CFMX - J2EE & clustering

Kai König
Chief software architect @ msg at.NET GmbH, Germany
Macromedia Certified Professional
Macromedia Certified Instructor

Preso given onsite @ CFUG WA, 02/05/2004

macromedia

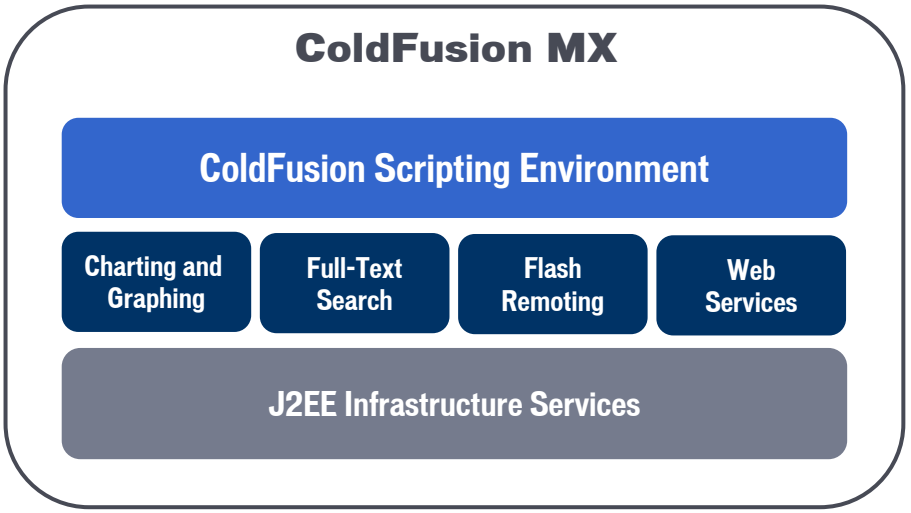
About Kai...

- Working as chief software architect at msg at.NET GmbH in Germany, the only Macromedia Premier Alliance Partner in Europe
- MMCP (CF 5, CF MX, Flash MX), MMCI
- User Group Manager CFUG Nordwest
- Blog in Black (www.bloginblack.de)
- Regular speaking on Macromedia related conferences worldwide
- Regular contributor to several German IT magazines

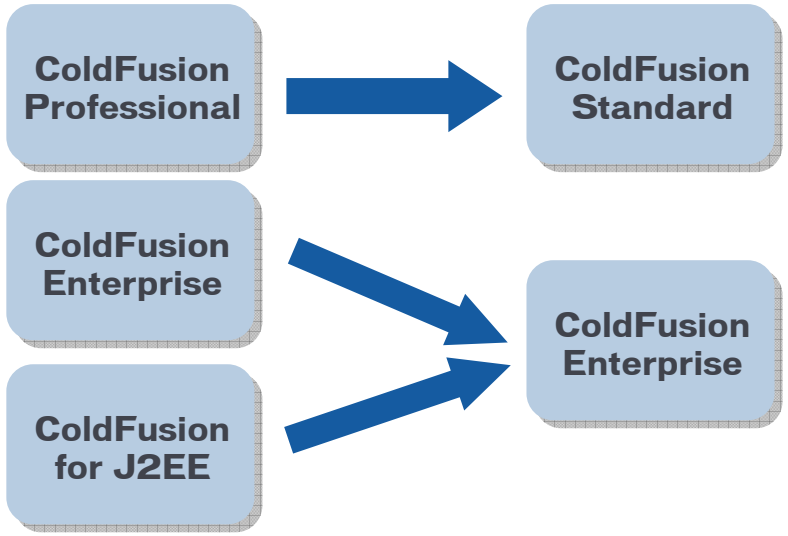
CFMX J2EE layer

macromedia

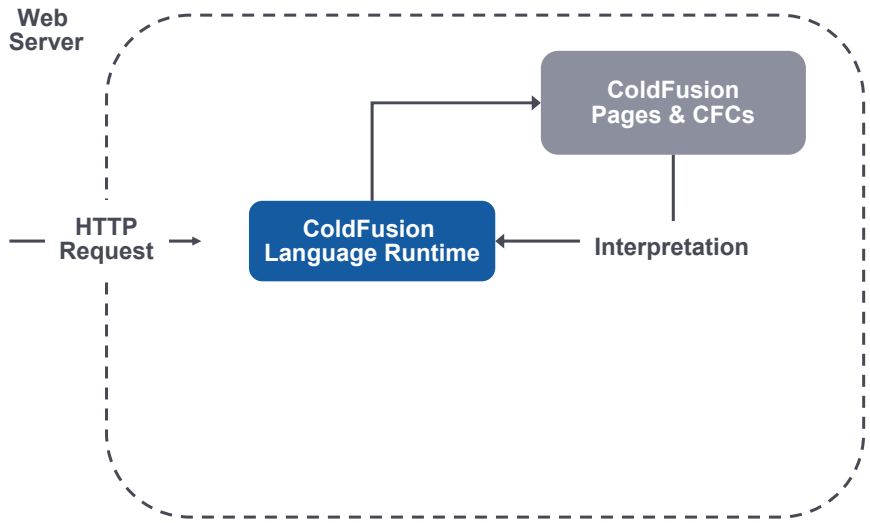
ColdFusion MX/MX 6.1 Architecture



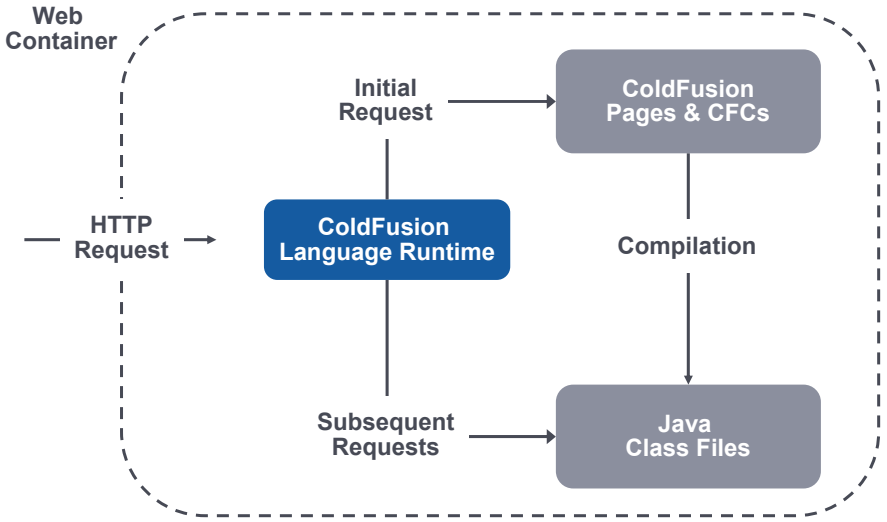
CFMX 6.1 - Consolidation



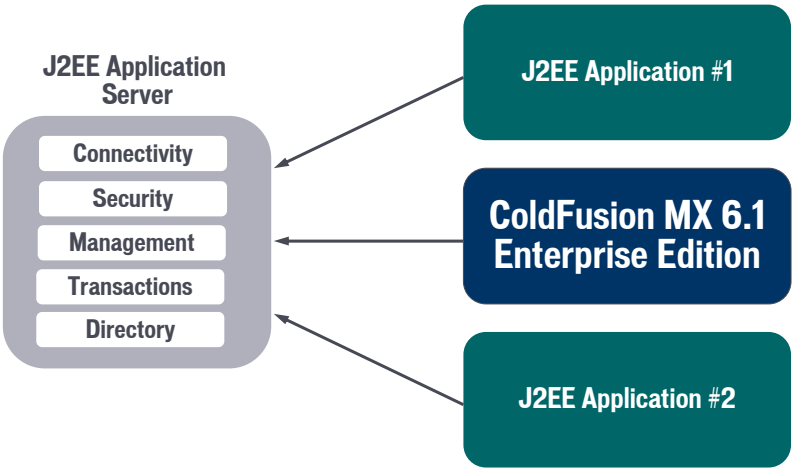
ColdFusion 5 Runtime Architecture



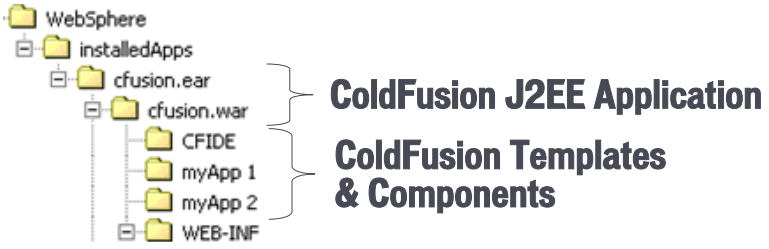
ColdFusion MX/MX 6.1 Runtime Architecture



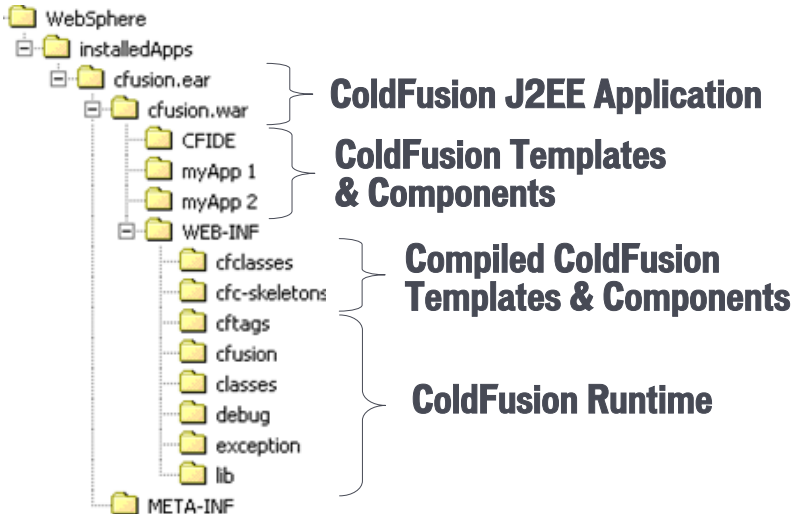
Deploying ColdFusion MX 6.1 on J2EE



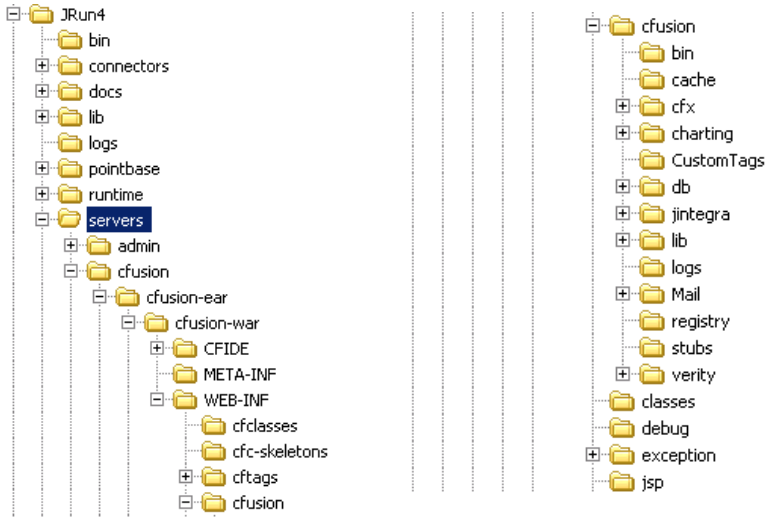
ColdFusion MX 6.1 deployed on WebSphere



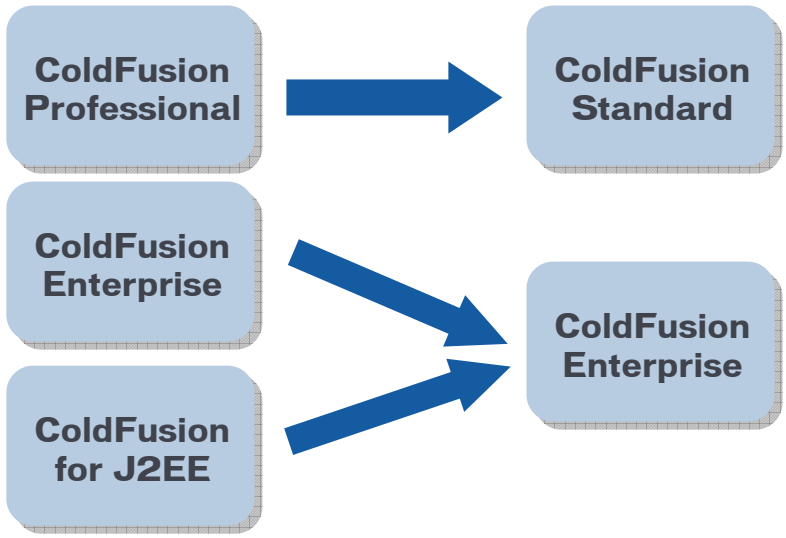
ColdFusion MX deployed on WebSphere



ColdFusion MX 6.1 Enterprise on JRun 4



CFMX 6.1 - Consolidation



Why clustering?

Properties and demands on web applications

- Availability – Reliability
- Performance
- Integrity of application data

Advantages of server clusters

- **Increase Reliability - one server box might die - your application won‘t**
- **Loadbalancing - spread user load equally over your cluster**
- **Seperate different applications regarding their runtime environment**
- **Might be cheaper to setup a farm of n Linux boxes instead of dealing with 2 Sun Solaris boxes**

J2EE clustering

- **One of the core elements of the J2EE SPI specification and its implementations is native cluster support**
  - API: Developer Interface for application development
  - SPI: Service Providing Interface - vendors like BEA and IBM etc. have to follow when implementing the J2EE standards in their application server
- **J2EE clustering means:**
  - Install multiple J2EE app servers and setup them with on ore more J2EE instances
  - Combine J2EE server instances to a J2EE cluster following some easy rules like cluster load, round robin etc.
  - Each J2EE server instance could run one or more J2EE applications which become part of the cluster

CFMXs role

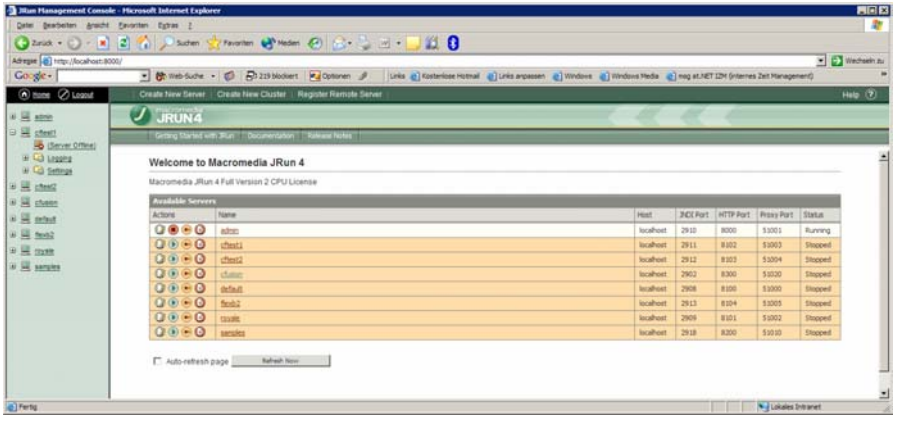
- CFMX 6.1 is a 100% certified compatible J2EE application, it‘s build in Java
- Why not try to cluster CFMX 6.1 similar to „common“ J2EE applications

Setting up multi-instance CFMX 6.1 Enterprise on JRun 4

Step 1: Create empty JRun 4 instances

- Take a JRun 4 installation with SP 2 or CFMX 6.1 Enterprise Editon in JRun 4 configuration
- Idea: Deploy CFMX 6.1 in two independent J2EE instances and form a cluster with the two of them
- Step: Create two J2EE instances (cftest1, cftest2) using „Create New Server“ in JRun 4 JMC, JRun 4 makes proposals for paths and port adresses

Step 1: Create empty JRun 4 instances



Step 2: Install CFMX 6.1 EE

- Use regular installer and select „J2EE configuration (.ear/.war-file)“ as installation method
- For JRun 4 deployment select .war-file, might differ for other J2EE app servers
- Use an installation path of your choice, it will just contain a template for your instances

Step 3: Deploy CFMX 6.1 to JRun 4

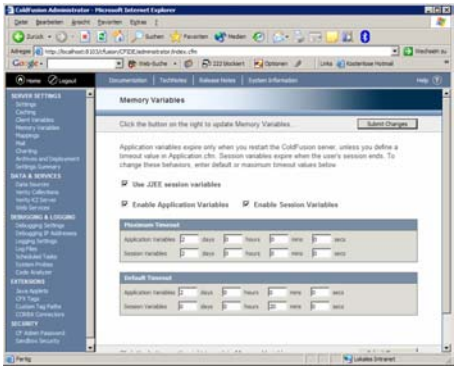
- Unzip cfusion.war-file to your new created J2EE instances...(d:\jrun4\servers\cftest1\cfusion, same for cftest2)
- „cfusion“ doesn‘t exist by default - it‘s the so called context root, important to separate between J2EE apps
- You should end up with CFIDE, META-INF and WEB-INF in your folders, cfusion.war could be deleted

Step 4: Start JRun instances

- Open JMC and start up the cftest1 and cftest 2 instances, the startup process automatially deploys the unzipped .war-file as a J2EE servlet
- Walk through the CFMX setup with calling http://localhost:<port>/cfusion/CFIDE/administrator/index.cfm - where „port“ is the HTTP port as defined in the JMC instance overview for cftest1 and cftest2

Step 5: Use J2EE session handling

- Setup „Use J2EE session handling“ in all CFMX 6.1 installations to build the cluster to force CF not to use it‘s own session handling



Step 5: Use J2EE session handling

- J2EE session handling allows using J2EE-based session replication for running user sessions in your cluster

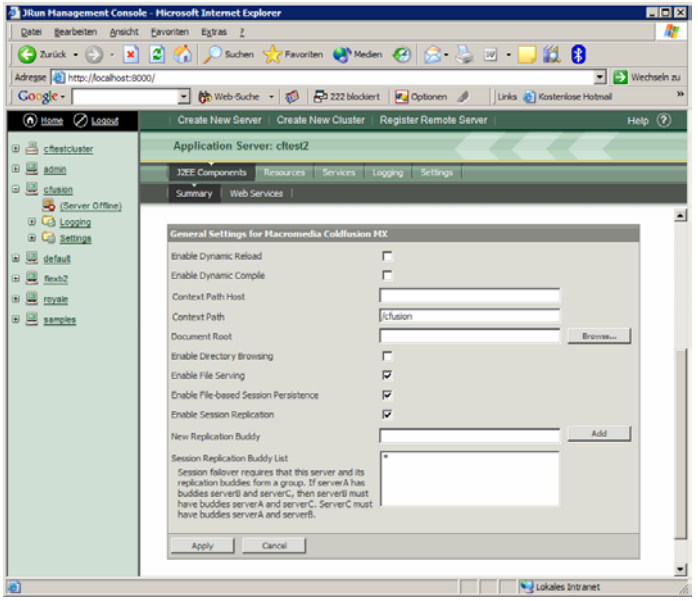
Step 6: Build cluster

- Use „Create New Cluster“ to build a cluster with cftest1 and cftest2
- All instances you want to add to the cluster have to run while adding

Step 7: Session replication

- The JRun cluster is dealing with a session repository for all cluster instances
- Setup Session replication in JMC for each cluster instance using the „Enable Session Replication“ checkbox and adding a Replication Buddy with the value * (this assumes your just running one cluster at a time on the machine)

Step 7: Session replication



Step 8: Editing deployment descriptors

- **Edit jrun-web.xml (located in the WEB-INF folders) and change the persistence-config entry so that it looks like**

```
<persistence-config>
  <active>false</active>
</persistence-config>
```

Step 9: Restart both instances

- **Restart cftest1 and cftest2 in JMC to update all settings**

Step 10: Bind HTTP server to cluster

- Use JRuns Web Server Configuration Tool to setup your HTTP server to point on the cluster
- Both instances are available for use via the internal JRun HTTP server already
- Go to d:\jrun4\lib and call java -jar wsconfig.jar to setup HTTP server for the cluster

Q&A