



JS7 JobScheduler Architecture

System Architecture:

Systems, Products, Platforms

**Information for
Interested Parties**



■ System Architecture

- System Architecture
- Product Architecture
- Secure Network Connections
- Supported Platforms

■ Cloud Setup

- JOC Cockpit and Controller High Availability
- Agent High Availability
- Hybrid Use of Agents

■ On Premises Setup

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

JOC Cockpit

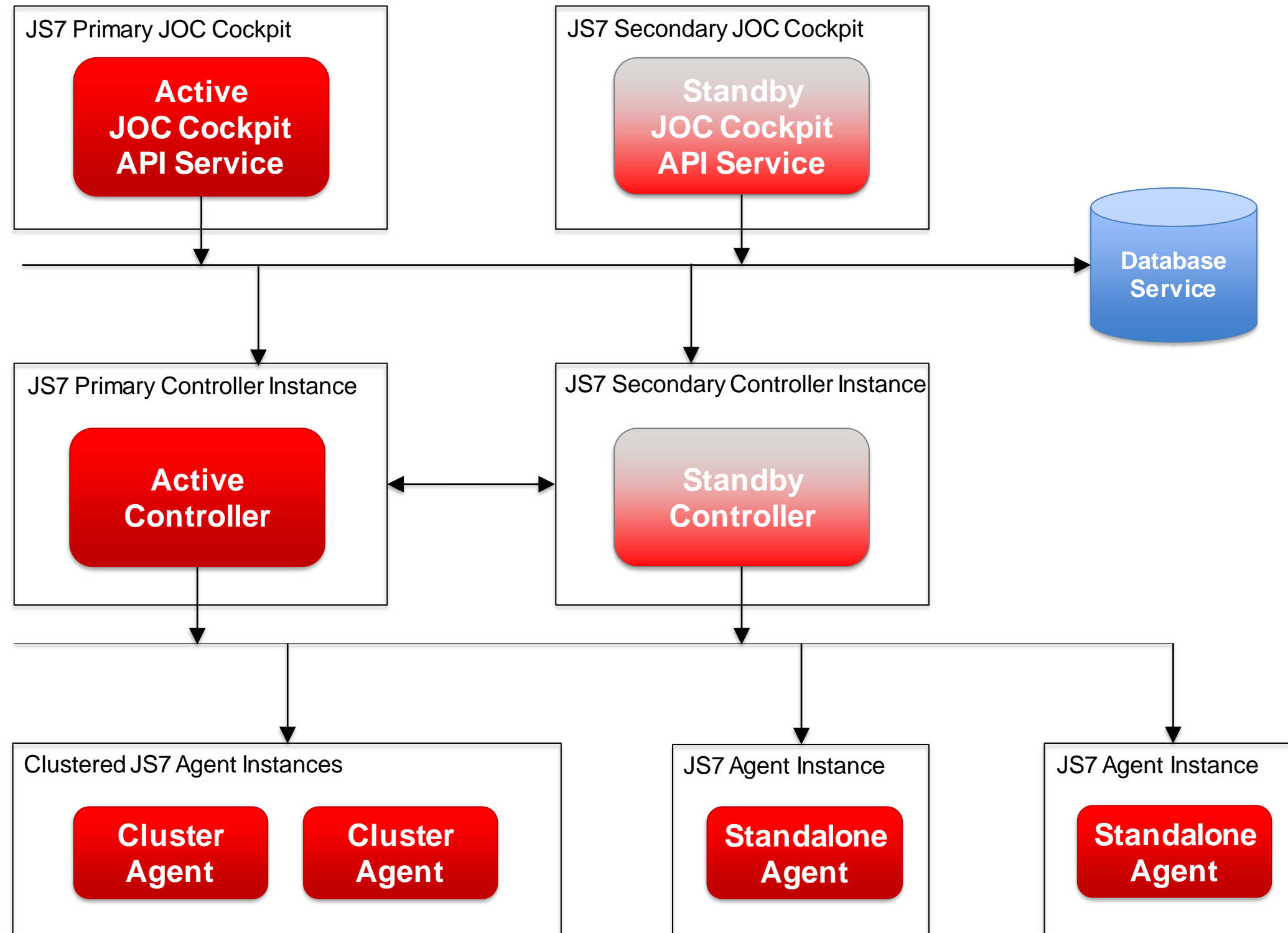
- JOC Cockpit is operated as a passive cluster or standalone and serves the User Interface and REST API Service
- Makes use of a database for persistence and for restart capabilities

Controller / Agents

- A Controller operated as a passive cluster or standalone orchestrates Agents
- Agents receive workflow configurations from a Controller, start workflows autonomously and report back execution results
- Agents are operated as a cluster or standalone

Connections

- Communication between products within the indicated direction of network connections



JOC Cockpit / API Service

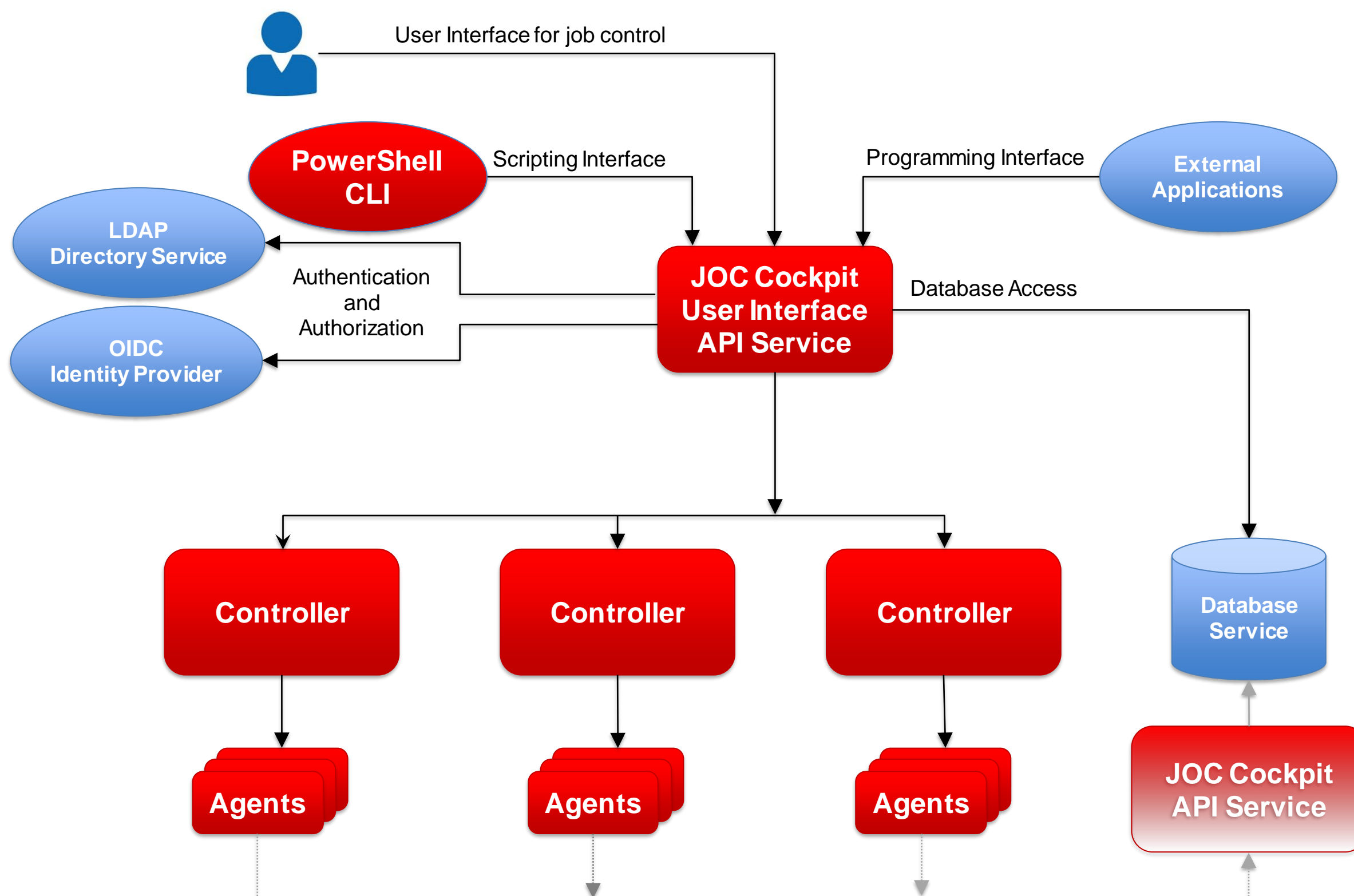
- The User Interface offers job management and control
- Users access JOC Cockpit from their browsers
- Access is subject to authentication and authorization optionally with LDAP, OIDC and other Identity Providers

Interfaces

- The PowerShell Command Line Interface and External Applications use the REST API Service for access to JOC Cockpit and Controller
- Authorization is available by roles/permissions (RBAC)

Controller / Agent

- The Controller holds the workflow configuration and orchestrates Agents
- Agents are deployed on top of any platform running the programs, scripts, services scheduled for execution



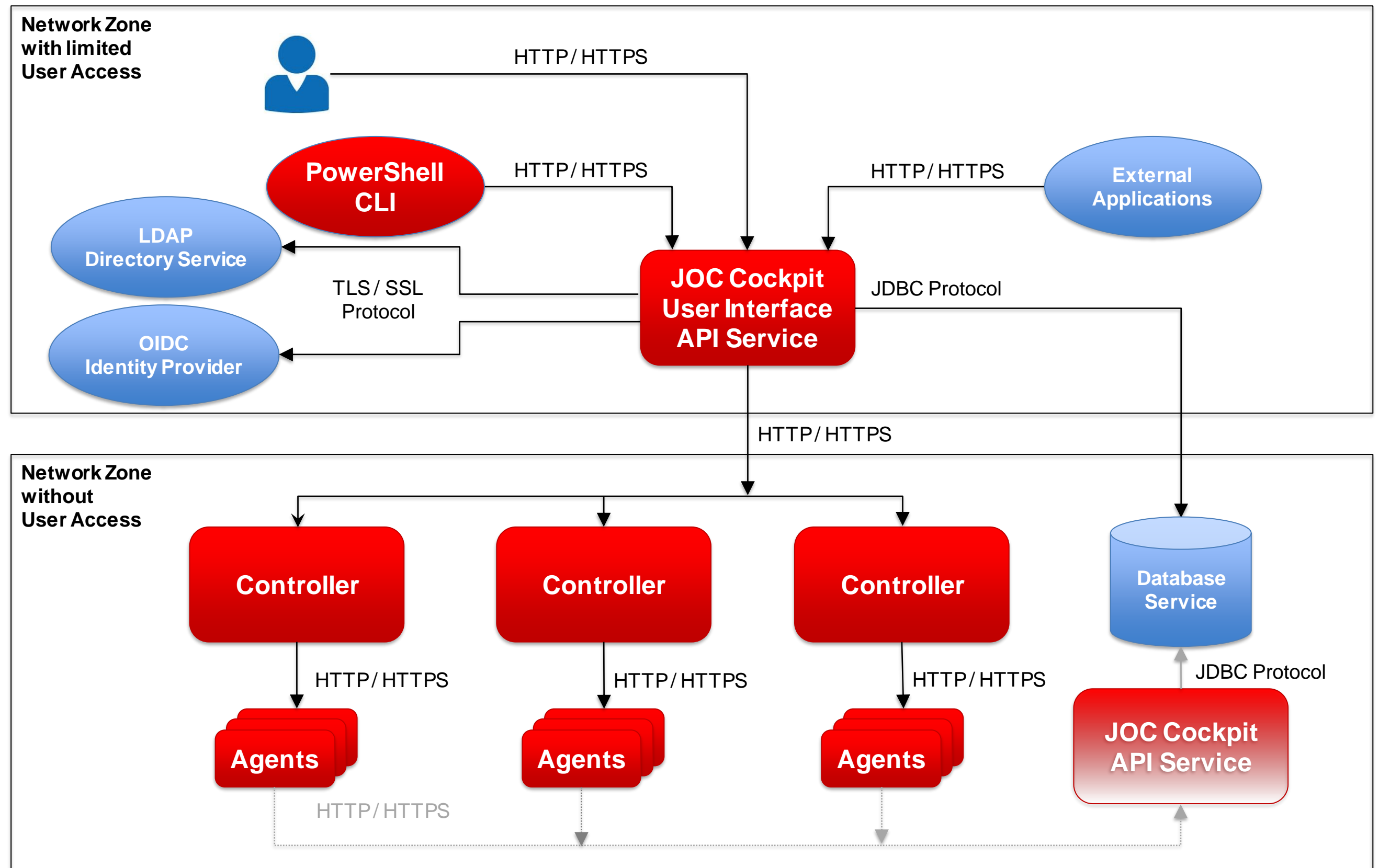
Secure Network Connections

Network Zone with restricted User Access

- Use of HTTPS for any connection to JOC Cockpit
- Access to JOC Cockpit requires authentication
- Access to JOC Cockpit is authenticated by the API Service using TLS/SSL

Network Zone without User Access

- Controller and Agent instances can be operated in a network zone without user access
- Controller instances are accessed exclusively by the JOC Cockpit API Service
- Agent instances are accessed exclusively by Controller instances
- Use of HTTPS for connections with client and server authentication certificates (mutual TLS authentication)



Supported Platforms

JOC Cockpit / API Service

- The JOC Cockpit and API Service are available for Container platforms, Linux and Windows

Controller / Agent

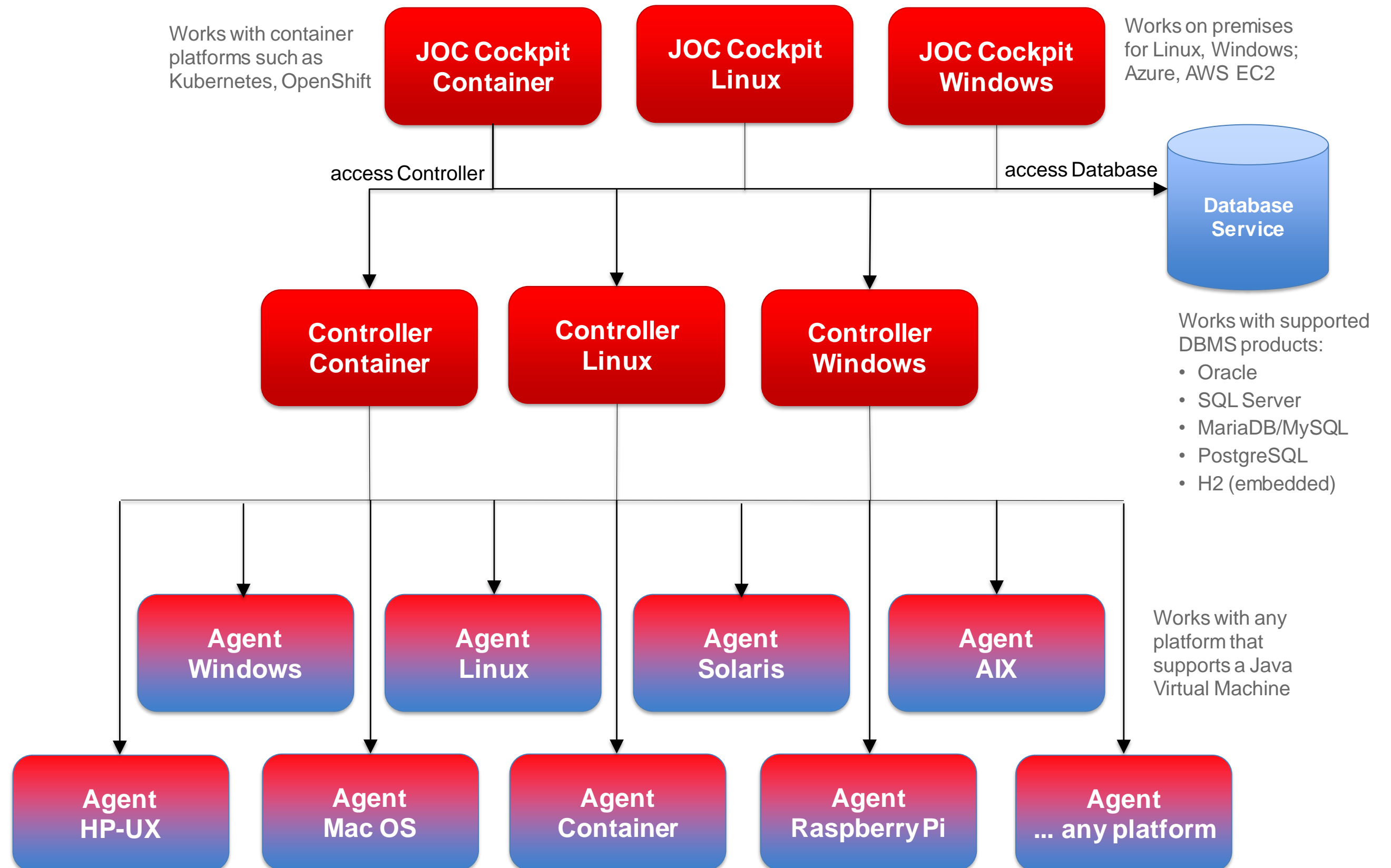
- The Controller is available for Container platforms, Linux and Windows
- Agents are available for any platform that supports a Java Virtual Machine including Containers

Database Service

- JOC Cockpit API Service makes use of a database service from any platform

Workflows

- Execution with Agents from any supported platform
- This includes mixed use of Agent platforms for parallel / sequential job execution





■ System Architecture

- System Architecture
- Product Architecture
- Secure Network Connections
- Supported Platforms

■ Cloud Setup

- Controller and JOC Cockpit High Availability
- Agent High Availability
- Hybrid Use of Agents

■ On Premises Setup

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

Cloud Setup: JOC Cockpit Cluster, Controller Cluster, Database Service Cluster

JOC Cockpit / API 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

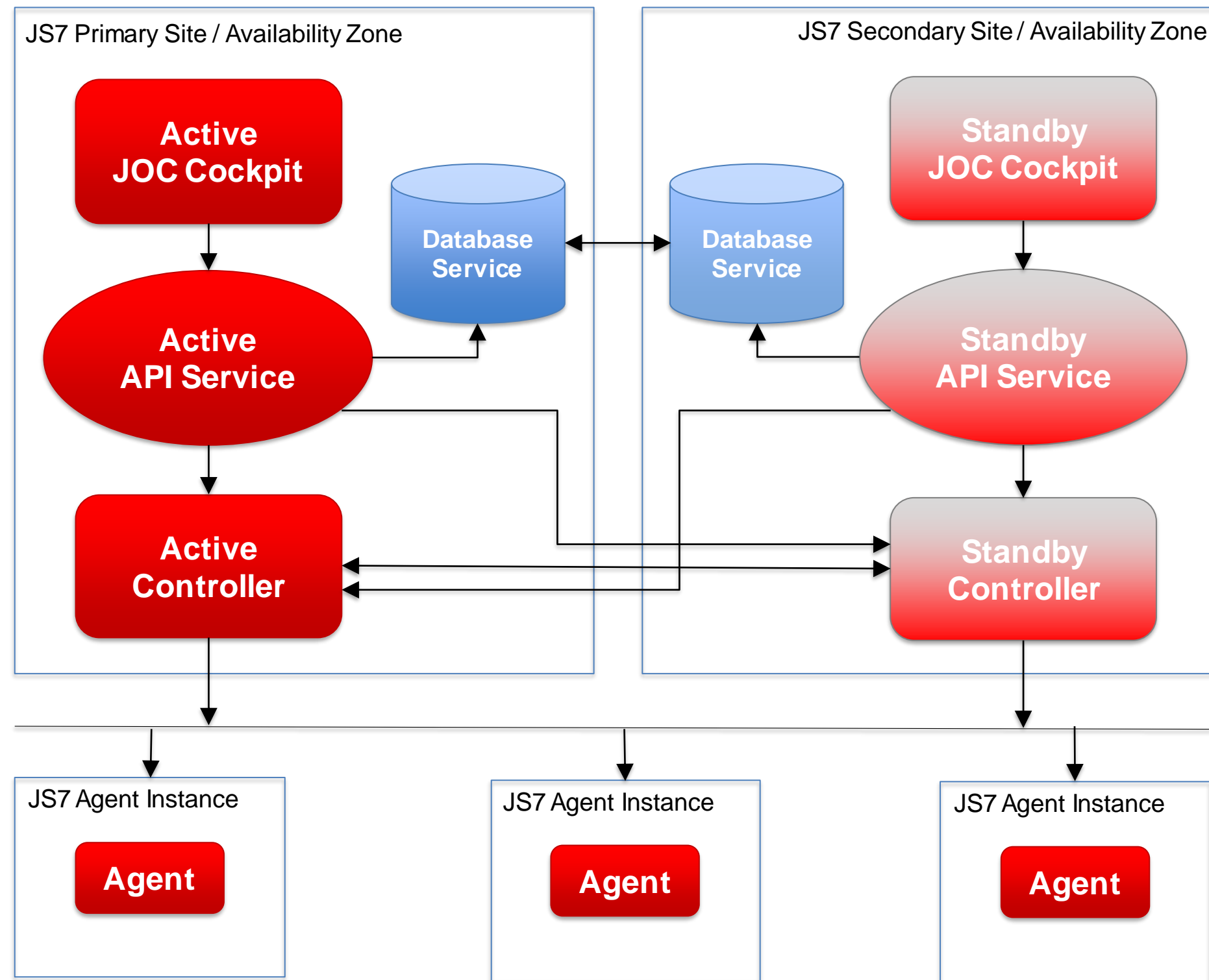
- Active / Standby Controller implement a passive cluster for automated fail-over

Agent

- Agents are deployed to any platforms and are accessed by the Active and Standby Controller instances

Database Service

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



JOC Cockpit / API Service

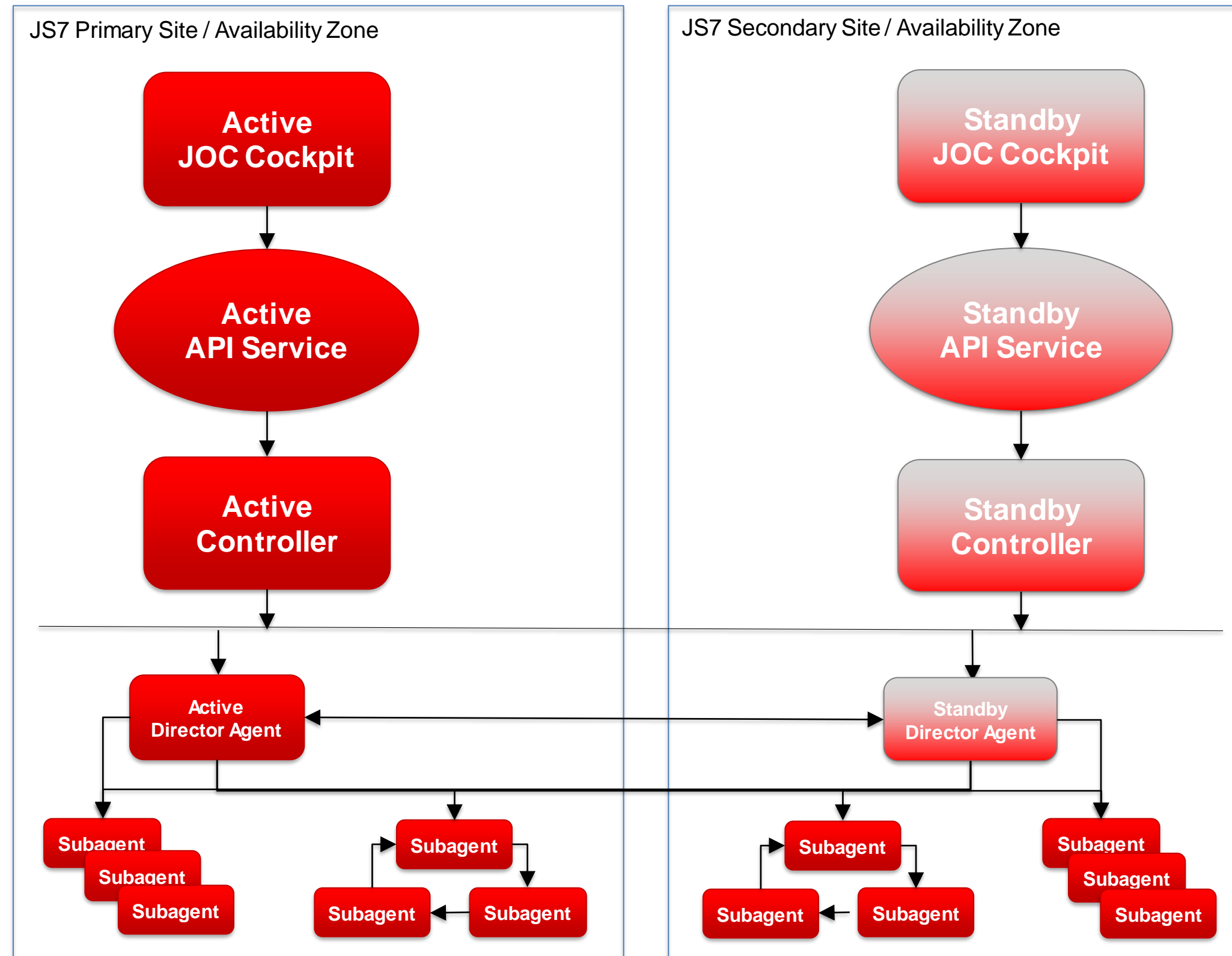
- JOC Cockpit is the User Interface for workflow management and control

Controller Cluster

- Active / Standby Controller implement a passive cluster for automated fail-over

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 unavailable the next Subagent is used
- Round-robin mode includes to execute each next job on the next Subagent



JOC Cockpit / API Service

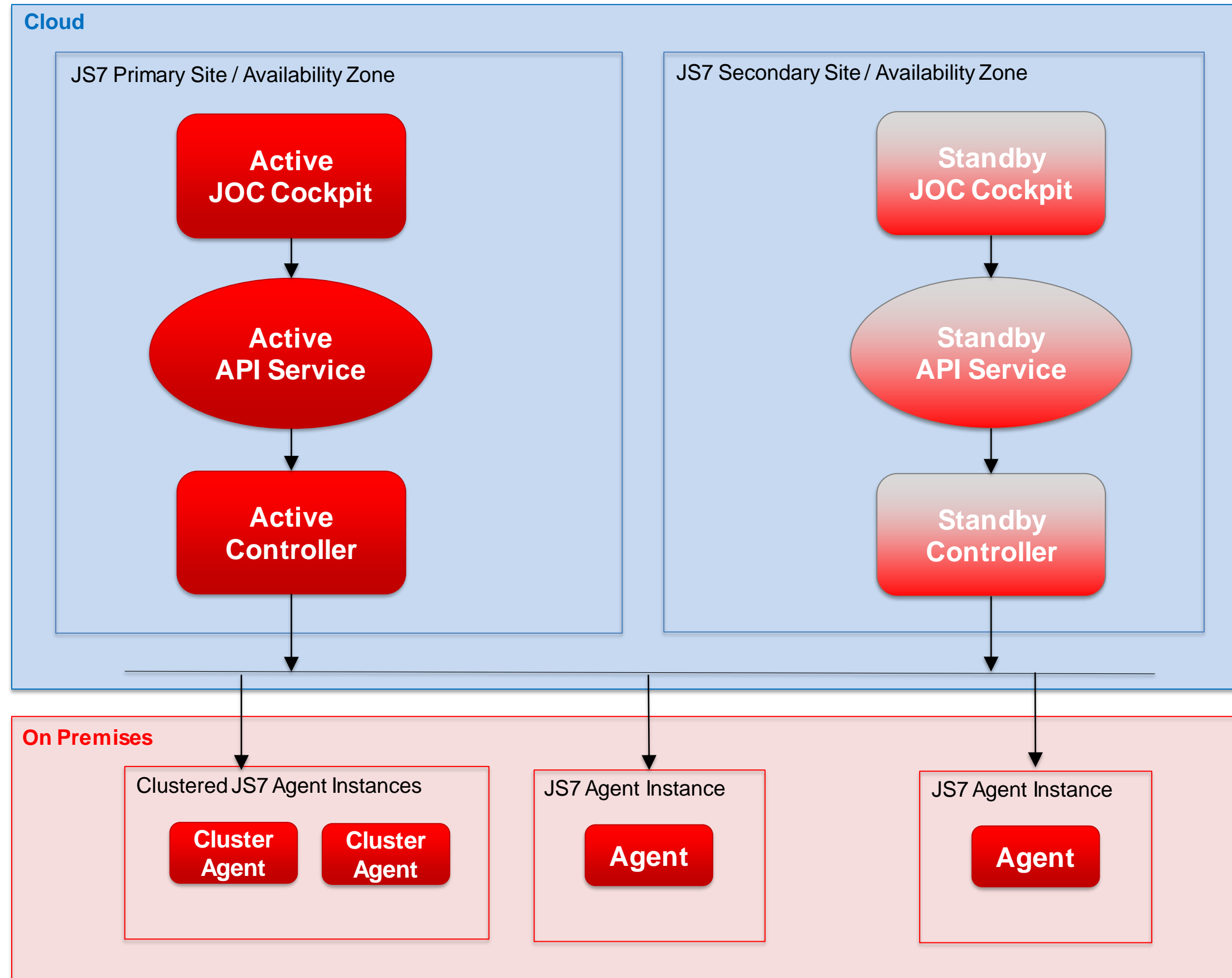
- JOC Cockpit is the User Interface for workflow management and control

Controller Cluster

- Active and Standby Controller implement a passive cluster for automated fail-over

Agents

- Any number of Cluster Agents and Standalone Agents can be operated on any platform used on premises
- Users set up a Virtual Private Cloud to allow the indicated connections
- Agents operated from cloud platforms and Agents operated on premises can be used in parallel





■ System Architecture

- System Architecture
- Product Architecture
- Secure Network Connections
- Supported Platforms

■ Cloud Setup

- JOC Cockpit and Controller High Availability
- Agent High Availability
- Hybrid Use of Agents

■ On Premises Setup

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

JOC Cockpit / API Service

- JOC Cockpit is the User Interface for workflow management and control
- Users access the JOC Cockpit from their browser

Controller

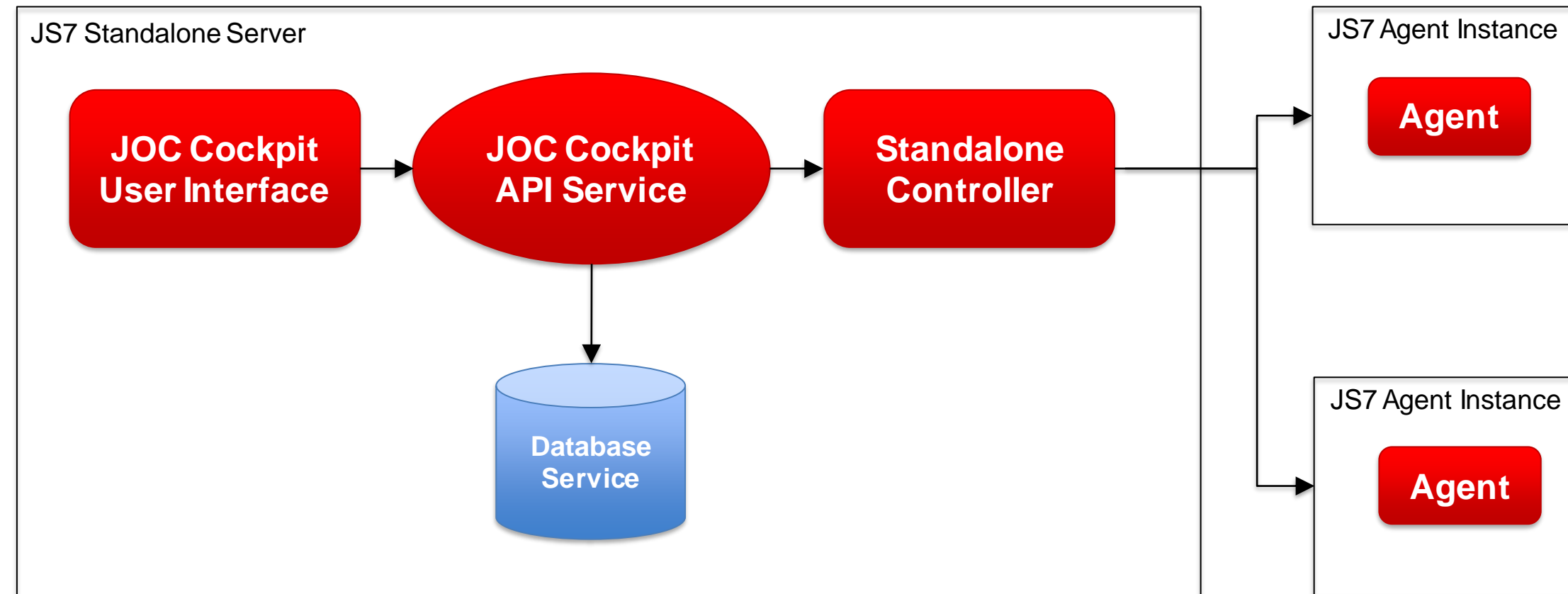
- The Controller orchestrates Agents for execution of workflows and jobs

Agent

- Agents are deployed on top of platforms running the programs, scripts, services scheduled for execution

Database Service

- The database stores the inventory and history of workflow execution



On Premises: Standalone Interface Server, Controller Cluster, Database Server

JOC Cockpit / API Service

- JOC Cockpit is the User Interface for workflow management and control
- The Controller cluster is managed by JOC Cockpit

Controller Cluster

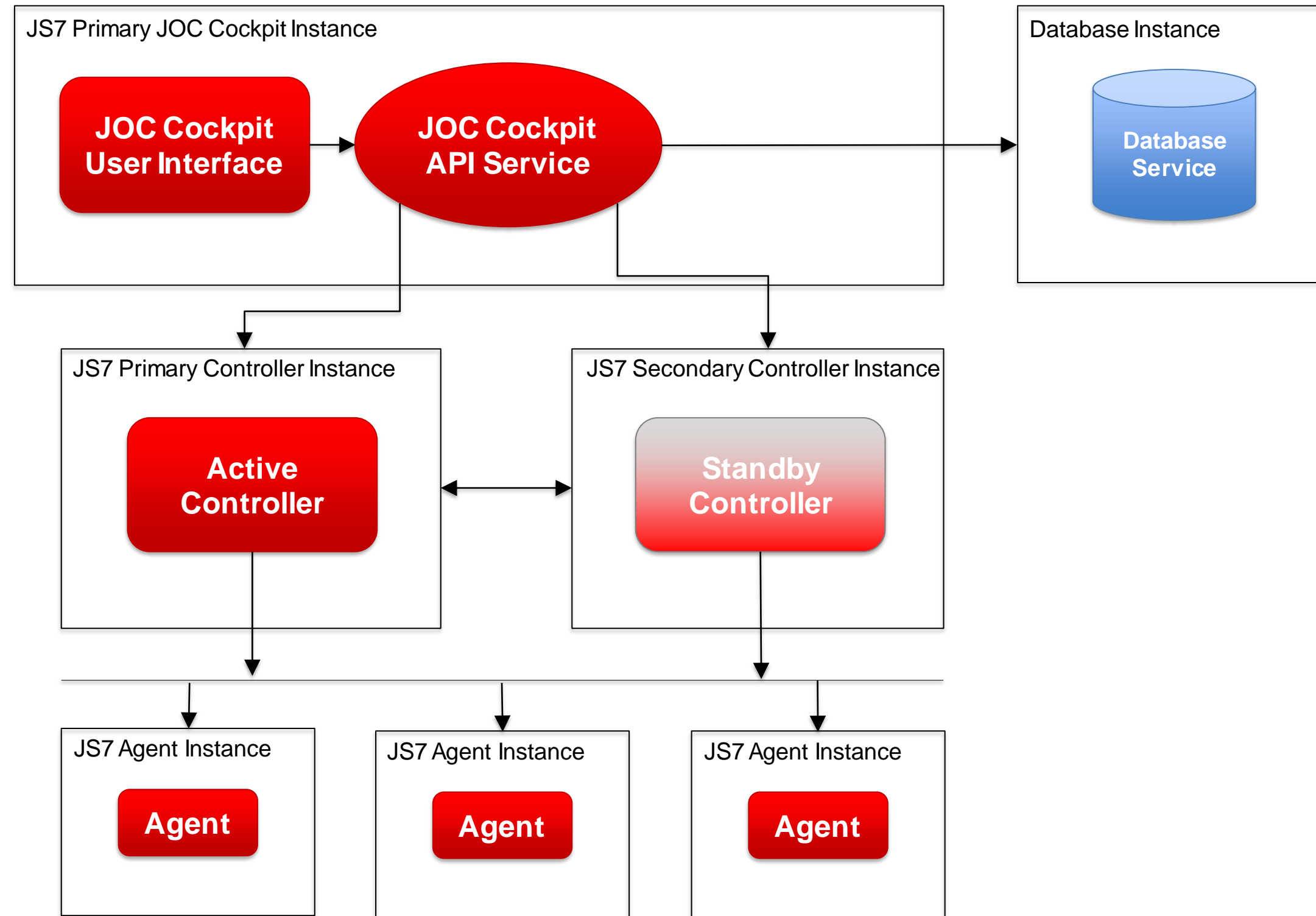
- Active and Standby Controller act as a cluster to synchronize status information for automated fail-over
- Active and Standby Controller are accessed by the JOC Cockpit API Service

Agent

- Agents are deployed on top of any platforms and are accessed by Active and Standby Controllers

Database Service

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



On Premises: JOC Cockpit Cluster, Controller Cluster, Database Server

JOC Cockpit / API 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

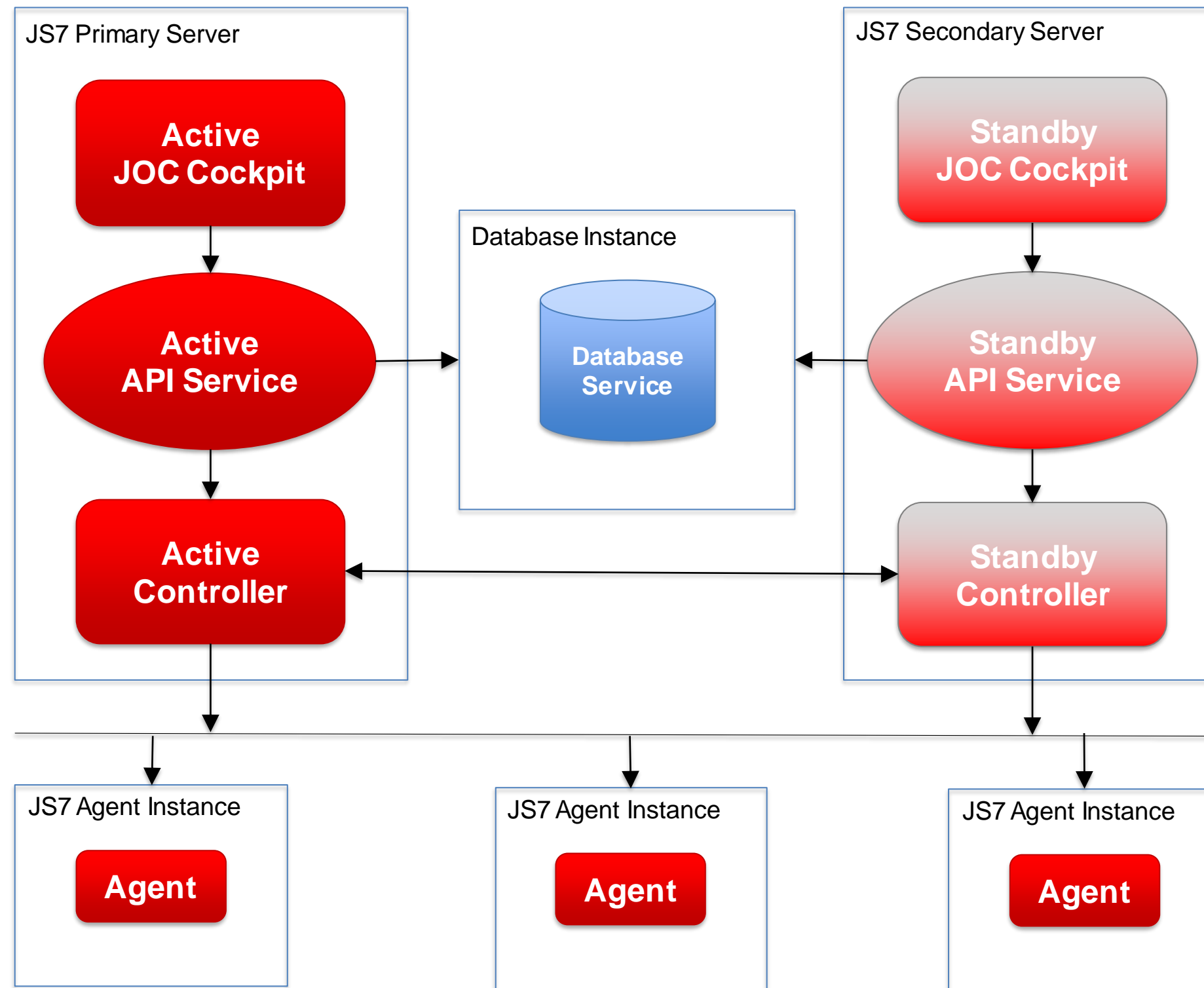
- Active / Standby Controller implement a passive cluster with automated fail-over

Agent

- Agents are deployed on top of any platform and are accessed by the Active and Standby Controller

Database Service

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



On Premises: Multi-Controller Instances

JOC Cockpit / API Service

- JOC Cockpit is the User Interface for workflow management and control
- Users can manage a number of Controllers in JOC Cockpit

Controller

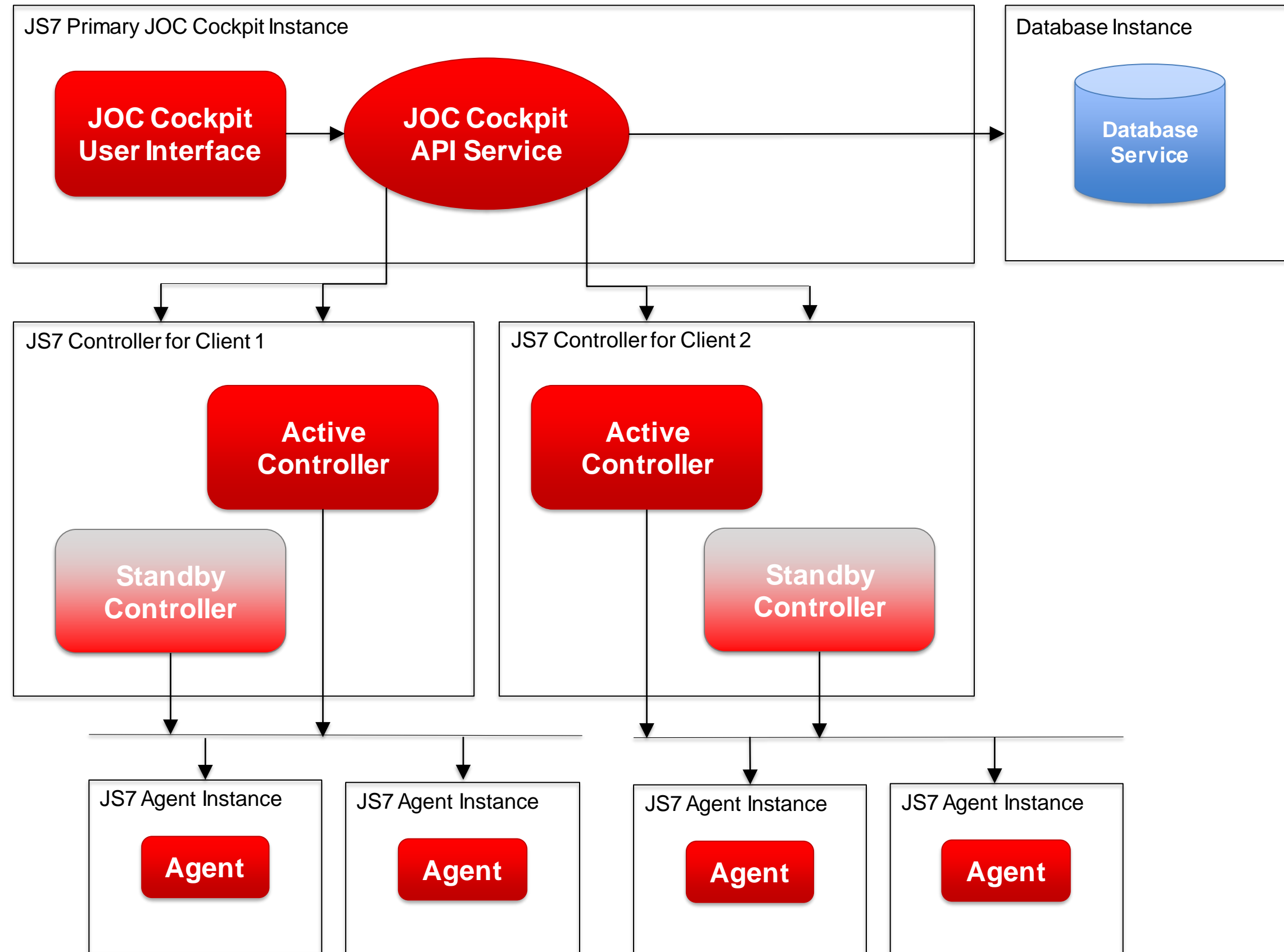
- Controllers are operated independently per Client
- Controllers can be operated as a cluster and standalone

Agent

- Agents are deployed on top of any platform and are accessed by a Controller
- Agents are dedicated to a Controller

Database Service

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



On Premises: Controller Cluster with Agent Cluster and Standalone Agents

Controller

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

Agents

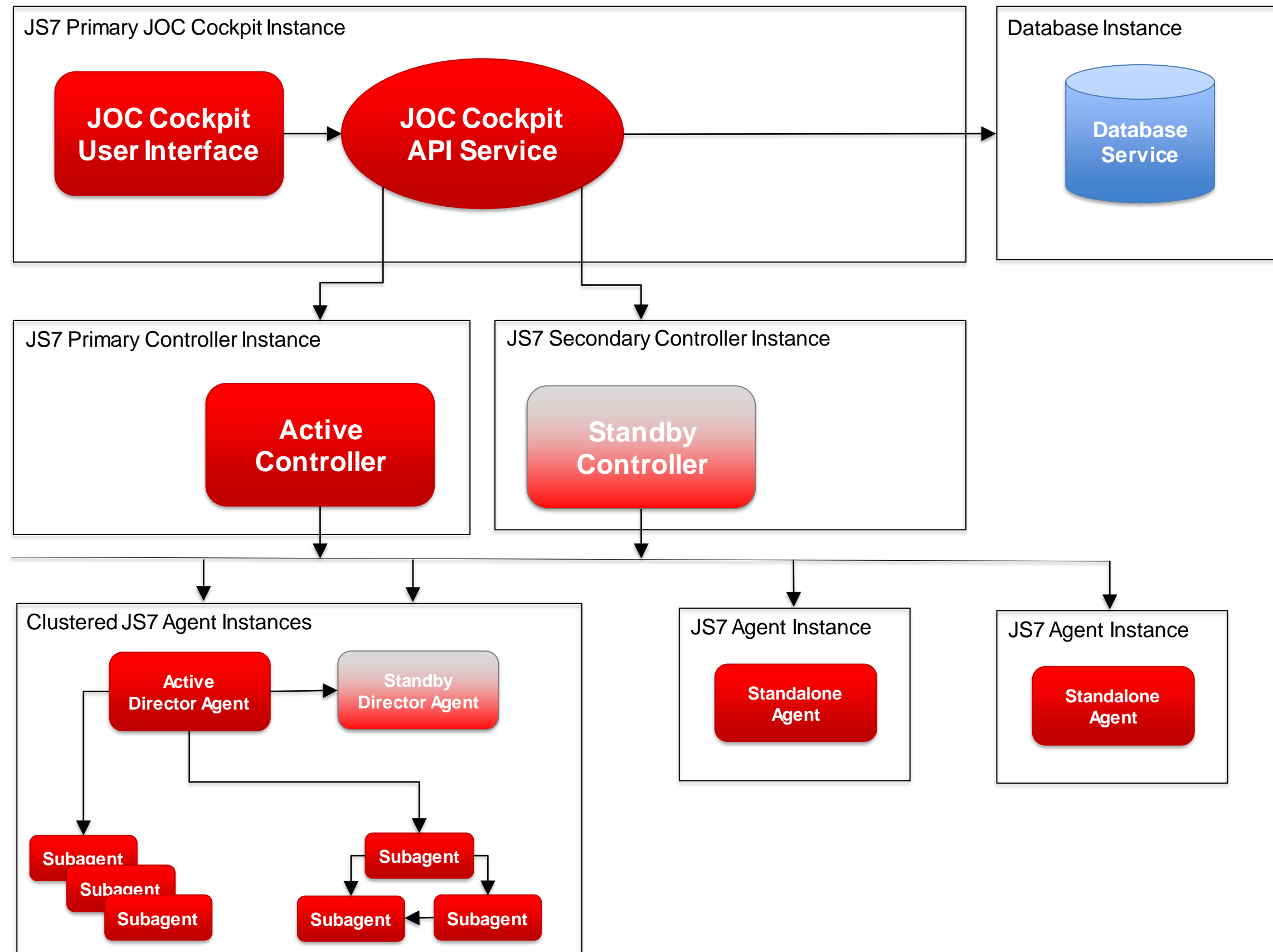
- Agents are deployed on top of any platform and are accessed by a Controller
- Agents are dedicated to a Controller

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 unavailable 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 can be operated on any platform





Questions?
Comments?
Feedback?

Software- und
Organisations-
Service GmbH

Giesebrechtstr. 15
D-10629 Berlin

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