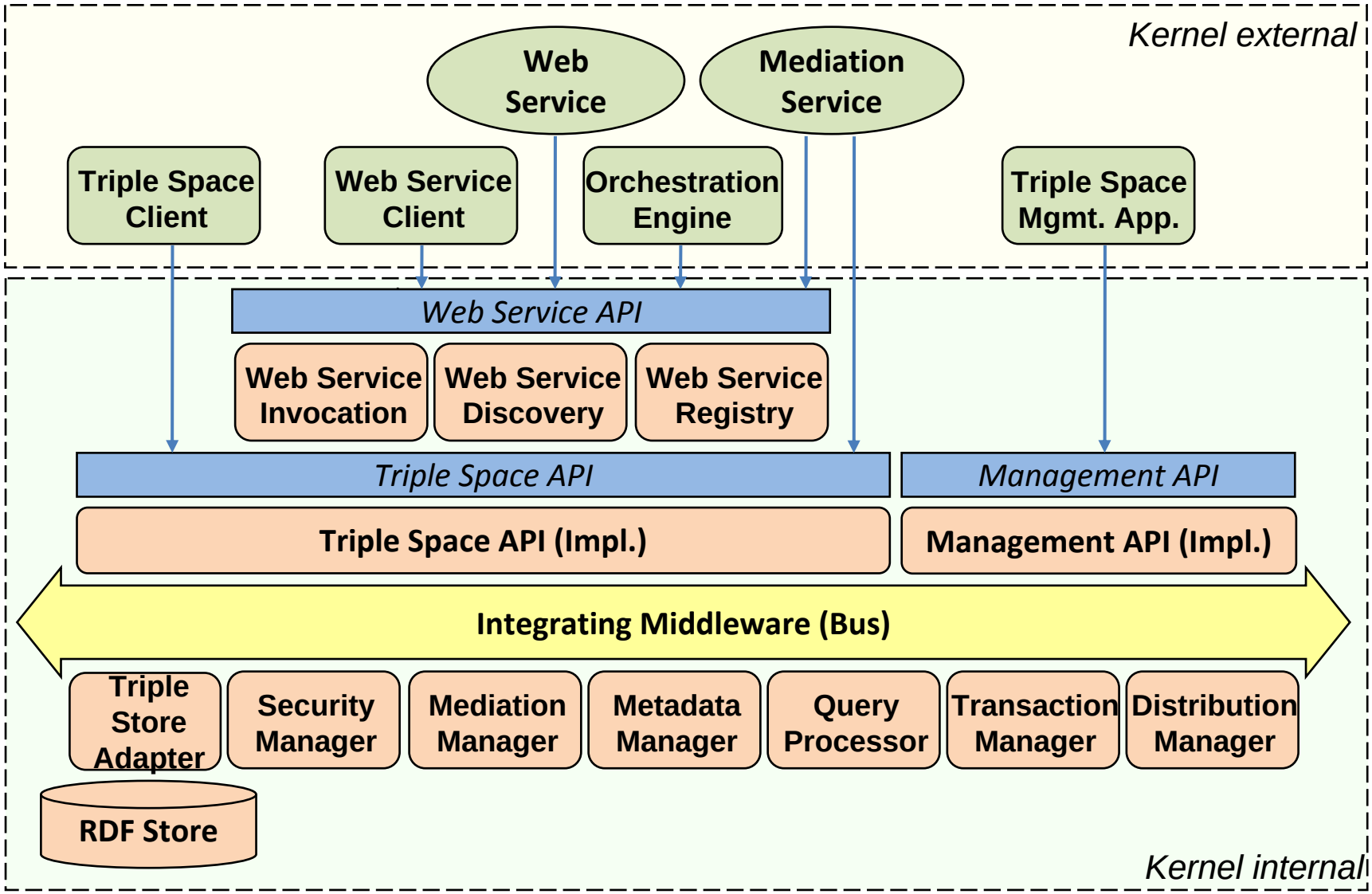


Triple Space Security Approach

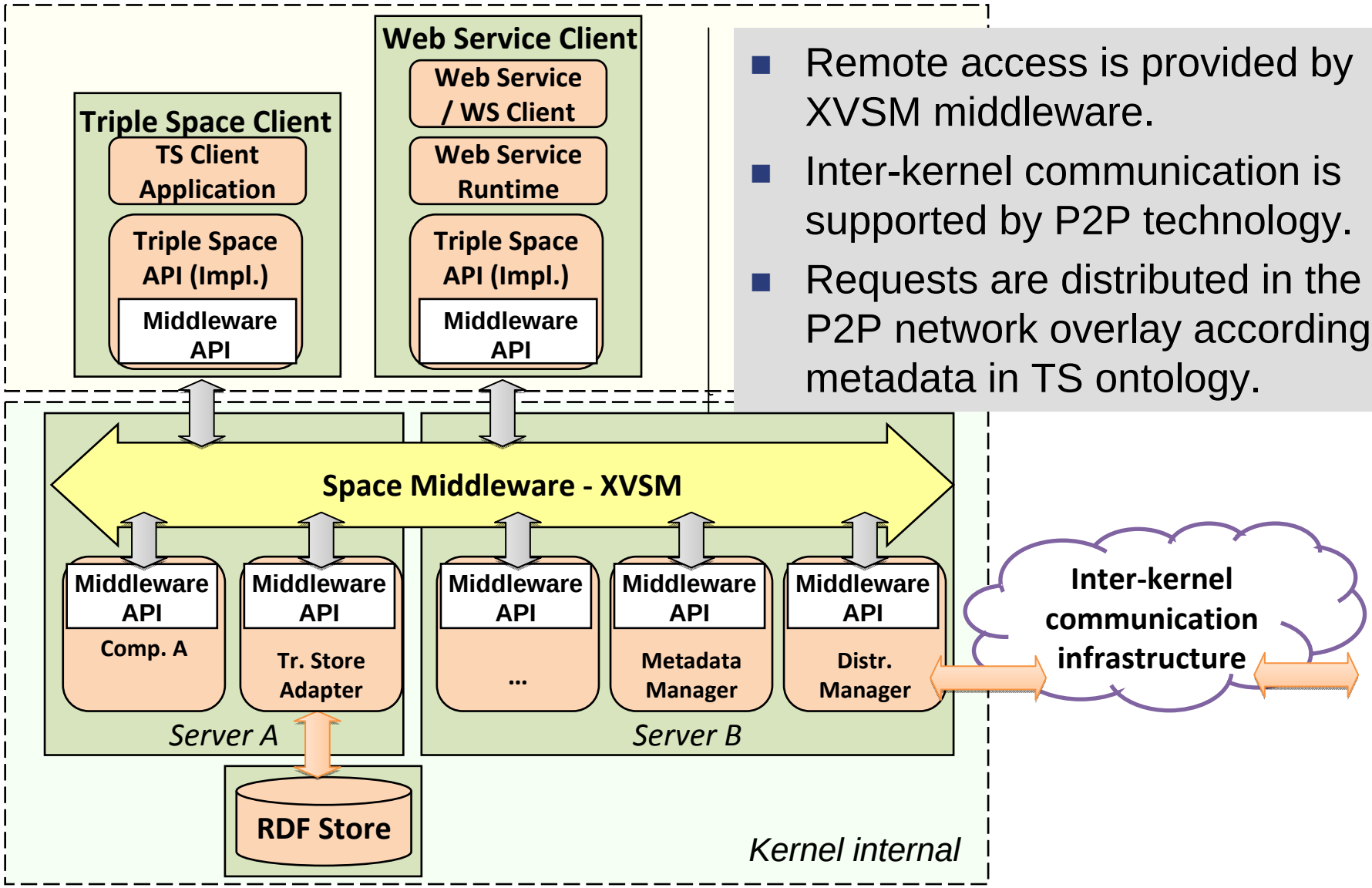
- Kernel boundaries are **security perimeters**
  - Security must be enforced when information crosses kernel boundaries
- Integrating middleware is the **security enforcement point**
- Kernel-internal components are trusted
- Security Manager
  - Cares for authentication and authorisation
  - Embodies the **security decision point**



Logical Kernel Architecture



Physical Kernel Architecture



- Remote access is provided by XVSM middleware.
- Inter-kernel communication is supported by P2P technology.
- Requests are distributed in the P2P network overlay according to metadata in TS ontology.

Towards a Distributed Architecture

- Triple Space requires a means to find and manage distributed kernels, e.g.
  - Centralised directory service
  - Hierarchical tree of directory servers (e.g. DNS, X.500)
  - Peer-to-peer network overlay (e.g. JXTA)
- We favour P2P, since:
  - It does not require a central coordination and controlling authority.
  - It does not prescribe a network structure but allows for dynamic joining and leaving of nodes.
  - P2P systems already have proven scalability and resiliency in many real-life applications.

Next Steps and Challenges

- Distribution mechanisms
  - Refine the requirements on scalability
  - Find a suitable distribution technology
  - Analyse mechanisms for reasoning on distributed data
- Kernel discovery
  - Analyse which metadata is required
- Component interaction
  - Define component interfaces
  - Define data flows between the components

e Outl

- High-level goals
- Requirements from use cases
- Triple Space architecture
  - Conceptual Triple Space architecture
  - Logical kernel architecture
  - Integration of kernel components with XVSM
  - Outlook on Web service integration and TS security
  - Physical kernel architecture
  - Towards a distributed architecture