Todos sabemos que JSF es el standard que Oracle define para la construcción de aplicaciones web basadas en controles . Ahora bien los standards siempre pueden mejorar con el paso del tiempo . Esto es algo que ha pasado con JSF . En estos momentos nos encontramos con dos versiones que conviven en el mercado JSF 1.2 y JSF 2.0 .

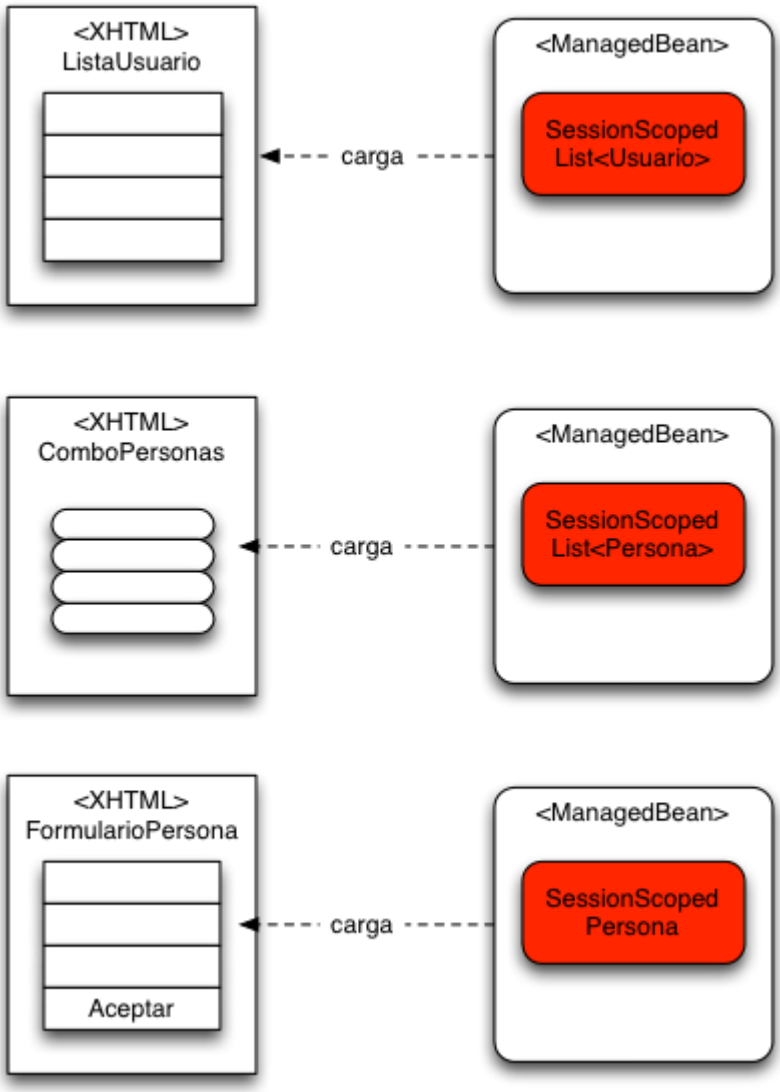
En la versión 1.2 de JSF existian 3 ambitos o scopes que nuestros ManagedBean podian tener

Request Scope: El ManagedBean tiene un ambito de petición

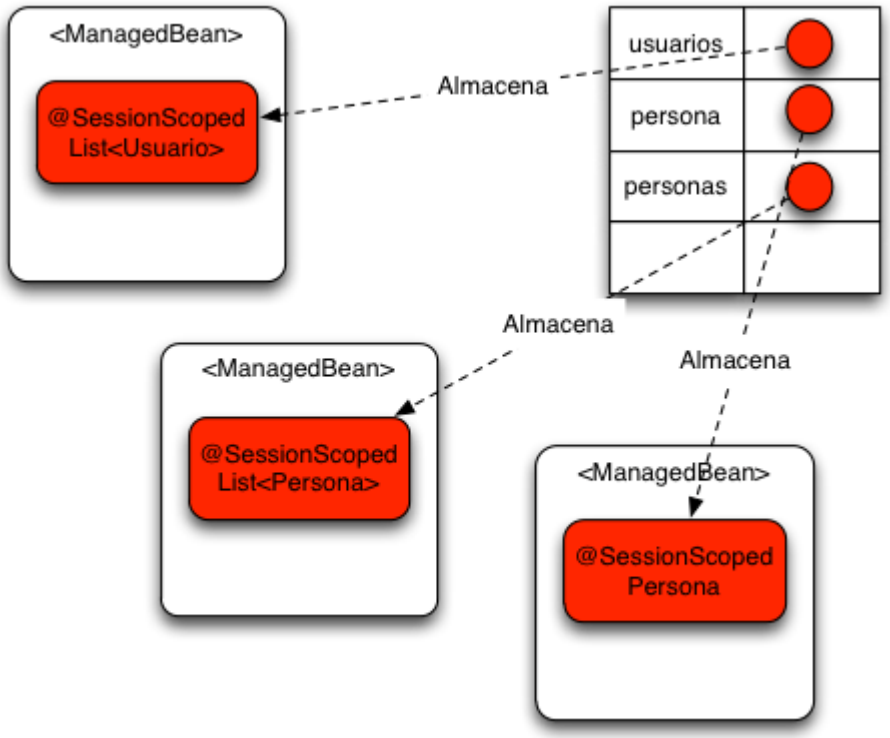
Session Scope : El ManagedBean vive durante la sesion del usuario

ApplicationScope: El ManagedBean vive durante la propia vida de la aplicación .

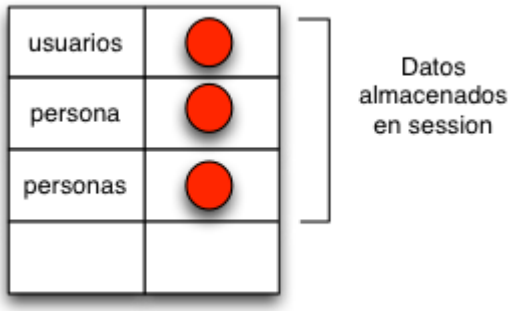
Muchas veces el propio funcionamiento de la aplicación con JSF hace que el programador trabaje de una forma mas sencilla utilizando ManagedBeans que tengan un ámbito de sesion como se muestra a continuación.



Lamentablemente todo lo que en JSF se asigne como sessionScoped es almacenado en la session del usuario .

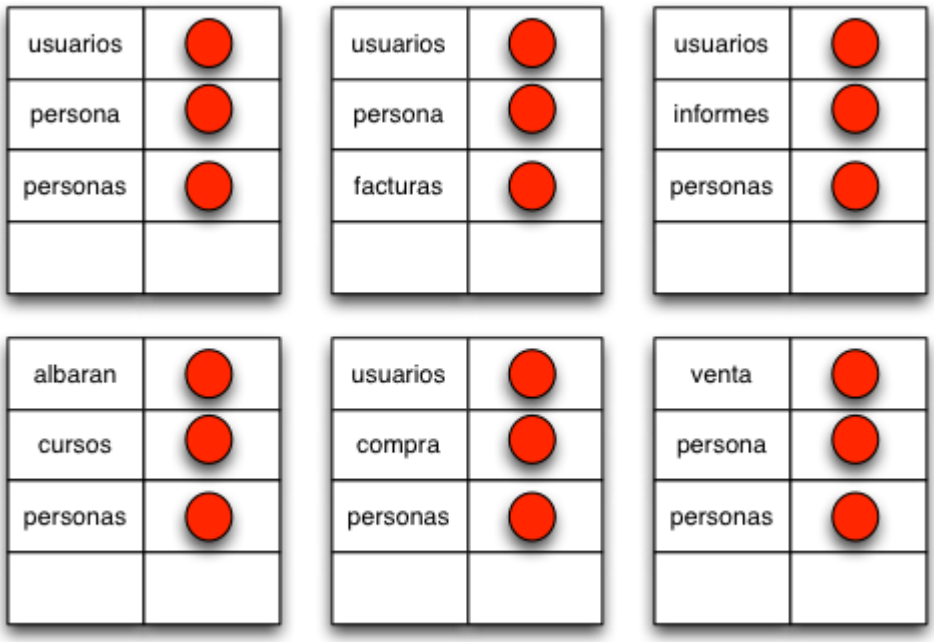


Existen unos casos puntuales como carritos de la compra o aplicaciones pequeñas etc en los cuales se trata de la operación correcta. Pero en muchos casos se acaban almacenando datos que no pertenecen al concepto propiamente dicho de Sesión sino que son simplemente listas de registros o cosas similares.

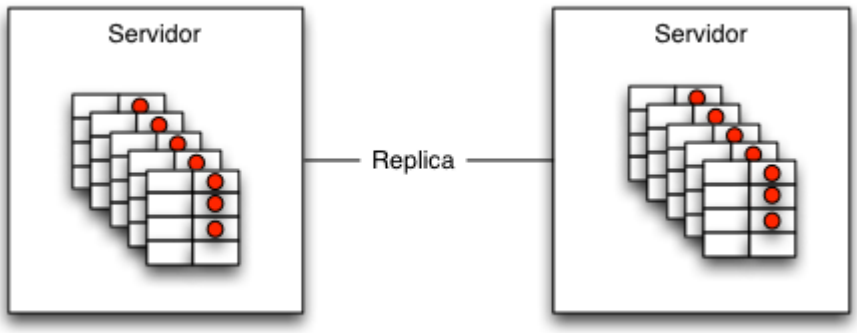


Esto al final acaba siendo un problema para la aplicación ya que aunque en el momento del

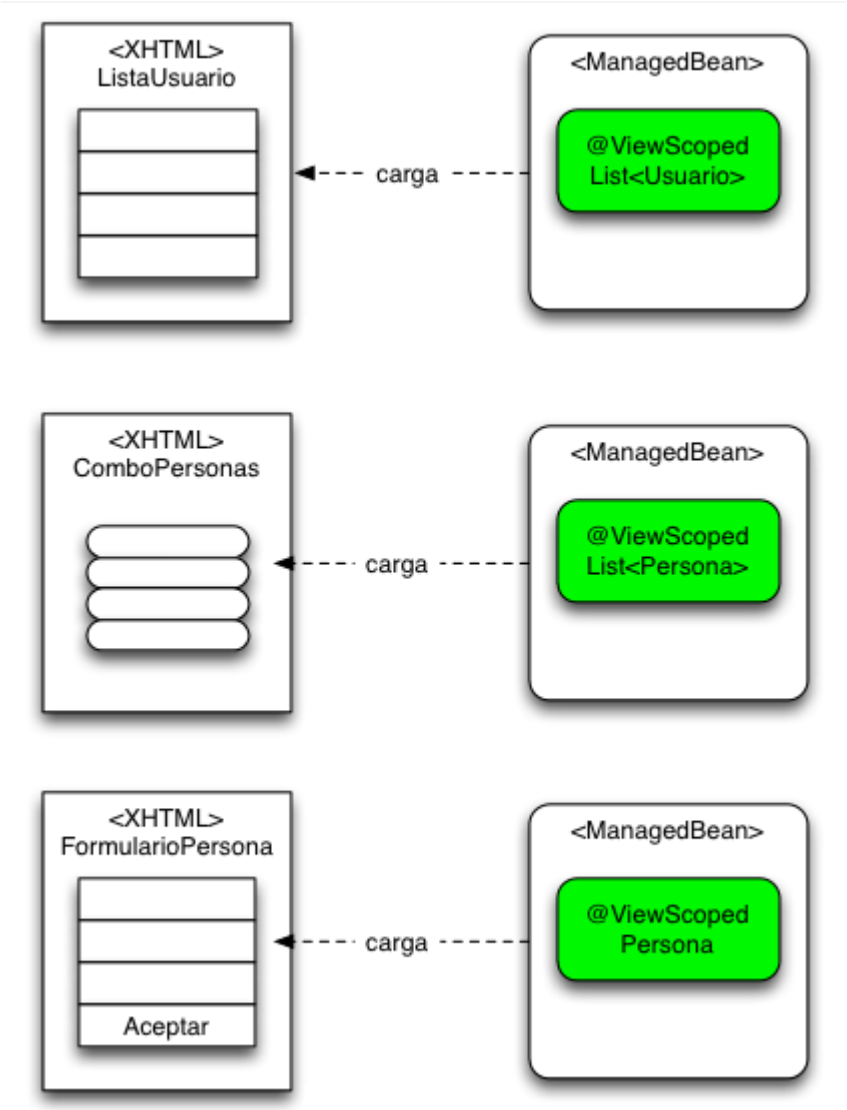
desarrollo no tendremos muchos problemas. Al ponerla en producción podremos observar un incremento importante de memoria consumida ,ya que podemos tener tranquilamente 500 o 1000 sessiones de usuario concurrentes y cada uno con una serie de datos en memoria.



Esto en principio nos puede parecer un puro problema de añadir mas memoria al servidor . Sin embargo según la aplicación se vaya complicando o necesite de una mayor escalabilidad el problema será mas dificil de resolver ya que obligaremos a nuestros servidores a replicar las sessiones en el cluster con los problemas de trafico de red y rendimiento que esto supondrá.



Para eliminar este problema JSF 2.o nos provee de un nuevo tipo de ambito denominado ViewScoped que permite almacenar los objetos mientras nos mantengamos en la misma vista . De esta forma podremos de una forma relativamente sencilla eliminar muchos de los datos que antiguamente almacenabamos en la sesión y mejorar la escalabilidad de nuestra aplicación.


<XHTML>
ListaUsuario
carga
<ManagedBean>
@ViewScoped
List<Usuario>
<XHTML>
ComboPersonas
carga
<ManagedBean>
@ViewScoped
List<Persona>
<XHTML>
FormularioPersona
Aceptar
carga
<ManagedBean>
@ViewScoped
Persona