Cluster and cache operation

This page describes how a Webdesk with several nodes (cluster, "distributed operation") is configured and operated, and how the caches behave in this setup. It is intended for system administrators and consultants. Prerequisites: all nodes access the same database, a reachable Redis server and, for cache replication via JMS, a JMS broker. The keys are set per node in webdesk.properties (see page "Konfigurationsdatei webdesk.properties und Umgebungsvariablen"). In addition, List of caches[1] describes the purpose of the caches, Active Jobs[2] describes job monitoring including node status, and Extended Tasks[3] describes the maintenance actions.

Overview: building blocks of cluster operation

Building block	Purpose	Switch
Cache replication (EhCache)	Synchronizes the EhCache caches between the nodes and transports service calls to the other nodes (saving jobs and job triggers, applying system parameters, publishing the cluster status). Transport is either RMI (multicast) or JMS.	isDistributed, useJmsReplication
Redis	Shared store and message channel (Pub/Sub) for the application caches defined as "replicated" (Caffeine + Redis), for the Hibernate provider redis-caffeine and for the HTTP sessions (Spring Session). The Redis connection is set up when isDistributed=true or redis.useAsSessionStore=true applies.	redis.*
Hibernate cache (second-level cache)	Cache for database entities. The provider can be switched globally.	webdesk.hibernate.cache.provider
HTTP sessions	Sessions held locally in memory, in Redis (Spring Session) or via the Tomcat session manager.	redis.useAsSessionStore, META-INF/context.xml
Jobs	Assignment of job triggers to a node.	PoOptions.clusterNodes, trigger field "Run on cluster node"

With isDistributed=false (default), all coordination between nodes is switched off; the caches then work purely locally. A useJmsReplication that is set is ignored without isDistributed=true.

Basic configuration of a node

Property	Description	Possible value(s)
isDistributed	Switches on distributed operation: cache replication, Redis connection for replicated caches, cluster heartbeat and a node-specific workflow engine name.	true/false (default: false)

	Must be set identically on all nodes.	
useJmsReplication	If true, the EhCache caches are replicated via JMS, otherwise via RMI with multicast discovery. Only effective with isDistributed=true. Must be set identically on all nodes.	true/false (default: false, i.e. RMI)
PoOptions.clusterNode	Name of this node; must be unique per node. If no name is set, the cluster status shows LOCAL. If isDistributed=true is set via an environment variable or JVM parameter and no node name is given there or in webdesk.properties, Webdesk uses the environment variable HOSTNAME as the node name at startup. If isDistributed is set only in webdesk.properties, this pre-assignment does not take place.	Text, e.g. node1
PoOptions.clusterNodes	List of all node names of the cluster. Maintained in the "System parameters" mask (bean PoOptions, property clusterNodes) and valid for all nodes. The entries must match exactly the PoOptions.clusterNode names of the nodes: the cluster status expects a status message from every entry, and the name of the current node is always taken into account in addition. The job field "Run on cluster node" offers the list only if it contains more than one entry.	List of names (default: ClusterNode1)
WfStartup.engineName	Name of the workflow engine of the node. If the default name is not changed with isDistributed=true, Webdesk builds the name from the default name and the node name (webdeskshark_<node name>); normally this key therefore does not need to be set.	Text (default: webdeskshark)
webdesk.cluster.heartbeatIntervalSeconds	Interval in seconds at which a node publishes its cluster status and its job monitoring snapshot to the other nodes (also immediately at startup). Runs only with isDistributed=true. 0 switches the	Number (default: 30)

	heartbeat off, a negative value uses the default.	
webdesk.cluster.skipIdleJobMonitoringSnapshots	If true, a node without jobs and with a stopped scheduler does not publish a job monitoring snapshot in the heartbeat.	true/false (default: true)

Note: The keys isDistributed, useJmsReplication, PoOptions.clusterNode, WfStartup.engineName, webdesk.cacheReplication.RMI.peerProvider.properties and webdesk.cacheReplication.JMS.peerProvider.properties can also be set as a JVM parameter or environment variable. The environment variable name is WD_ plus the key in upper case, with dots replaced by underscores (e.g. WD_ISDISTRIBUTED, WD_POOPTIONS_CLUSTERNODE). Empty environment variables are ignored.

Cache replication: RMI or JMS

The transport applies globally to all replicated EhCache caches and is selected with useJmsReplication. In addition, Webdesk uses the replication to send service calls to the other nodes. This concerns saving jobs and job triggers, applying system parameters (re-injection of the bean properties) and publishing the cluster status. If the replication is disturbed, these changes do not reach the other nodes.

Property	Description	Possible value(s)
webdesk.cacheReplication.RMI.peerProvider.properties	Provider properties with RMI. The default is automatic discovery via multicast. timeToLive defines the reach: 0 same host, 1 same subnet, 32 same site, 64 same region, 128 same continent, 255 unrestricted. The nodes must therefore be in the same multicast group and multicast must be allowed in the network.	Default: peerDiscovery=automatic,multicastGroupAddres
webdesk.cacheReplication.peerListener.properties	Settings of the RMI listener (RMI only).	Default: socketTimeoutMillis=120000
webdesk.cacheReplication.JMS.peerProvider.properties	Provider properties JMS broker (JMS only): address of the broker as well as user and password of a valid broker account. Webdesk does not start a broker of its own. The names of the topic and queue are fixed and cannot be changed. The example value in the template contains default credentials, which must be replaced by the real credentials.	Template: providerURL=tcp://localhost:61616/,userName=<user>,password=<
webdesk.cacheReplication.excludedCaches	Comma-separated list of cache names that are not to be replicated. Applies to the caches defined in ehcache.xml files and, in Hibernate mode ehcache2, to Hibernate regions.	List of cache names (default: empty)

webdesk.cacheReplication.listener.replicatePuts, .replicateUpdates, .replicateUpdatesViaCopy, .replicateRemovals, .replic	What is replicated and how.	true/false (default: true, in each case)
webdesk.cacheReplication.maxElementsInMemory, .timeToIdleSeconds, .timeToLiveSeconds, .eternal, .overflowToDisk	Default values for EhCache caches without their own definition.	Default: 100 / 300 / 300 / false / true

Warning: Setting webdesk.cacheReplication.listener.replicatePuts=false globally is not permitted. The sending of service calls between the nodes relies on the put of an entry into the internal communication cache being replicated; with replicatePuts=false, job changes, system parameters and the cluster status would no longer arrive at the other nodes.

Warning: The older key webdesk.cacheReplication.peerProvider.properties (without RMI) is no longer evaluated. Multicast addresses or ports set with it do not apply; the default (230.0.0.1, port 4446) takes effect.

The default values for caches can additionally be overridden by a file defaultCacheConfiguration.properties on the classpath (WEB-INF/classes); it takes precedence over webdesk.properties. The fact that overflowToDisk is active by default can lead to longer response times for large caches, because entries are then read from disk; caches with their own bean definition can switch this off with the bean property overflowToDisk (example: PtmPermissionCacheBean.overflowToDisk=false in webdesk.properties). The override via <Bean>.<property> only works for Spring cache beans, not for the regions of the ehcache.xml files used for the Hibernate cache.

Redis

Redis is used for sessions (Spring Session), for replicated application caches (Caffeine as the local level, Redis as the shared level, invalidation messages via the Pub/Sub channel webdesk:cache-sync) and for the Hibernate provider redis-caffeine (channel webdesk:hibernate-cache-coherence). The connection is set up as soon as isDistributed=true or redis.useAsSessionStore=true applies. If neither redis.connection.url nor redis.clusterNodes is set, Webdesk connects to localhost port 6379.

Property	Description	Possible value(s)
redis.useAsSessionStore	If true, the HTTP sessions are held in Redis via Spring Session.	true/false (default: false)
redis.connection.url	Address of a single Redis server. Host, port and database number are taken over; a password contained in the URL is used only if redis.connection.password is empty.	e.g. redis://redis-host:6379/0 (default: empty, then localhost:6379)
redis.clusterNodes	Comma-separated list of the nodes of a Redis cluster. If set, it takes precedence over redis.connection.url; the timeout for Redis commands is then 2 seconds.	e.g. host1:6379,host2:6379 (default: empty)
redis.connection.password	Password for Redis. Do not pass it on in documentation, tickets or logs.	<your-redis-password> (default: empty)
redis.credentials.encrpted	If true, redis.connection.password is stored encrypted with the Webdesk method. The key name is spelled this way in the product	true/false (default: false)

	("encrpted") and must be used exactly like this.	
redis.session.cookie.jvmRoute	Route suffix of the session cookie JSESSIONID for load balancers that route by this suffix (e.g. Apache with mod_jk, worker name such as worker2). If the value is empty, Webdesk reads the jvmRoute attribute of the Engine element from the Tomcat conf/server.xml. The value is not automatically the same as PoOptions.clusterNode. If no route is found, a warning is logged and the cookie is written without a suffix.	Text (default: empty)

The redis.* keys can also be set as a JVM parameter or environment variable (e.g. WD_REDIS_USEASSESSIONSTORE, WD_REDIS_CONNECTION_URL); these take precedence over webdesk.properties.

Hibernate cache (second-level cache)

The provider of the Hibernate cache is selected with webdesk.hibernate.cache.provider in webdesk.properties. The choice applies globally and independently of isDistributed. The redis-caffeine variant has been available as an option since Webdesk 4.64.

Value	Effect
ehcache2 (default, also for an empty value)	EhCache-based cache; with isDistributed=true replicated via RMI or JMS. The regions are defined in the ehcache.xml files of the modules.
redis-caffeine	Redis as the authoritative shared store, supplemented by a local Caffeine near cache. Requires a Redis connection, i.e. isDistributed=true or redis.useAsSessionStore=true; otherwise startup fails with a message that the Redis connection is not available.
disabled	Switches off the second-level cache and the query cache.

Any other value prevents startup with an error message that names the permitted values.

Rules for operation

- All nodes of an environment must use the same value.
- Changing the value requires a restart of the entire environment; mixed operation of different providers is to be avoided.
- For redis-caffeine, the Redis server must have the eviction policy maxmemory-policy noeviction (or an equivalent, documented behavior for managed Redis). Webdesk checks this at startup using CONFIG GET maxmemory-policy and aborts startup for other policies (e.g. allkeys-lru, volatile-lru) or if CONFIG GET is denied. Even with noeviction, the memory consumption of Redis should be monitored, because a full Redis leads to an outage.
- Local EhCache2 cache files (disk state) must not be reused when the provider is changed.

Switching to redis-caffeine

1. Define the value for webdesk.hibernate.cache.provider for the entire environment and plan a maintenance window so that no mixed operation occurs.
2. Check that Redis runs with maxmemory-policy noeviction.
3. Delete the Redis keys of the Hibernate cache (prefix wd:hibernate:l2:) before starting, because earlier entries are not compatible. Example for a single Redis server (for Redis Cluster, run it per primary node):

```
redis-cli --scan --pattern 'wd:hibernate:l2:*' | xargs -r redis-cli del
```

4. Set the value in webdesk.properties of all nodes and start the nodes.
5. Check in the log that there is no persistent flood of decoding warnings for Hibernate cache keys.

Rolling back or switching to ehcache2 / disabled

1. Stop all nodes.
2. If the previous mode was redis-caffeine, delete the keys wd:hibernate:l2:*.
3. Set the value in webdesk.properties of all nodes to the target value.
4. Start all nodes again in one go.

Tuning the near cache (redis-caffeine)

Mutable entities are cached locally by default; Redis remains authoritative. If an event is missed, a node can serve a value that is out of date for at most valueTtlMillis. The following keys are optional; the default values are designed for normal operation. Where these keys are stored in an installation should be agreed with Webdesk support.

Property	Description	Possible value(s)
wd.cache.hibernate.nearCache.valueTtlMillis	Validity of a locally read value in milliseconds.	Number (default: 3000)
wd.cache.hibernate.nearCache.coherenceStateTtlSeconds	Time in seconds for which the locally observed coherence state is retained.	Number (default: 120000)
wd.cache.hibernate.nearCache.maxEntries	Maximum number of local entries.	Number (default: 20000)
wd.cache.hibernate.nearCache.reconnectBehaviorStrategy	Behavior strategy on a connection loss to Redis: local values are discarded. The values LOAD and NONE are rejected.	CLEAR (default)
wd.cache.hibernate.nearCache.excludedMutableRegions	Comma-separated list of regions that must never be read locally (for particularly consistency-critical regions).	List (default: empty)
wd.cache.hibernate.nearCache.queryRegions	Comma-separated list of query regions for which local reading is allowed.	List (default: empty)
wd.cache.hibernate.region.entryTtlMillis	Lifetime of an entry in Redis in milliseconds.	Number (default: 3600000)
wd.cache.hibernate.readWrite.softLockTtlMillis	Lifetime of the soft locks in Redis in milliseconds.	Number (default: 60000)

The metrics webdesk.hibernate.l2.redis.publish.failures, webdesk.hibernate.l2.redis.event.apply.failures and webdesk.hibernate.l2.redis.subscription.disconnects should stay at 0 during normal operation; webdesk.hibernate.l2.redis.near.cache.disabled should be 0.

HTTP sessions in the cluster

The "Session mode" field of the "Cluster Status" mask shows which type of session handling is active (see below). There are two independent settings:

- **Spring Session with Redis:** active with redis.useAsSessionStore=true (default: false). The session cookie remains JSESSIONID, optionally with a route suffix (redis.session.cookie.jvmRoute). If a node fails, the remaining node finds the session even if the cookie still carries the suffix of the failed node. Certain session attributes are deliberately not stored in Redis but stay on the node that created them (among others the states of Cocoon forms and flowscripts, the Vaadin application HR Expert, the logged-in and the switched user, the skin). The load balancer should therefore assign the requests of a session to the same node where possible (sticky routing).
- **Tomcat session manager (memcached-session-manager with Redis):** is switched on exclusively via a <Manager className="de.javakaffee.web.msm.MemcachedBackupSessionManager" .../> element in the META-INF/context.xml of Webdesk. In the context.xml delivered in the WAR, this element is commented out. The Webdesk Docker image, however, brings its own context.xml, which is copied into the web application and contains the element in active form (with memcachedNodes="redis://redis"); the libraries of the session manager are present in the image. The "Cluster Status" mask reports "memcached-session-manager / Redis" as soon as the element is found in the context.xml, even if redis.useAsSessionStore=true is additionally set.

Warning: Before going live, it must be clarified per environment, and verified in the "Session mode", which mechanism is actually active. With Docker operation in particular, the supplied context.xml must be taken into account.

For forms there are also the system parameters enableFormStateClusterFailover (default: false) and useSessionToStoreFormState (default: true) of the bean PoOptions. With failover enabled, the most recently saved state of certain forms is restored after a node switch; with useSessionToStoreFormState=true the state is stored in the session (and thus in Redis), otherwise in a replicated EhCache region.

Jobs in the cluster

- Whether and with how many threads jobs are executed on a node is determined by PoJobServiceTarget.active and noOfSchedulerThreads (see System parameters Portal & Organization[4]).
- If Webdesk works as a cluster (isDistributed=true, more than one entry in PoOptions.clusterNodes and the own node name is contained), the job editor shows the field "Run on cluster node" for the triggers. A trigger is scheduled only on the node whose name is selected there. In this constellation, a trigger without a node assignment is scheduled on no node (except for the one-time start via "Start Job").
- The monitoring of the jobs of all nodes is described in Active Jobs[5].

What is not synchronized automatically between the nodes

Area	Behavior and measure
MetaForm cache	Is excluded from the replication. After changes to form definitions that were not picked up on all nodes, run the action "Invalidate MetaForm cache on all cluster nodes" in the "Extended Tasks" mask. It clears the MetaForm cache on the current node and on the other nodes.

Other non-replicated caches	Some caches are deliberately local: among others the workflow process definition caches (webdesk_processdefinitions, webdesk_processdefinitions2hide, webdesk_wfprocessdefinitiondao_methodcache), the CMIS sessions of document management and the cache at.workflow.webdesk.po.STATIC_NON_CLUSTERED. Changes to them take effect only locally on each node; see Extended Tasks[6] (refresh workflow cache) and Job refresh Cache[7].
Lucene search index	The full-text index resides in the local file system of each node and is updated when the node starts (key webdesk.luceneIndexing, default: true). The actions "Refresh Lucene search index" and "Clean Lucene search index" in "Extended Tasks" only affect the node on which they are executed. The subfolder of the index is random by default; it can be fixed with the JVM parameter webdesk.lucene.index.subdir.
Clearing menus, permissions and other caches by hand	The clearing actions in the "List of caches" mask and in "Extended Tasks" are executed on the node on which they are called. Targeted forwarding to the other nodes is provided only for the MetaForm action; for other caches, the effect on other nodes depends on whether the cache is replicated (column "Distributed").
Session attributes	See section "HTTP sessions in the cluster".

"List of caches" mask (po_showCaches)

The mask shows all caches of the node that receives the request. The list can be filtered with the text field "Show only caches which names contain". For each cache, the following are shown: "Name", "Entries", "Efficiency" (hit rate in percent), "Cache type", "Distributed" and the configuration (maximum entries, time to idle, time to live, eternal). With the option "Display memory consumption", the columns "Memory, in KB" and "Max. Memory, in KB" as well as the sums are additionally calculated.

Column "Distributed"	Meaning
Local	EhCache cache without a replication listener: is not synchronized between nodes.
Replicated	EhCache cache with RMI/JMS replication, or Caffeine+Redis cache synchronized via Redis.
Caffeine (Local)	Caffeine cache without synchronization via Redis.
Terracotta-clustered / Unknown	EhCache cache with a Terracotta connection, or an unrecognized cache type.

If distributed operation is active, the mask additionally shows the text "This webdesk instance is part of a cluster having the name" with the node name and the button "Show Replication Peers". It lists the partners known to the replication (with RMI URL and GUID, with JMS the name); if no partners are known, "No Caching Peers available" appears. With "Clear all caches" or the "Clear cache" icon per row, caches of

the current node are cleared (after clearing all caches, the workflow cache has to be refreshed, see List of caches[8]).

"Cluster Status" mask (po_showClusterStatus)

When opened, the node publishes its own status and waits up to 2.5 seconds for the messages of the configured nodes to arrive. If configured nodes still appear as missing afterwards, the mask shows the message "Cluster replication is still warming up." with the names under "Configured nodes not visible yet". During normal operation, all nodes send their status in the heartbeat (default: every 30 seconds).

The header line shows "Cluster summary" (responding nodes / configured nodes), the counts "Stale" and "Missing", and the topology:

- "Cluster mode": "Distributed" or "Single-server".
- "Cache replication topology": "RMI", "JMS" or "None".
- "Session mode": "Spring Session / Redis", "memcached-session-manager / Redis", "Local in-memory" or "Unknown".

The table lists per node "Cluster node", "State", "Job service" (started or stopped), "Threads", "Refreshed at", "Active sessions", "Active jobs", "Executing jobs", memory (free, used, maximum) as well as "Version", "Revision" and "Branch". Differing versions between the nodes are immediately visible in the columns "Version", "Revision" and "Branch". The values in the "State" column:

State	Meaning
CURRENT	The node whose mask is currently displayed.
RESPONDING	Configured node whose last status message is not older than 180 seconds.
STALE	The last status message is older than 180 seconds (possible causes: node failed, heartbeat switched off or replication interrupted).
MISSING	The node is listed in PoOptions.clusterNodes, but no status message is available: the node has never reported (not started, different name, replication does not work) or its last message has expired (status messages are removed from the cache after 300 seconds, so a failed node changes from STALE to MISSING).
OBSERVED_ONLY	The node sends status but is not listed in PoOptions.clusterNodes (e.g. spelling of the name or a missing entry).

Typical symptoms and checks

Symptom	Check
The entry "ClusterNode1" or a node stays on MISSING although all nodes are running	PoOptions.clusterNodes still contains the default value, or names that differ from PoOptions.clusterNode. Compare the names on all nodes and in the list.
Other nodes are not visible (MISSING/STALE), topology shows "None"	isDistributed=true on all nodes and restarted? With RMI: same multicast group and port, multicast allowed in the network, suitable timeToLive. With JMS: broker reachable, credentials valid,

	useJmsReplication=true on all nodes. Check the replication peers in the "List of caches" mask.
A change to a job or system parameter takes effect on one node only	The forwarding runs via the cache replication. Check the cluster status and the replication peers; replicatePuts must not be set to false.
A job runs on no node	Triggers without an entry in the field "Run on cluster node" are not scheduled in the cluster; check the node status in Active Jobs[9].
Forms or masks show old fields on one node	Invalidate the MetaForm cache on all nodes using the action.
Startup aborts with "Unsafe Redis maxmemory-policy ..."	Set Redis to maxmemory-policy noeviction (only with redis-caffeine).
Startup aborts: "Unsupported value ... webdesk.hibernate.cache.provider"	Permitted values are ehcache2, redis-caffeine, disabled.
The login page appears after switching to another node	Check the "Session mode" in the cluster status. With Spring Session: same Redis connection on all nodes, redis.session.cookie.jvmRoute matching the route of the load balancer (a warning is logged if no route is found).
Outdated data after changes on another node (redis-caffeine)	Check the metrics for publish, apply and connection errors; if necessary, add the region to excludedMutableRegions.

Commissioning checklist

1. Define a unique PoOptions.clusterNode per node (WfStartup.engineName normally stays at the default), set isDistributed=true and, if desired, useJmsReplication=true with the broker account.
2. Check the network: with RMI, multicast (default 230.0.0.1:4446) between the nodes; with JMS, access to the broker; in addition, access of all nodes to Redis.
3. Set the Redis connection (redis.connection.url or redis.clusterNodes) on all nodes; take the password only from a protected source.
4. Decide which session type is used (redis.useAsSessionStore or Tomcat session manager) and coordinate the route of the load balancer (redis.session.cookie.jvmRoute).
5. Define the Hibernate provider uniformly (webdesk.hibernate.cache.provider); with redis-caffeine, take the Redis policy noeviction and the deletion of the keys wd:hibernate:l2:* into account.
6. Start the nodes; in the "System parameters" mask, fill the list PoOptions.clusterNodes with all node names.
7. Check in the "Cluster Status" mask: all nodes RESPONDING or CURRENT, topology as planned, uniform versions.
8. Jobs: define the node in the field "Run on cluster node" for every trigger and check it in "Active Jobs".

1. /daisy/webdesk-manual-admin/2561-dsy.html?language=4
2. /daisy/webdesk-manual-admin/1146-dsy.html?language=4
3. /daisy/webdesk-manual-admin/463-dsy.html?language=4
4. /daisy/webdesk-manual-admin/5655-dsy.html?language=4
5. /daisy/webdesk-manual-admin/1146-dsy.html?language=4
6. /daisy/webdesk-manual-admin/463-dsy.html?language=4
7. /daisy/webdesk-manual-admin/3779-dsy.html?language=4

8. /daisy/webdesk-manual-admin/2561-dsy.html?language=4
9. /daisy/webdesk-manual-admin/1146-dsy.html?language=4