Homogénéisation d'applications

Sujet de stage :

Réalisation de briques logicielles Java d'homogénéisation de l'apparence et de l'utilisation de suites logicielles sous Linux, Mac OS X et Windows

COLUCCI Brice

Plan de la présentation

- Présentation générale
- Travail réalisé
- Mise en oeuvre
- Bilan du stage

Le LAOG et le JMMC

Le LAOG

- Unité mixte de recherche
- Recherches
  - Formation stellaire et planétaire
  - Astrophysique moléculaire
  - Phénomènes à haute énergie
  - Recherche instrumentale

Le JMMC

- Groupement de laboratoires
- Interférométrique optique
- Centre de réalisation au LAOG

Le sujet de stage

- Besoins recencés
  - Homogénéisation des interfaces et de l'utilisation des applications
  - Demande de développement croissante

=> Librairie JMCS

Environnement et outils

- Poste de travail et serveur Linux
- Gestion de fichiers en configuration
- UML
- Patrons de conception
- Architecture MCS

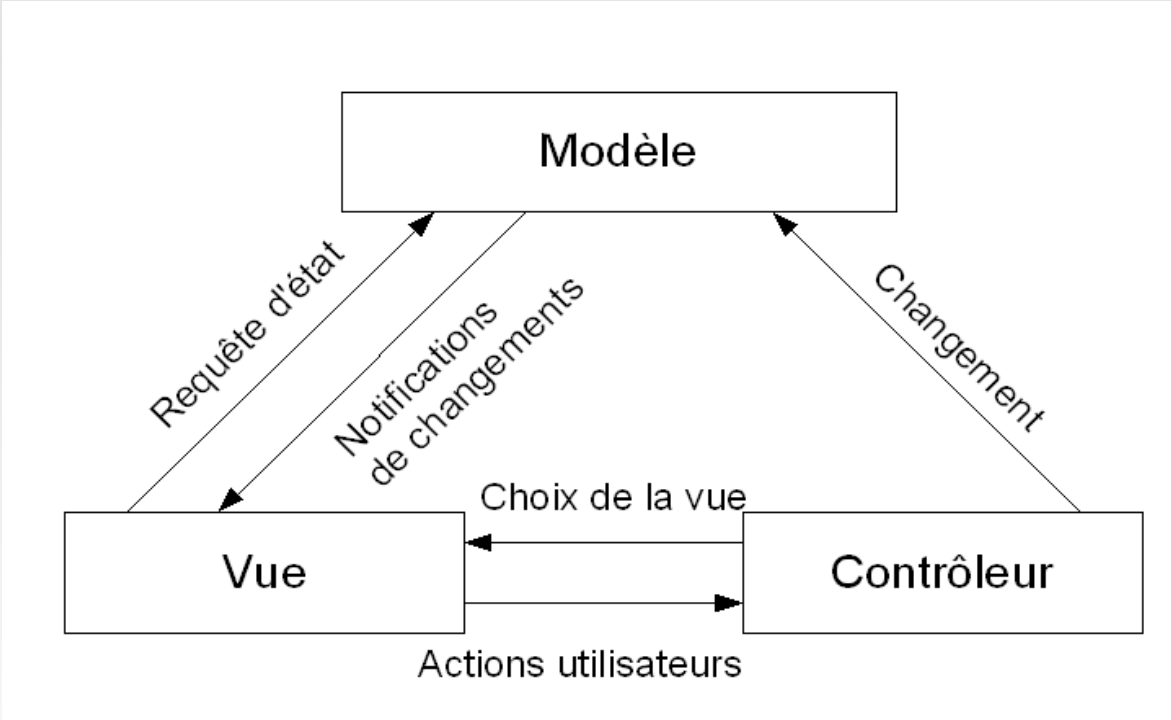
Architecture MCS

- Groupement par thématique
- Modularité du code
- Développement encadré
- Makefile
- Normes de programmation

```
/jmcs
|- bin
|- config
|- doc
|- errors
|- include
|- lib
|- object
|- src
|  |- fr
|     |- jmmc
|        |- mcs
|           |- gui
|- test
|- tmp
```

Les patrons de conception

- Modèle-Vue-Contrôleur (MVC)



- Observateur/Observable

JMCS

- Développée en Java 1.5 et ultérieur
- Compatible avec les applications autonomes (Jars)
- Compatible avec les applications Java WebStart

Travail réalisé

- Ecran de démarrage et fenêtre d'a-propos
- Gestion des journaux d'exécution
- Gestion des retours d'erreurs et commentaires
- Génération et visualisation de l'aide utilisateur
- L'interface ”ligne de commande”
- Uniformisation des menus et raccourcis claviers
- Documentation

Fenêtre d'à-propos et écran de démarrage

Ecran de démarrage

Fenêtre d'à-propos

Ecran de démarrage et fenêtre d'à-propos

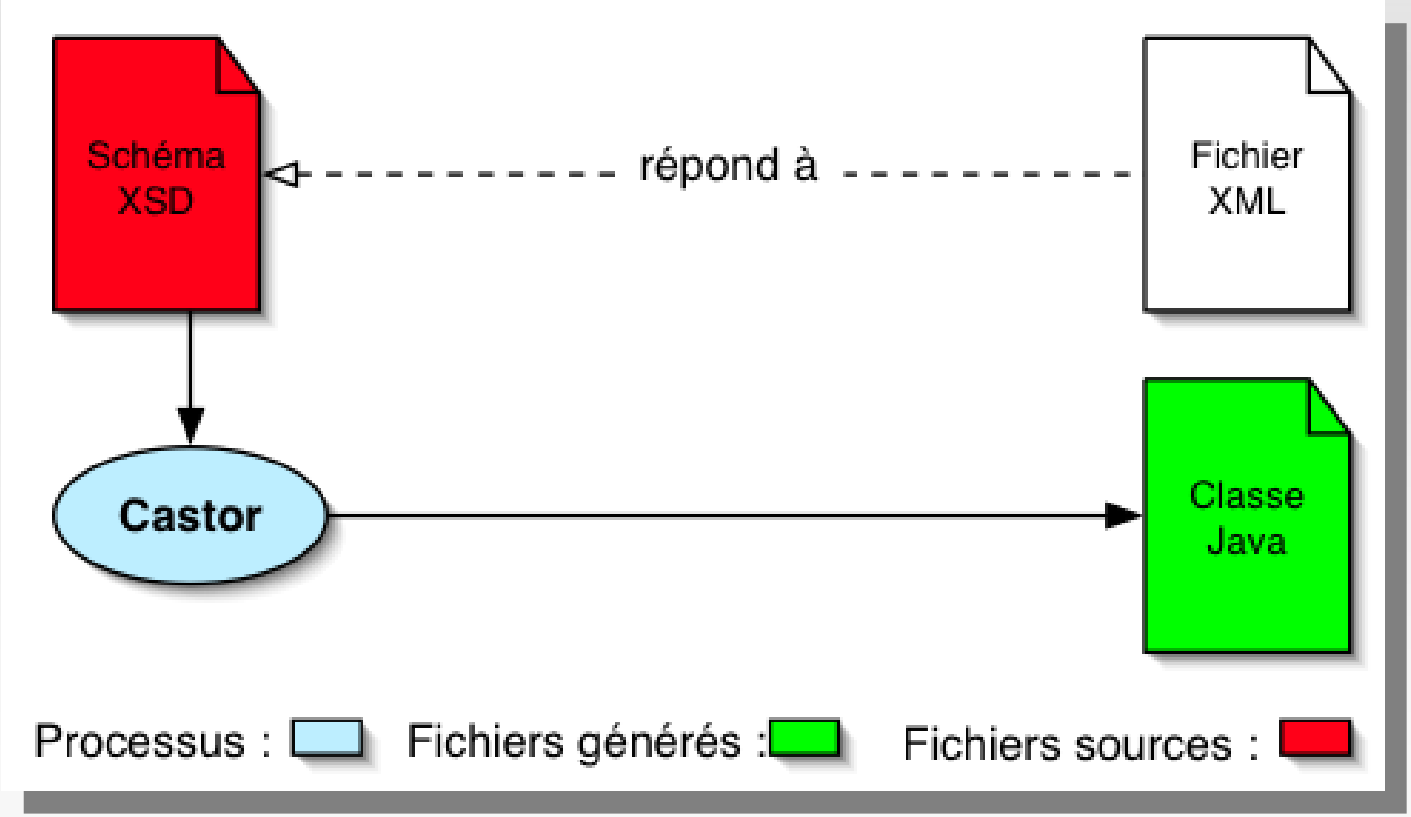
Structure du fichier XML

```
<ApplicationData link="">
	<program name="" version=""/>
	<compilation date="" compiler=""/>
	<text></text>
	<copyright></copyright>
	<dependencies>
		<package name="" description="" link=""/>
	</dependencies>
</ApplicationData>
```

Ecran de démarrage et fenêtre d'à-propos

Librairie Java opensource

Castor



Gestion des journaux d'exécution

- Besoins :
  - Récupérer tous les logs de toutes les classes
  - Stocker l'ensemble des logs pour de futures utilisations
  - Fixer le niveau de verbosité des logs en cours d'exécution
  - Filtrer les logs selon la classe à laquelle ils appartiennent

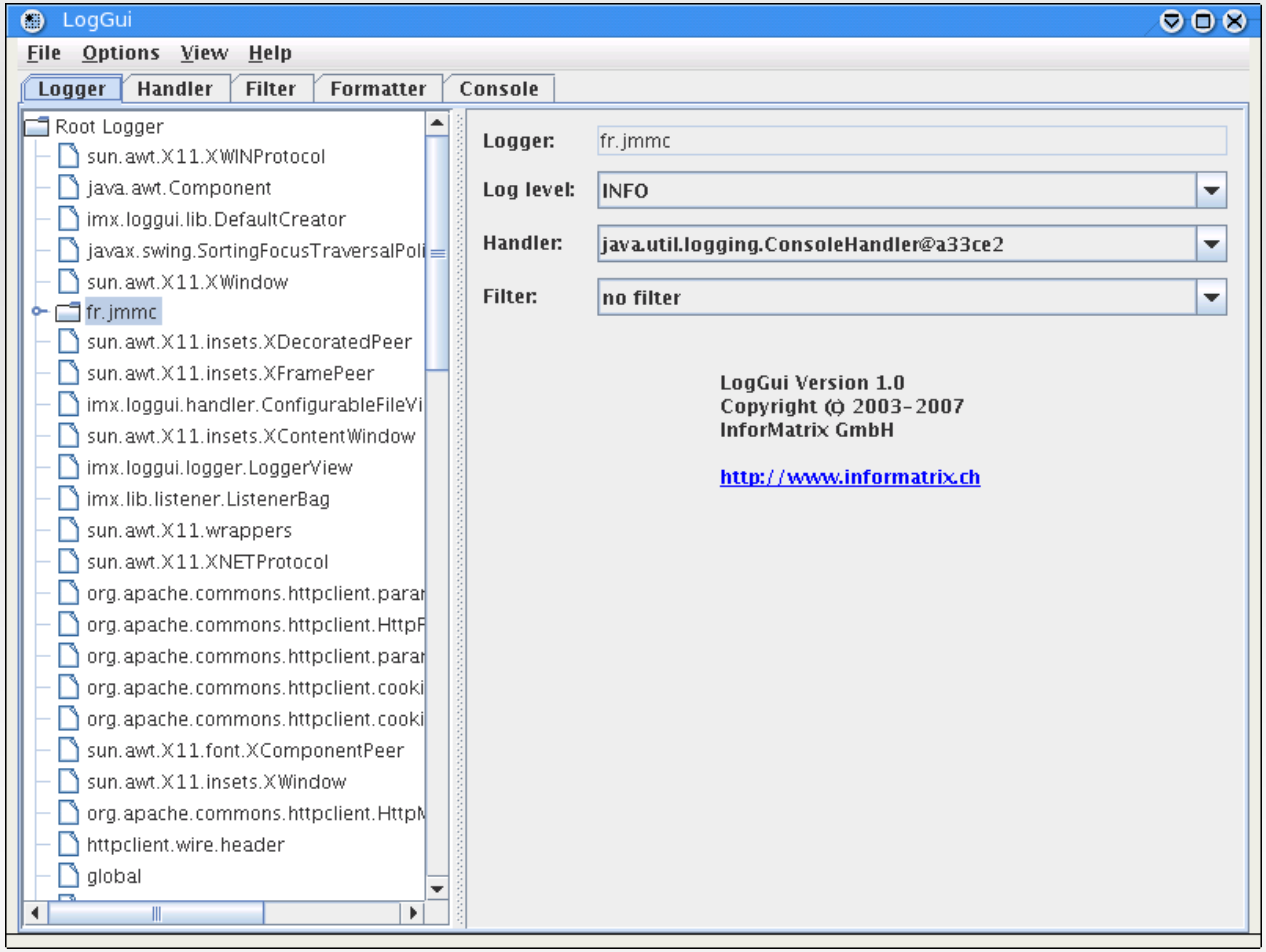
```
29 mai 2008 14:04:09 fr.jmmc.mcs.gui.App interpretArguments
INFO: Set logger level to 5
29 mai 2008 14:04:09 fr.jmmc.mcs.gui.App <init>
FINE: Application arguments interpreted
29 mai 2008 14:04:10 fr.jmmc.mcs.gui.App showSplashScreen
INFO: Show SplashScreen
29 mai 2008 14:04:10 fr.jmmc.mcs.gui.ApplicationDataModel getLogoURL
INFO: Logo URL = logo.png
```

Gestion des journaux d'exécution

Librairie Java opensource

LogGui

- Changer le niveau de verbosité à la volée
- Filtrer les traces selon les classes



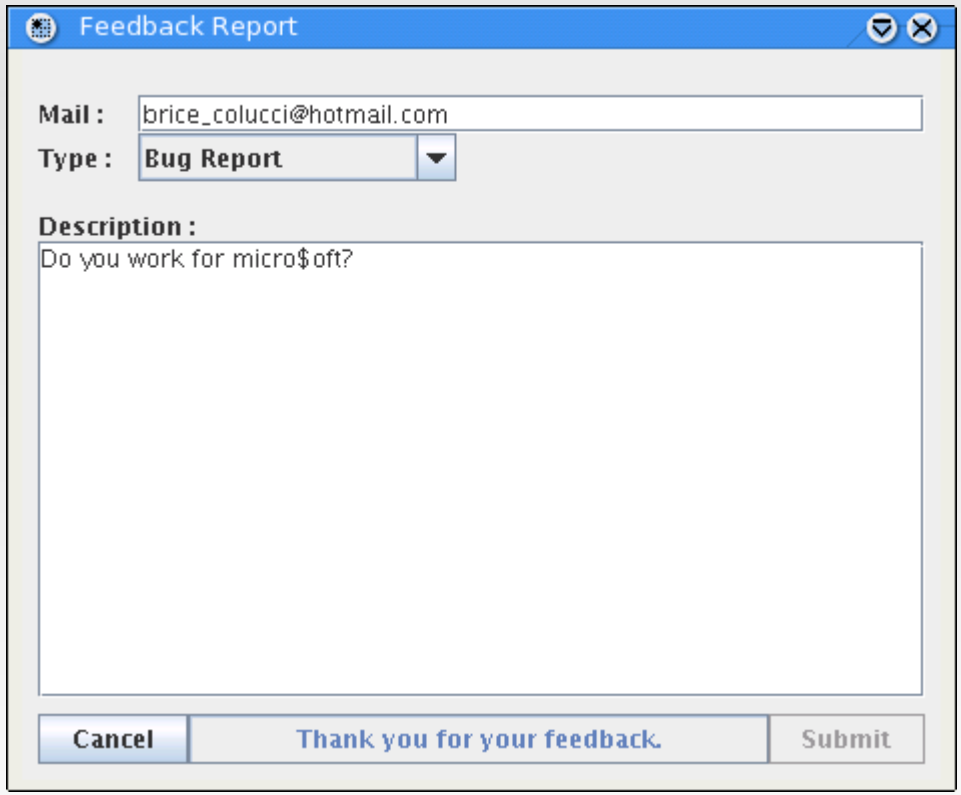
Gestion des retours d'erreurs et commentaires

- Besoins :
  - Corriger les bugs et donc améliorer la qualité des logiciels
  - Prendre en compte les remarques des utilisateurs à travers le retour de commentaire

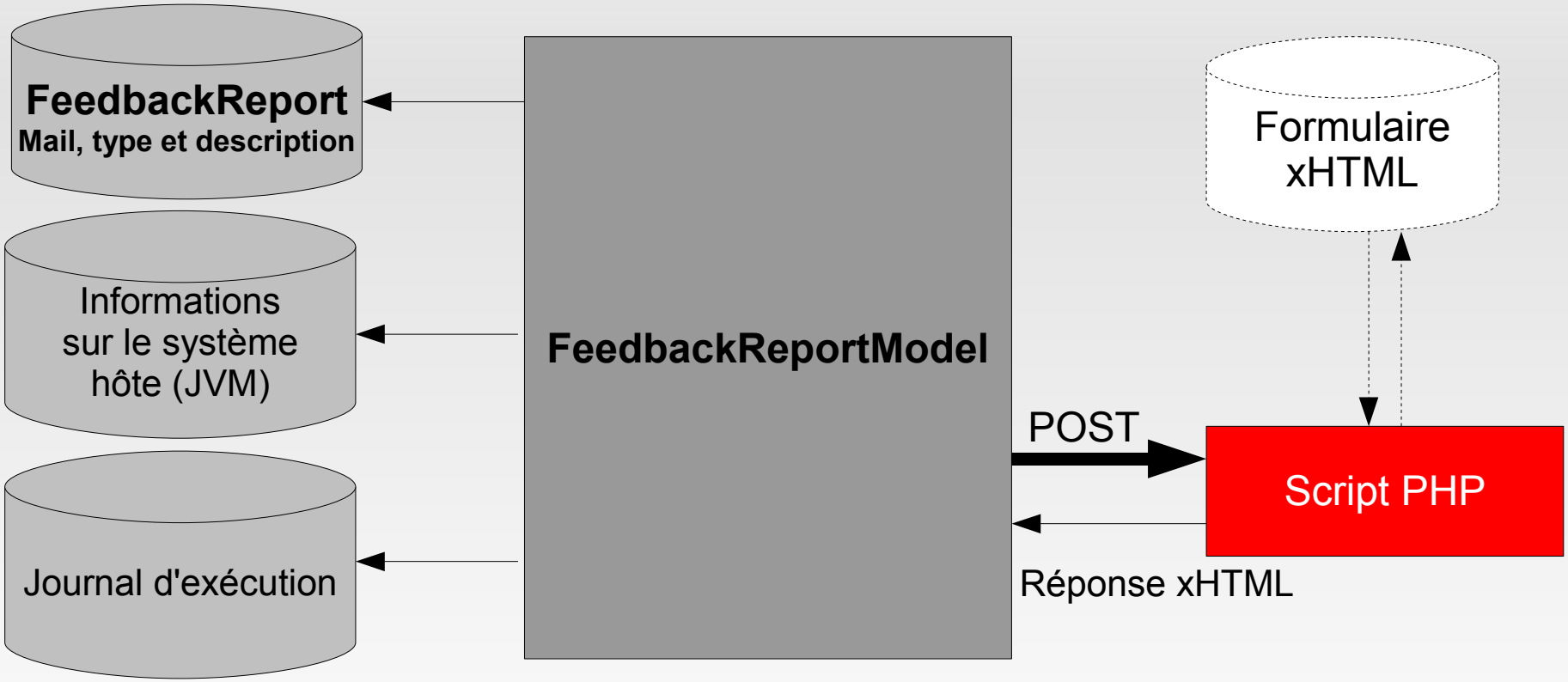
Gestion des retours d'erreurs et commentaires

Informations à transmettre :

- Adresse e-mail
- Type d'erreur
- Commentaire
- Propriétés du système hôte
- Journal d'exécution
- Informations spécifiques



Gestion des retours d'erreurs et commentaires



Génération et visualisation de l'aide utilisateur

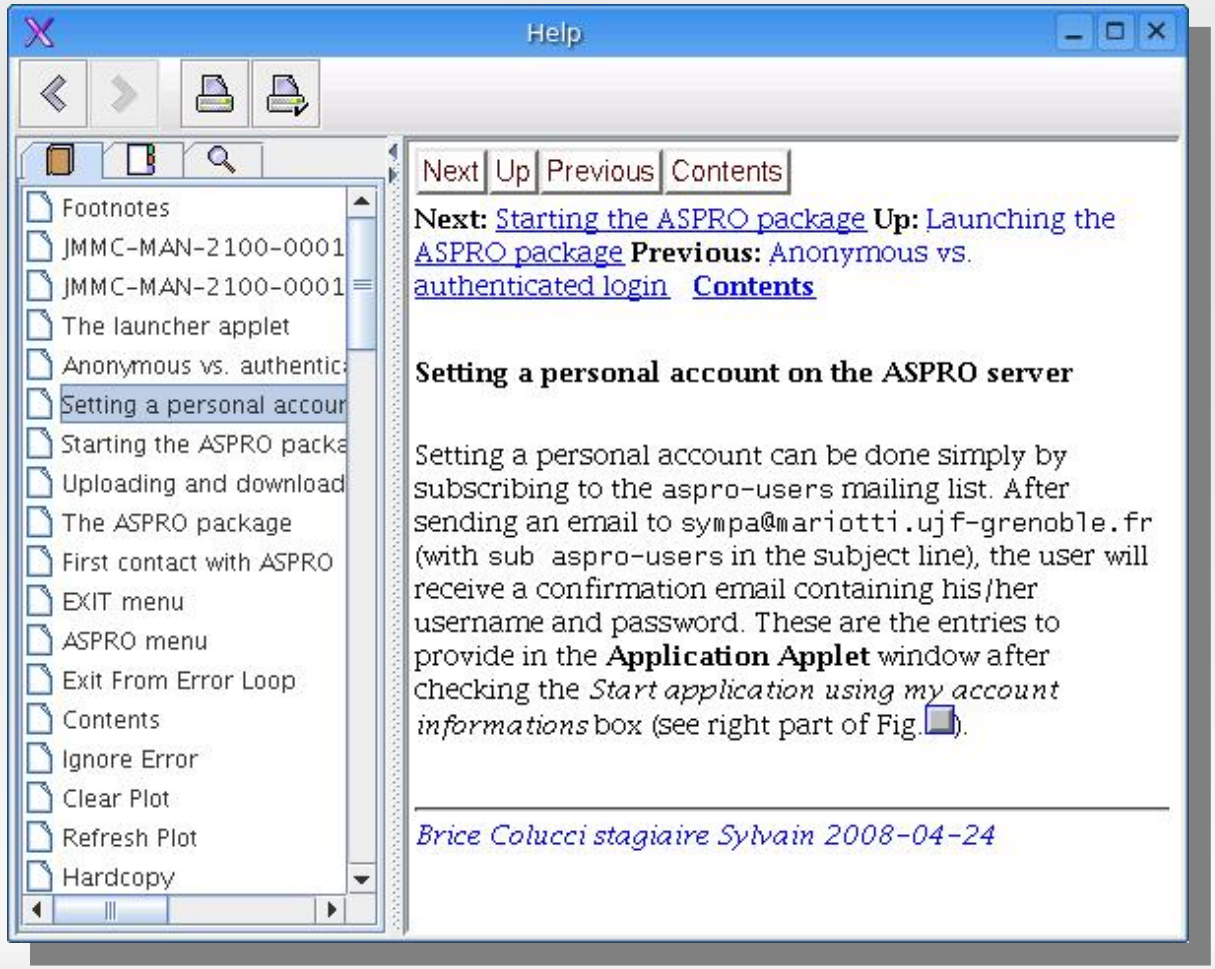
- Chaque module dispose d'une documentation (LaTeX)

- Une application peut utiliser plusieurs modules

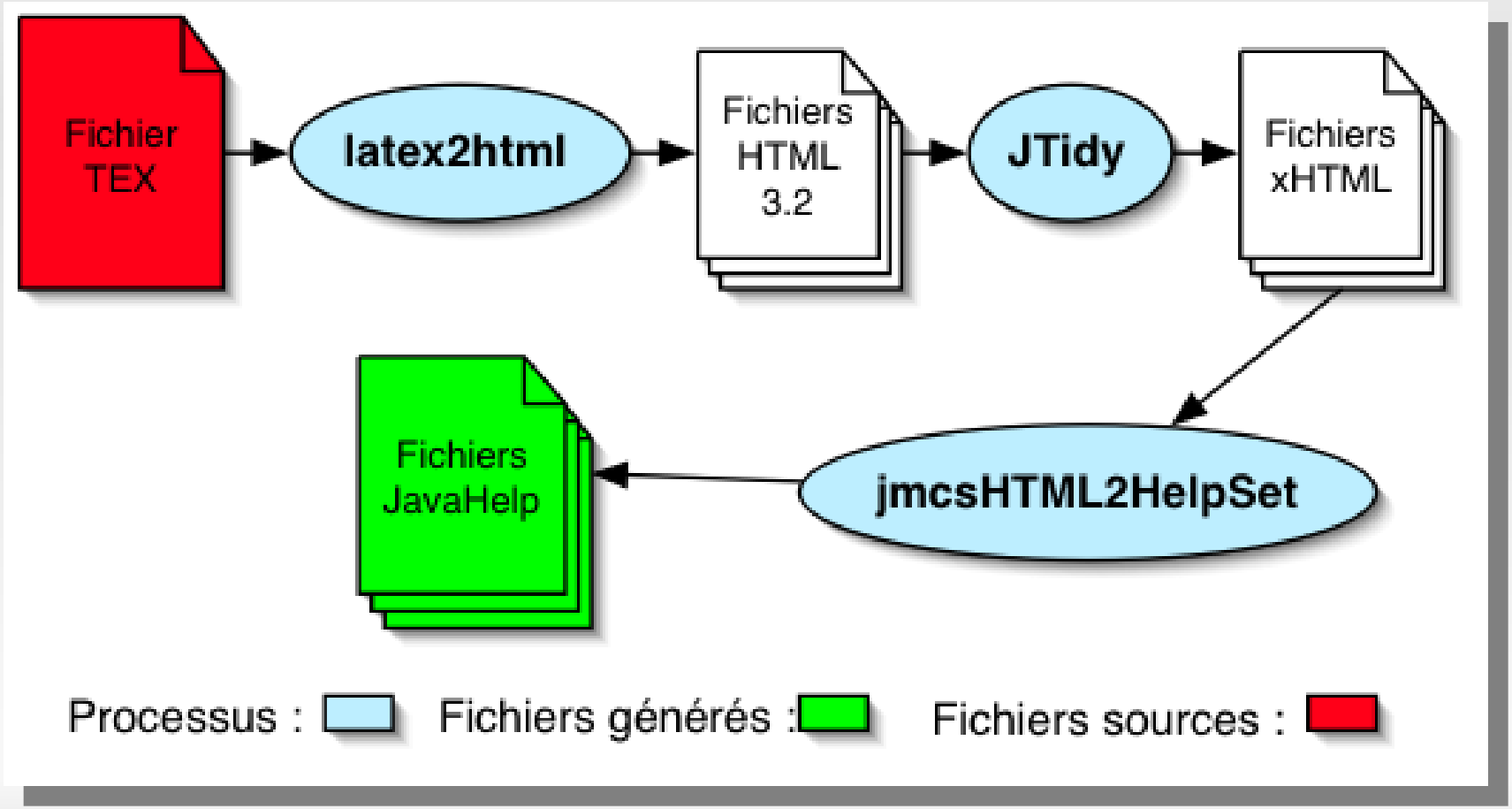
=> JavaHelp (Sun Microsystems)

  - TOC (Table Of Content)
  - JHM (Java Help Map)
  - HS (HelpSet)
  - Fichiers xHTML

=> JHelpDev



Génération et visualisation de l'aide utilisateur



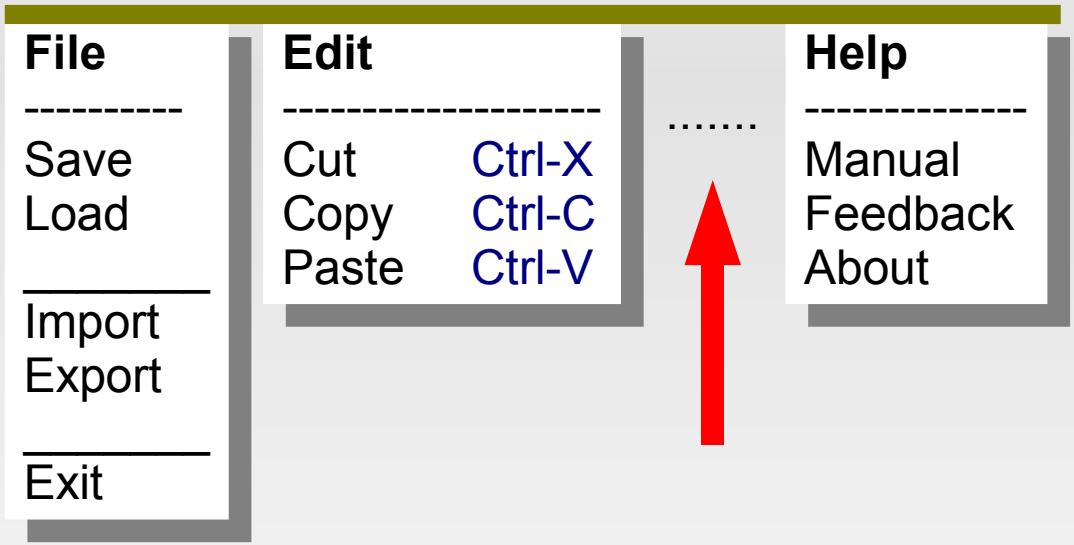
L'interface ”ligne de commande”

- Besoins :
  - Fixer le niveau de verbosité des traces
  - Afficher les informations étendues sur l'application
  - Afficher l'aide en ligne sur les arguments

Getopt — Librairie Java opensource

- -version : Nom et version de l'application
- -v[0|1|2...|5] : Verbosité
- -h : Aide en ligne

Uniformisation des menus et raccourcis claviers



- Générer automatiquement tous les menus standarts
- Pouvoir intercaler des menus dans l'ordre désiré

Uniformisation des menus et raccourcis claviers

Appel courant d'une fonction d'un objet

```
retour = monObjet.uneFonction(argument[])
```

Introspection Java (les classes sont compilées ...)

```
retour = uneFonction.invoke(monObjet, argument[])
```

L'objet ici est une instance d'une classe, il faut donc avoir au préalable récupéré la classe ...

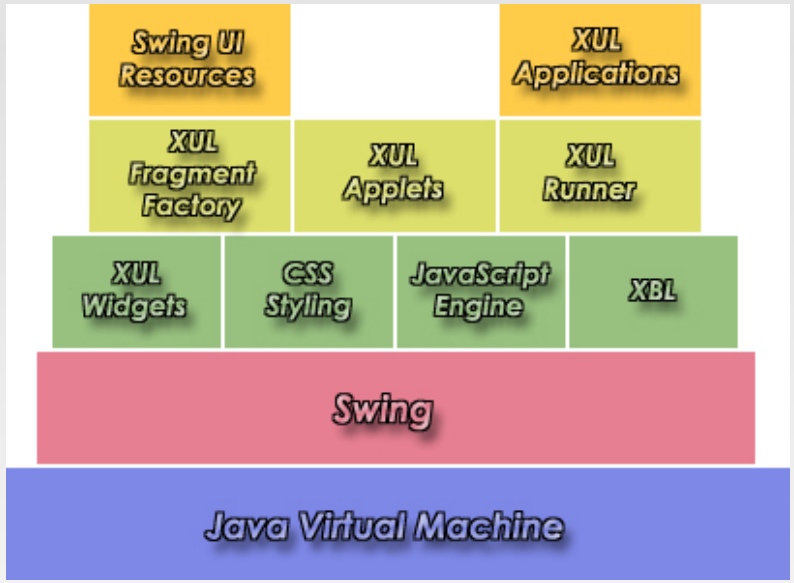
Très séduisant mais ne répond pas à tous les besoins!

Uniformisation des menus et raccourcis claviers

Librairie de conception d'applications basées sur un formalisme XML

JXUL (fonctionne avec le noyau XUL ...)

- Puissantes fonctionnalités
- Formalisme XML simple
- Implémentation invasive



Couche d'abstraction à SWING

=> Impossible d'utiliser en partie seulement les fonctionnalités offertes

Uniformisation des menus et raccourcis claviers

=> Ajout du formalisme des menus à notre fichier XML

```
<ApplicationData link="">
...
<menubar>
      <menu label="">
            <menuitem classpath="" action="" description="">
            <menuitem classpath="" action="" checkbox="" accelerator="">
            <menuitem/>
            <menuitem classpath="" action="" icon="" accelerator="">
      </menu>
</menubar>
</ApplicationData>
```

Utilisation de l'introspectivité mais plus ”simplement”

Documentation

- Documentation de développement

  Utilitaire de génération de documentation extraite du code source (comme javadoc)

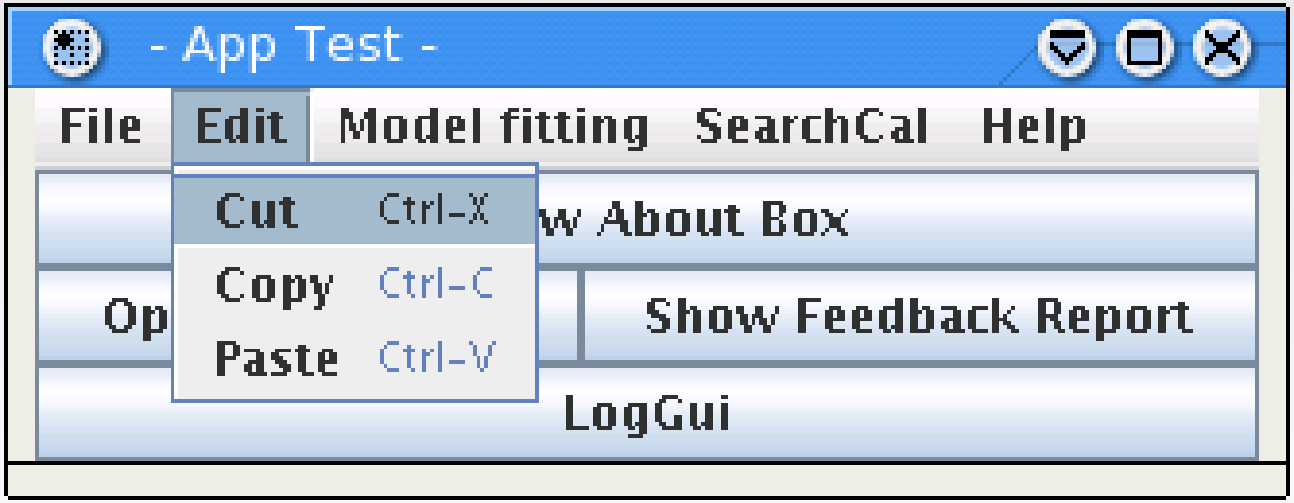
  Doxygen (doc API en pages HTML)

- Documentation utilisateur

=> Le code et les commentaires sont écrits en anglais

Mise en oeuvre

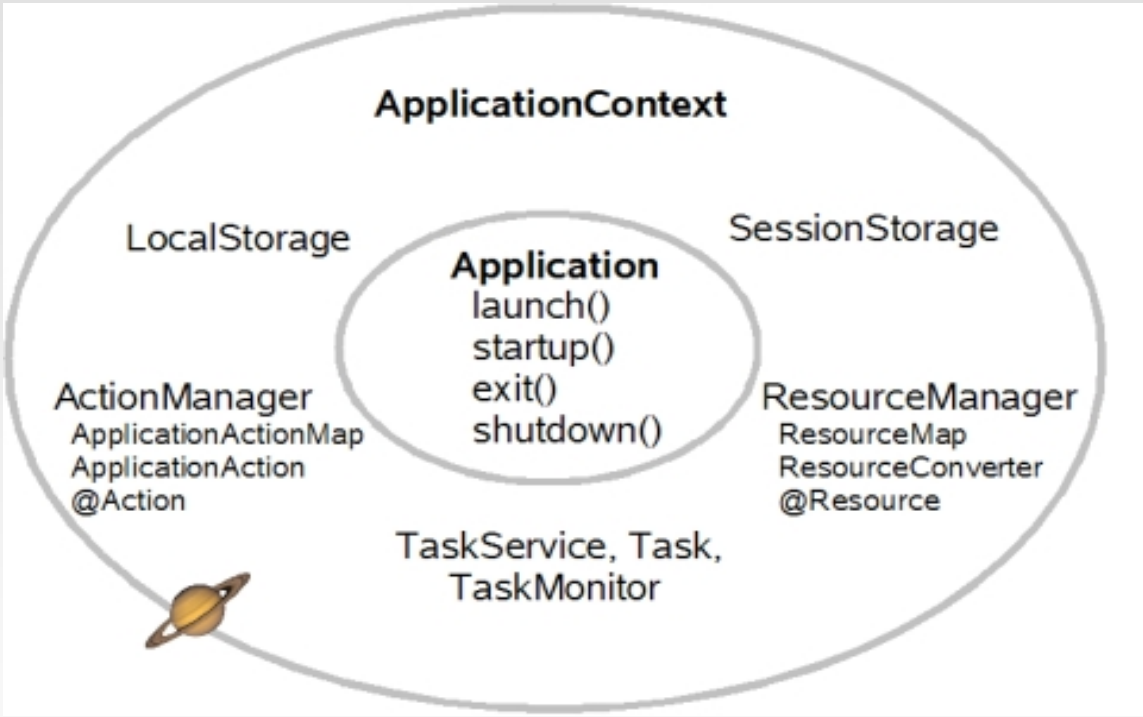
- Centraliser l'ensemble des fonctionnalités développées
- Généraliser leurs appels
- Pouvoir utiliser l'ensemble pour toute application



Mise en oeuvre

Framework de développement d'applications graphiques Java

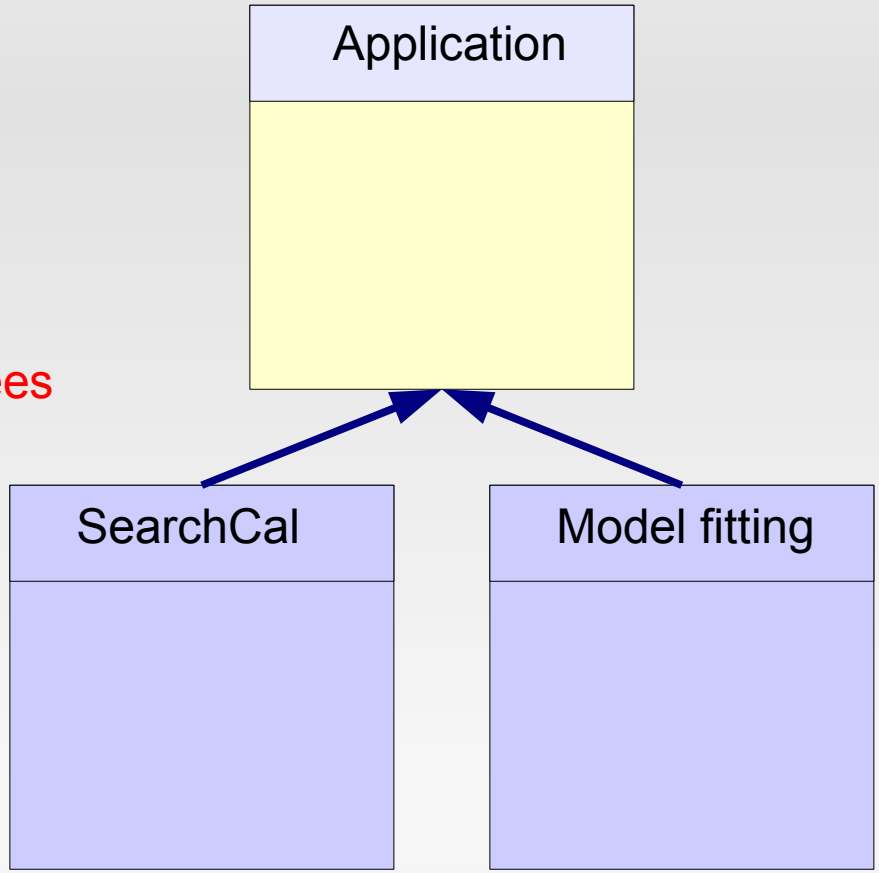
AppFramework (*Sun Microsystems*)



Mise en oeuvre

Centraliser l'ensemble des fonctionnalités => **dérivation**

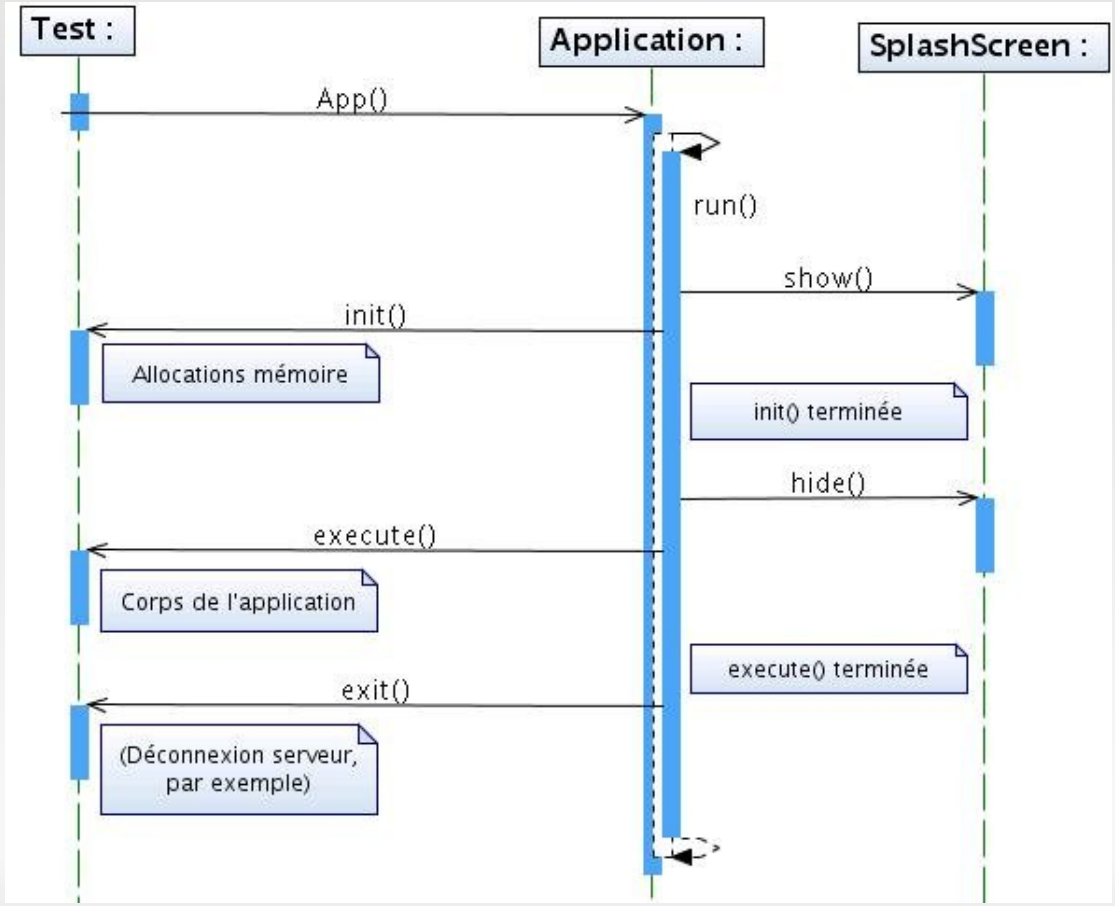
=> Hériter de toutes les fonctionnalités créées



Mise en oeuvre

Généraliser leurs appels => **abstraction**

- Imposer une structure unique
- Gérer le cyle de vie



Bilan du stage

- Rigueur de développement
- Utilisation de librairies
- Organisation :
  - Réunions d'équipe hebdomadaires
  - Lectures de code croisées
- Ambiance du laboratoire
- Domaine intéressant

Bilan du stage

Merci de votre attention!