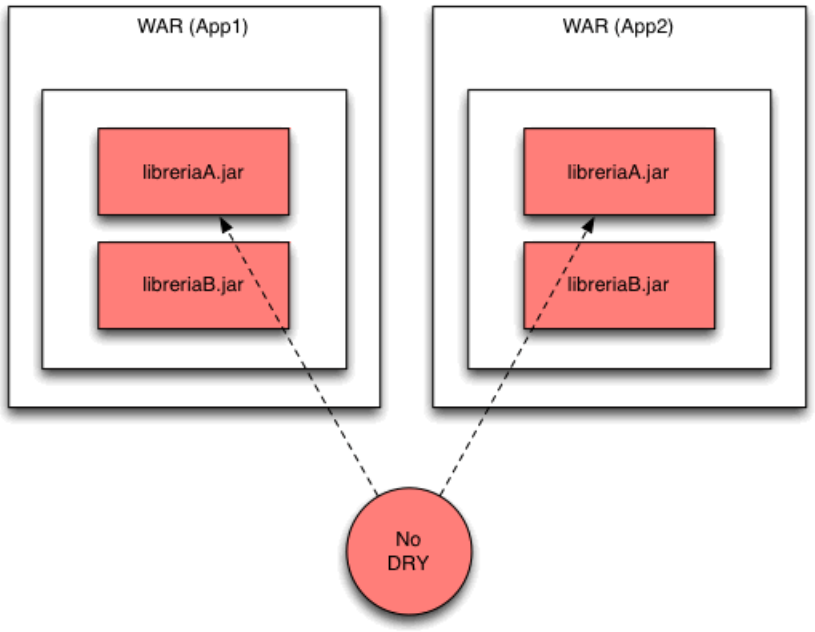
Uno de los servidores JEE que mas se usa a día de hoy es JBoss ya sea a traves de su sistema de licencias o como versión de comunidad. En su versión 6.0 EAP o 7.0 de comunidad ha incorporado un nuevo concepto denominado JBoss Modules que vuelve loco a la mayoría de la gente . Vamos a hablar de ello un poco a detalle en este articulo .

WAR y administradores de sistemas

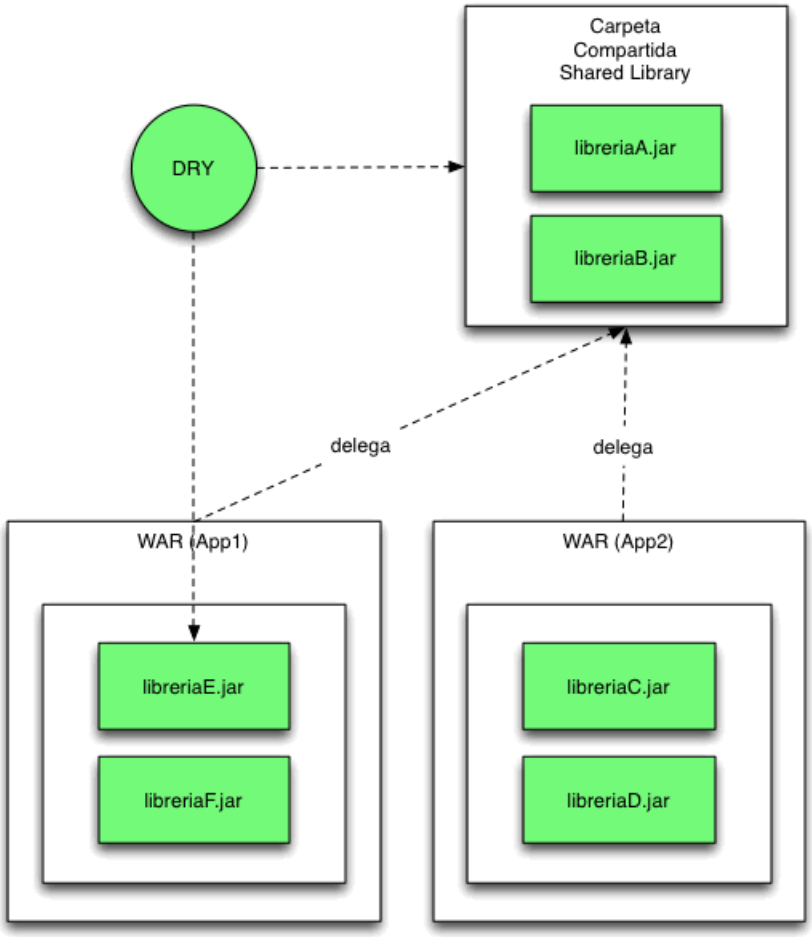
Una de las cosas que mas se quejan los administradores de sistemas es del problema de las librerias java (ficheros JAR) y su uso en las distintas aplicaciones . Ya que en muchas casos nos encontramos con librerias repetidas en cada aplicación lo cual para ellos no cumple con el principio DRY de no repetir pero en este caso no código sino ficheros JAR.



Librerías Compartidas

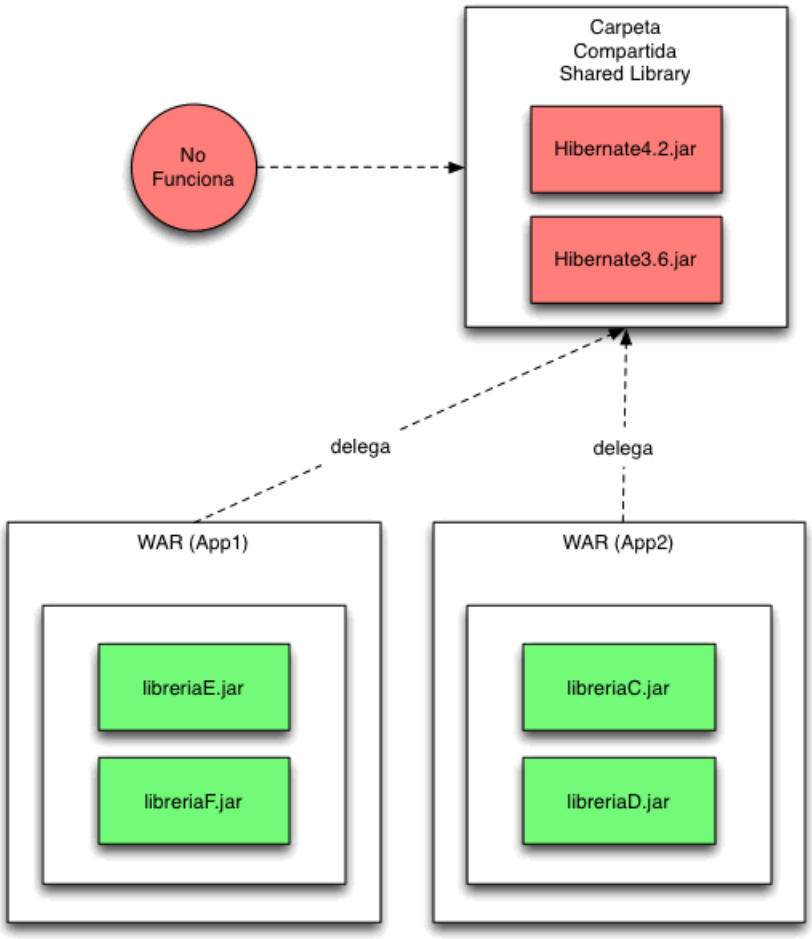
Para evitar este tipo de problemas los administradores optan habitualmente por usar algún tipo de carpeta compartida (shared library) que el servidor soporte . De esta manera pueden tener las librerías ubicadas en un único lugar y obligar a que todas las aplicaciones

primero revisen esta carpeta a la hora de cargar las distintas clases.



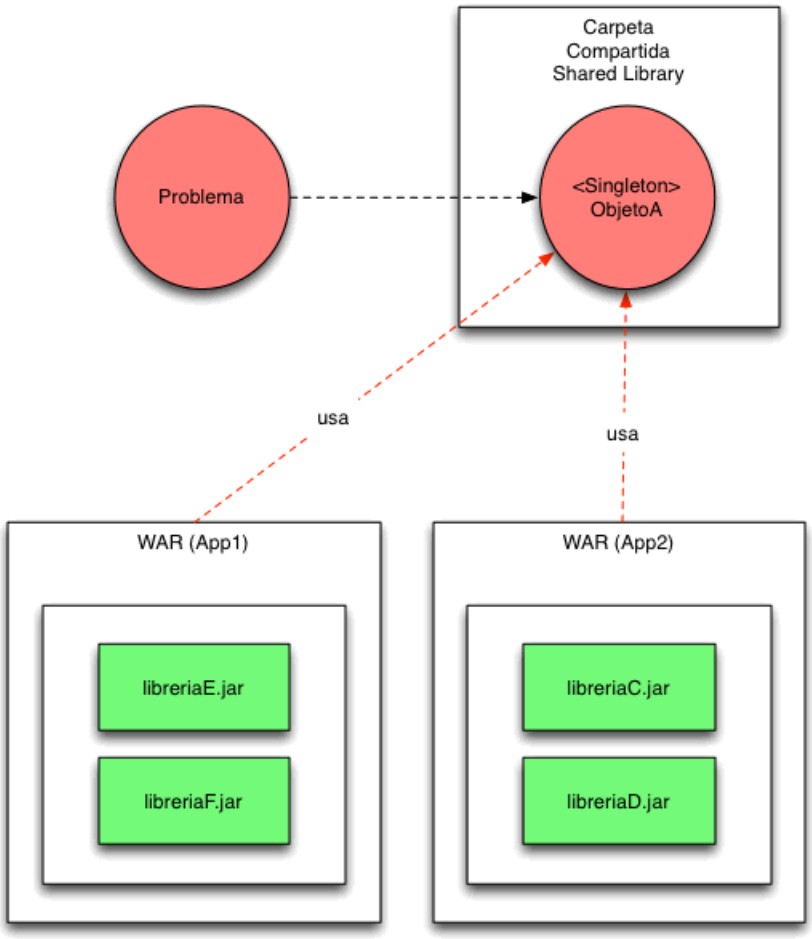
Librerias y versionado

Lamentablemente esto que parece una buena solución como punto de partida acaba no siéndolo en la practica debido a dos problemas esenciales . El primero es el problema de versionado ya que muchas veces tenemos la necesidad de tener varias versiones de librerias en un mismo servidor de aplicaciones ya que cada aplicación usa unas distintas . Por ejemplo una aplicación usa Hibernate 3.6 y otra mas moderna Hibernate 4.2.



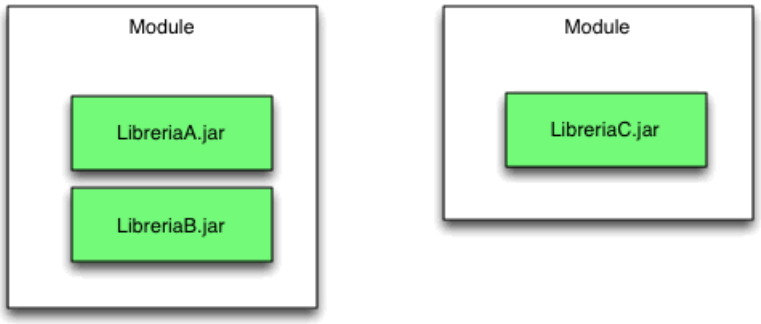
Librerias y Singletons

Otro de los problemas mas habituales es el uso de objetos singletons por parte de los frameworks . Cuando estas clases se instancian unicamente crean un objeto que es único para toda la aplicación . Son habituales los casos de objetos que sirven para configurar cosas. El problema que añaden los singleton es que si los ponemos en una carpeta compartida en el servidor . Serán únicos para todas las aplicaciones lo cual causará problemas.

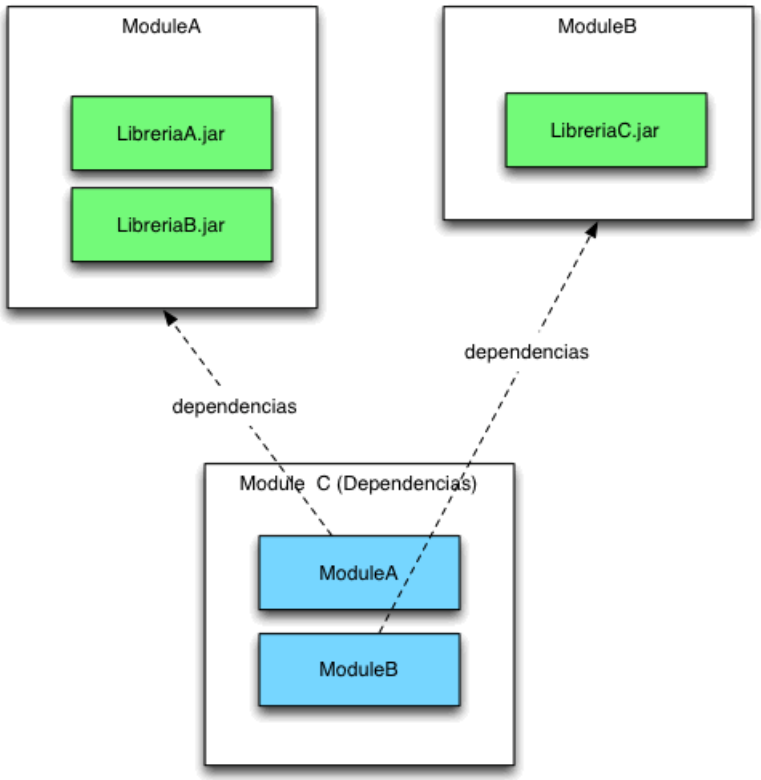


JBoss Modules

En JBoss han optado por una solución que parece de entrada interesante .Han definido un nuevo concepto el concepto de Módulo . Un Modulo puede incluir uno o varios jars



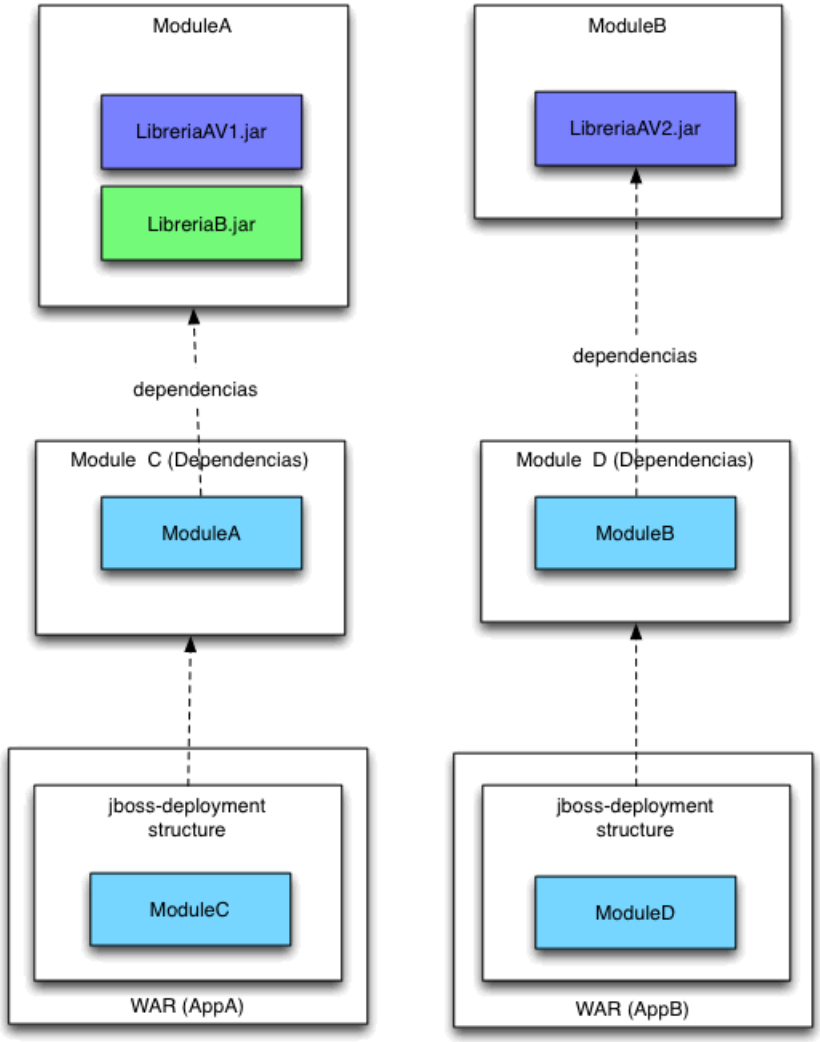
Ademas un módulo puede depender de otros módulos y agrupar así un conjunto de JARs



Jboss-deployment-structure.xml

Una vez entendido el concepto de módulo una aplicación puede a traves del fichero de configuración jboss-deployment-structure.xml añadir los módulos que desee y por lo tanto

los JARs . De esta forma un administrador puede tener instaladas diferentes versiones de librerías (cada versión en un módulo) . Evitando tener ficheros repetidos de librerías cargados varias veces.



Modules y DRY

Con este diseño JBoss consigue facilitar a los administradores la gestión de las librerías utilizando el principio DRY (Dont Repeat YourSelft) a los ficheros JAR de cada una de las librerías que se encontrarán ubicados solo una vez en el servidor en el módulo que

corresponda . Aunque esto parece óptimo tiene también sus problemas que abordaremos en artículos posteriores.