JS7 JobScheduler Architecture

System Architecture:

Systems, Components, Platforms

Information for Interested Parties

Software- und Organisations-Service

JS7 JobScheduler

- System Architecture
  - Basic System Architecture
  - Components and Connections
  - Network Connections
  - Supported Platforms

- Setup Scenarios
  - Standalone Server
  - Controller High Availability
  - Controller and JOC Cockpit High Availability
  - Multi-Client Capability
  - Agent High Availability

System Architecture

Basic System Architecture

JOC Cockpit

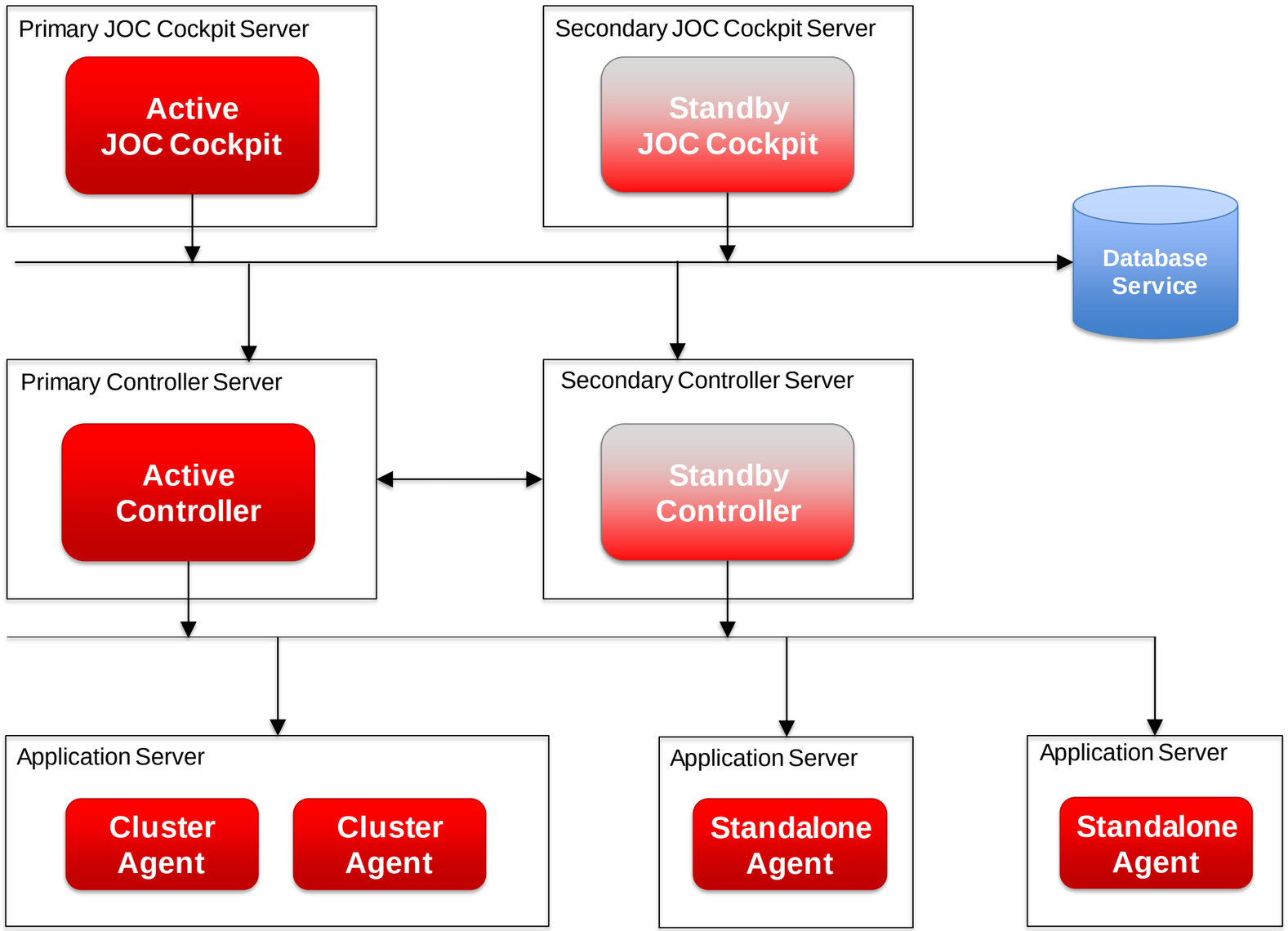
- JOC Cockpit is operated as a passive cluster and serves the User Interface and REST Web Service
- JOC Cockpit Services make use of a database for restart capabilities

Controller / Agents

- The Controller is operated as a passive cluster to orchestrate Agents
- Agents receive workflow configurations from a Controller, execute jobs autonomously and report back execution results
- Agents are operated as a cluster or standalone

Connections

- Communication between components within the scope of the indicated network connections



Components

Components and Connections

JOC Cockpit / Web Service

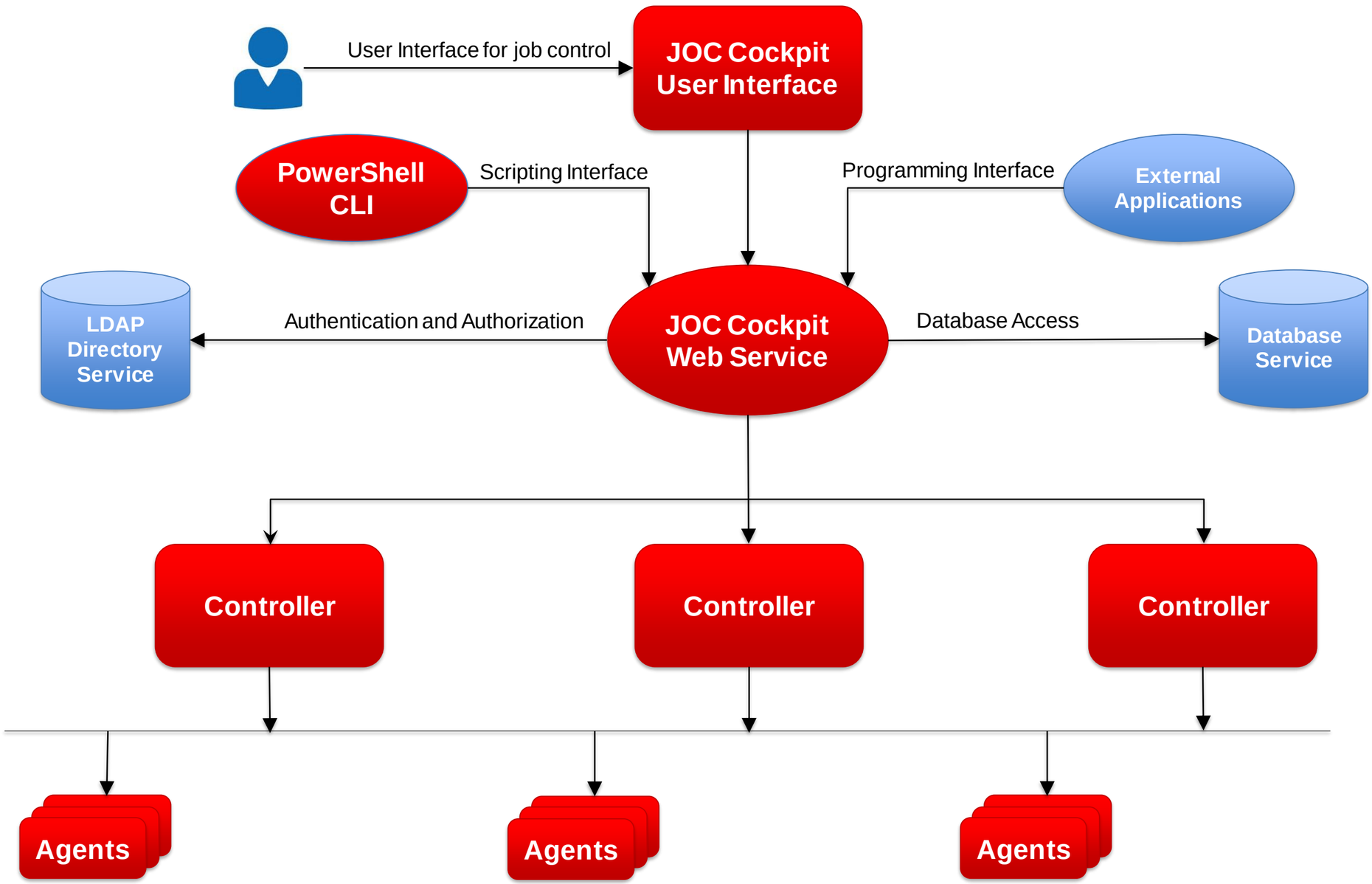
- The user interface offers job management and control
- Users access the JOC Cockpit from their browsers
- User access is subject to authentication and authorization – optionally with an LDAP Directory Service

Interfaces

- The PowerShell Command Line Interface and External Applications use the same Web Service for access to any Controller
- Authorization is available for individual permissions on operations grouped by roles

Controller / Agent

- The Controller holds the workflow configuration and orchestrates Agents
- Agents are deployed on top of existing servers running the programs and scripts scheduled for execution



Network Connections

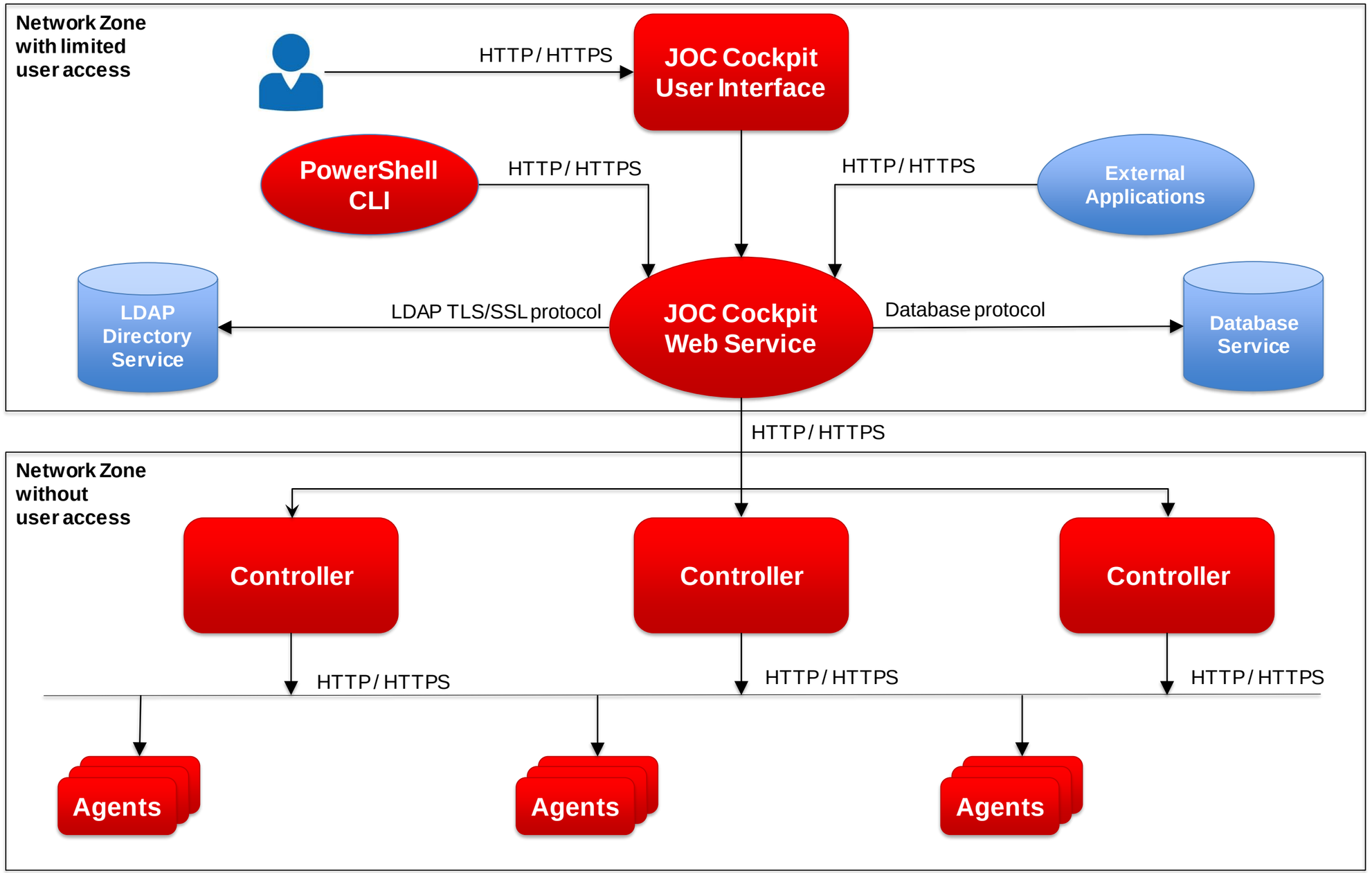
Secure Network Connections

Network Zone with restricted user access

- Users have limited access that requires authentication
- Connections to JOC Cockpit are authenticated by the Web Service that can be configured to use LDAP over TLS or SSL
- Use of HTTPS for network connections with client and server authentication

Network Zone without user access

- Controller and Agent instances are operated in a network zone without direct user access
- Controller instances are accessed exclusively by the JOC Cockpit Web Service
- Agent instances are accessed exclusively by Controller instances



Platforms

Supported Platforms

Cockpit / Web Service

- The JOC Cockpit and REST Web Service are available for Windows and Linux

Controller / Agent

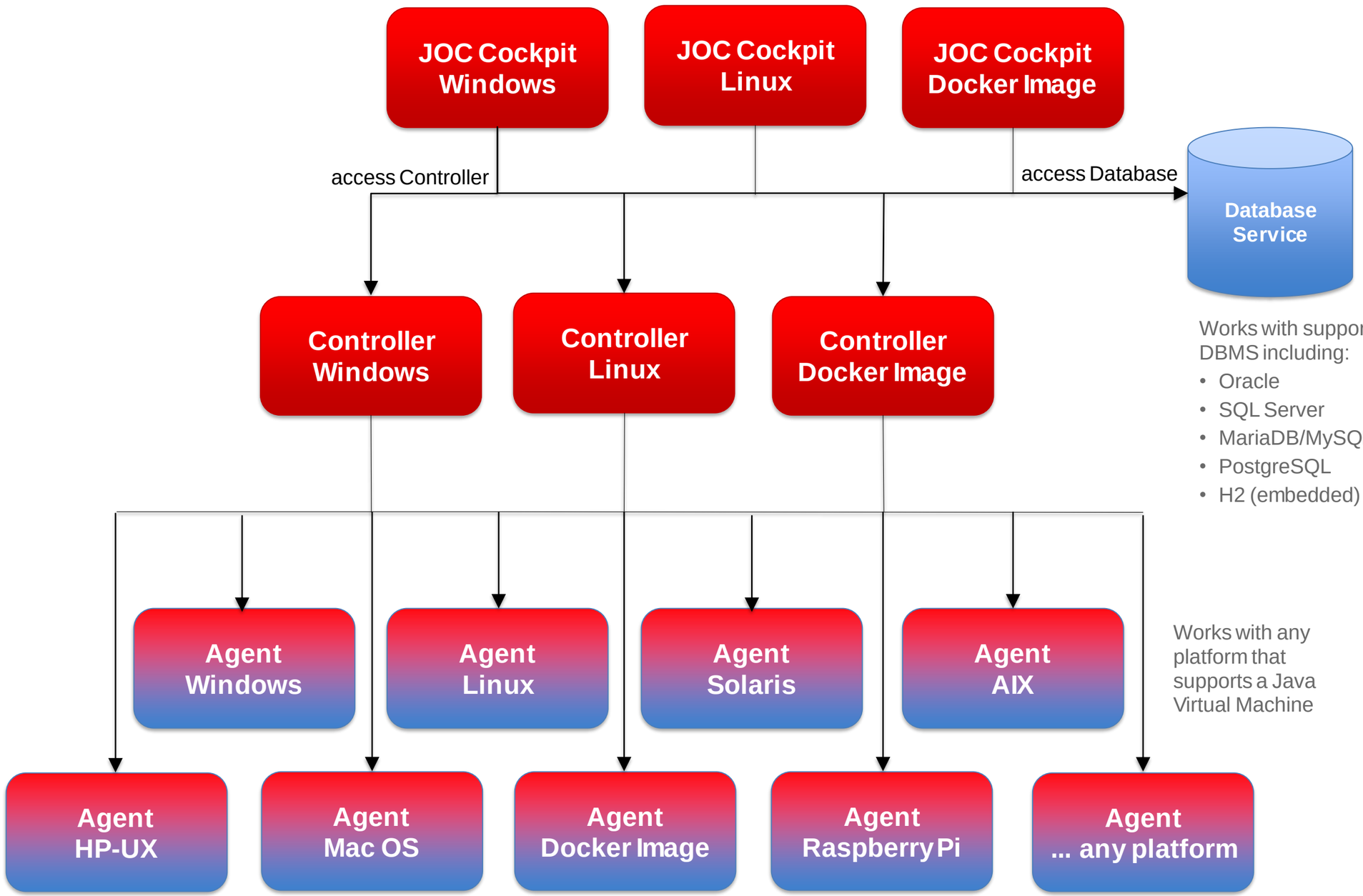
- The Controller is available for Windows and Linux
- Agents are available for any platform that supports a Java Virtual Machine

Database Service

- The JOC Cockpit Web Service uses a database from any platform

Workflows

- Workflows can be executed with any Agent



Works with supported DBMS including:

- Oracle
- SQL Server
- MariaDB/MySQL
- PostgreSQL
- H2 (embedded)

Works with any platform that supports a Java Virtual Machine

Software- und Organisations-Service

JS7 JobScheduler

- System Architecture
  - Basic System Architecture
  - Components and Connections
  - Network Connections
  - Supported Platforms

- Setup Scenarios
  - Standalone Server
  - Controller High Availability
  - Controller and JOC Cockpit High Availability
  - Multi-Client Capability
  - Agent High Availability

Setup Scenario: Standalone Server

Scenario: Standalone Server for User Interface, Controller and Database Service

JOC Cockpit / Web Service

- JOC Cockpit is the user interface for workflow management and control
- Users access the JOC Cockpit from their browser

Controller

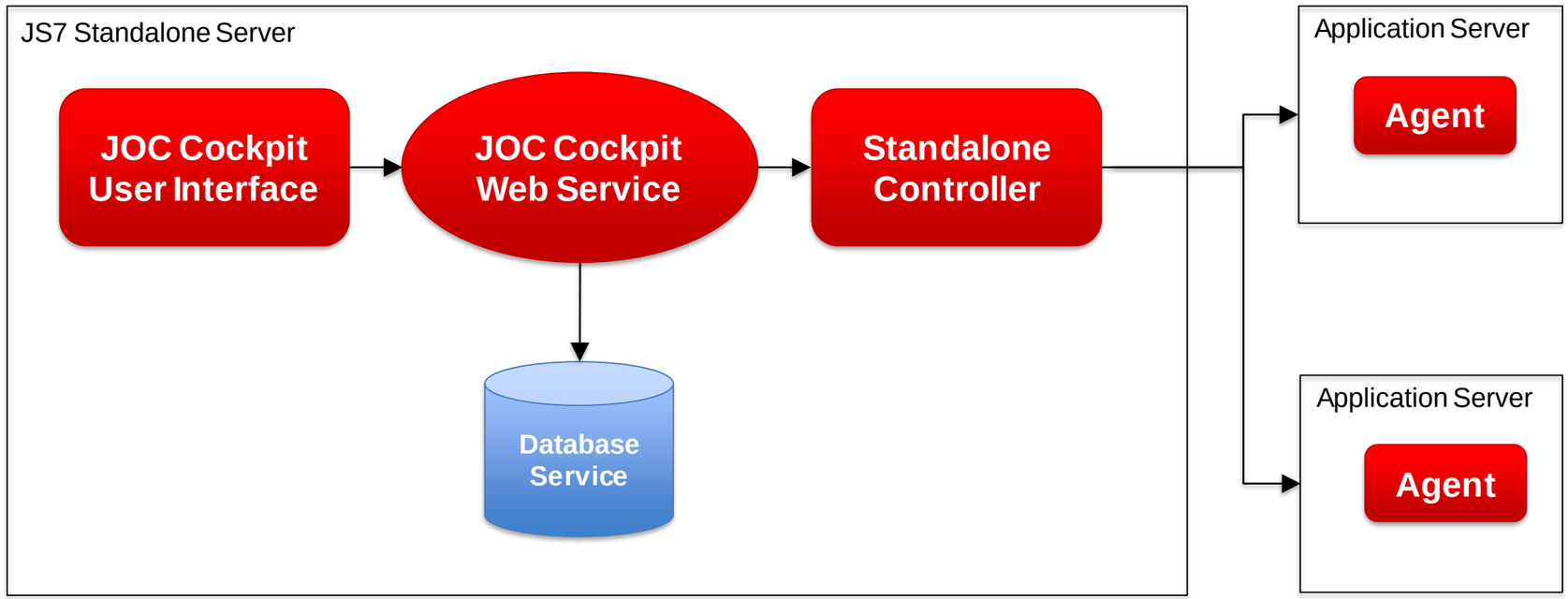
- The Controller orchestrates Agents for execution of workflows

Agent

- Agents are deployed on top of existing servers running the programs and scripts scheduled for execution

Database Service

- The database stores the inventory and history information for workflows



Setup Scenario: Controller High Availability

Scenario: Standalone Interface Server, Controller Cluster, Database Server

JOC Cockpit / Web Service

- JOC Cockpit is the user interface for workflow management and control
- Users access the Controller cluster using a Web Service

Controller Cluster

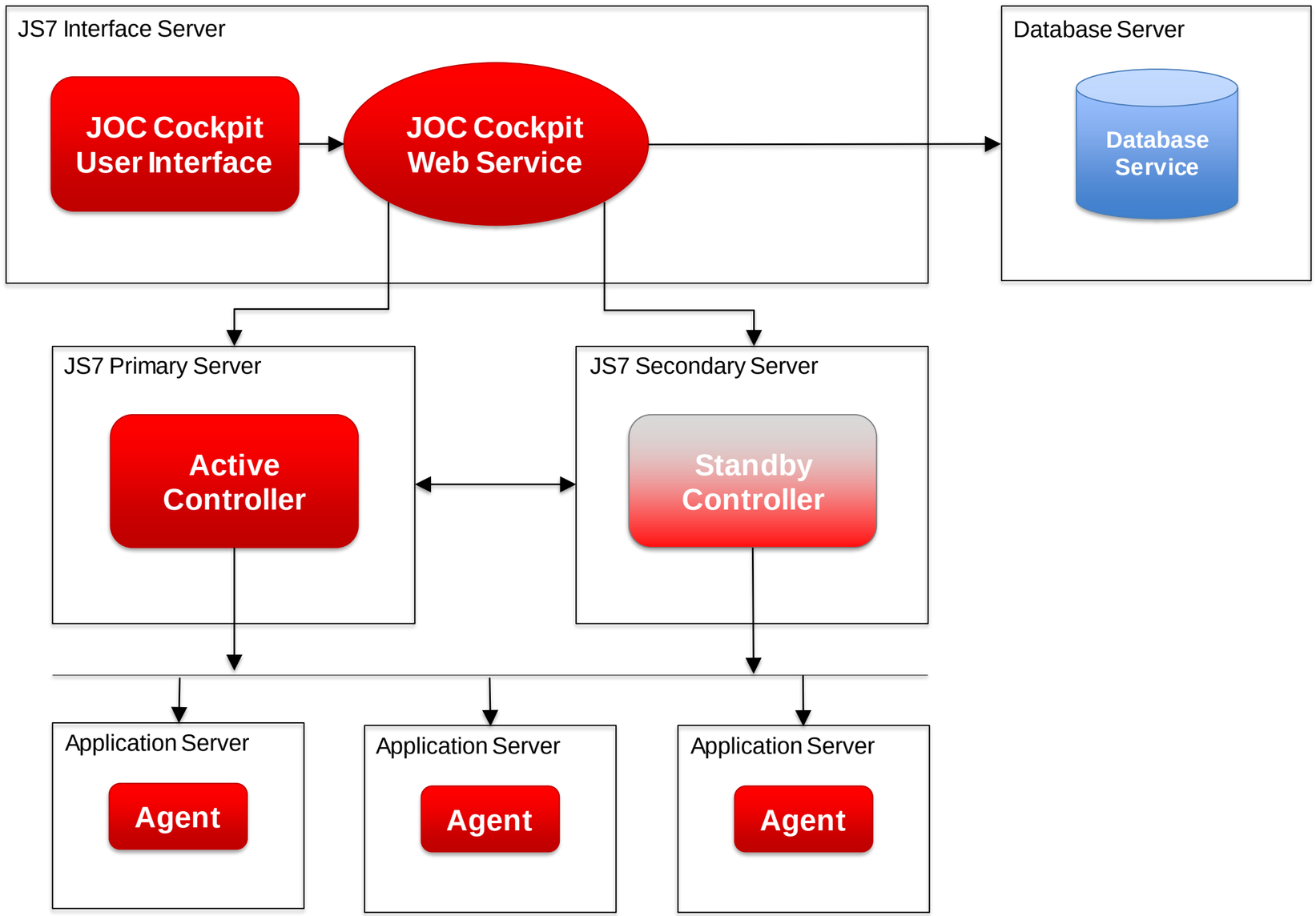
- Active and Standby Controllers act as a cluster to synchronize their journal for automated fail-over
- Active and Standby Controller are accessed by the JOC Cockpit Web Service

Agent

- Agents are deployed on top of existing servers and can be accessed by the Active and Standby Controllers

Database Service

- JOC Cockpit makes use of a database for persistence and restart capabilities



Setup Scenario: Controller and JOC Cockpit High Availability

Scenario: JOC Cockpit Cluster, Controller Cluster, Database Server

JOC Cockpit / Web Service

- JOC Cockpit is the user interface for workflow management and control
- A number of JOC Cockpit instances can be operated as a passive cluster
- Each JOC Cockpit instance has access to the Active and Standby Controller

Controller Cluster

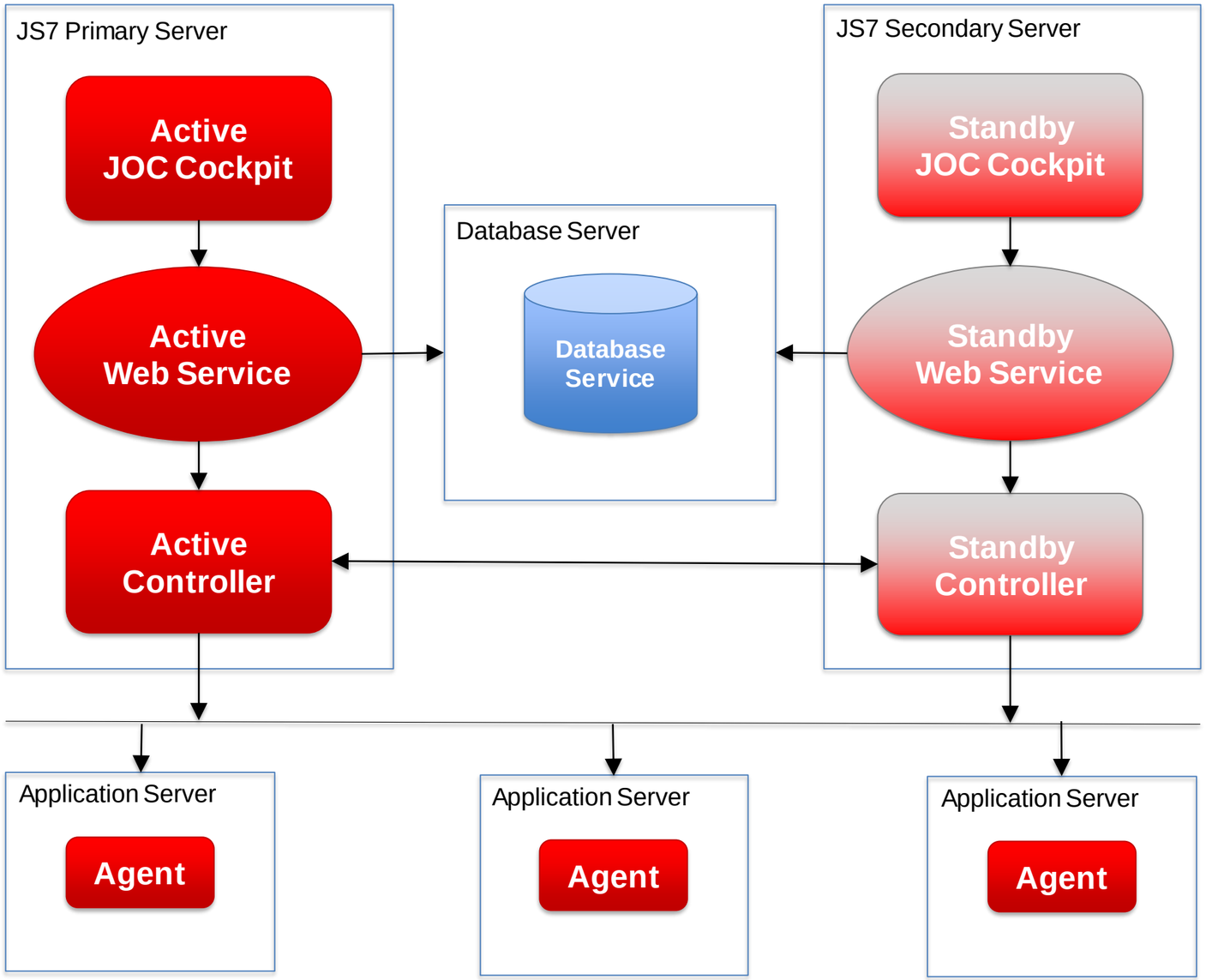
- Active and Standby Controllers implement a cluster for automated fail-over

Agent

- Agents are deployed on top of existing servers and can be accessed by the Active and Standby Controller

Database Service

- JOC Cockpit makes use of a database for persistence and restart capabilities



Setup Scenario: Multi-Client Capability

Scenario: Interface Server, Multi-Controller Servers, Database Server

JOC Cockpit / Web Service

- JOC Cockpit is the user interface for workflow management and control
- Users access the Controller Cluster using a Web Service

Controller

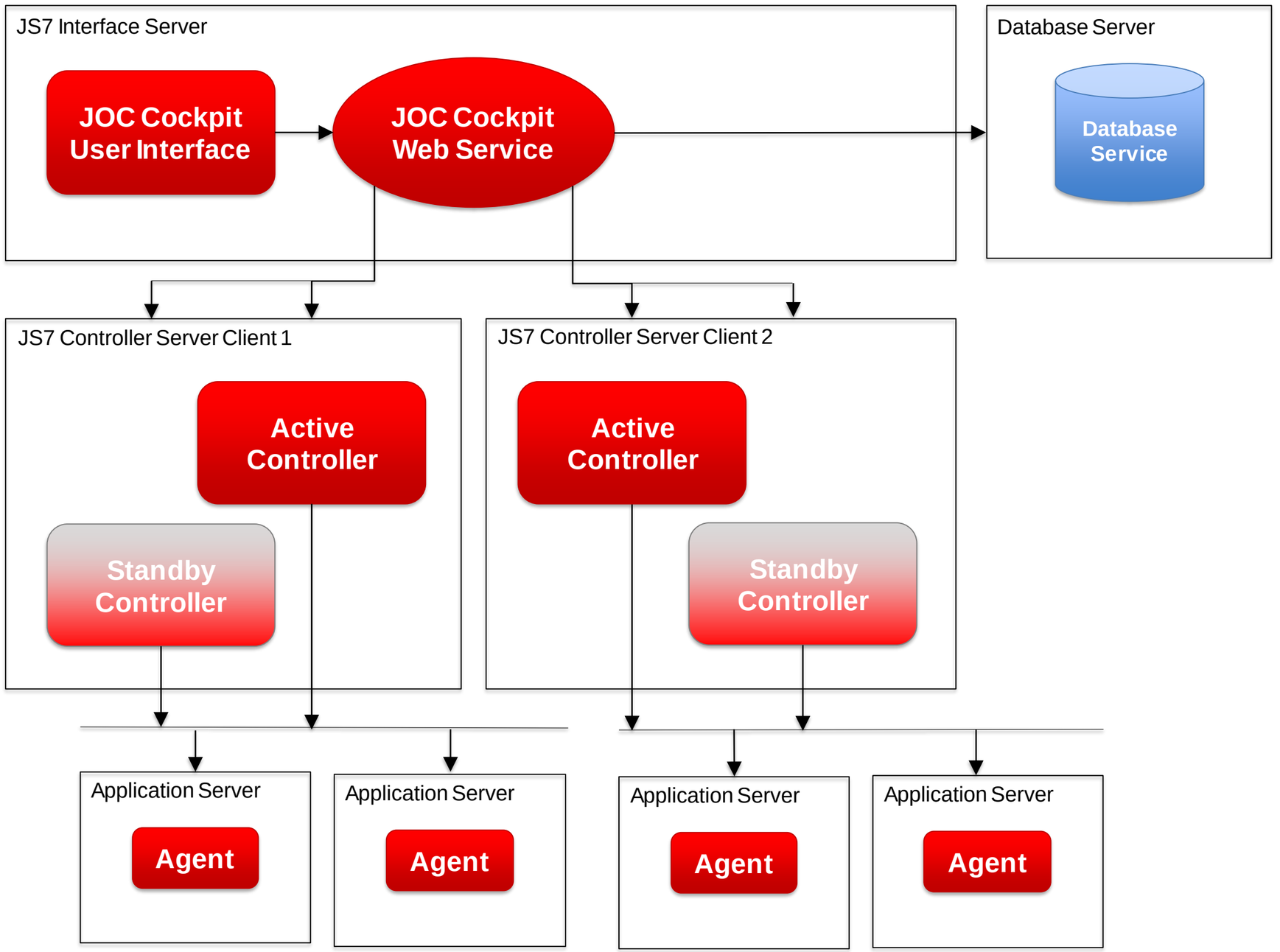
- Controller instances are operated and assigned per client, each Controller can be operated as a cluster
- Controller instances are accessed by the JOC Cockpit Web Service

Agent

- Agents are deployed on top of existing servers and are accessed by a Controller
- Agents are dedicated for use by specifc Controller cluster instances acting for a client

Database Service

- JOC Cockpit makes use of a database for persistence and restart capabilities



Setup Scenario: Agent High Availability

Scenario: Interface Server, Database Server, Controller Cluster, Agent Cluster, Standalone Agents

Controller

- The Controller connects to an Agent Cluster and to Standalone Agents

Agents

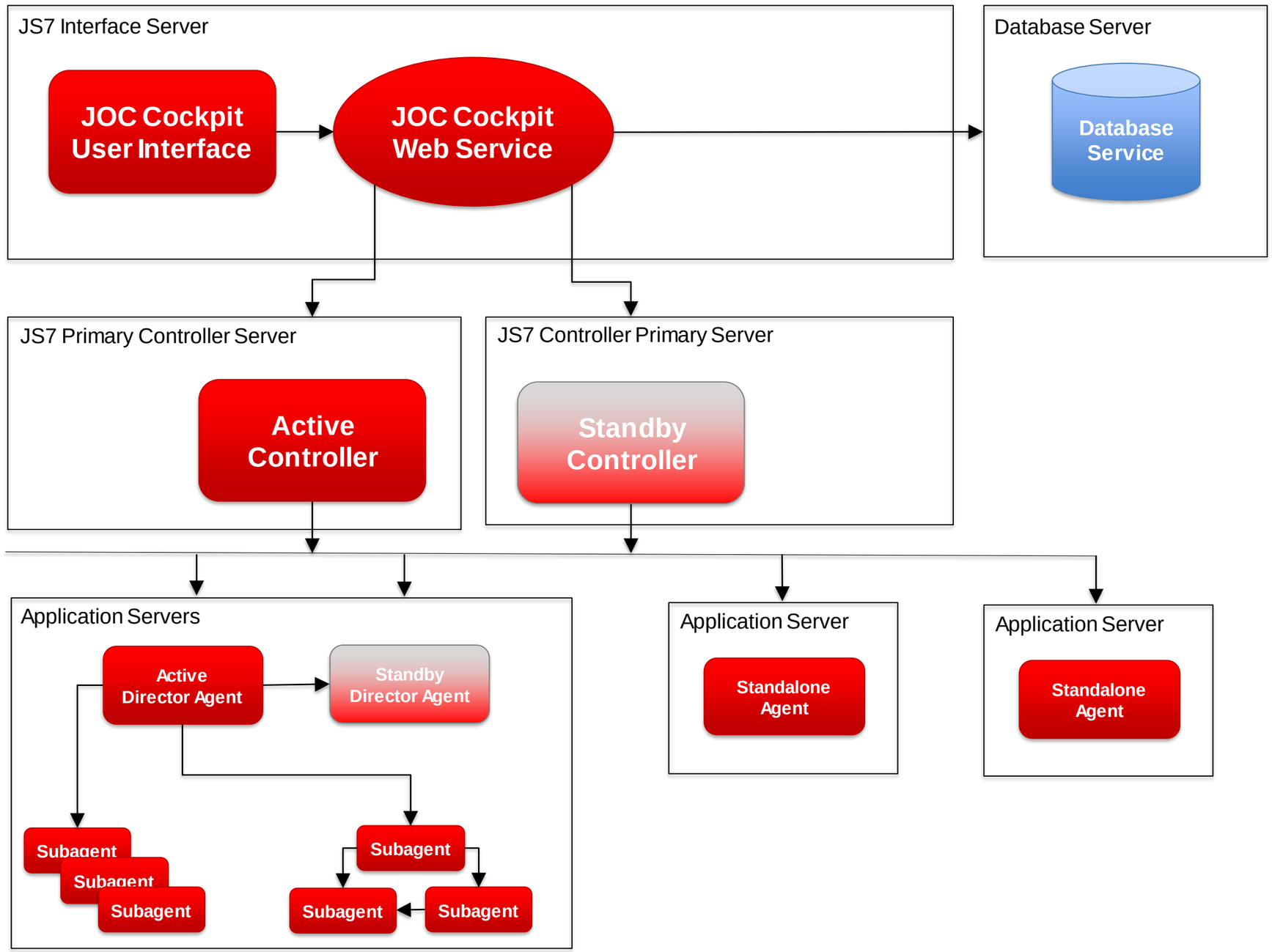
- Agents are deployed on top of existing servers and are accessed by a Controller
- Agents are dedicated for use by a specifc Controller

Agent Cluster

- A Director Agent holds the active role and orchestrates Subagents for job execution
- Fixed-priority mode includes to execute jobs with the first Subagent, only if unavailale the next Subagent is used
- Round-robin mode includes to execute each next job on the next Subagent

Standalone Agents

- Any number of Standalone Agents are operated on individual application servers



Questions?

Comments?

Feedback?

Software- und
Organisations-
Service GmbH

Giesebrechtstr. 15
D-10629 Berlin

info@sos-berlin.com
https://www.sos-berlin.com