

# Résumé - Partie I

Qualité des Systèmes d'Information

ISTQB Foundation Level

## 1 La Qualité d'un SI

### 1.1 Définitions

**Qualité du SI :** Concordance aux besoins demandés par les différents intervenants du processus d'informatisation. Elle relève de l'existence ou l'absence des dysfonctionnements.

**Principe clé :** Le niveau de qualité est inversement proportionnel au risque. Quand la qualité croît, les risques décroissent.

### 1.2 Six composantes du succès d'un SI (Delone et McLean, 1992)

#### 1. Qualité du système technique

- Accès facile
- Temps de réponse court
- Outil pratique et efficace

#### 2. Qualité de l'information produite

- Précision de l'information
- Accessibilité
- Fiabilité

#### 3. Degré d'utilisation

- Temps réel d'utilisation
- Nombre de fonctionnalités utilisées

#### 4. Satisfaction de l'utilisateur

- Attitude de l'usager envers le produit

#### 5. Impact sur la performance individuelle

- Effet sur le temps et la qualité de la décision
- Gains en productivité
- Amélioration de la qualité de vie au travail

#### 6. Impact sur la performance de l'organisation

## 2 Qualité du Produit vs Qualité du Processus

### 2.1 Qualité du produit

**Produit :** Bien, service ou idée réunissant des attributs tangibles et intangibles qui satisfait les consommateurs.

**Qualité du produit SI :** Fondée sur la qualité de ses composants matériels et logiciels, sa conception, sa performance et sa capacité à répondre efficacement aux besoins.

### 2.2 Qualité du processus

**Processus** : Ensemble de moyens et d'activités liés qui transforment des éléments entrants en éléments sortants.

**Qualité du processus** : Ensemble des démarches nécessaires au développement, gestion, exploitation, maintenance et amélioration continue tout au long du cycle de vie.

## 2.3 Standards et référentiels

La gestion de la qualité des processus SI se base sur :

- **ITIL** (Information Technology Infrastructure Library)
- **CMMI** (Capability Maturity Model Integration)
- **ISO 27001** (sécurité de l'information)

## 2.4 Quand agir sur la qualité ?

**En amont** Plus efficace (prévention)

**En aval** Plus facile (correction)

# 3 Non-Qualité et Coût de Non-Qualité (CNQ)

## 3.1 Qu'est-ce que la non-qualité ?

**Non-qualité** : Écart entre la satisfaction attendue (ce que le client espère) et la satisfaction réelle (ce que le produit offre).

C'est la conséquence des dysfonctionnements lors d'une phase de réalisation du produit.

## 3.2 Types de non-qualité

**Sous-qualité** Performances inférieures au besoin

- Insatisfaction du client
- Rejet ou désengagement

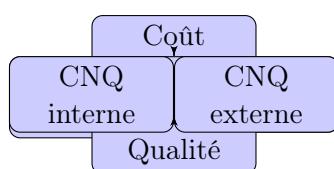
**Sur-qualité** Performances dépassant les besoins réels

- Produit peut sembler inutile ou complexe
- Ressources et coûts gaspillés

## 3.3 Chiffres clés

- La non-qualité peut atteindre **20% du CA** ou **35% de la valeur ajoutée** (entreprises européennes et américaines)
- Peut excéder **50% du budget de fonctionnement** (entreprises marocaines)
- Une démarche qualité coûte environ **10% du CA**
- **La non-qualité est plus coûteuse que la qualité**

## 3.4 Structure du Coût de Non-Qualité



### 3.5 Coûts de Non-Qualité Internes

Coûts liés aux dysfonctionnements **avant livraison au client** :

- Erreurs découvertes en qualification nécessitant correction et tests supplémentaires
- Retards dans le planning de livraison
- Fonctionnalités non opérationnelles sur certains navigateurs
- Réunions pour revoir les erreurs dans la documentation

### 3.6 Coûts de Non-Qualité Externes

Coûts liés aux dysfonctionnements **après livraison au client** :

- Développement et déploiement de correctifs d'urgence
- Adaptation de l'application pour résoudre les incompatibilités
- Mobilisation excessive des équipes d'assistance
- Coûts pour restaurer les données et renforcer la sécurité
- Pénalités financières en cas de violation des réglementations

### 3.7 Coûts de Prévention

Investissements pour anticiper et minimiser les dysfonctionnements :

- Outils d'analyse de code (SonarQube, Pylint)
- Méthodologies Agile ou DevOps
- Formation des développeurs aux bonnes pratiques
- Systèmes d'alerte et de monitoring

### 3.8 Coûts de Détection

Actions pour identifier les non-conformités avant livraison :

- Tests unitaires et fonctionnels
- Tests de montée en charge et de performance
- Revues et relectures du code
- Inspections et contrôles qualité

### 3.9 Formule de calcul du CNQ

$$\text{CNQ} = \text{CNQ interne} + \text{CNQ externe} + \text{Coûts de la qualité (détectio} n + \text{prévention})$$

## 4 Norme ISO/IEC 25000 - SQuaRE

### 4.1 Présentation

**SQuaRE** : System and Software Quality Requirements and Evaluation

**Objectif** : Créer un cadre pour l'évaluation de la qualité des produits logiciels

**Évolution** : Résultat de l'évolution des normes ISO/IEC 9126 (modèle de qualité) et ISO/IEC 14598 (processus d'évaluation)

## 4.2 Les cinq divisions de la norme

1. **Division du management de la qualité (ISO/IEC 2500n)**
  - ISO/IEC 25000 - Guide de SQuaRE
  - ISO/IEC 25001 - Planification et gestion
2. **Division des modèles de qualité (ISO/IEC 2501n)**
  - ISO/IEC 25010 - Modèles de qualité des systèmes et logiciels
  - ISO/IEC 25012 - Modèle de qualité des données
3. **Division de la mesure de la qualité (ISO/IEC 2502n)**
  - ISO/IEC 25020 - Modèle et guide de référence de mesure
  - ISO/IEC 25021 - Éléments de mesure de la qualité
4. **Division des exigences de qualité (ISO/IEC 2503n)**
  - ISO/IEC 25030 - Exigences de qualité
5. **Division de l'évaluation de la qualité (ISO/IEC 2504n)**
  - ISO/IEC 25040 - Modèle de référence et guide
  - ISO/IEC 25041 - Guide d'évaluation
  - ISO/IEC 25042 - Modules d'évaluation
  - ISO/IEC 25045 - Module d'évaluation de la récupérabilité

## 5 Caractéristiques de qualité ISO/IEC 25010

### 5.1 1. Adéquation fonctionnelle

Mesure dans laquelle un produit fournit des fonctions répondant aux besoins.

**Complétude fonctionnelle** L'ensemble des fonctions couvre toutes les tâches spécifiées

- Exemple : Trello sans gestion des deadlines affecte la complétude

**Exactitude fonctionnelle** Le produit fournit des résultats exacts

- Exemple : Application de caisse doit gérer les arrondis au centime près

**Pertinence fonctionnelle** Les fonctions facilitent l'accomplissement des tâches

- Exemple : Careem permet de réserver, suivre et payer facilement

### 5.2 2. Efficacité des performances

Performances en termes de temps et utilisation des ressources.

**Comportement temporel** Temps de réponse et débit adéquats

- Exemple : Édition d'un reçu > 10s = mauvais comportement temporel

**Utilisation des ressources** Quantités et types de ressources appropriés

- Exemple : CPU à 90% = mauvaise gestion des ressources

**Capacité** Limites maximales répondent aux exigences

- Exemple : YouTube diffuse des millions de vidéos simultanément

### 5.3 3. Compatibilité

Capacité à échanger des informations et fonctionner avec d'autres produits.

**Coexistence** Fonctionne avec d'autres produits sans impact négatif

- Exemple : Excel ne doit pas planter lors de l'ouverture de Word

**Interopérabilité** Échange d'informations avec d'autres produits

- Exemple : iOS et Android échangent du contenu multimédia

## 5.4 4. Capacité d'interaction

Degré d'utilisation par des utilisateurs pour accomplir des tâches.

**Appropriation-Reconnaissabilité** Les utilisateurs reconnaissent si le produit est approprié

- Exemple : Symboles universels (loupe pour recherche, panier)

**Capacité d'apprentissage** Les fonctions peuvent être apprises rapidement

- Exemple : Démonstrations lors du premier accès

**Exploitabilité** Facilité d'exploitation et de contrôle

- Exemple : Glovo permet des commandes rapides en peu de clics

**Protection contre les erreurs** Empêche les erreurs d'utilisation

- Exemple : Messages d'erreur instantanés pour champs non conformes

**Engagement de l'utilisateur** Interface attrayante et motivante

- Exemple : Reward récompense les utilisateurs qui marchent

**Inclusivité** Utilisable par un public diversifié

- Exemple : WhatsApp utilisable par tous âges et cultures

**Assistance à l'utilisateur** Aide disponible au besoin

- Exemple : Chatbots dans les sites de vente en ligne

**Autodescription** Informations présentées de manière évidente

- Exemple : "Vous n'êtes pas autorisé" au lieu de "404"

## 5.5 5. Fiabilité

Fonctionnement sans défaillance pendant une période donnée.

**Absence de défaillance** Exécution sans défaillance en conditions normales

- Exemple : Systèmes bancaires sans erreur de transaction

**Disponibilité** Opérationnel et accessible quand nécessaire

- Exemple : Site BLS souvent indisponible pour les RDV visa

**Tolérance aux pannes** Fonctionne malgré les pannes

- Exemple : Serveur de secours, accès hors ligne WhatsApp

**Récupérabilité** Récupération des données après interruption

- Exemple : Sauvegarde automatique Microsoft Office

## 5.6 6. Sécurité

Protection contre les attaques et accès non autorisés.

**Confidentialité** Données accessibles uniquement aux autorisés

- Exemple : Messages WhatsApp chiffrés de bout en bout

**Intégrité** Protection contre modification non autorisée

- Exemple : Gestion des autorisations dans Drive

**Non-répudiation** Preuve que des actions ont eu lieu

- Exemple : Google Docs trace toutes les modifications

**Responsabilité** Actions retracables jusqu'à l'entité

- Exemple : Logs pour tracer les utilisateurs et leurs actions

**Authenticité** Identité peut être prouvée

- Exemple : Double authentification Gmail

**Résistance** Reste opérationnel si attaqué

- Exemple : CNSS n'a pas résisté à la cyberattaque

## 5.7 7. Maintenabilité

Efficacité de modification pour amélioration ou correction.

**Modularité** Composé d'éléments distincts avec impact minimal

- Exemple : Module "Produits" séparé du module "Paiement"

**Réutilisabilité** Utilisable dans plus d'un système

- Exemple : Fonction de calcul TVA réutilisable

**Analysabilité** Facilité d'évaluer l'impact des modifications

- Exemple : Traçabilité détaillée dans les logs

**Modifiabilité** Modification sans introduire de défauts

- Exemple : Ajout de fonctionnalité sans dégrader les autres

**Testabilité** Facilité d'établir et exécuter des tests

- Exemple : Fonctionnalités principales facilement testables

## 5.8 8. Flexibilité

Adaptation à l'évolution des exigences et contextes.

**Adaptabilité** Transfert vers différents environnements

- Exemple : WhatsApp mobile, web et desktop

**Évolutivité** Gestion de charges croissantes ou décroissantes

- Exemple : Extensibilité des serveurs cloud

**Installabilité** Installation/désinstallation efficace

- Exemple : Installation facile d'Office 365

**Remplaçabilité** Peut remplacer un autre produit

- Exemple : Office 365 et Google Workplace

## 5.9 9. Sûreté

Éviter les états dangereux pour la vie, la santé ou l'environnement.

**Contrainte opérationnelle** Limite le fonctionnement à des états sûrs

- Exemple : Facebook désactive le compte si connexions suspectes

**Identification des risques** Détecte les événements à risque

- Exemple : Glovo bloque après multiples modifications de carte

**Sécurité** Retour automatique à un état sûr

- Exemple : PayPal bloque le compte si transaction suspecte

**Avertissement de danger** Alerte sur les risques

- Exemple : Navigateurs alertent sur sites non HTTPS

**Intégration sûre** Maintien de la sécurité lors de l'intégration

- Exemple : Interfaçage ONCF avec CMI sécurisé

# 6 Outils et Méthodes de Gestion de la Qualité

## 6.1 QQCOQP

Méthode pour analyser un problème :

**Quoi ?** Quel est le problème ?

**Qui ?** Qui est concerné ?

**Où ?** En quel lieu ?

**Quand ?** À quel moment ?

**Comment ?** Sous quelles formes ?

**Pourquoi ?** Quelles raisons de le résoudre ?

## 6.2 Diagramme d'Ishikawa (Arête de poisson)

Identifier les causes possibles d'un problème.

**Catégories 6M :**

- **Matière** : Matériaux ou données utilisées
- **Main-d'œuvre** : Personnes et compétences
- **Méthode** : Processus ou méthodologies
- **Matériel** : Outils, logiciels, machines
- **Milieu** : Environnement de travail
- **Mesure** : Métriques ou contrôles qualité

**Version adaptée (développement logiciel) :**

- Méthode, Matériel, Main-d'œuvre, Milieu, Mesure, **Money** (budget)

## 6.3 Diagramme de Pareto

**Principe 80/20 :** 80% des effets sont dus à 20% des causes

**Objectif :** Prioriser les causes ayant le plus d'impact

## 6.4 Cycle PDCA (Roue de Deming)

Méthode cyclique d'amélioration continue :

1. **Plan (Planifier)** : Identifier le problème et préparer un plan
2. **Do (Développer)** : Mettre en œuvre les actions à petite échelle
3. **Check (Contrôler)** : Mesurer les résultats et comparer aux objectifs
4. **Act (Agir)** : Standardiser les solutions efficaces ou ajuster

## 6.5 Méthode des 5 Pourquoi

Poser "Pourquoi ?" de manière itérative pour identifier la cause racine.

**Bonnes pratiques :**

- Impliquer les personnes directement concernées
- Rester factuel et précis
- Communiquer clairement les résultats