Conseils stratégiques (développement JEE)

Tester très régulièrement/fréquemment (au fur et à mesure)

- Il est assez facile de trouver une anomalie dans 15 nouvelles lignes de code.
  Il est par contre très difficile de localiser une anomalie au sein d'un grand paquets de nouveaux fichiers non testés.
  ==> il faut donc mettre en place des tests unitaires au fur et à mesure de l'avancement de la programmation.
- Au moins un test unitaire par service métier (+éventuellement des tests unitaires pour les "DAO").
- Les pages JSP et les beans associés (actions ou managedBeans) doivent également être testés au fur et à mesure de l'avancement de la programmation.

Stabiliser l'architecture le plus tôt possible .

- Le code , le comportement et la configuration d'un composant ne sont parfaitement stabilisés qu'au sein d'une architecture bien mise en place.
  Autrement dit, remettre en cause l'architecture globale d'une application peut avoir d'énormes répercussions .
- L'architecture elle même est globalement validée par un assemblage réussi de différentes technologies (ex: JSF + Spring + Hibernate/JPA , avec ou sans maven) .
- Pour au moins vérifier le bon fonctionnement de l'architecture dans un cas simple il est primordial de mettre en oeuvre un squelette/prototype de l'application le plus tôt possible.
- Ce prototype doit être de type "90% technique , 10% fonctionnel" (c.a.d. 1 ou 2 tables relationnelles , 1 ou 2 entités persistantes , 1 seul service métier (+ 1 éventuel DAO) , 1 ou 2 tests unitaires , 1 bean "web" , 1 ou 2 pages JSP/XHTML et tous les fichiers de configurations nécessaires)

améliorations progressives (incréments fonctionnels)

- Apporter de nouvelles fonctionnalités que si les premières sont opérationnelles
- Relancer régulièrement tous les tests unitaires pour vérifier une "non régression" de l'application.
- Effectuer régulièrement des sauvegardes (.zip du code ou ...) de façon à pouvoir revenir en arrière dans un état de marche si par accident tout a été cassé .
- Améliorer les aspects techniques progressivement (+sécurité , + look évolué , ...)
- Ne pas hésiter à restructurer l'application si nécessaire (refactoring) .

En résumé (dans l'ordre):

1. Modéliser (fonctionnellement et techniquement) l'application (avec UML et ...)
2. Mettre en oeuvre la configuration (maven ou eclipse ou .....) du projet (bonnes librairies , ...)
3. Coder rapidement un embryon opérationnel de l'application (chaîne complète : "IHM + Services + persistance + DataBase") + tests unitaires .
4. Placer ce prototype dans un référentiel ("SVN" ou "GIT" , ...) et continuer incrémentalement (versions "0.1" , "0.2" , ... "0.9" , "1.0") .