Topologies and Network flows

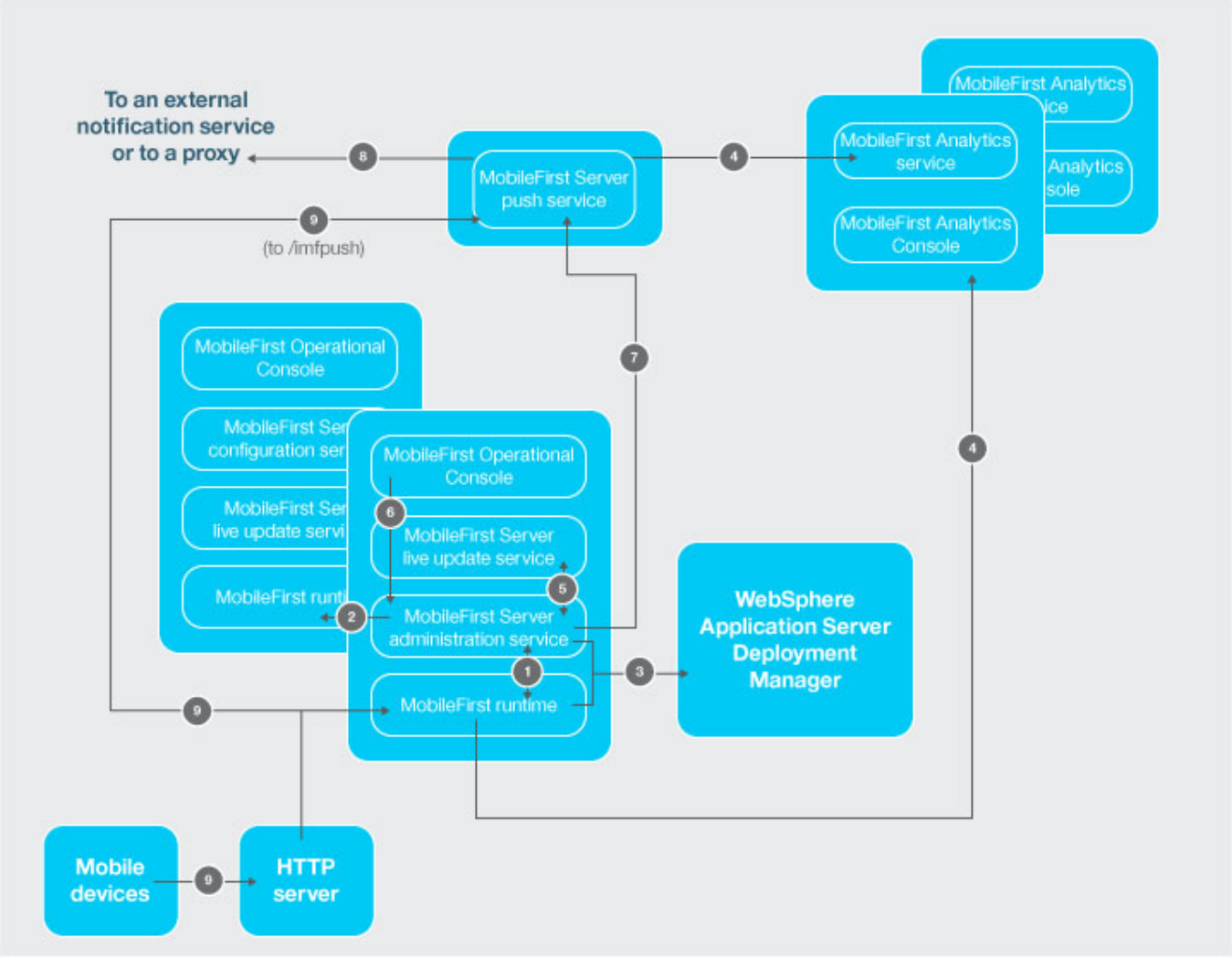
Overview

The information presented here details possible server topologies for MobileFirst Server components, as well as available network flows.
The components are deployed according to the server topology that you use. The network flows explain to you how the components communicate with one another and with the end-user devices.

Jump to

Network flows between the MobileFirst Server components

The MobileFirst Server components can communicate with each other over JMX or HTTP. You need to configure certain JNDI properties to enable the communications.
The network flows between the components and the device can be illustrated by the following image:



The flows between the various MobileFirst Server components, IBM MobileFirst Analytics, the mobile devices, and the application server are explained in the following sections:

1. MobileFirst runtime to MobileFirst Server administration service
2. MobileFirst Server administration service to MobileFirst runtime in other servers
3. MobileFirst Server administration service and MobileFirst runtime to the deployment manager on WebSphere Application Server Network Deployment
4. MobileFirst Server push service and MobileFirst runtime to MobileFirst Analytics
5. MobileFirst Server administration service to MobileFirst Server live update service
6. MobileFirst Operations Console to MobileFirst Server administration service
7. MobileFirst Server administration service to MobileFirst Server push service, and to the authorization server
8. MobileFirst Server push service to an external push notification service (outbound)
9. Mobile devices to MobileFirst runtime

MobileFirst runtime to MobileFirst Server administration service

The runtime and the administration service can communicate with each other through JMX and HTTP. This communication occurs during the initialization phase of the runtime. The runtime contacts the administration service local to its application server to get the list of the adapters and applications that it needs to serve. The communication also happens when some administration operations are run from MobileFirst Operations Console or the administration service. On WebSphere Application Server Network Deployment, the runtime can contact an administration service that is installed on another server of the cell. This enables the non-symmetric deployment (see Constraints on MobileFirst Server administration service, MobileFirst Server live update service and MobileFirst runtime). However, on all other application servers (Apache Tomcat, WebSphere Application Server Liberty, or stand-alone WebSphere Application Server), the administration service must be running on the same server as the runtime.

The protocols for JMX depend on the application server:

- Apache Tomcat - RMI
- WebSphere Application Server Liberty - HTTPS (with the REST connector)
- WebSphere Application Server - SOAP or RMI

For the communication via JMX, it is required that these protocols are available on the application server. For more information about the requirements, see Application server prerequisites (../appserver/#application-server-prerequisites).

The JMX beans of the runtime and the administration service are obtained from the application server. However, in the case of WebSphere Application Server Network Deployment, the JMX beans are obtained from the deployment manager. The deployment manager has the view of all the beans of a cell on WebSphere Application Server Network Deployment. As such, some configurations are not needed on WebSphere Application Server Network Deployment (such as the farm configuration), and non-symmetric deployment is possible on WebSphere Application Server Network Deployment. For more information, see Constraints on MobileFirst Server administration service, MobileFirst Server live update service and MobileFirst runtime.

To distinguish different installation of MobileFirst Server on the same application server or on the same WebSphere Application Server cell, you can use an environment ID, which is a JNDI variable. By default, this variable has an empty value. A runtime with a given environment ID communicates only with an administration service that has the same environment ID. For example, the administration service has an environment ID set to X, and the runtime has a different environment ID (for example, Y), then the two components do not see each other. The MobileFirst Operations Console shows no runtime available.

An administration service must be able to communicate with all the MobileFirst runtime components of a cluster. When an administration operation is run, such as uploading a new version of an adapter, or changing the active status of an application, all runtime components of the cluster must be notified about the change. If the application server is not WebSphere Application Server Network Deployment, this communication can happen only if a farm is configured. For more information, see [Constraints on MobileFirst Server administration service, MobileFirst Server live update service and MobileFirst runtime].

The runtime also communicates with the administration service through HTTP or HTTPS to download large artifacts such as the adapters. A URL is generated by the administration service and the runtime opens and outbound HTTP or HTTPS connection to request an artifact from this URL. It is possible to override the default URL generation by defining the JNDI properties (mfp.admin.proxy.port, mfp.admin.proxy.protocol, and mfp.admin.proxy.host) in the administration service. The administration service might also need to communicate with the runtime through HTTP or HTTPS to get the OAuth tokens that are used to run the push operations. For more information, see MobileFirst Server administration service to MobileFirst Server push service, and to the authorization server.

The JNDI properties that are used for the communication between the runtime and the administration service are as follows:

MobileFirst Server administration service

- JNDI properties for administration services: JMX (../server-configuration/#jndi-properties-for-administration-service-jmx)
- JNDI properties for administration services: proxies (../server-configuration/#jndi-properties-for-administration-service-proxies)
- JNDI properties for administration services: topologies (../server-configuration/#jndi-properties-for-administration-service-topologies)

MobileFirst runtime

- List of JNDI properties for MobileFirst runtime (../server-configuration/#list-of-jndi-properties-for-mobilefirst-runtime)

MobileFirst Server administration service to MobileFirst runtime in other servers

As described in MobileFirst runtime to MobileFirst Server administration service, it is required to have the communication between an administration service and all the runtime components of a cluster. When an administration operation is run, all the runtime components of a cluster can then be notified about this modification. The communication is through JMX.

On WebSphere Application Server Network Deployment, this communication can occur without any specific configuration. All the JMX MBeans that correspond to the same environment ID are obtained from the deployment manager.

For a cluster of stand-alone WebSphere Application Server, WebSphere Application Server Liberty profile, or Apache Tomcat, the communication can happen only if a farm is configured. For more information, see Installing a server farm (../appserver/#installing-a-server-farm).

MobileFirst Server administration service and MobileFirst runtime to the deployment manager on WebSphere Application Server Network Deployment

On WebSphere Application Server Network Deployment, the runtime and the administration service obtain the JMX MBeans that are used in MobileFirst runtime to MobileFirst Server administration service and MobileFirst Server administration service to MobileFirst runtime in other servers by communicating with

the deployment manager. The corresponding JNDI properties are **mfp.admin.jmx.dmgr.*** in JNDI properties for administration services: JMX (../server-configuration/#jndi-properties-for-administration-service-jmx).

The deployment manager must be running to allow the operations that require JMX communication between the runtime and the administration service. Such operations can be a runtime initialization, or the notification of a modification performed through the administration service.

MobileFirst Server push service and MobileFirst runtime to MobileFirst Analytics

The runtime sends data to MobileFirst Analytics through HTTP or HTTPS. The JNDI properties of the runtime that are used to define this communication are:

- **mfp.analytics.url** - the URL that is exposed by MobileFirst Analytics service to receive incoming analytics data from the runtime. Example: `http://<hostname>:<port>/analytics-service/rest`

  When MobileFirst Analytics is installed as a cluster, the data can be sent to any member of the cluster.

- **mfp.analytics.username** - the user name that is used to access MobileFirst Analytics service. The analytics service is protected by a security role.
- **mfp.analytics.password** - the password to access the analytics service.
- **mfp.analytics.console.url** - the URL that is passed to MobileFirst Operations Console to display a link to MobileFirst Analytics Console. Example: `http://<hostname>:<port>/analytics/console`

  The JNDI properties of the push service that are used to define this communication are:

- **mfp.push.analytics.endpoint** - the URL that is exposed by MobileFirst Analytics service to receive incoming analytics data from the push service. Example: `http://<hostname>:<port>/analytics-service/rest`

  When MobileFirst Analytics is installed as a cluster, the data can be sent to any member of the cluster.

- **mfp.push.analytics.username** - the user name that is used to access MobileFirst Analytics service. The analytics service is protected a security role.
- **mfp.push.analytics.password** - the password to access the analytics service.

MobileFirst Server administration service to MobileFirst Server live update service

The administration service communicates with the live update service to store and retrieve configuration information about the MobileFirst artifacts. The communication is performed through HTTP or HTTPS.

The URL to contact the live update service is automatically generated by the administration service. Both services must be on the same application server. The context root of the live update service must define in this way: `<adminContextRoot>config`. For example, if the context root of the administration service is

mfpadmin, then the context root of the live update service must be **mfpadminconfig**. It is possible to override the default URL generation by defining the JNDI properties (**mfp.admin.proxy.port**, **mfp.admin.proxy.protocol**, and **mfp.admin.proxy.host**) in the administration service.

The JNDI properties to configure this communication between the two services are:

- **mfp.config.service.user**
- **mfp.config.service.password**
- And those properties in JNDI properties for administration services: proxies (../server-configuration/#jndi-properties-for-administration-service-proxies).

MobileFirst Operations Console to MobileFirst Server administration service

MobileFirst Operations Console is a web user interface and acts as the front end to the administration service. It communicates with the REST services of the administration service through HTTP or HTTPS. The users who are allowed to use the console, must also be allowed to use the administration service. Each user that is mapped to a certain security role of the console must also be mapped to the same security role of the service. With this setup, the requests from the console can thus be accepted by the service.

The JNDI properties to configure this communication are in JNDI properties for the MobileFirst Operations Console (../server-configuration/#jndi-properties-for-mobilefirst-operations-console).

> Note: The **mfp.admin.endpoint** property enables the console to locate the administration service. You can use the asterisk character "*" as wildcard for specifying that the URL, generated by the console to contact the administration services, use the same value as the incoming HTTP request to the console. For example: `*://*:*/mfpadmin` means use the same protocol, host, and port as the console, but use **mfpadmin** as context root. This property is specified for the console application.

MobileFirst Server administration service to MobileFirst Server push service, and to the authorization server

The administration service communicates with the push service to request various push operations. This communication is secured through the OAuth protocol. Both services need to be registered as confidential clients. An initial registration can be performed at installation time. In this process, both services need to contact an authorization server. This authorization server can be MobileFirst runtime.

The JNDI properties of the administration service to configure this communication are:

- **mfp.admin.push.url** - the URL of the push service.
- **mfp.admin.authorization.server.url** - the URL of the MobileFirst authorization server.
- **mfp.admin.authorization.client.id** - the client ID of the administration service, as an OAuth confidential client.
- **mfp.admin.authorization.client.secret** - the secret code that is used to get the OAuth-based tokens.

> Note: The **mfp.push.authorization.client.id** and **mfp.push.authorization.client.secret** properties of the administration service can be used to register the push service automatically as a confidential client when the administration service starts. The push service must be configured with the same values.

The JNDI properties of the push service to configure this communication are:

- **mfp.push.authorization.server.url** - the URL of the MobileFirst authorization server. Same as the property **mfp.admin.authorization.server.url**.
- **mfp.push.authorization.client.id** - the client ID of the push service to contact the authorization server.
- **mfp.push.authorization.client.secret** - the secret code that is used to contact the authorization server.

MobileFirst Server push service to an external push notification service (outbound)

The push service generates outbound traffic to the external notification service such as Apple Push Notification Service (APNS) or Google Cloud Messaging (GCM). This communication can also be done through a proxy. Depending on the notification service, the following JNDI properties must be set:

- **push.apns.proxy.***
- **push.gcm.proxy.***

For more information, see List of JNDI properties for MobileFirst Server push service (../server-configuration/#list-of-jndi-properties-for-mobilefirst-server-push-service).

Mobile devices to MobileFirst runtime

The mobile devices contact the runtime. The security of this communication is determined by the configuration of the application and the adapters that are requested. For more information, see MobileFirst security framework (../../../authentication-and-security).

Constraints on the MobileFirst Server components and MobileFirst Analytics

Understand the constraints on the various MobileFirst Server components and MobileFirst Analytics before you decide your server topology.

- Constraints on MobileFirst Server administration service, MobileFirst Server live update service and MobileFirst runtime
- Constraints on MobileFirst Server push service

Constraints on MobileFirst Server administration service, MobileFirst Server live update service and MobileFirst runtime

Find out the constraints and the deployment mode of the administration service, live update service, and the runtime per server topology.

The live update service must be always installed with the administration service on the same application server as explained in MobileFirst Server administration service to MobileFirst Server live update service. The context root of the live update service must define in this way: `/<adminContextRoot>config`. For example, if the context root of the administration service is /**mfpadmin**, then the context root of the live update service must be /**mfpadminconfig**.

You can use the following topologies of application servers:

- Stand-alone server: WebSphere Application Server Liberty profile, Apache Tomcat, or WebSphere Application Server full profile

- Server farm: WebSphere Application Server Liberty profile, Apache Tomcat, or WebSphere Application Server full profile
- WebSphere Application Server Network Deployment cell
- Liberty collective

Modes of deployment

Depending on the application server topology that you use, you have two modes of deployment choice for deploying the administration service, the live update service and the runtime in the application server infrastructure. In asymmetric deployment, you can install the runtimes on different application servers from the administration and the live update services.

Symmetric deployment
In symmetrical deployment, you must install the MobileFirst administration components (MobileFirst Operations Console, the administration service, and the live update service applications) and the runtime on the same application server.

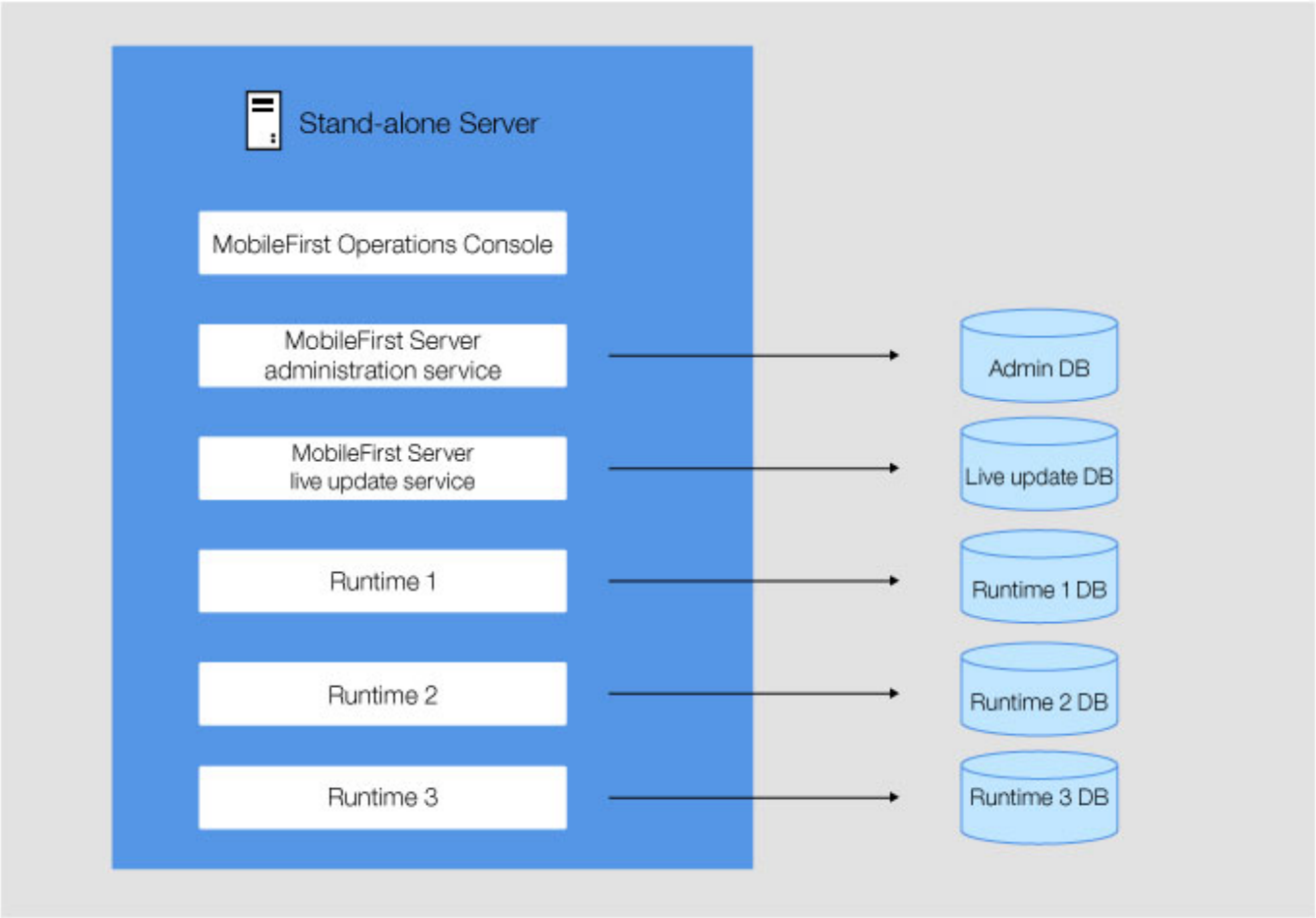
Asymmetric deployment
In asymmetric deployment, you can install the runtimes on different application servers from the MobileFirst administration components.
Asymmetric deployment is only supported for WebSphere Application Server Network Deployment cell topology and for Liberty collective topology.

Select a topology

- Stand-alone server topology
- Server farm topology
- Liberty collective topology
- WebSphere Application Server Network Deployment topologies
- Using a reverse proxy with server farm and WebSphere Application Server Network Deployment topologies

Stand-alone server topology

You can configure a stand-alone topology for WebSphere Application Server full profile, WebSphere Application Server Liberty profile, and Apache Tomcat. In this topology, all the administration components and the runtimes are deployed in a single Java Virtual Machine (JVM).



With one JVM, only symmetric deployment is possible with the following characteristics:

- One or several administration components can be deployed. Each MobileFirst Operations Console communicates with one administration service and one live update service.
- One or several runtimes can be deployed.
- One MobileFirst Operations Console can manage several runtimes.
- One runtime is managed by only one MobileFirst Operations Console.
- Each administration service uses its own administration database schema.
- Each live update service uses its own live update database schema.
- Each runtime uses its own runtime database schema.

Configuration of JNDI properties

Some JNDI properties are required to enable Java Management Extensions (JMX) communication between the administration service and the runtime, and to define the administration service that manages a runtime. For details about these properties, see List of JNDI properties for MobileFirst Server administration service (../server-configuration/#list-of-jndi-properties-for-mobilefirst-server-administration-service) and List of JNDI properties for MobileFirst runtime (../server-configuration/#list-of-jndi-properties-for-mobilefirst-runtime).

Stand-alone WebSphere Application Server Liberty profile server
The following global JNDI properties are required for the administration services and the runtimes.

JNDI properties	Values
mfp.topology.platform	Liberty
mfp.topology.clustermode	Standalone

JNDI properties	Values
mfp.admin.jmx.host	The host name of the WebSphere Application Server Liberty profile server.
mfp.admin.jmx.port	The port of the REST connector that is the port of the httpsPort attribute declared in the element of the server.xml file of WebSphere Application Server Liberty profile server. This property has no default value.
mfp.admin.jmx.user	The user name of the WebSphere Application Server Liberty administrator, which must be identical to the name defined in the element of the server.xml file of the WebSphere Application Server Liberty profile server.
mfp.admin.jmx.pwd	The password of the WebSphere Application Server Liberty administrator user.

Several administration components can be deployed to enable the same JVM to run on separate administration components that manage different runtimes.

When you deploy several administration components, you must specify:

- On each administration service, a unique value for the local **mfp.admin.environmentid** JNDI property.
- On each runtime, the same value for the local **mfp.admin.environmentid** JNDI property as the value defined for the administration service that manages the runtime.

Stand-alone Apache Tomcat server The following local JNDI properties are required for the administration services and the runtimes.

JNDI properties	Values
mfp.topology.platform	Tomcat
mfp.topology.clustermode	Standalone

JVM properties are also required to define Java Management Extensions (JMX) Remote Method Invocation (RMI). For more information, see Configuring JMX connection for Apache Tomcat (../appserver/#apache-tomcat-prerequisites).

If the Apache Tomcat server is running behind a firewall, the **mfp.admin.rmi.registryPort** and **mfp.admin.rmi.serverPort** JNDI properties are required for the administration service. See Configuring JMX connection for Apache Tomcat (../appserver/#apache-tomcat-prerequisites).

Several administration components can be deployed to enable the same JVM to run on separate administration components that manage different runtimes.
When you deploy several administration components, you must specify:

- On each administration service, a unique value for the local mfp.admin.environmentid JNDI property.
- On each runtime, the same value for the local mfp.admin.environmentid JNDI property as the value defined for the administration service that manages the runtime.

Stand-alone WebSphere Application Server
The following local JNDI properties are required for the administration services and the runtimes.

JNDI properties	Values
mfp.topology.platform	WAS
mfp.topology.clustermode	Standalone
mfp.admin.jmx.connector	The JMX connector type; the value can be SOAP or RMI.

Several administration components can be deployed to enable the same JVM to run on separate administration components that manage different runtimes.
When you deploy several administration components, you must specify:

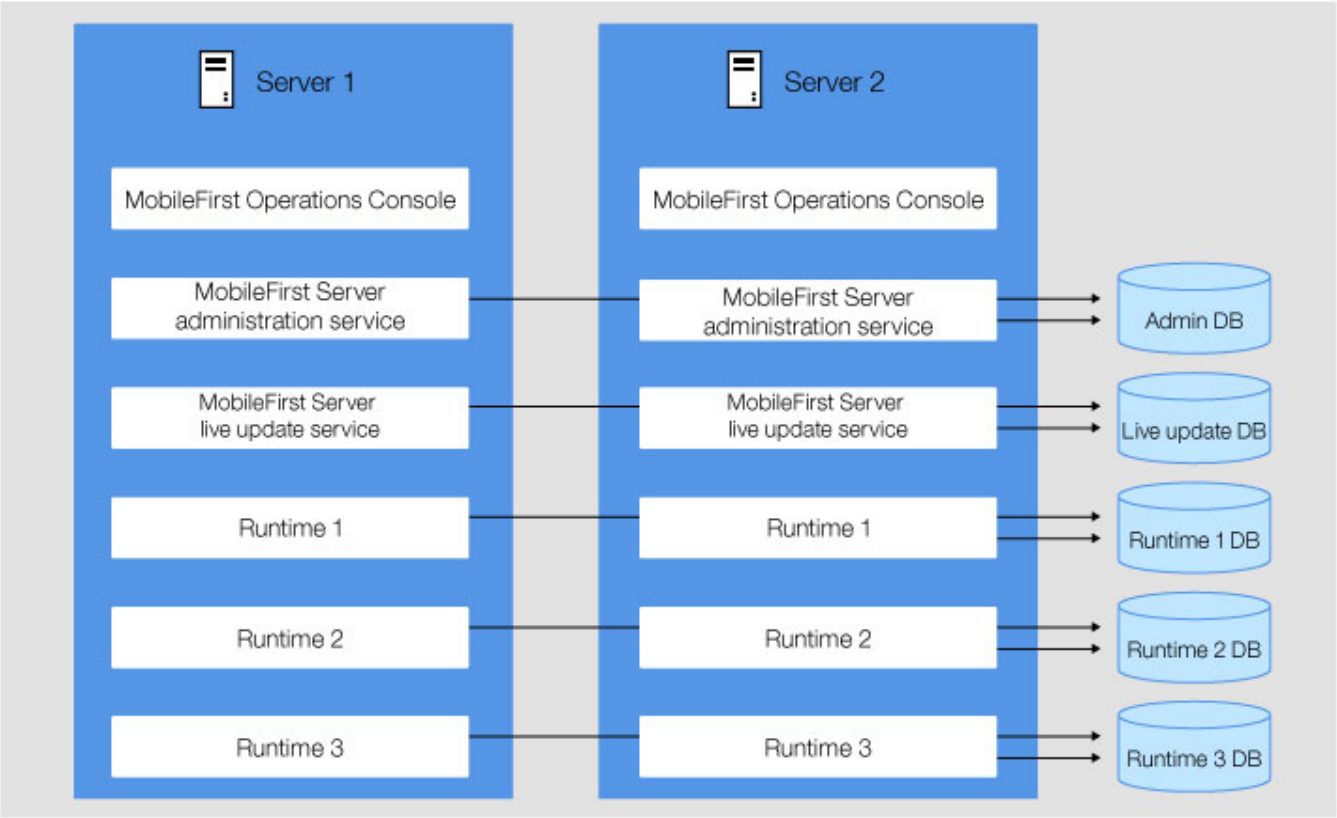
- On each administration service, a unique value for the **local mfp.admin.environmentid** JNDI property.
- On each runtime, the same value for the local **mfp.admin.environmentid** JNDI property as the value defined for the administration service that manages the runtime.

Server farm topology

You can configure a farm of WebSphere Application Server full profile, WebSphere Application Server Liberty profile, or Apache Tomcat application servers.

A farm is a set of individual servers where the same components are deployed and where the same administration service database and runtime database are shared between the servers. The farm topology enables the load of MobileFirst applications to be distributed across several servers. Each server in the farm must be a Java virtual machine (JVM) of the same type of application server; that is, a homogeneous server farm. For example, a set of several Liberty servers can be configured as a server farm. Conversely, a mix of Liberty server, Tomcat server, or stand-alone WebSphere Application Server cannot be configured as a server farm.

In this topology, all the administration components (MobileFirst Operations Console, the administration service, and the live update service) and the runtimes are deployed on every server in the farm.



This topology supports only symmetric deployment. The runtimes and the administration components must be deployed on every server in the farm. The deployment of this topology has the following characteristics:

- One or several administration components can be deployed. Each instance of MobileFirst Operations Console communicates with one administration service and one live update service.
- The administration components must be deployed on all servers in the farm.
- One or several runtimes can be deployed.
- The runtimes must be deployed on all servers in the farm.
- One MobileFirst Operations Console can manage several runtimes.
- One runtime is managed by only one MobileFirst Operations Console.
- Each administration service uses its own administration database schema. All deployed instances of the same administration service share the same administration database schema.
- Each live update service uses its own live update database schema. All deployed instances of the same live update service share the same live update database schema.
- Each runtime uses its own runtime database schema. All deployed instances of the same runtime share the same runtime database schema.

Configuration of JNDI properties

Some JNDI properties are required to enable JMX communication between the administration service and the runtime of the same server, and to define the administration service that manages a runtime. For convenience, the following tables list these properties. For instructions about how to install a server farm, see Installing a server farm (../appserver/#installing-a-server-farm). For more information about the JNDI properties, see List of JNDI properties for MobileFirst Server administration service (../server-configuration/#list-of-jndi-properties-for-mobilefirst-server-administration-service) and List of JNDI properties for MobileFirst runtime (../server-configuration/#list-of-jndi-properties-for-mobilefirst-runtime).

WebSphere Application Server Liberty profile server farm
The following global JNDI properties are required in each server of the farm for the administration services and the runtimes.

JNDI properties	Values
mfp.topology.platform	Liberty
mfp.topology.clustermode	Farm
mfp.admin.jmx.host	The host name of the WebSphere Application Server Liberty profile server
mfp.admin.jmx.port	The port of the REST connector that must be identical to the value of the httpsPort attribute declared in the element of the **server.xml** file of the WebSphere Application Server Liberty profile server. `<httpEndpoint id="defaultHttpEndpoint" httpPort="9080" httpsPort="9443" host="*" />`

mfp.admin.jmx.user	The user name of the WebSphere Application Server Liberty administrator that is defined in the element of the **server.xml** file of the WebSphere Application Server Liberty profile server.
	`<administrator-role>`
	`  <user>MfpRESTUser</user>`
	`</administrator-role>`

mfp.admin.jmx.pwd	The password of the WebSphere Application Server Liberty administrator user.

The **mfp.admin.serverid** JNDI property is required for the administration service to manage the server farm configuration. Its value is the server identifier, which must be different for each server in the farm.

Several administration components can be deployed to enable the same JVM to run on separate administration components that manage different runtimes.

When you deploy several administration components, you must specify:

- On each administration service, a unique value for the local mfp.admin.environmentid JNDI property.
- On each runtime, the same value for the local **mfp.admin.environmentid** JNDI property as the value defined for the administration service that manages the runtime.

Apache Tomcat server farm
The following global JNDI properties are required in each server of the farm for the administration services and the runtimes.

JNDI properties	Values
mfp.topology.platform	Tomcat
mfp.topology.clustermode	Farm

JVM properties are also required to define Java Management Extensions (JMX) Remote Method Invocation (RMI). For more information, see Configuring JMX connection for Apache Tomcat (../appserver/#apache-tomcat-prerequisites).

The **mfp.admin.serverid** JNDI property is required for the administration service to manage the server farm configuration. Its value is the server identifier, which must be different for each server in the farm.

Several administration components can be deployed to enable the same JVM to run on separate administration components that manage different runtimes.

When you deploy several administration components, you must specify:

- On each administration service, a unique value for the local mfp.admin.environmentid JNDI property.
- On each runtime, the same value for the local **mfp.admin.environmentid** JNDI property as the value defined for the administration service that manages the runtime.

WebSphere Application Server full profile server farm
The following global JNDI properties are required on each server in the farm for the administration services and the runtimes.

JNDI properties	Values
mfp.topology.platform WAS	WAS
mfp.topology.clustermode	Farm
mfp.admin.jmx.connector	SOAP

The following JNDI properties are required for the administration service to manage the server farm configuration.

JNDI properties	Values
mfp.admin.jmx.user	The user name of WebSphere Application Server. This user must be defined in the WebSphere Application Server user registry.
mfp.admin.jmx.pwd	The password of the WebSphere Application Server user.
mfp.admin.serverid	The server identifier, which must be different for each server in the farm and identical to the value of this property used for this server in the server farm configuration file.

Several administration components can be deployed to enable the same JVM to run on separate administration components that manage different runtimes.

When you deploy several administration components, you must specify the following values:

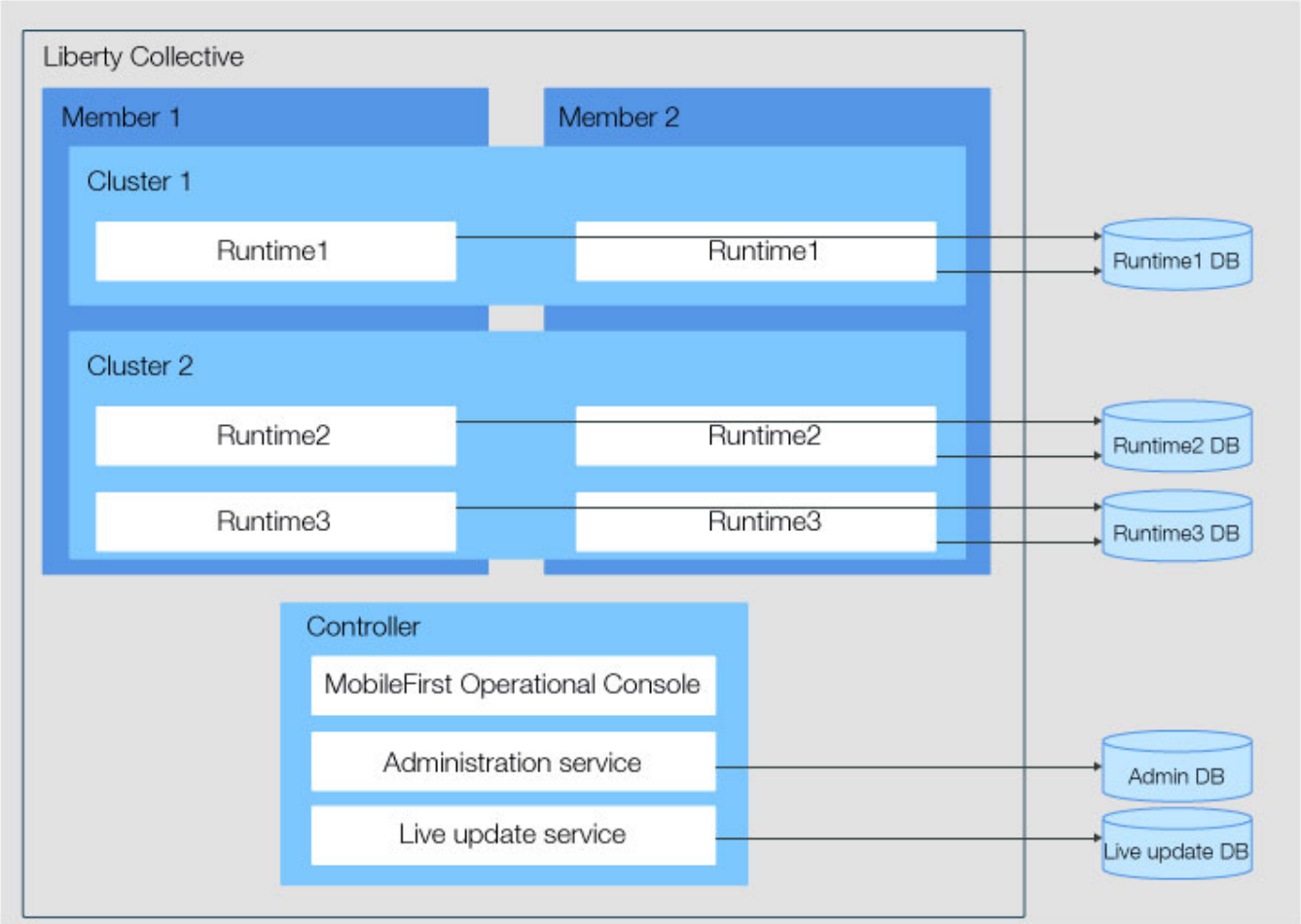
- On each administration service, a unique value for the local **mfp.admin.environmentid** JNDI property.
- On each runtime, the same value for the local **mfp.admin.environmentid** JNDI property as the value defined for the administration service that manages the runtime.

Liberty collective topology

You can deploy the MobileFirst Server components in a Liberty collective topology.

In the Liberty collective topology, the MobileFirst Server administration components (MobileFirst Operations Console, the administration service, and the live update service) are deployed in a collective

controller and the MobileFirst runtimes in collective member. This topology supports only asymmetric deployment, the runtimes cannot be deployed in a collective controller.



The deployment of this topology has the following characteristics:

- One or several administration components can be deployed in one or several controllers of the collective. Each instance of * * MobileFirst Operations Console communicates with one administration service and one live update service.
- One or several runtimes can be deployed in the cluster members of the collective.
- One MobileFirst Operations Console manages several runtimes that are deployed in the cluster members of the collective.
- One runtime is managed by only one MobileFirst Operations Console.
- Each administration service uses its own administration database schema.
- Each live update service uses its own live update database schema.
- Each runtime uses its own runtime database schema.

Configuration of JNDI properties

The following tables list the JNDI properties are required to enable JMX communication between the administration service and the runtime, and to define the administration service that manages a runtime. For more information about these properties, see List of JNDI properties for MobileFirst Server administration service (../server-configuration/#list-of-jndi-properties-for-mobilefirst-server-administration-service) and List of JNDI properties for MobileFirst runtime (../server-configuration/#list-of-jndi-properties-for-mobilefirst-runtime). For instructions about how to install a Liberty collective manually, see Manual installation on WebSphere Application Server Liberty collective (../appserver/#manual-installation-on-websphere-application-server-liberty-collective).

The following global JNDI properties are required for the administration services:

JNDI properties	Values
mfp.topology.platform	Liberty
mfp.topology.clustermode	Cluster
mfp.admin.serverid	controller
mfp.admin.jmx.host	The host name of the Liberty controller.
mfp.admin.jmx.port	The port of the REST connector that must be identical to the value of the **httpsPort** attribute declared in the element of the server.xml file of the Liberty controller. **<httpEndpoint** id="defaultHttpEndpoint" httpPort="9080" httpsPort="944 3" host="*"**/>**
mfp.admin.jmx.user	The user name of the controller administrator that is defined in the element of the **server.xml** file of the Liberty controller. **<administrator-role> <user>**MfpRESTUser**</user> </administrator-ro le>**
mfp.admin.jmx.pwd	The password of the Liberty controller administrator user.

Several administration components can be deployed to enable the controller to run separate administration components that manage different runtimes.

When you deploy several administration components, you must specify on each administration service, a unique value for the local **mfp.admin.environmentid** JNDI property.

The following global JNDI properties are required for the runtimes:

JNDI properties	Values
mfp.topology.platform	Liberty
mfp.topology.clustermode	Cluster
mfp.admin.serverid	A value that identifies uniquely the collective member. It must be different for each member in the collective. The value `controller` cannot be used as it is reserved for the collective controller.
mfp.admin.jmx.host	The host name of the Liberty controller.
mfp.admin.jmx.port	The port of the REST connector that must be identical to the value of the **httpsPort** attribute declared in the element of the server.xml file of the Liberty controller. **<httpEndpoint** id="defaultHttpEndpoint" httpPort="9080" httpsPort="944 3" host="*"**/>**

mfp.admin.jmx.user	The user name of the controller administrator that is defined in the element of the **server.xml** file of the Liberty controller. **<administrator-role> <user>**MfpRESTUser**</user> </administrator-role>**
mfp.admin.jmx.pwd	The password of the Liberty controller administrator user.

The following JNDI property is required for the runtime when several controllers (replicas) using the same administration components are used:

JNDI properties	Values
mfp.admin.jmx.replica	Endpoint list of the different controller replicas with the following syntax: `replica-1 hostname:replica-1 port, replica-2 hostname:replica-2 port,..., replica-n hostname:replica-n port`

When several administration components are deployed in the controller, each runtime must have the same value for the local **mfp.admin.environmentid** JNDI property as the value that is defined for the administration service that manages the runtime.

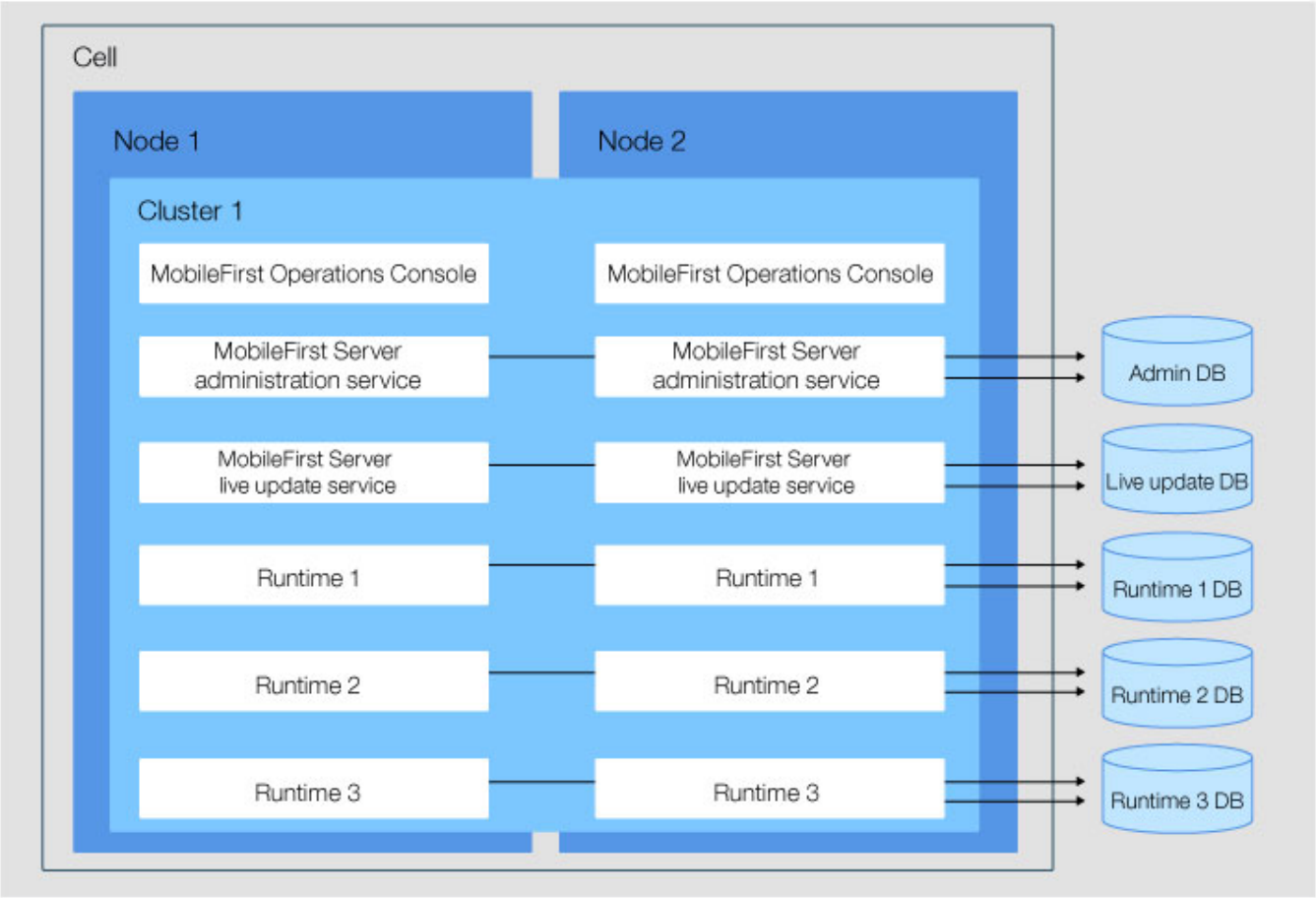
WebSphere Application Server Network Deployment topologies

The administration components and the runtimes are deployed in servers or clusters of the WebSphere Application Server Network Deployment cell.

Examples of these topologies support either asymmetric or symmetric deployment, or both. You can, for example, deploy the administration components (MobileFirst Operations Console, the administration service, and the live update service) in one cluster and the runtimes managed by these components in another cluster.

Symmetric deployment in the same server or cluster

The diagram below shows symmetric deployment where the runtimes and the administration components are deployed in the same server or cluster.

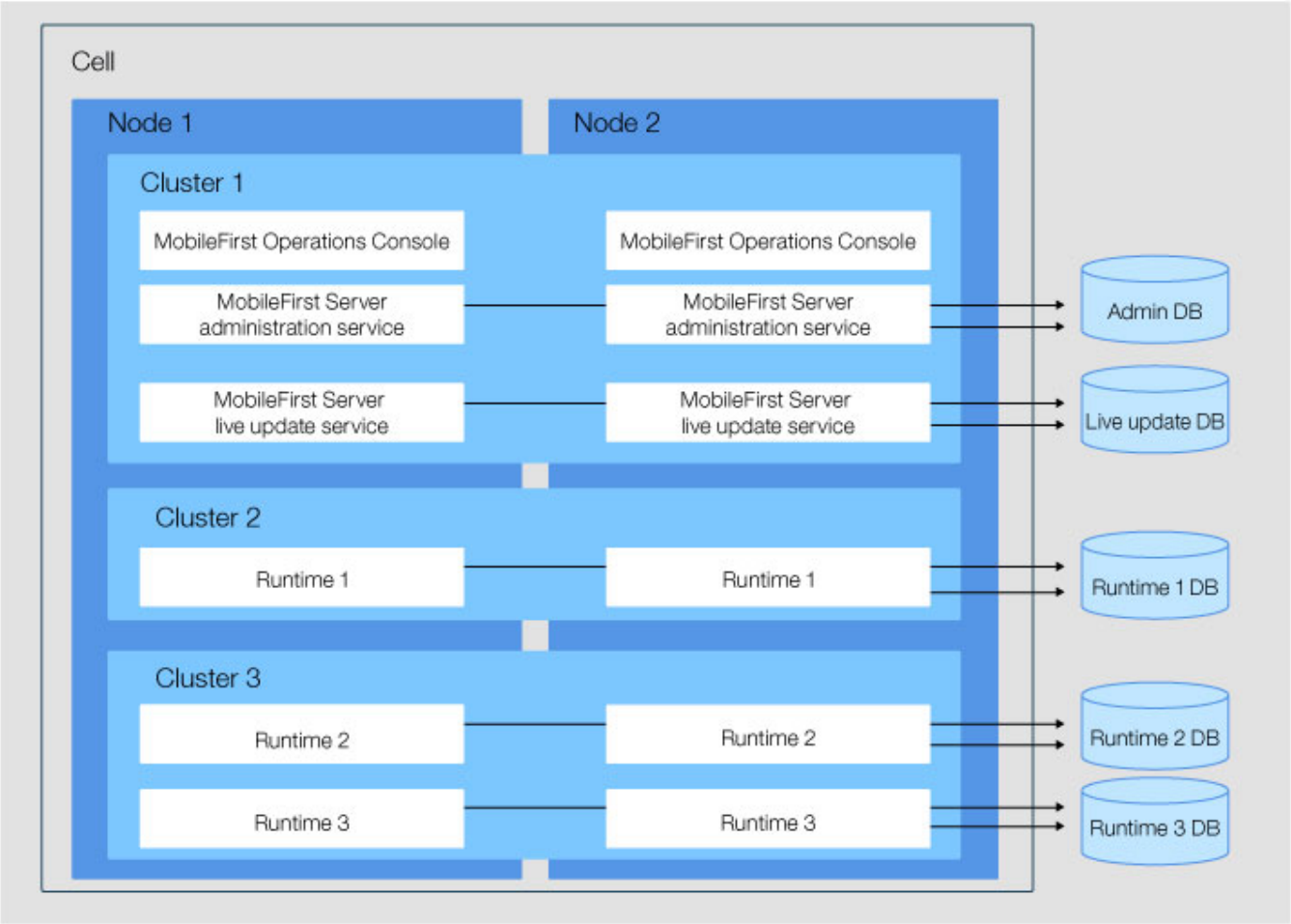


The deployment of this topology has the following characteristics:

- One or several administration components can be deployed in one or several servers or clusters of the cell. Each instance of * MobileFirst Operations Console communicates with one administration service and one live update service.
- One or several runtimes can be deployed in the same server or cluster as the administration components that manage them.
- One runtime is managed by only one MobileFirst Operations Console.
- Each administration service uses its own administration database schema.
- Each live update service uses its own live update database schema.
- Each runtime uses its own runtime database schema.

Asymmetric deployment with runtimes and administration services in different server or cluster

The diagram below shows a topology where the runtimes are deployed in a different server or cluster from the administration services.



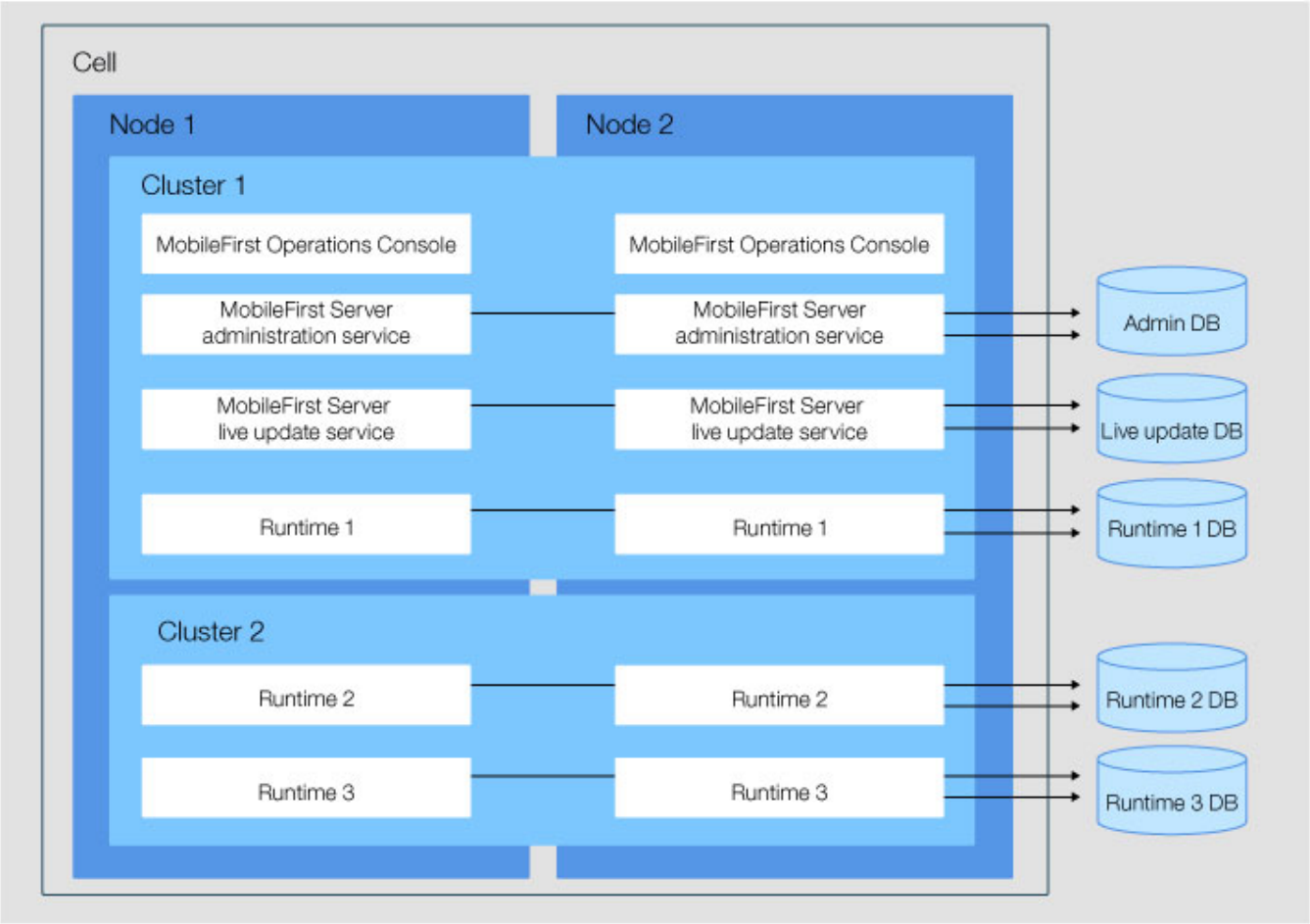
The deployment of this topology has the following characteristics:

- One or several administration components can be deployed in one or several servers or clusters of the cell. Each instance of * MobileFirst Operations Console communicates with one administration service and one live update service.
- One or several runtimes can be deployed in other servers or clusters of the cell.
- One MobileFirst Operations Console manages several runtimes deployed in the other servers or clusters of the cell.
- One runtime is managed by only one MobileFirst Operations Console.
- Each administration service uses its own administration database schema.
- Each live update service uses its own live update database schema.
- Each runtime uses its own runtime database schema.

This topology is advantageous, because it enables the runtimes to be isolated from the administration components and from other runtimes. It can be used to provide performance isolation, to isolate critical applications, and to enforce Service Level Agreement (SLA).

Symmetric and asymmetric deployment

The diagram below shows an example of symmetric deployment in Cluster1 and of asymmetric deployment in Cluster2, where Runtime2 and Runtime3 are deployed in a different cluster from the administration components. MobileFirst Operations Console manages the runtimes deployed in Cluster1 and Cluster2.



The deployment of this topology has the following characteristics:

- One or several administration components can be deployed in one or several servers or clusters of the cell. Each instance of * MobileFirst Operations Console communicates with one administration service and one live update service.
- One or several runtimes can be deployed in one or several servers or clusters of the cell.
- One MobileFirst Operations Console can manage several runtimes deployed in the same or other servers or clusters of the cell.
- One runtime is managed by only one MobileFirst Operations Console.
- Each administration service uses its own administration database schema.
- Each live update service uses its own live update database schema.
- Each runtime uses its own runtime database schema.

Configuration of JNDI properties

Some JNDI properties are required to enable JMX communication between the administration service and the runtime, and to define the administration service that manages a runtime. For details about these properties, see List of JNDI properties for MobileFirst Server administration service (../server-configuration/#list-of-jndi-properties-for-mobilefirst-server-administration-service) and List of JNDI properties for MobileFirst runtime (../server-configuration/#list-of-jndi-properties-for-mobilefirst-runtime).

The following local JNDI properties are required for the administration services and for the runtimes:

JNDI properties	Values
mfp.topology.platform	WAS
mfp.topology.clustermode	Cluster

JNDI properties	Values
mfp.admin.jmx.connector	The JMX connector type to connect with the deployment manager. The value can be SOAP or RMI. SOAP is the default and preferred value. You must use RMI if the SOAP port is disabled.
mfp.admin.jmx.dmgr.host	The host name of the deployment manager.
mfp.admin.jmx.dmgr.port	The RMI or the SOAP port used by the deployment manager, depending on the value of mfp.admin.jmx.connector.

Several administration components can be deployed to enable you to run the same server or cluster with separate administration components managing each of the different runtimes.

When several administration components are deployed, you must specify:

- On each administration service, a unique value for the local **mfp.admin.environmentid** JNDI property.
- On each runtime, the same value for the local **mfp.admin.environmentid** as the value defined for the administration service that manages that runtime.

If the virtual host that is mapped to an administration service application is not the default host, you must set the following properties on the administration service application:

- **mfp.admin.jmx.user**: the user name of the WebSphere Application Server administrator
- **mfp.admin.jmx.pwd**: the password of the WebSphere Application Server administrator

Using a reverse proxy with server farm and WebSphere Application Server Network Deployment topologies

You can use a reverse proxy with distributed topologies. If your topology uses a reverse proxy, configure the required JNDI properties for the administration service.

You can use a reverse proxy, such as IBM HTTP Server, to front server farm or WebSphere Application Server Network Deployment topologies. In this case, you must configure the administration components appropriately.

You can call the reverse proxy from:

- The browser when you access MobileFirst Operations Console.
- The runtime when it calls the administration service.
- The MobileFirst Operations Console component when it calls the administration service.

If the reverse proxy is in a DMZ (a firewall configuration for securing local area networks) and a firewall is used between the DMZ and the internal network, this firewall must authorize all incoming requests from the application servers.

When a reverse proxy is used in front of the application server infrastructure, the following JNDI properties must be defined for the administration service.

JNDI properties	Values
mfp.admin.proxy.protocol	The protocol that is used to communicate with the reverse proxy. It can be HTTP or HTTPS.

JNDI properties	Values
mfp.admin.proxy.host	The host name of the reverse proxy.
mfp.admin.proxy.port	The port number of the reverse proxy.

The **mfp.admin.endpoint** property that references the URL of the reverse proxy is also required for MobileFirst Operations Console.

Constraints on MobileFirst Server push service

The push service can be on the same application server as the administration service or the runtime, or can be on a different application server. The URL used by the client apps to contact the push service is the same as the URL used by the client apps to contact the runtime, excepted that the context root of the runtime is replaced by imfpush. If you install the push service on a different server than the runtime, your HTTP server must direct the traffic to the /imfpush context root to a server where the push service runs.

For more information about the JNDI properties that are needed to adapt the installation to a topology, see MobileFirst Server administration service to MobileFirst Server push service, and to the authorization server. The push service must be installed with the context root **/imfpush**.

Multiple MobileFirst runtimes

You can install multiple runtimes. Each runtime must have its own context root, and all of these runtimes are managed by the same MobileFirst Server administration service and MobileFirst Operations Console.

The constraints as described in Constraints on MobileFirst Server administration service, MobileFirst Server live update service and MobileFirst runtime applies. Each runtime (with its context root) must have its own database tables.

> For instructions, see Configuring multiple runtimes (../server-configuration/#configuring-multiple-runtimes).

Multiple instances of MobileFirst Server on the same server or WebSphere Application Server cell

By defining a common environment ID, multiple instances of MobileFirst Server are possible to be installed on the same server.

You can install multiple instances of MobileFirst Server administration service, MobileFirst Server live update service, and MobileFirst runtime on the same application server or WebSphere Application Server cell. However, you must distinguish their installations with the JNDI variable: **mfp.admin.environmentid**, which is a variable of the administration service and of the runtime. The administration service manages only the runtimes that have the same environment identifier. As such, only the runtime components and the administration service that have the same value for **mfp.admin.environmentid** are considered as part of the same installation.

Last modified on