Disseny i implementació d'un marc de treball per la capa de presentació J2EE

Salvador Morlà Murillo
Enginyeria Informàtica

Josep Maria Camps Riba
17 de Juny de 2013

Justificació i context

- Les tecnologies de la informació juguen un paper clau tant al mon empresarial com al món personal.
- Existeix una alta demanda de professionals experts en desenvolupament de programari Web.
- J2EE és una de les tecnologies més esteses per afrontar aquest tipus de desenvolupaments i compte amb un gran número d'usuaris.
- Els marcs de treball faciliten la feina del desenvolupadors d'aplicacions, donant estructura, fomentant la reutilització i assegurant la mantenibilitat.

Objectius del projecte

Els objectius que ens marcam per aquest PFC són els següents:

- Aprofundir en el coneixements de la tecnologia J2EE i en la part de la capa de presentació.
- Avaluar els marcs comercials existents per veure les seves característiques més rellevants.
- Dissenyar i implementar un marc de treball de capa de presentació
- Implementar una aplicació de test que faci servir el marc de treball dissenyat.

Planificació del projecte

El projecte se planifica en tres fases els objectius de les quals se detallen a continuació:

- Fase I: estudi teòric de la tecnologia i estudi de marcs de treball de capa de presentació existents al mercat.
- Fase II: Anàlisi, disseny i implementació de un marc de treball de capa de presentació propi.
- Fase III: Implementació d'una aplicació amb el framework dissenyat.

Fites rellevants:

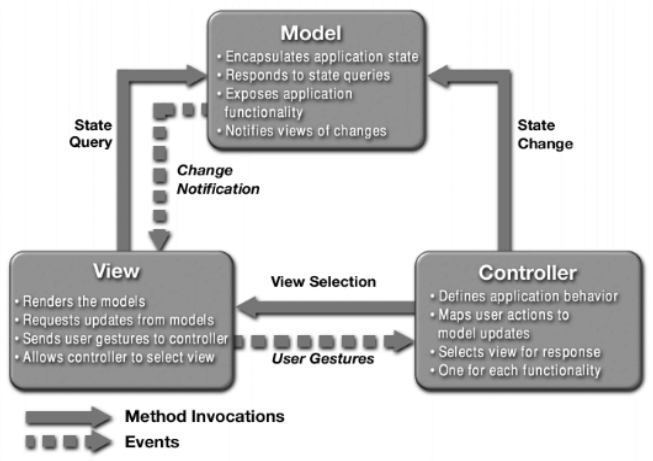
- 11 març 2013: Introducció, objectius i planificació.
- 15 abril 2013: Documentació de les conclusions de l'estudi de marcs existents.
- 3 juny 2013: Framework propi.
- 17 juny 2013: Aplicació Web amb framework propi i memòria.

Planificació del projecte

Detall de la planificació del projecte:

Nombre de tarea	Duración	Comienzo	Fin
PCF	**71 días**	**lun 11/03/13**	**lun 17/06/13**
Estudi de frameworks de presentació	**26 días**	**lun 11/03/13**	**lun 15/04/13**
JavaServer Faces	**8 días**	**lun 11/03/13**	**mié 20/03/13**
Investigació i estudi del framework	5 días	lun 11/03/13	vie 15/03/13
Documentació sobre el framework	3 días	lun 18/03/13	mié 20/03/13
Struts	**8 días**	**jue 21/03/13**	**lun 01/04/13**
Investigació i estudi del framework	5 días	jue 21/03/13	mié 27/03/13
Documentació sobre el framework	3 días	jue 28/03/13	lun 01/04/13
Spring MVC	**8 días**	**mar 02/04/13**	**jue 11/04/13**
Investigació i estudi del framework	5 días	mar 02/04/13	lun 08/04/13
Documentació sobre el framework	3 días	mar 09/04/13	jue 11/04/13
Documentació final del lliurament	2 días	vie 12/04/13	lun 15/04/13
Fi d'estudi de frameworks de presentació	0 días	lun 15/04/13	lun 15/04/13
Framework J2EE de capa de presentació	**35 días**	**mar 16/04/13**	**lun 03/06/13**
Anàlisi	6 días	mar 16/04/13	mar 23/04/13
Disseny	6 días	mié 24/04/13	mié 01/05/13
Implementació	21 días	jue 02/05/13	jue 30/05/13
Proves	2 días	vie 31/05/13	lun 03/06/13
Lliurament del framework	0 días	lun 03/06/13	lun 03/06/13
Aplicació d'exemple	3 días	mar 04/06/13	jue 06/06/13
Memòria	7 días	vie 07/06/13	lun 17/06/13
Lliurament final	0 días	lun 17/06/13	lun 17/06/13

Patró d'arquitectura Model – Vista – Controlador (MVC)



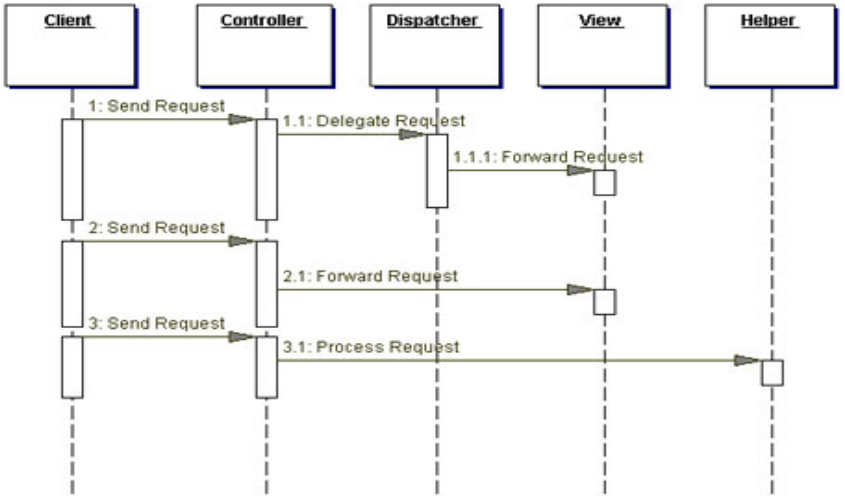
- **Model**: és la representació específica de la informació i les regles de negoci que governa l'accés i l'actualització d'aquesta informació. El model no és conscient de com s'han de presentar les dades (a un navegador per exemple).

- **Vista**: representa la capa de presentació de l'aplicació. Empra els procediments del model per obtenir els continguts i els representa. La vista no té dependència de la lògica de l'aplicació i no se veu alterada si la lògica de negoci canvia.

- **Controlador**: té la responsabilitat d'interceptar les peticions que se realitzen des de la vista i passar-les al model per que executi l'acció corresponent que obté la resposta adequada a la petició i finalment presentar a la vista apropiada el resultat de la petició.

Patró de disseny Front Controller

- Es el patró més emprat com a estratègia per gestionar les peticions dels clients. Àmpliament implementat als marcs de treball avaluats.
- Sistematitza les entrades dels usuaris de l'aplicació interceptant les peticions.
- Remet les peticions d'usuari al gestor adequat per satisfer-les en funció de la petició mitjançant el que s'anomena un *dispatcher.*
- Construeix una resposta adequada a la petició



STRUTS

- Esdevé un dels primers marcs de treball de capa de presentació J2EE
- Elements del seu disseny destacables:
  - Controlador basat en el patró FrontController basat en un servlet anomenat ActionServlet.
  - Basa la seva configuració en un fitxer descriptor en llenguatge XML (struts-config.xml)
  - Actions: contenen la lògica que s'ha de processar per satisfer les peticions de la capa de presentació.
  - ActionForms: JavaBeans que tenen la funció de contenir les dades processades a les Actions per posar-les a disposició de les vistes per ser representades.
  - Molt orientat a emprar JSP com a tecnologia per la capa de Vista.
- Proveu de funcionalitats per la validació i conversió de dades, internalització o suport multi-idioma i una llibreria de *tags* per ajudar a representar les dades a les vistes.

Spring MVC

- Mòdul independent del framework Spring per construcció d'aplicacions Web seguint el patró d'arquitectura de Model-Vista-Controlador.
- Elements del seu disseny destacables:
  - Orientat a peticions i dissenyat al voltant d'un servlet central que despatxa peticions (DispatcherServlets) que invoca Controllers per resoldre qualsevol lògica de negoci de l'aplicació.
  - Implementa el patró FrontController.
  - ModelAndView és l'objecte de retorn dels Controller com a resultat de la petició. Aquesta classe és simplement una classe contenidora (Container) que guarda la informació del model i la vista.
  - Obert a adoptar qualsevol tecnologia per implementar la vista.
  - Dona suport a la configuració mitjançant anotacions des de versió 2.5
- Proveu de funcionalitats per la validació i conversió de dades, internalització o suport multi-idioma. Corva d'aprenentatge alta.

JavaServer Faces

- Mòdul independent del framework Spring per construcció d'aplicacions Web seguint el patró d'arquitectura de Model-Vista-Controlador.
- Elements del seu disseny destacables:
  - Orientat a components i no a peticions. Sistema basat en un controlador (dissenyat seguint el patró de FrontController) i implementat com un servlet (FacesServlet)
  - Se completa amb unes llibreries de tags per facilitar la presentació de la informació del model i complementades amb components d'interfície d'usuari (UI Components).
  - ManagedBeans: són components de programari lligats als UI Components i que implementen el processament dels esdeveniments
  - Baixa necessitat de configuracions basades en descriptors (XML) gràcies al suport en l'ús d'injecció de dependències amb anotacions

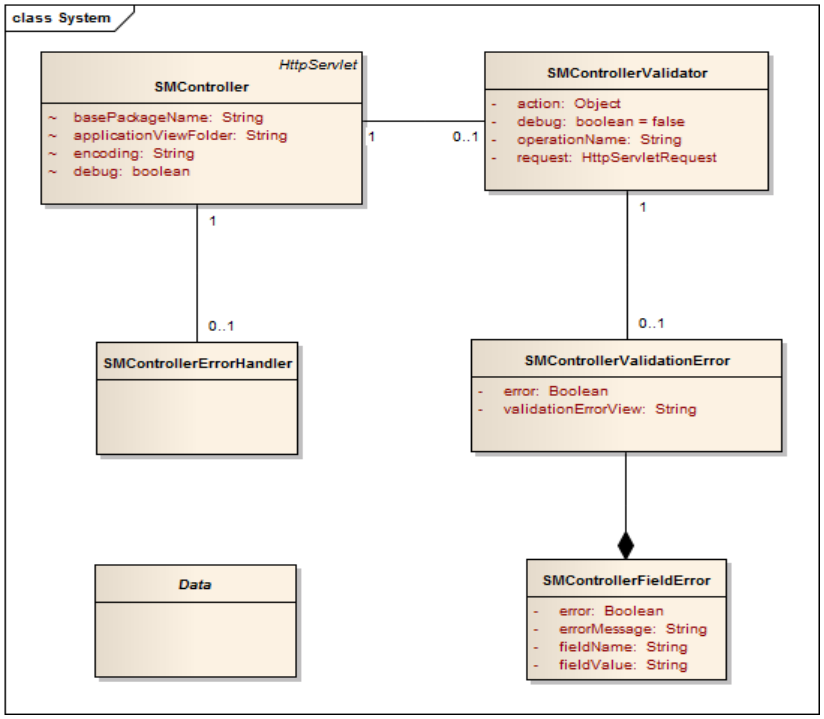
Objectius perseguits. Declaració d'abast

- Disseny d'un marc de treball propi de capa de presentació J2EE.
- Se persegueix la facilitat de configuració i la senzillesa d'ús:
    - No necessitat de configuració mitjançant fitxers XML.
- Oferir un controlador que gestioni el fluxe complet MVC de les peticions d'usuari.
- Addicionalment donar suport a:
    - Validació de dades
    - Control de seguretat d'accés a accions i als seus mètodes

Tecnologies clau per aconseguir els objectius:

- Ús d'anotacions.
- Exploració de classes en temps d'execució mitjançant Java Reflection.

Classes principals:



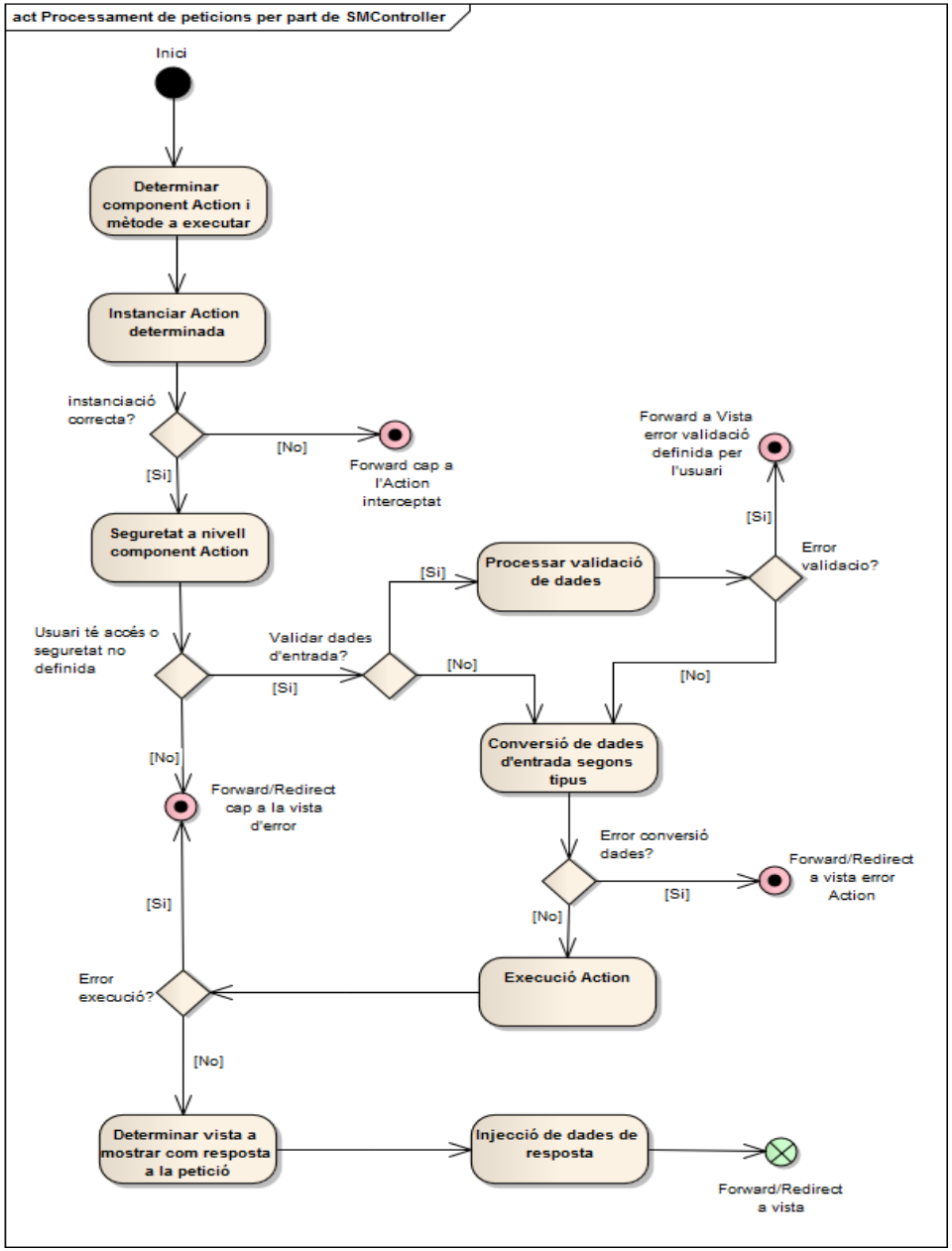
SMController: implementa el patró FrontController mitjançant un servlet. Invoca a les classes de component de controlador Action per satisfer les peticions.

SMControllerValidator: recolza a la classe SMController i ho fa realitzant la validació de dades prèvies a l'execució dels mètodes d'un component de controlador Action.

SMControllerErrorHandler: Aquesta és la classe que s'encarrega de processar una excepció generada durant el processament d'una petició d'usuari.

Data: Representa un contenidor de dades del framework.

Disseny de SMFramework. Processament de peticions



Configuració de classes de component de controlador Action

- Mitjançant anotacions inspeccionades en temps d'execució pel controlador mitjançant Java Reflection.
- Anotacions de SMFramework:
  - **@SMControllerBean**: els objectes anotats amb aquesta anotació constitueixen els components per passar informació entre la vista i el model
  - **@Forward i @Redirect**: indiquen la vista que ha de carregar el controlador una vegada satisfet el processament de la petició d'usuari. En funció de l'anotació indicada s'empra un mètode o un altre.
  - **@ErrorDestination**: indicarà al controlador a quina vista s'ha de redirigir la navegació quan se produeixi un error en el processament de la petició.
  - **@RolesAllowed:** permet establir quins rols de l'aplicació poden executar un mètode d'un component de controlador Action.
  - **@Request i @Response:** permeten injectar a la classe de component de controlador Action els objectes Request i Response per si es necessari manipular-los.
  - **@ValidateBean, @ValidateMethod i @ValidateField**: permeten especificar la validació de les dades d'entrada d'una petició.

Validació de dades:

- SMController se recolza al component SMControllerValidator.
- El marc de treball inclou una serie de tipus de validació estàndard que poden ser configurats per adaptar-se a noves necessitats i poden ser estesos per l'usuari.

Conversió de dades:

- A la vista totes les dades són text.
- SMController s'encarrega de la conversió de dades que provenen de la vista com a paràmetres d'entrada d'una petició.
- SMController s'ajuda de la llibreria d'ApacheCommons BeanUtils.
- Se suporten els tipus: Integer, Double, Float, BigDecimal, Date i Timestamp.

Classes de component de controlador Action

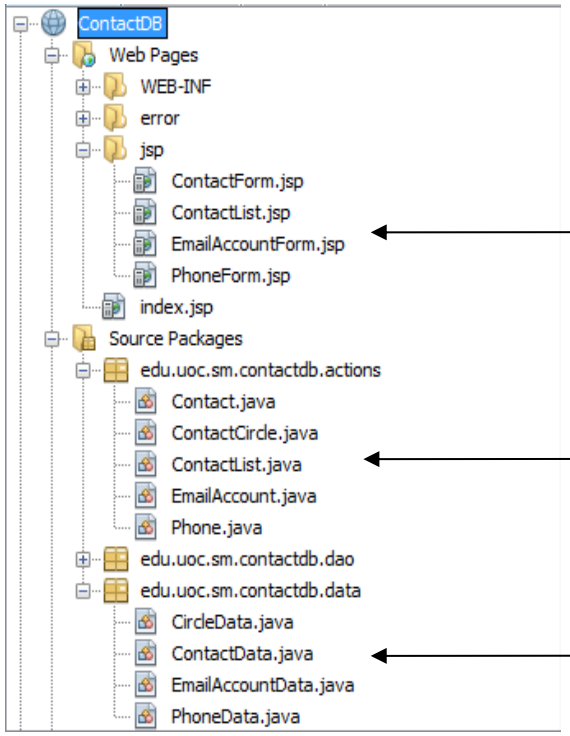
- Classes POJO que sobre les que el controlador delega la responsabilitat de processar les dades del model.
- En elles s'especifica el comportament que ha de seguir el controlador SMController per satisfer la petició:
  - Objectes SMControllerBean per rebre dades i enviar dades des de i cap a les vistes
  - Vista de destí en la que mostrar el resultat del processament de la petició d'usuari.
  - Vista de destí a la que redirigir en cas de que se produeixi un error en el processament de la petició.

```
@SMControllerBean (requestGetParameter=false, create=false)
List<UnaClasse> llista

@Forward (“/vistes/LlistaUnaClasse.jsp”)
@ErrorDestination (“/error/jsp”)
public void execute() {
    llista = new ArrayList<UnaClasse>();
    UnaClasseDAO unaclasseDao  = new  UnaClasseDAO();
    llista = unaclasseDao.list();
}
```

Aplicació d'exemple

- Base de de dades de contactes amb funcionalitats bàsiques per mantenir telèfons i comptes de correu electrònic.



Vistes

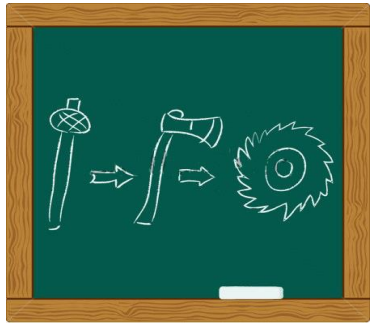
Classes de component de controlador Action

Contenidors de dades

- Experiència positiva. Senzillesa i fàcil configuració.
- Validació de dades millorable

Millores:

- Millora del sistema de validació de dades: redissenyar el seu funcionament per ser més flexible i òptim per l'usuari.
- Millora del disseny de la conversió de dades: el marc s'hauria de redissenyar per permetre l'extensió dels tipus suportats per l'usuari.
- Millora del tractament d'excepcions
- Introducció de suport de localització: se poden incloure suport per la implementació d'aplicacions multi-idioma.
- Incorporació de funcionalitats per facilitar la presentació de dades a les vistes.
- Eina de generació de codi: dissenyar una eina paral·lela al marc que a partir de meta informació sigui capaç de generar l'estructura de projecte amb els contenidors de dades i les classes de component de controlador Action.



Conclusions finals

- L'ús de marcs de treball per resoldre la capa de presentació és imprescindible per donar estructura i mantenibilitat a les aplicacions Web.
- El disseny de marcs de treball és un problema altament complexe.
- Requereix aprofundir molt en la tecnologia sobre la que dona suport el marc de treball.
- Existeix una tendència evolutiva a la configuració alternativa als fitxers XML. Hem intentat aportar el nostre enfocament.
- Un dels punts més complexes per al disseny i implementació ha estat la conversió de dades provinents de la vista.
- S'han adquirit coneixements de J2EE i patrons de disseny. Experiència molt positiva.

Salvador Morlà Murillo
Enginyeria Informàtica

Josep Maria Camps Riba
17 de Juny de 2013