

Qualité logicielle

V. Baudin

Veronique.Baudin@laas.fr

06/03/2008 16:37

1 Définition générale

En génie logiciel, la qualité est définie en termes de facteurs, parfois contradictoires, dont le choix doit s'effectuer en fonction du contexte.

Les facteurs généralement retenus sont :

- Validité : aptitude d'un produit logiciel à remplir exactement ses fonctions, définies par le cahier des charges et les spécifications.
- Fiabilité (ou robustesse) : aptitude d'un produit logiciel à fonctionner dans des conditions anormales.
- Extensibilité : facilité avec laquelle un logiciel se prête à une modification ou à une extension des fonctions qui lui sont demandées.
- Réutilisabilité : aptitude d'un logiciel à être réutilisé, en tout ou en partie, dans de nouvelles applications.
- Compatibilité : facilité avec laquelle un logiciel peut être combiné avec d'autres logiciels.
- Efficacité : Utilisation optimales des ressources matérielles.
- Portabilité : facilité avec laquelle un logiciel peut être transférée sous différents environnements matériels et logiciels.
- Vérifiabilité : facilité de préparation des procédures de test.
- Intégrité : aptitude d'un logiciel à protéger son code et ses données contre des accès non autorisés.
- Facilité d'emploi : facilité d'apprentissage, d'utilisation, de préparation des données, d'interprétation des erreurs et de rattrapage en cas d'erreur d'utilisation.

Par exemple pour un logiciel de type « gestion de rendez-vous », on privilégiera les facteurs facilité d'emploi et fiabilité par rapport à un critère portabilité, si la cible d'utilisateurs visée est essentiellement des linuxiens.

2 Spécificités pour DEVA

Dans le cadre de DEVA, deux facteurs de qualité importants seront la facilité d'emploi et l'extensibilité, dans la mesure où les logiciels proposés feront l'objet de diffusion à la fois auprès d'utilisateurs finaux, et auprès de contributeurs participant à la maintenance et à l'amélioration de ces logiciels.

A partir de là, on peut identifier les documents et informations qui devront enrichir le logiciel lui-même.

- Cahier des charges du logiciel (pourquoi, pour qui, environnement d'utilisation)
- Limitations dans l'utilisation du logiciel
- Document de conception du logiciel
- Description des outils nécessaires pour intervenir sur le code (environnement de développement, compilateurs, type de matériels et d'OS,
- Description de l'environnement nécessaire pour l'utilisateur final (installation de logiciels pré-requis, configuration spécifique de machines,
- Structure claire des différents fichiers : par exemple arborescence classique du type

```
PROJET/  
  build/  
  config/  
  docs/  
  generated/  
  lib/  
  log/  
  src/
```

- Le(s) manuel(s) d'installation et d'utilisation avec des exemples d'installation et d'utilisation.
- Des jeux de test permettant de vérifier l'installation et/ou des modifications de code

On peut compléter cette liste par quelques recommandations, en termes de codage :

- ◆ utiliser les normes définies souvent par langage,
- ◆ commenter le code,
- ◆ dans la mesure du possible utiliser un outil classique et OpenSource de gestion de version (SVN ou autre)

3 Bibliographie

Conventions de codage pour le langage Java : <http://java.sun.com/docs/codeconv/>

Des conseils pour le codage Java : <http://www.developer.com/java/data/article.php/3612756>

Conventions et conseils pour le codage en C++ http://www.research.att.com/~bs/bs_faq2.html

Projet **PLUME**

<http://www-sop.inria.fr/dream/rapports/devprocess-2003.pdf>