Consulting Services

JobScheduler
Architecture Decision Template

Information for Consulting Parties

Contents

Consulting Services

Architecture Decisions

Architecture Decision Template

Agent Cluster

- Agent Cluster
- Fixed Priority and Round-Robin Scheduling: Redundancy and automated fail-over

Passive Cluster

- Primary & Backup JobScheduler
- Redundancy and automated fail-over

Active Cluster

- Active Cluster JobScheduler
- Redundancy and load sharing

Master/ Agent Cluster

- Master/Agent Cluster JobScheduler
- Redundancy, load sharing, load distribution

Supervisor JobScheduler

- Passive & Active Cluster Support,
- Master/Agent Cluster Support, Unclustered JobScheduler Support
- Central Configuration

Components: JOC Cockpit / Web Service / Master / Agent

Overview: Components

JOC Cockpit / Web Service

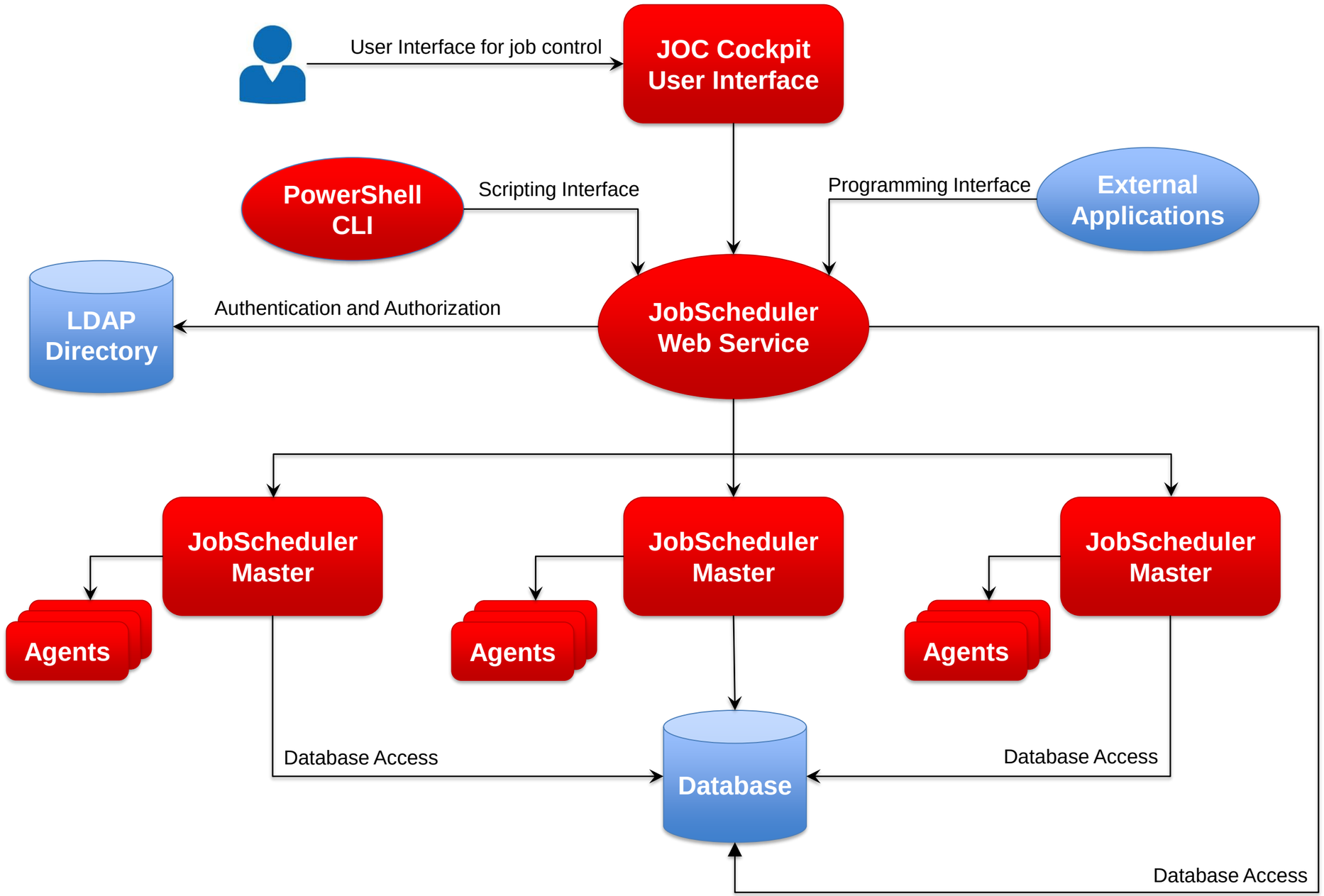
- The JOC Cockpit is the user interface for job control with browsers
- Users access the Master using a Web Service that performs authentication and authorization – optionally against an LDAP directory

Interfaces

- The PowerShell Command Line Interface and External Applications use the same Web Service for access to a JobScheduler Master
- Authorization is available for individual requests to the JobScheduler Master

Master / Agent

- The JobScheduler Master executes tasks and orchestrates Agents
- Agents are deployed on top of existing servers running the programs and scripts that should be scheduled



Security: JOC Cockpit / Web Service / Master / Agent

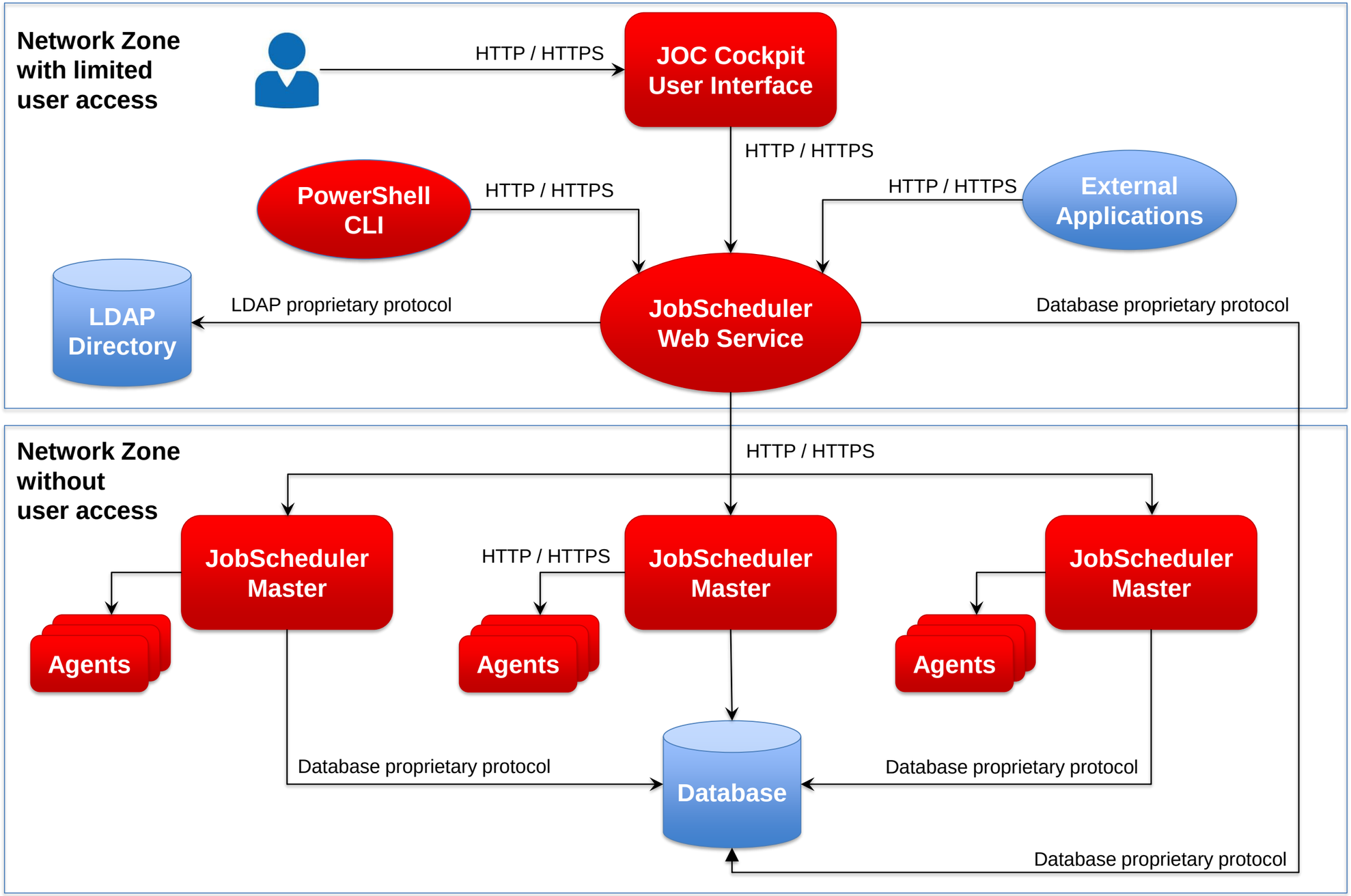
Overview: Security

Network Zone with restricted user access

- Users have limited access that requires authentication
- Any connection to a Master is authenticated by the Web Service that can be configured to use LDAP
- Use of HTTPS for connections can be enforced

Network Zone without user access

- Master and Agent instances are operated in this zone without direct user access
- The Master instances are accessed exclusively by the Web Service
- The Agent instances are accessed exclusively by Master instances



Platforms: JOC Cockpit / Web Service / Master / Agent

Overview: Supported Platforms

Cockpit / Web Service

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

Master / Agent

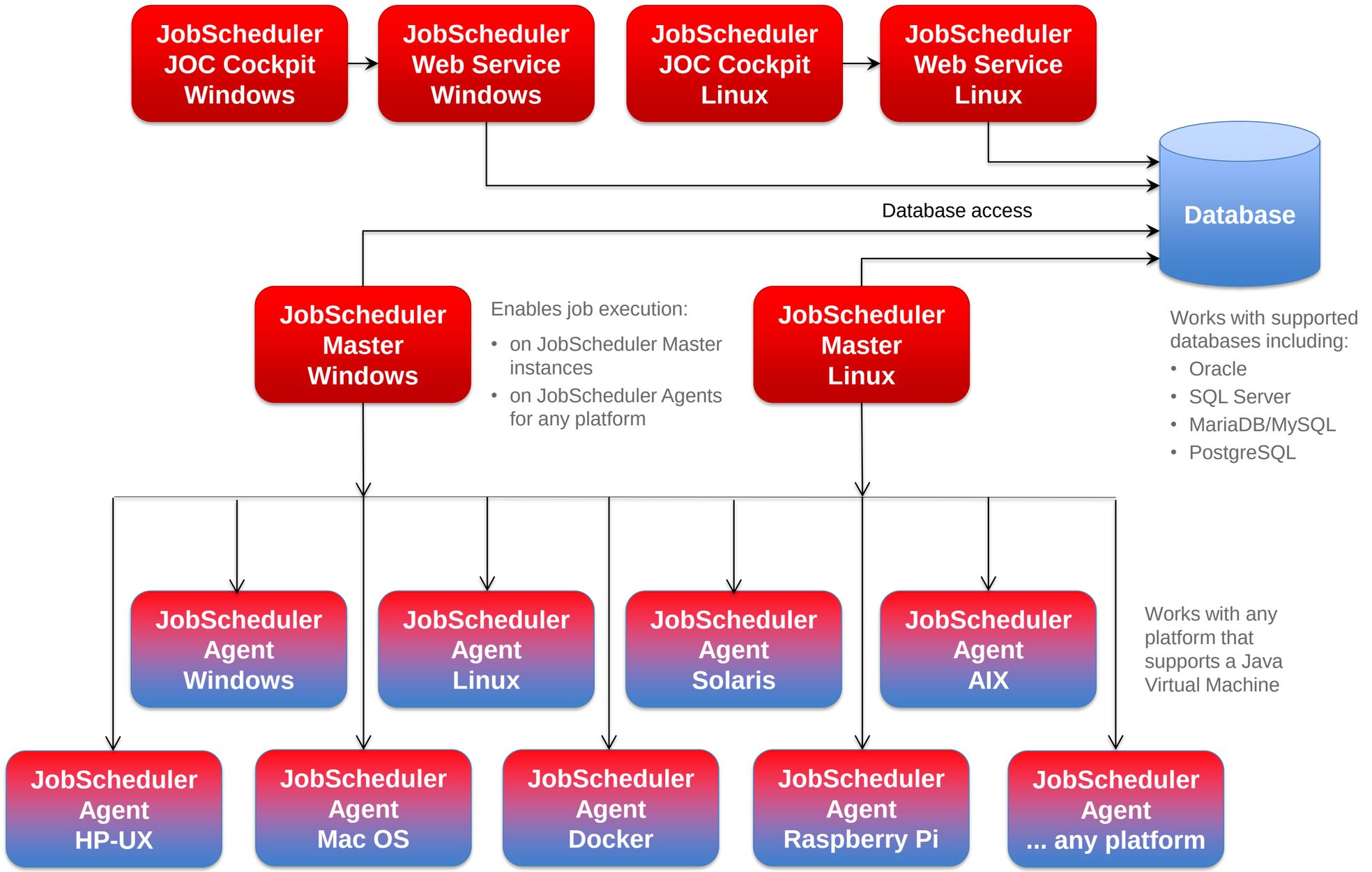
- JobScheduler Master is available for Windows and Linux
- JobScheduler Agents are available for any platform that supports a Java Virtual Machine

Database

- The JobScheduler REST Web Service and Master use a database on any platform

Jobs

- Jobs can be executed locally on the Master
- Jobs can be executed on any JobScheduler Agent



Setup Scenario: Standalone JobScheduler Server

Scenario: Standalone JobScheduler Server for Interface, Master and Database

JOC Cockpit / Web Service

- The JOC Cockpit is the user interface for job control
- Users access the Master using a Web Service that performs authentication and authorization

Master

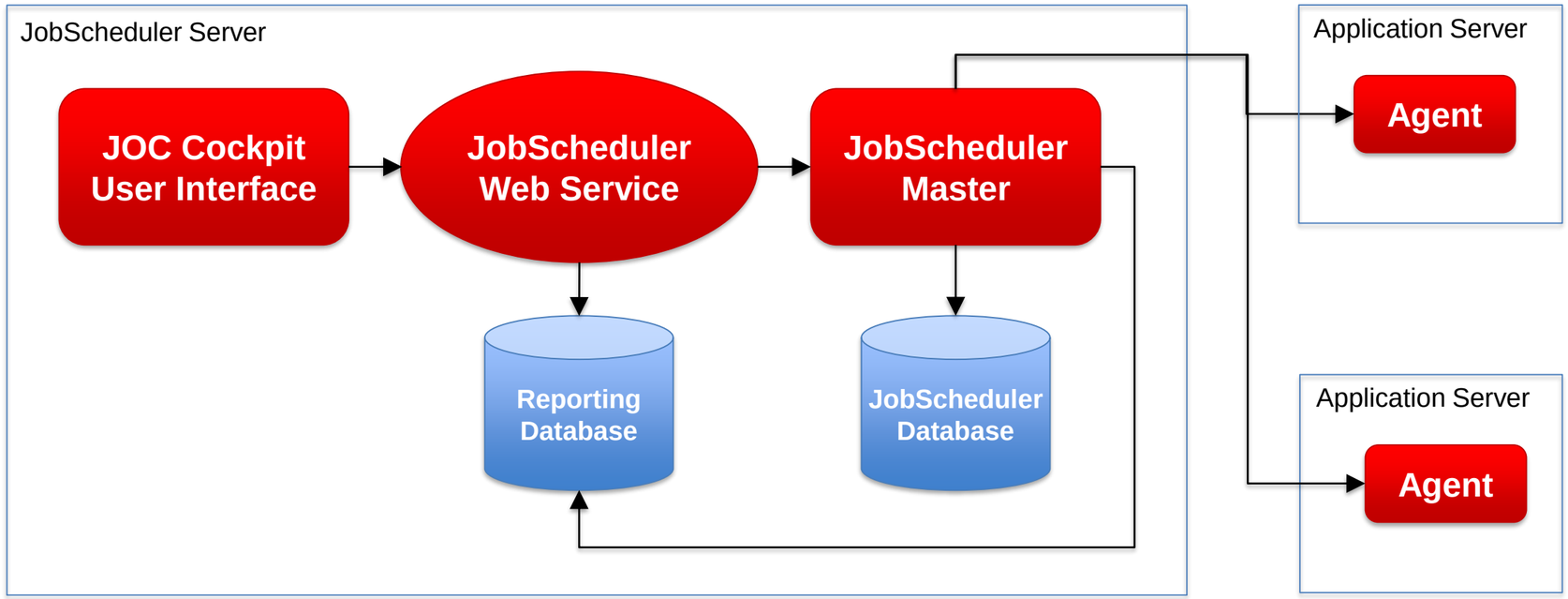
- The JobScheduler Master executes local tasks and orchestrates Agents for execution of remote tasks

Database

- The JobScheduler Database stores run-time information
- The Reporting Database stores the inventory and history information of jobs
- Databases can be mapped to a single database with a common schema

Agent

- Agents are deployed on top of existing servers running the programs and scripts that should be scheduled



Setup Scenario: JobScheduler Server, Database Server

Scenario: Standalone JobScheduler Server for Interface and Master, separate Database Server

JOC Cockpit / Web Service

- The JOC Cockpit is the user interface for job control
- Users access the Master using a Web Service

Master

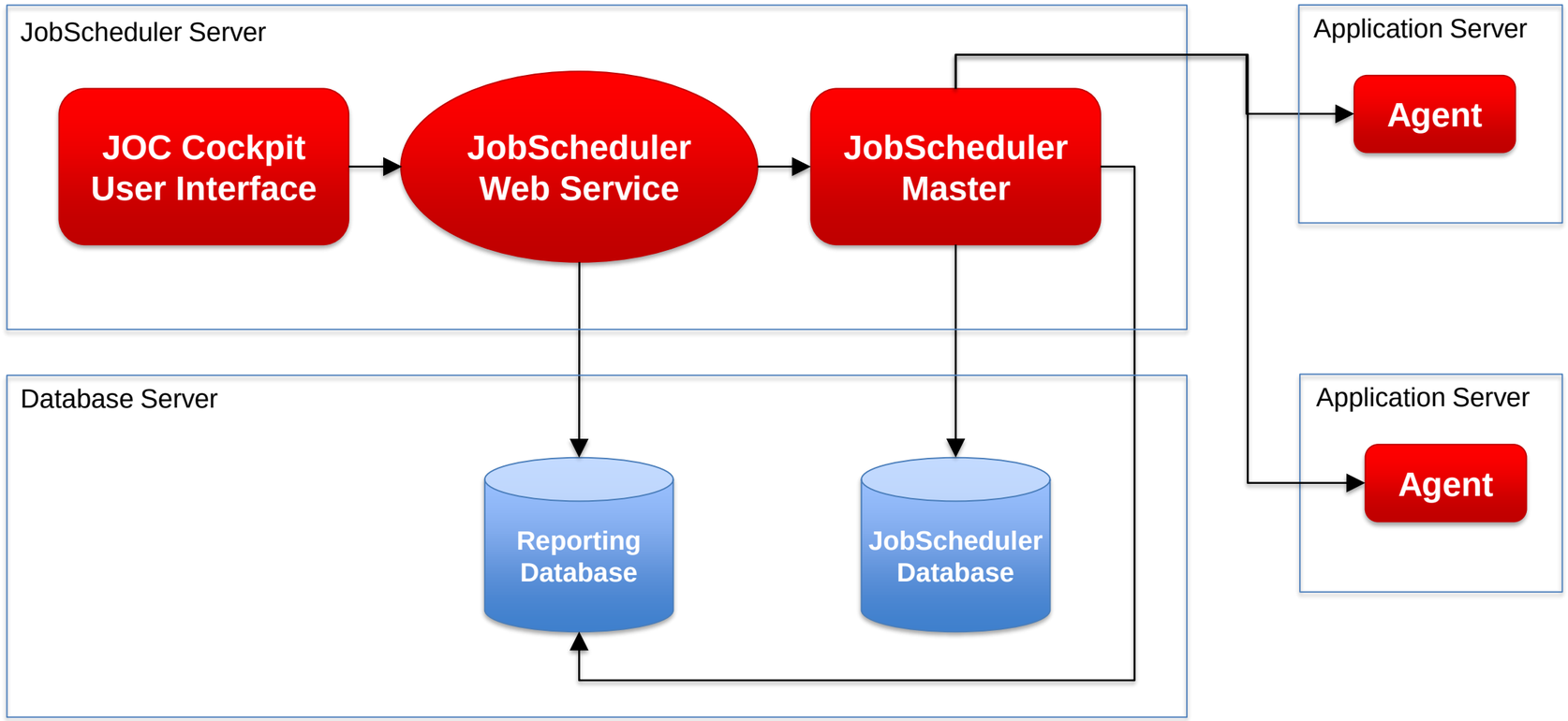
- The JobScheduler Master executes local tasks and orchestrates Agents for execution of remote tasks

Database

- The JobScheduler Database stores run-time information
- The Reporting Database stores the inventory and history information of jobs
- Databases can be operated from a database server and can be mapped to a single database instance with a common schema

Agent

- Agents are deployed on top of existing servers running the programs and scripts that should be scheduled



Setup Scenario: High Availability

Scenario: Standalone Interface Server, Master Cluster, Database Server

JOC Cockpit / Web Service

- The JOC Cockpit is the user interface for job control
- Users access the Master using a Web Service

Master Cluster

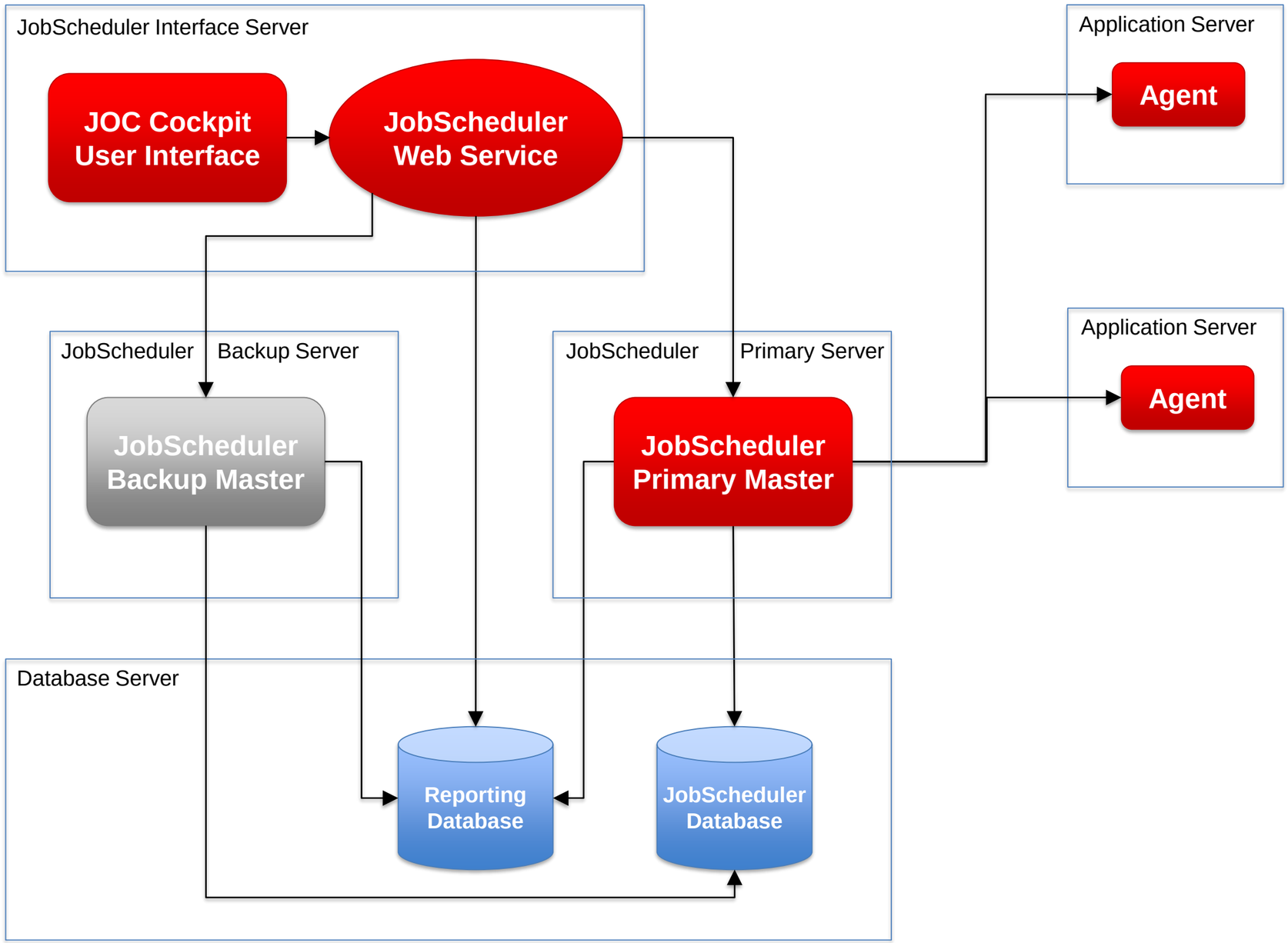
- Primary and Backup Master implement an automated failover in case of failure
- Primary and Backup Master are accessed by the Web Service
- Primary and Backup Master use a clustered database

Database

- JobScheduler and Reporting Databases are available in a database cluster

Agent

- Agents are deployed on top of existing servers and can be accessed by the Primary and Backup Master



Setup Scenario: High Availability

Scenario: Master Passive Cluster, JOC Cockpit Active Cluster, Database Server

JOC Cockpit / Web Service

- The JOC Cockpit is the user interface for job control
- A number of JOC Cockpit instances are operated as a passive or active cluster
- Each JOC Cockpit instance can access any Master

Master Cluster

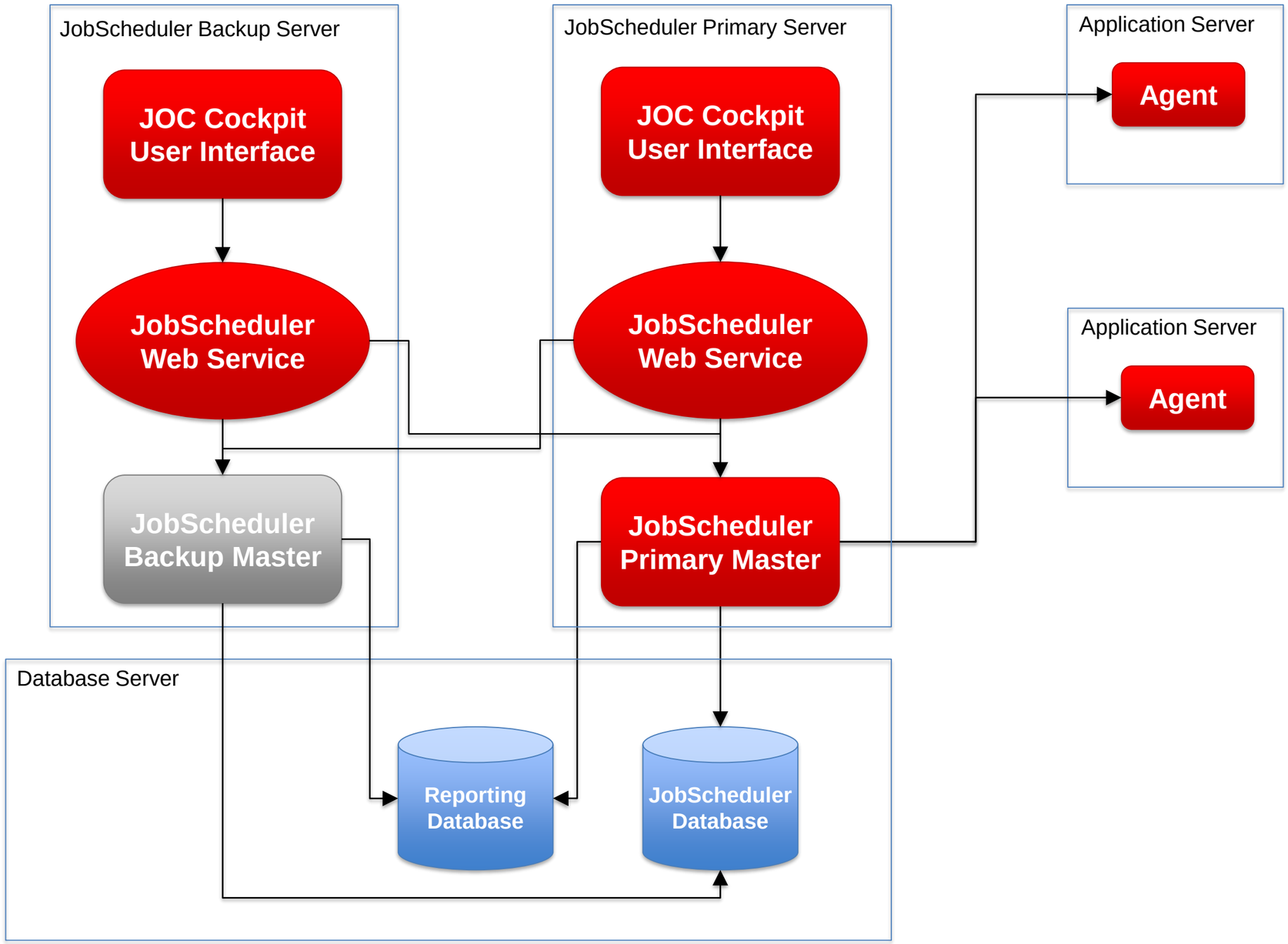
- Primary and Backup Master implement an automated failover in case of failure
- Primary and Backup Master use a clustered database

Database

- JobScheduler and Reporting Databases are available in a database cluster

Agent

- Agents are deployed on top of existing servers and can be accessed by the Primary and Backup Master



Setup Scenario: Multi Master

Scenario: Interface Server, Multi Master Servers with local Databases, Reporting Database Server

JOC Cockpit / Web Service

- The JOC Cockpit is the user interface for job control
- Users access the Master using a Web Service

Master

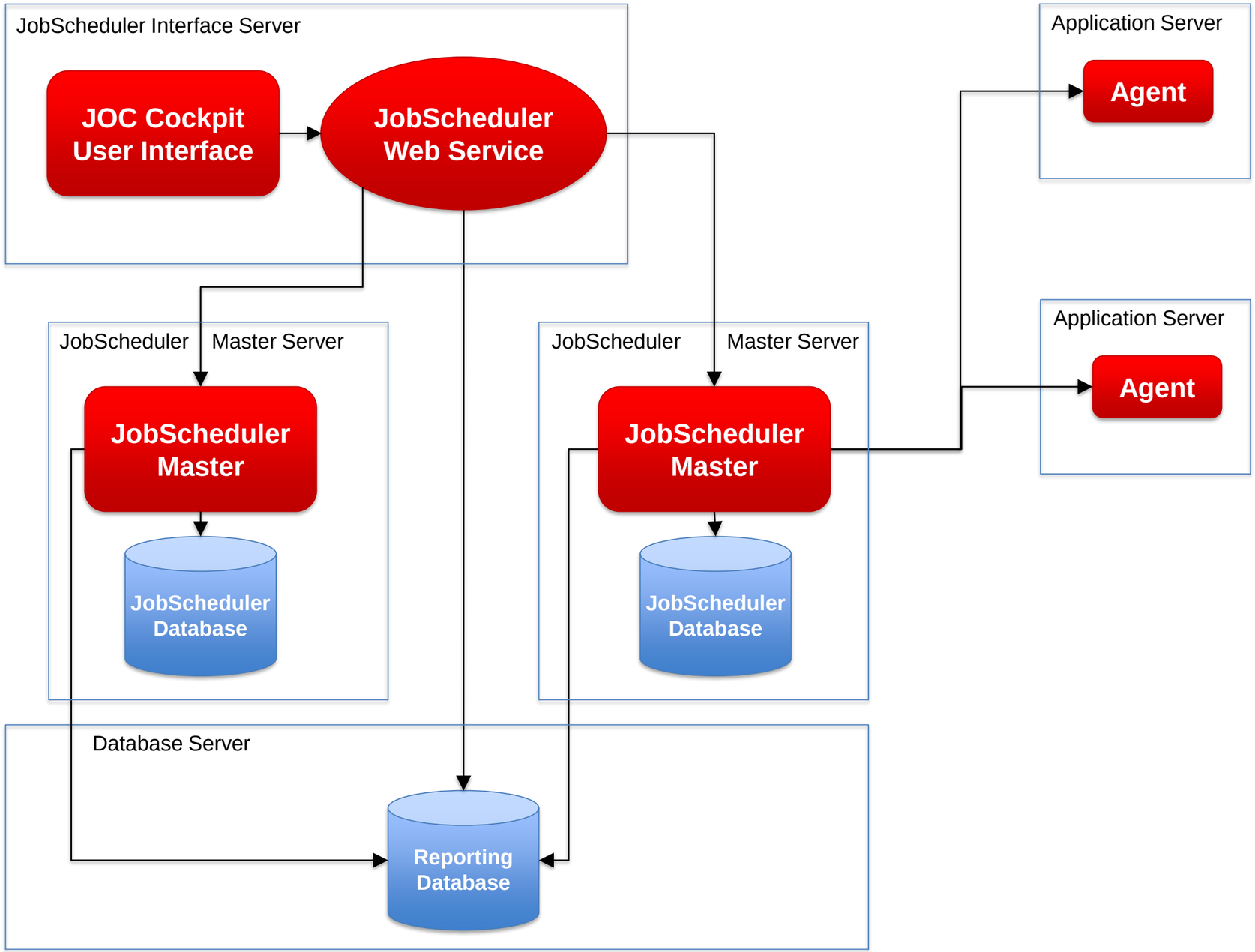
- Multiple Master instances are accessed by the JOC Cockpit user interface

Database

- The JobScheduler Database stores run-time information and is operated locally per each Master instance
- The Reporting Database stores the inventory and history information of jobs
- Failure of the Reporting Database does not prevent a Master from running jobs

Agent

- Agents are deployed on top of existing servers and can be accessed by any Master



Architecture: JobScheduler Agent Cluster

Architecture Decision Templates: Agent Cluster

Master/Agent Platforms

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

Agent Cluster

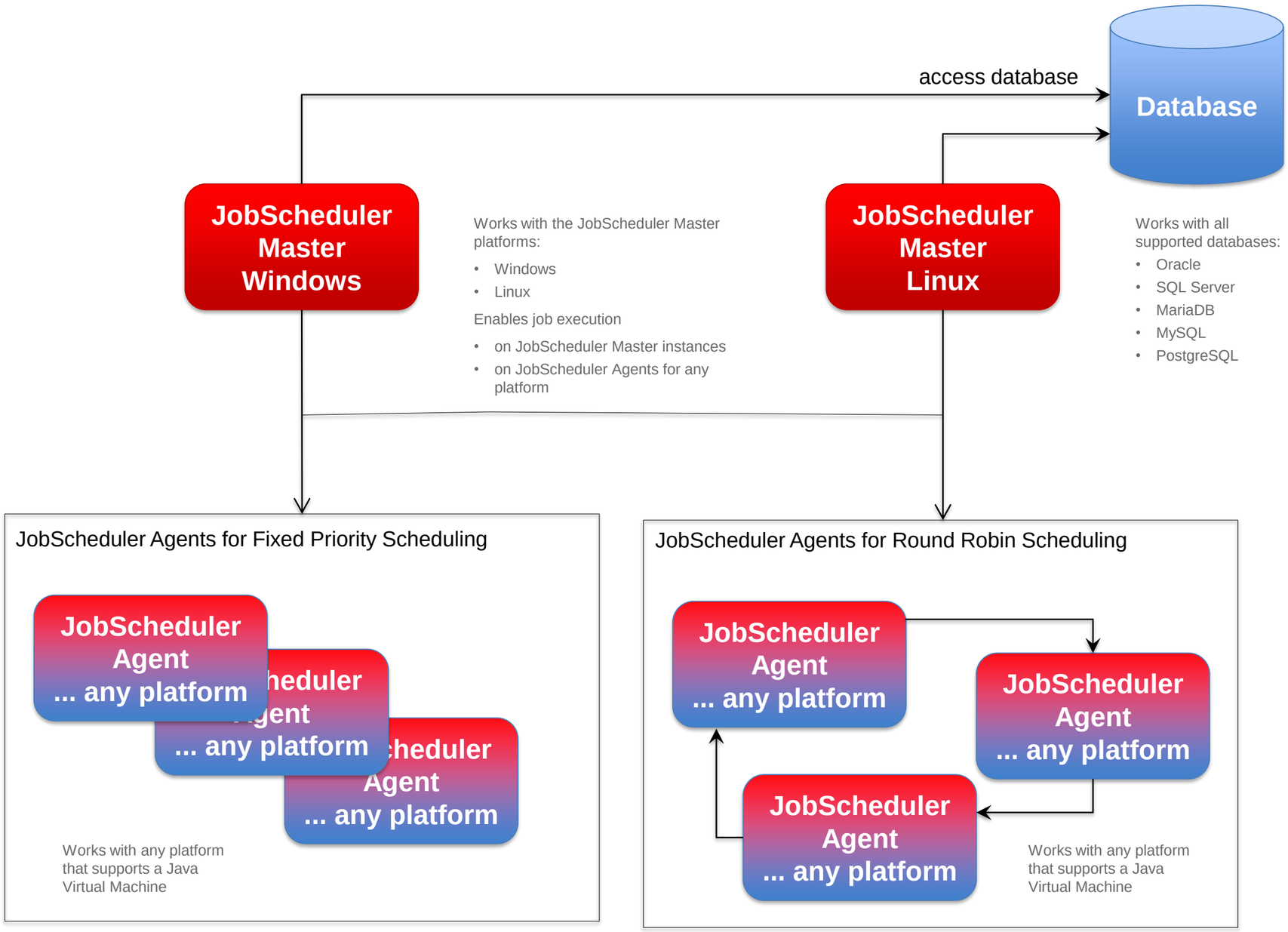
- Agents can be configured to work in a Cluster

Fixed Priority Scheduling

- JobScheduler Master selects the first available Agent from a Cluster for job execution
- Should an Agent not be available then the next available Agent is used

Round-Robin Scheduling

- JobScheduler Master switches the Agent used for each job execution
- Should an Agent not be available then the next available Agent is used



Architecture: Primary JobScheduler Master

Architecture Decision Templates: Master Passive Cluster

Passive Cluster

- Primary and Backup JobScheduler Master use the same database
- Primary JobScheduler Master is monitored by its failover instance
- Failover instance operates in stand-by mode
- All connections to servers use the SSH protocol

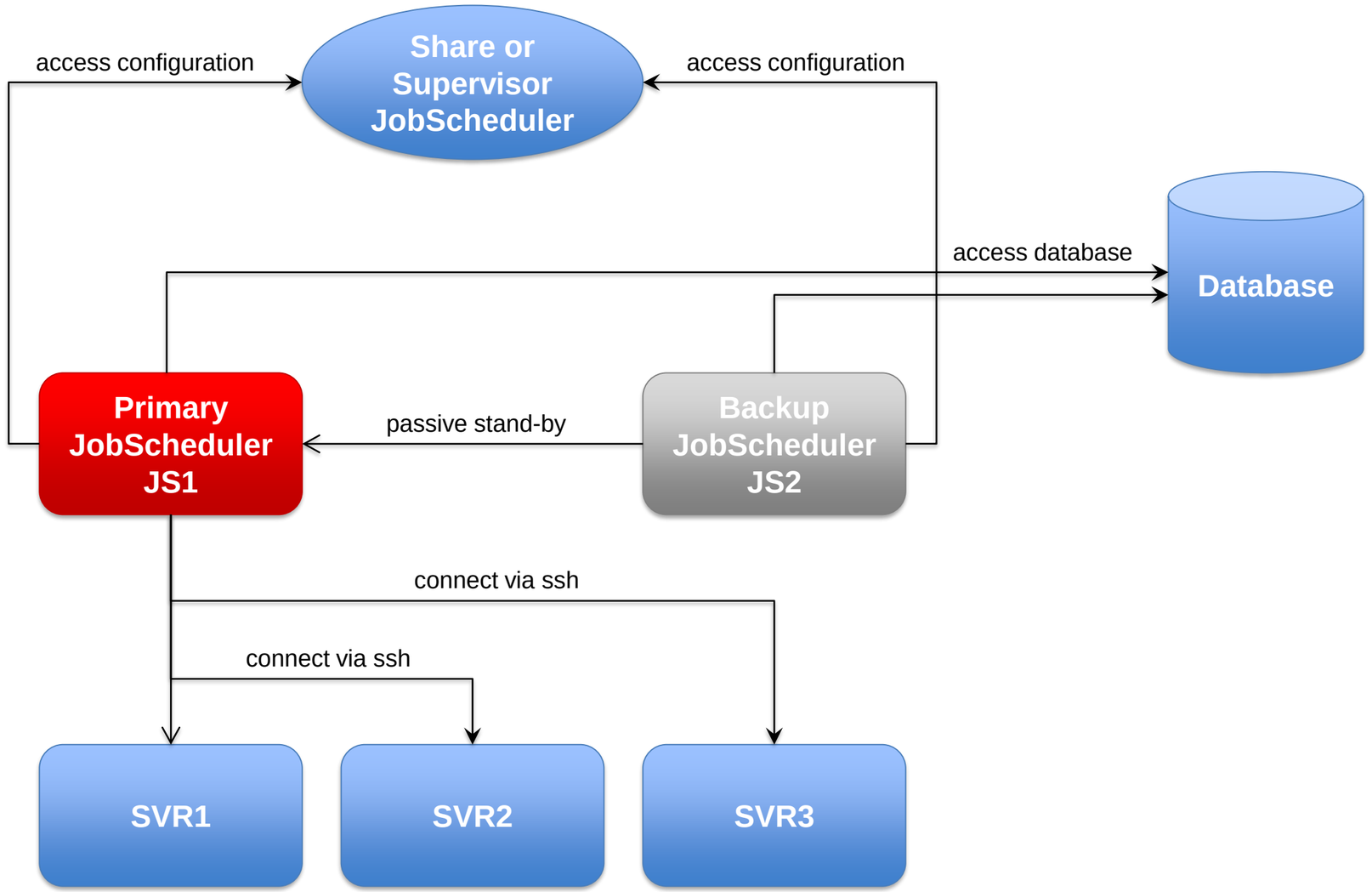
SSH Connections

JITL Jobs

- Requires a JVM per task
- Memory resources

SSH Client

- No pre-/post-processing
- No substitution of parameters in script files
- Script files have to be provided on the target system



Architecture: Backup JobScheduler Master

Architecture Decision Templates: Master Passive Cluster

Passive Cluster

- Primary and Backup JobScheduler Master both use the same database
- Backup JobScheduler Master is active after failure of Primary instance
- Primary instance operates in stand-by mode
- All connections to servers use the SSH protocol

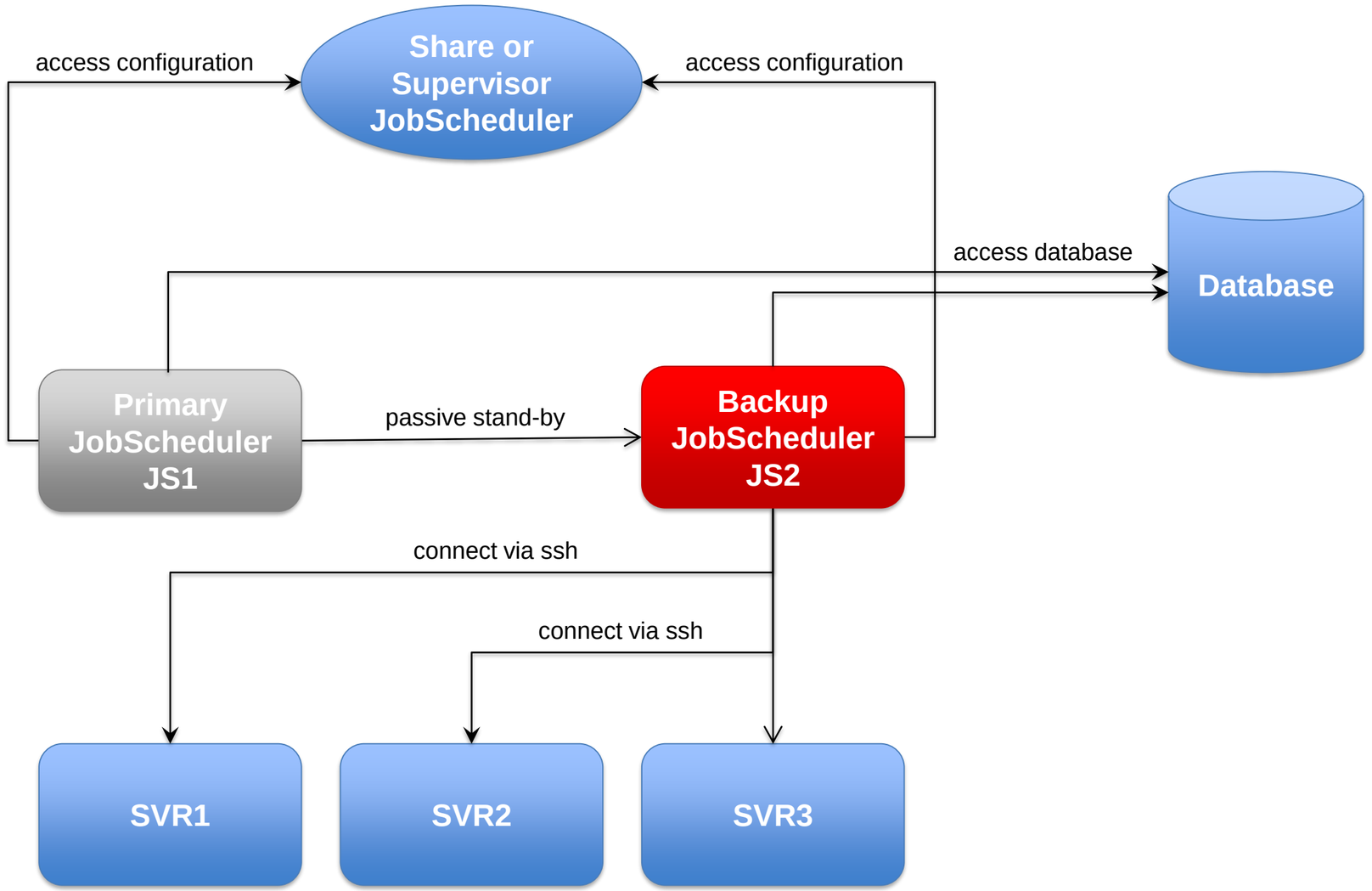
SSH Connections

JITL Jobs

- Requires a JVM per task
- Memory resources

SSH Client

- No pre-/post-processing
- No substitution of parameters in script files
- Script files have to be provided on the target system



Architecture: Active Cluster JobScheduler Master

Architecture Decision Templates: Master Active Cluster

Active Cluster

- JobScheduler Cluster members use the same database
- JobScheduler Cluster members share the workload of jobs
- All Instances operate in active mode
- All connections to servers use the ssh protocol

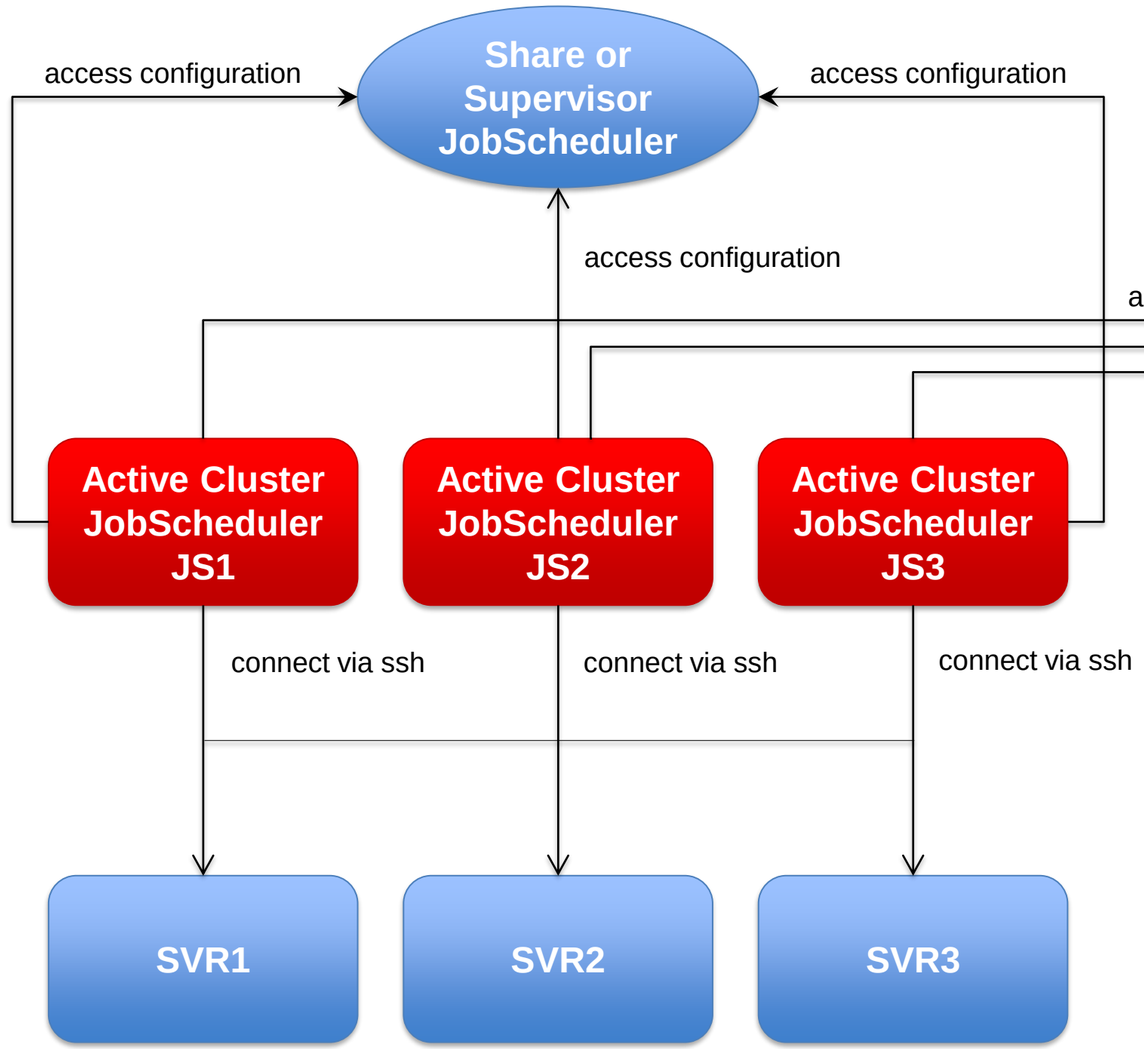
SSH Connections

JITL Jobs

- Requires a JVM per task
- Memory resouces

SSH Client

- No pre-/post-processing
- No substitution of parameters in script files
- Script files have to be provided on the target system



Architecture: Active Cluster JobScheduler with failed instance

Architecture Decision Templates: Master Active Cluster

Active Cluster

- JobScheduler Cluster members use the same database
- JobScheduler Cluster members share the workload of jobs
- All Instances operate in active mode
- All connections to servers use the ssh protocol

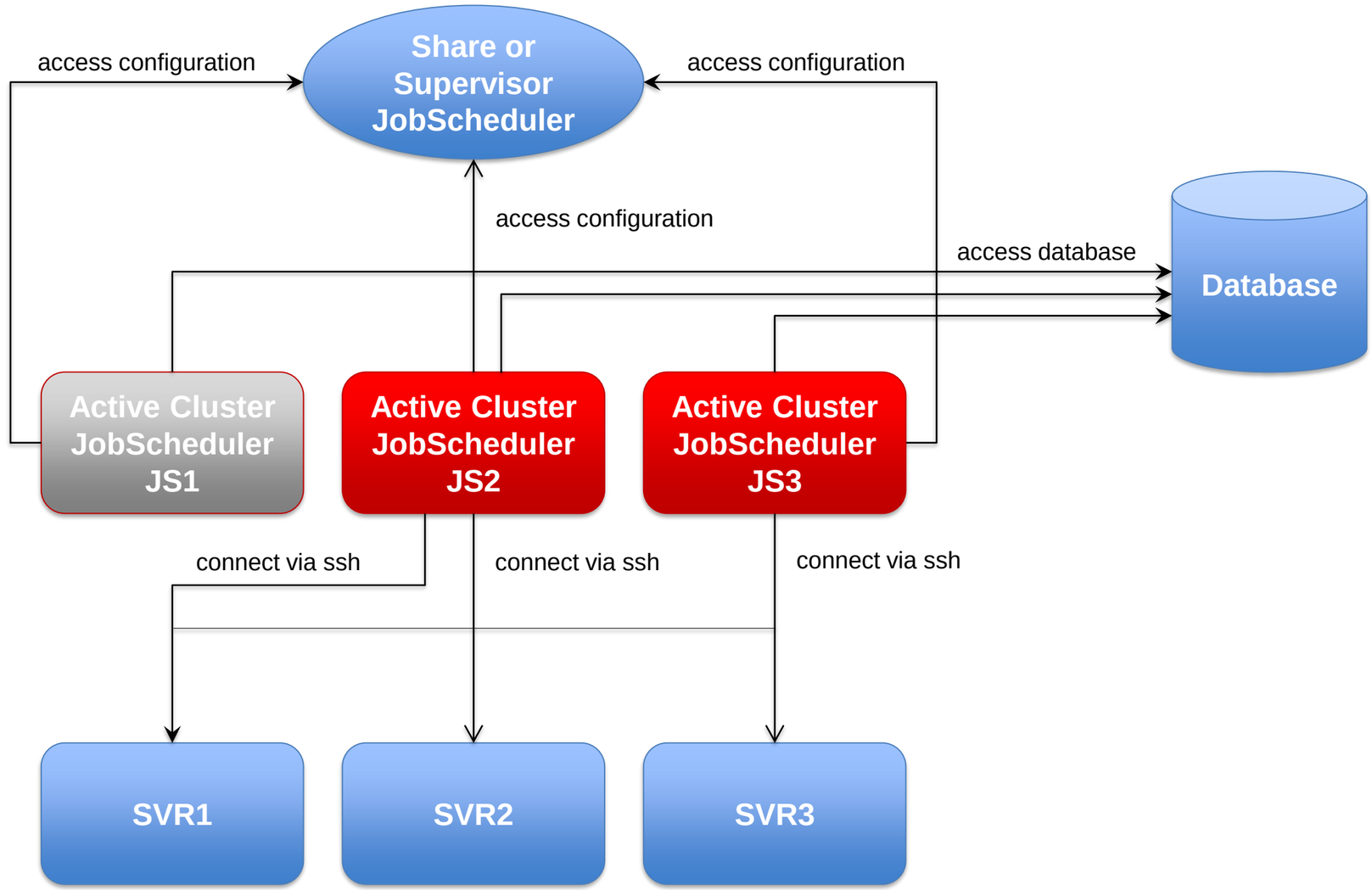
SSH Connections

JITL Jobs

- Requires a JVM per task
- Memory resources

SSH Client

- No pre-/post-processing
- No substitution of parameters in script files
- Script files have to be provided on the target system



Architecture: Master/Agent Passive Cluster JobScheduler

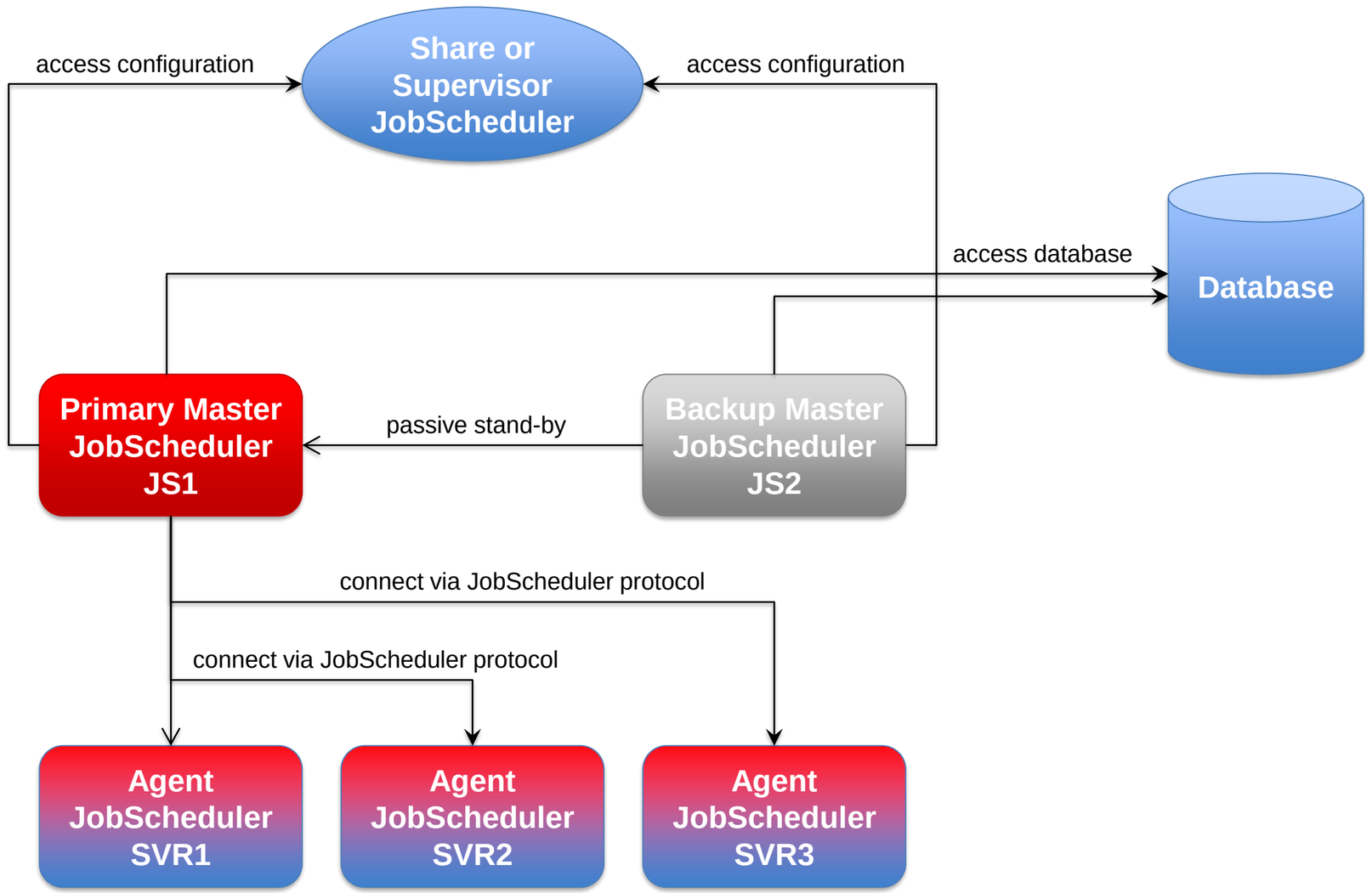
Architecture Decision Templates: Master/Agent Passive Cluster

Master/Agent Passive Cluster

- Primary and Backup JobScheduler use the same database
- Primary JobScheduler is monitored by its Backup instance
- Backup instance operates in stand-by mode
- All Cluster instances use Agents to execute jobs on remote servers
- Connections to servers use the internal protocol

Job Execution

- Jobs are executed locally per JobScheduler Agent.
- No central resources required for job execution
- Pre-/post-processing
- Use of JITL Jobs or script files with parameter substitution



Architecture: Master/Agent Active Cluster JobScheduler

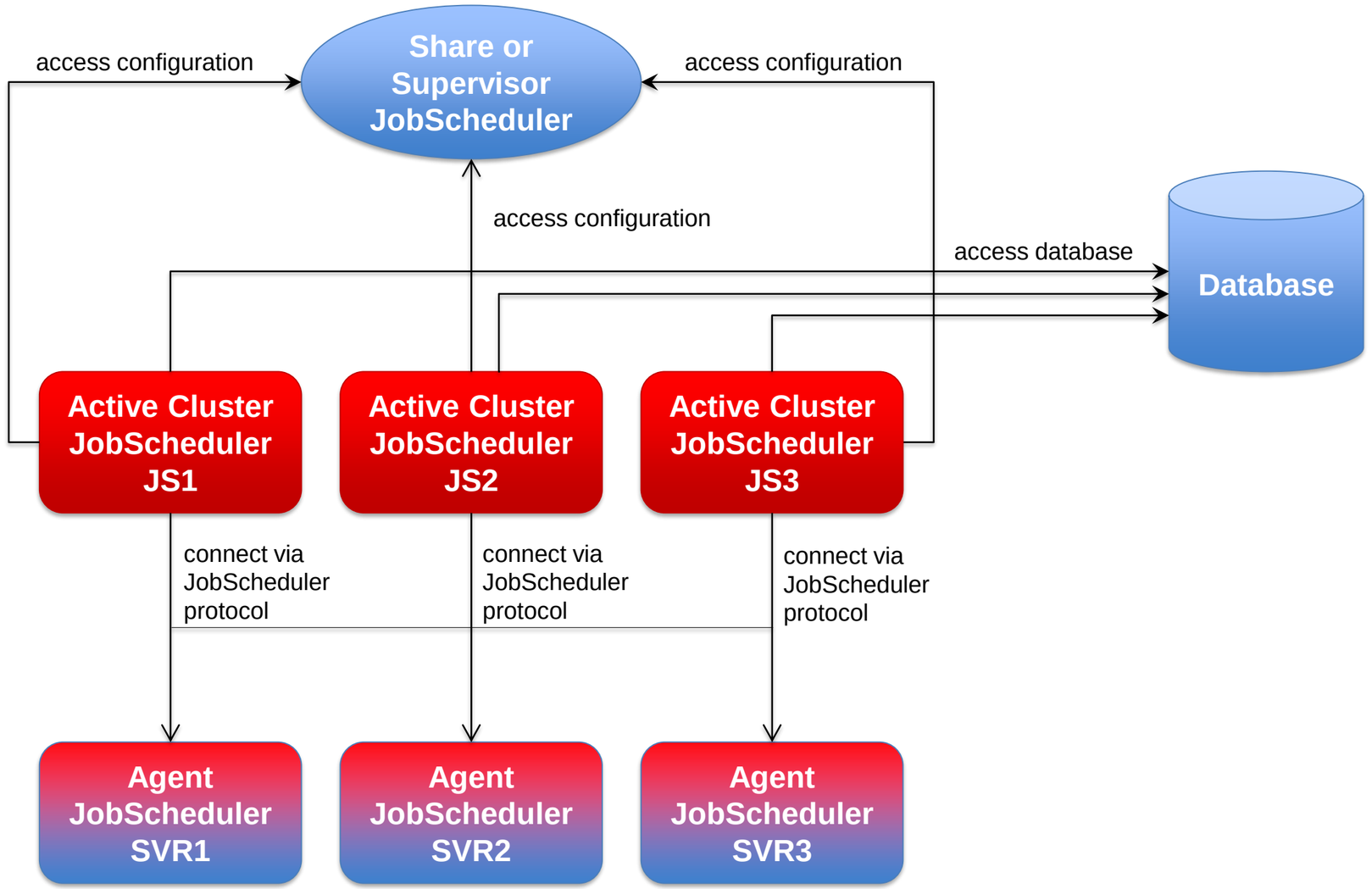
Architecture Decision Templates: Master/Agent Active Cluster

Master/Agent Active Cluster

- JobScheduler Cluster members use the same database
- JobSchedulers Cluster members share the workload of jobs
- All Instances operate in active mode
- All Cluster instances use Agents to execute jobs on remote servers

Job Execution

- Jobs are executed locally per JobScheduler Agent.
- No central resources required for job execution
- Pre-/post-processing
- Use of JITL Jobs or script files with parameter substitution



Architecture: Supervisor for Master Passive Cluster

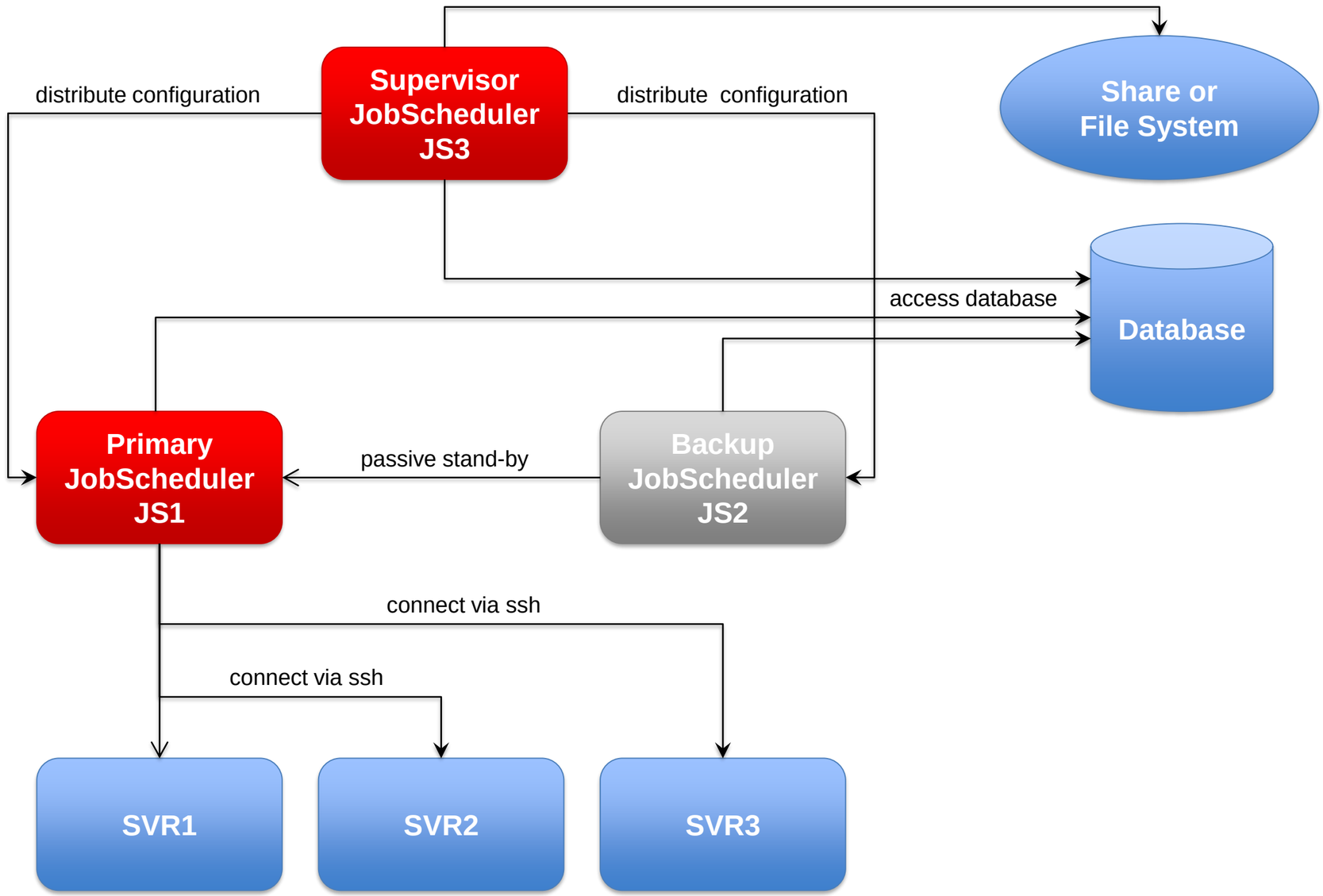
Architecture Decision Templates: Supervisor JobScheduler

Passive Cluster

- Primary and Backup JobScheduler use the same database
- Primary JobScheduler is monitored by its Backup instance
- Backup instance operates in stand-by mode
- All connections to servers use the ssh protocol

Supervisor JobScheduler

- Distribute configuration to Primary and Backup JobScheduler instances



Architecture: Supervisor for Master Active Cluster

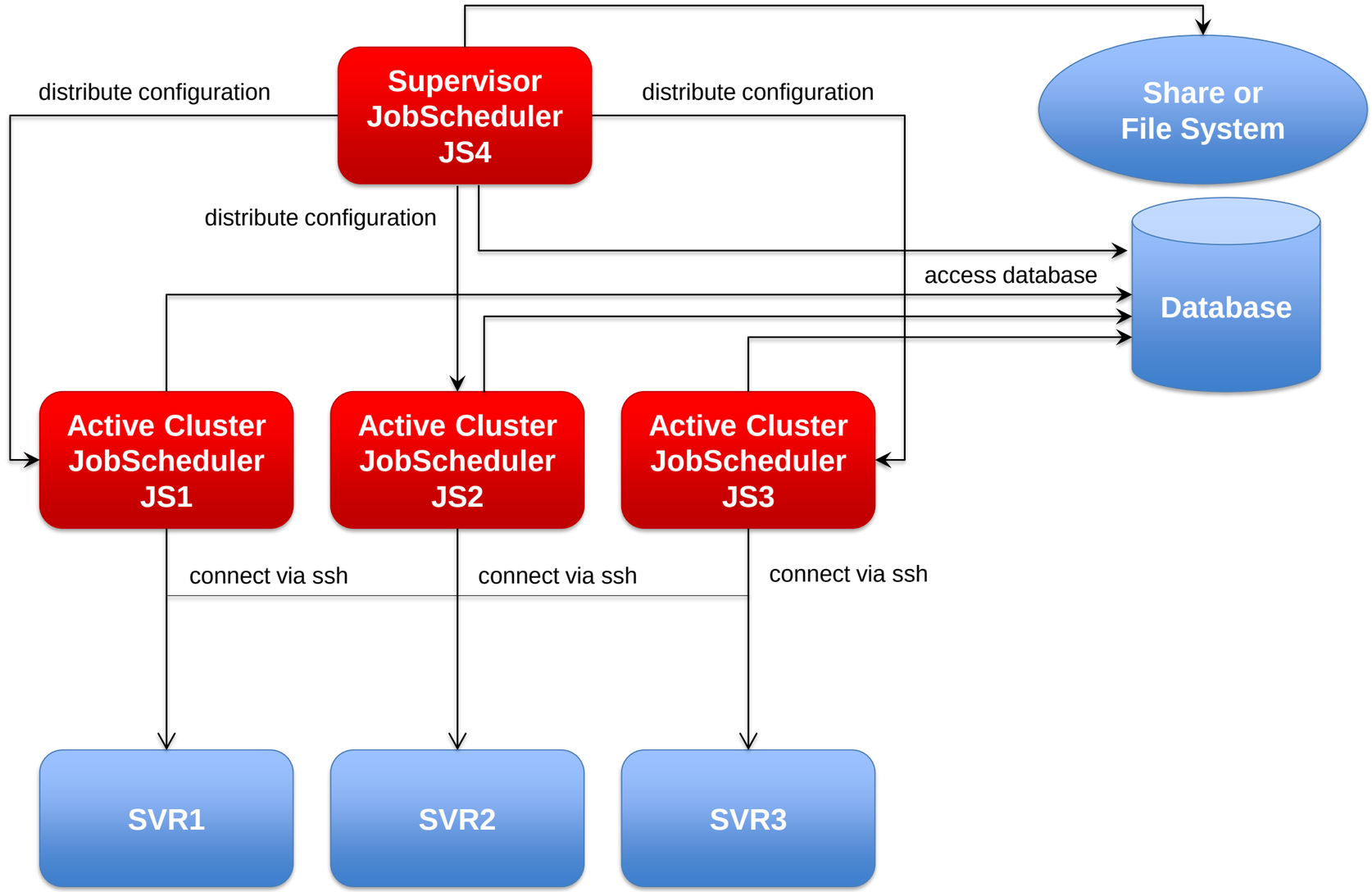
Architecture Decision Templates: Supervisor JobScheduler

Active Workload JobScheduler Cluster

- JobScheduler Cluster members use the same database
- JobScheduler Cluster members share the workload of jobs
- All Instances operate in active mode
- All connections to servers use the ssh protocol

Supervisor JobScheduler

- Distribute configuration to Cluster JobScheduler instances



Architecture: Supervisor for Master/Agent Active Cluster

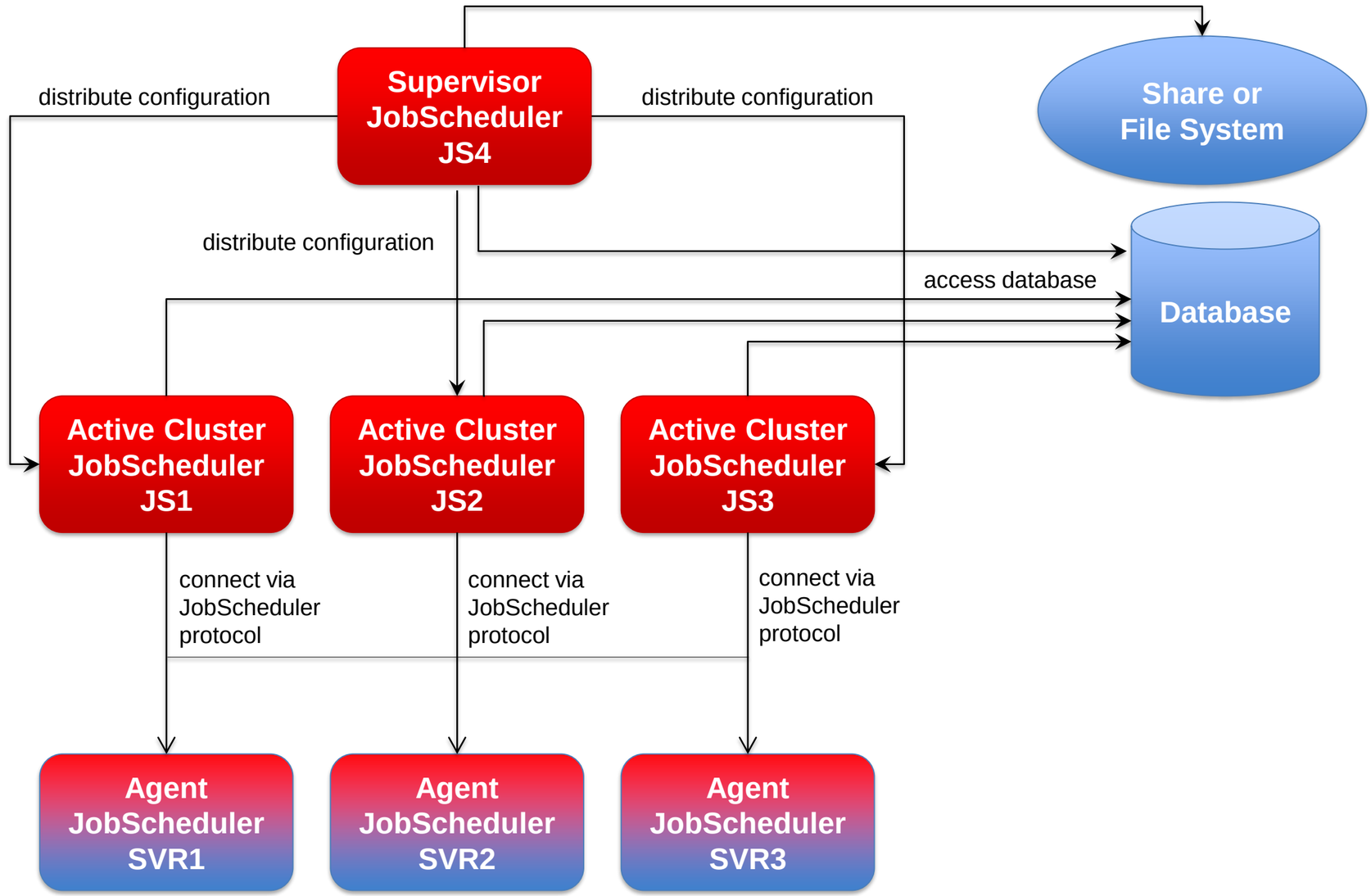
Architecture Decision Templates: Supervisor JobScheduler

Master/Agent Active Cluster

- JobScheduler Cluster members use the same database
- JobScheduler Cluster members share the workload of jobs
- All Instances operate in active mode
- All Cluster instances use Agents to execute jobs on remote servers

Supervisor JobScheduler

- Distribute configuration to Cluster JobScheduler instances



Architecture: Supervisor for Unclustered JobScheduler

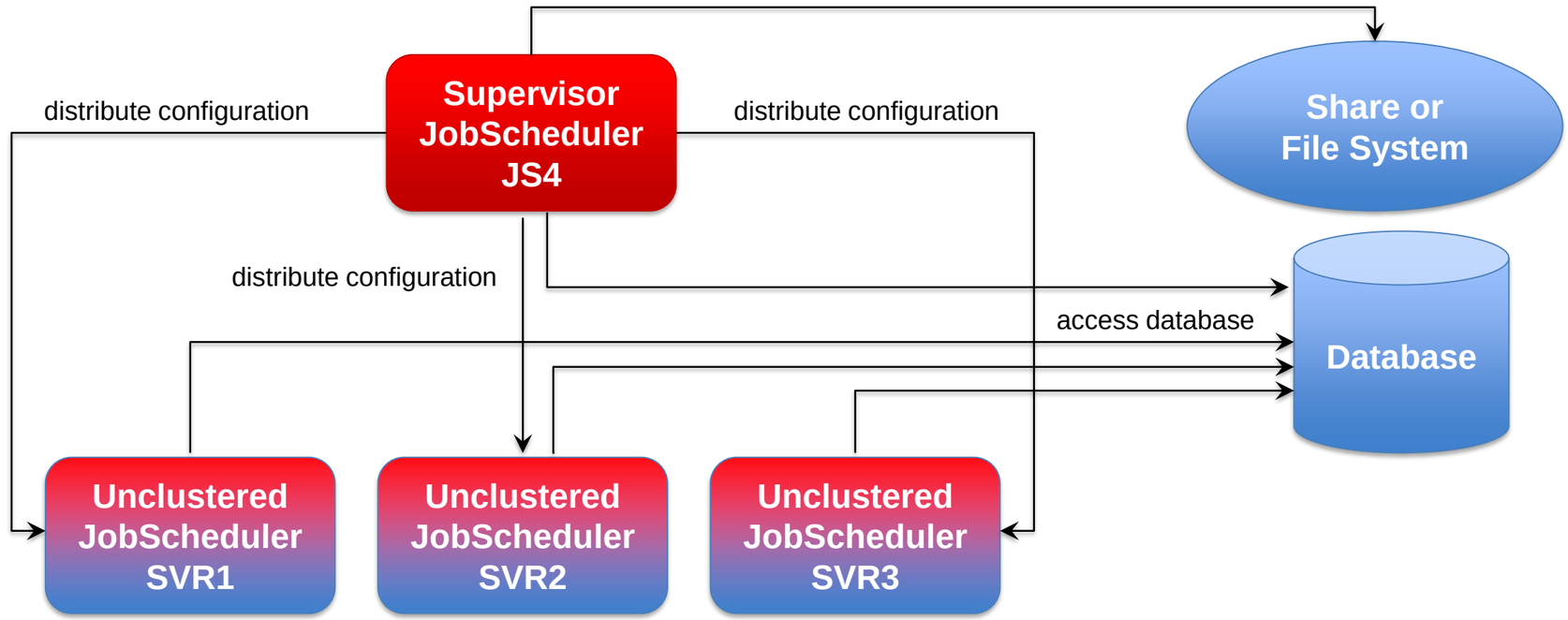
Architecture Decision Templates: Supervisor JobScheduler

Unclustered JobSchedulers

- JobSchedulers use the same database
- JobSchedulers operate independently from each other
- All Instances operate in active mode

Supervisor JobScheduler

- Distribute configuration to JobScheduler instances



Questions?

Comments?

Feedback?

Software- und
Organisations-
Service GmbH

Giesebrechtstr. 15
D-10629 Berlin

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