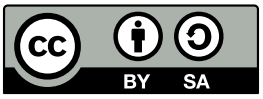
Un glossaire du Génie Logiciel

mailto:equipe-gl@telecom-bretagne.eu

V1.1.1



Ce document recense environ 100 mots ou expressions utilisés en génie logiciel. Quelques lignes définissent chaque entrée en faisant référence à d'autres mots du glossaire. Les références apparaissent en bleu et sont actives dans le document pdf.

Nous avons évité les termes techniques relatifs à un paradigme particulier. Quelques mots sont regroupés dans l'entrée mots techniques sans être définis.

Une classification subjective est proposée à l'aide de <tags> dont la liste suit.

Général management, artefact* [1], propriété, phase*, cycle de vie*, processus*, rôle, gestion de projet, qualité*, risque*

Point de vue ingénierie, méthode*, architecture*, moa*

Cycle de vie analyse*, besoin*, spécification*, conception*, réalisation*, test*, maintenance*

Niveau n1 à n3 [2]

Glossaire

agile Approche itérative de développement qui met en œuvre plusieurs pratiques du Génie Logiciel (GL) comme impliquer le client, livrer fréquemment des versions, à chaque itération, le besoin est ré-évalué, et tout ce qui est livré possède des tests. ⟨n2⟩ ⟨ingénierie⟩ ⟨cycle de vie⟩ ⟨management⟩.

analyse Phase initiale à tout développement. Elle consiste à *comprendre le problème*. Elle permet d'identifier les besoins et de les exprimer sous la forme d'exigences. Elle se termine en général par la production d'un cahier des charges. ⟨n1⟩ ⟨ingénierie⟩ ⟨phase⟩ ⟨ingénierie⟩.

architecture Décrit les éléments structuraux importants d'un système. Pour un logiciel on met en évidence les composants et les connecteurs qui les relient. On identifie des styles d'architecture. ⟨n2⟩ ⟨ingénierie⟩ ⟨artefact⟩ ⟨conception⟩.

1. Les tags avec une étoile sont eux-mêmes des entrées du glossaire.
2. Niveau 1 - connaissance de base pour l'ingénieur - ; Niveau 2 - vocabulaire génie logiciel général - ; au niveau 3 - vocabulaire génie logiciel expert informatique -

architecture (phase d') Début de la conception, cette phase consiste à identifier et évaluer différentes architectures de solution. La meilleure au regard des qualités recherchées est étudiée en profondeur dans la conception détaillée. ⟨n2⟩ ⟨phase⟩.

artefact Quelque chose produit par l'homme [3] : un document, un modèle, du code source, un schéma, etc. ⟨n2⟩ ⟨modèle⟩.

besoin Partie essentielle de l'analyse, l'identification des besoins permet de dresser la liste des fonctions attendues, de leurs propriétés et contraintes. Les besoins peuvent être formalisés par des exigences. ⟨n1⟩ ⟨ingénierie⟩ ⟨phase⟩ ⟨besoin⟩.

bogue Voir bug. ⟨n1⟩ ⟨test⟩.

boîte blanche (test en) Se dit d'un test effectué sur un système dont on connait la structure interne. On exploite cette connaissance pour guider les tests. Bien adapté à la vérification. ⟨n1⟩ ⟨test⟩ ⟨vérification⟩.

boîte grise Se dit d'un composant spécifié par un contrat. La partie blanche est celle exposée dans le contrat, la partie noire est celle cachée comme des choix de conception ou de réalisation. ⟨n3⟩ ⟨spécification⟩.

boîte noire (test en) Se dit d'un test effectué sur un système dont on ignore la structure interne. On observe le comportement sans faire d'hypothèse sur la structure interne. Bien adapté à la validation. ⟨n1⟩ ⟨test⟩ ⟨validation⟩.

bug Terme générique pour désigner un dysfonctionnement. On évite d'en produire par prévention à l'aide de méthodes et de processus rigoureux et de qualité. On les détecte et les corrige avant la recette avec des tests puis après lors de la maintenance. ⟨n1⟩ ⟨test⟩.

cahier des charges Document de référence produit à la fin de l'analyse. Il décrit propriétés fonctionnelles et les propriétés non fonctionnelles du système. Il synthétise les besoins, les exigences et les contraintes. ⟨ingénierie⟩ ⟨moa⟩ ⟨n1⟩ ⟨artefact⟩ ⟨besoin⟩ ⟨analyse⟩ ⟨qualité⟩.

client Le client (Maître d'ouvrage (MOA)) exprime ses besoins dans un cahier des charges lors de la phase d'analyse. Il participe également à la validation du produit en s'assurant que les besoins sont satisfaits. Dans un cycle de vie dit agile, le client peut également être sollicité dans les phases de conception et réalisation. ⟨n1⟩ ⟨moa⟩ ⟨rôle⟩.

code Concrétisation du travail de réalisation, le code est l'ensemble des fichiers (code source, exécutable, configuration, données) qui répondent aux besoins exprimés par le client. Il doit passer des tests de validation et de vérification. Pour gérer tous ces fichiers, on utilise un outil de gestion de configuration. ⟨n1⟩ ⟨réalisation⟩ ⟨artefact⟩.

code source Partie du code produit par des humains, ce qui exclut les exécutables (binaires) et les données. Ce sont donc les programmes des fonctions du système mais aussi de construction et configuration. ⟨n1⟩ ⟨réalisation⟩ ⟨artefact⟩.

3. En traitement du signal, d'image ou en électronique, un artefact est aussi produit par l'homme mais caractérise un défaut à éliminer. Ce n'est pas l'interprétation usuelle en génie logiciel.

cohésion Mesure informelle[4] du degré de focalisation qu'un composant logiciel a pour réaliser une fonction. On cherche généralement à augmenter la cohésion, en identifiant les responsabilités par exemple, pour simplifier la maintenance. De nombreuses métriques de cohésion existent. ⟨n2⟩ ⟨propriété⟩ ⟨architecture⟩.

collaboration Identifier les collaborations d'un composant permet de réfléchir à son couplage aux autres composants. ⟨n2⟩ ⟨propriété⟩ ⟨architecture⟩.

commentaire Partie du code écrit en langage naturel, à destination de diverses parties prenantes pour faciliter la compréhension et la maintenance du code. ⟨n1⟩ ⟨réalisation⟩⟨artefact⟩.

complexité cyclomatique Métrique qui compte le nombre de chemins d'exécution possibles dans une fonction. La probabilité qu'une erreur de programmation existe dans une fonction augmentent rapidement au-delà d'une complexité cyclomatique de 5. Invite donc à une décomposition fine des fonctions. ⟨n2⟩ ⟨propriété⟩ ⟨test⟩.

composant Unité de composition qui peut être spécifiée, réalisée, testée, déployée et assemblée avec d'autres composants à l'aide de connecteurs. Voir aussi module. ⟨n3⟩ ⟨architecture⟩ ⟨artefact⟩.

conception Phase qui suit l'analyse. Elle consiste à *trouver des solutions* au problème identifié. Les solutions sont évaluées au regard des exigences du client. On distingue souvent la conception globale ou architecture (phase d') et la conception détaillée. ⟨n1⟩ ⟨ingénierie⟩ ⟨phase⟩.

conception détaillée Seconde étape de la phase de conception, la conception détaillée consiste à retravailler (raffiner) avec plus de détails la solution retenue. ⟨n1⟩ ⟨ingénierie⟩ ⟨phase⟩ ⟨conception⟩.

configuration Pour être exécuté, le système doit être configuré. Cette opération consiste à décrire l'environnement d'exécution du code. Par exemple, l'adresse de la machine, l'emplacement d'une base de donnée, un mot de passe administrateur, etc. Il y a des configurations spécifiques aux développement, aux tests ou aux déploiements en conditions opérationnelles chez le client. ⟨n2⟩ ⟨ingénierie⟩ ⟨réalisation⟩.

connecteur Élement d'architecture qui permet d'assembler des composants. Des connecteurs classiques : l'appel de procédures/méthodes, les pipes Unix, des protocoles comme CORBA, RMI, SOAP, ... ⟨n2⟩ ⟨architecture⟩ ⟨artefact⟩.

construction Pendant la phase de réalisation, la construction consiste à produire le code du système à partir du code source. Cette activité est parfois décrite par un programme qui compile le code source, qui produit les composants qui seront livrés, et qui exécute des tests dans la configuration de développement. ⟨n2⟩ ⟨ingénierie⟩ ⟨réalisation⟩.

contrainte Expression dans un cahier des charges de règles qui doivent être respectées. On distingue des contraintes de temps, de coûts, techniques, etc. ⟨n1⟩ ⟨ingénierie⟩ ⟨besoin⟩.

4. Cohésion et couplage sont des concepts importants, mais compliqués. Une étude de 2008 analyse en profondeur l'évolution de ces concepts et conclut à l'absence de définition consensuelle.

contrat Ensemble des contraintes qui s'appliquent à un élément logiciel. Pour une fonction (ou une méthode), le contrat est un couple précondition/postcondition qui définissent les états acceptables avant (pré) et après (post) l'exécution de la fonction. On peut distinguer 4 niveaux de contrat (1) syntaxique, la signature de la fonction, (2) sémantique, le couple pré/postcondition, (3) synchronisation, l'ordre d'usage des fonctions dans une classe par exemple et (4) la qualité de service. ⟨n3⟩ ⟨besoin⟩ ⟨spécification⟩ ⟨conception⟩.

couplage Mesure informelle[4] du degré de connexion qu'un composant logiciel a avec d'autres. On cherche généralement à réduire le couplage, en identifiant les collaborations par exemple, pour simplifier la maintenance. De nombreuses métriques de couplage existent. ⟨n2⟩ ⟨ingénierie⟩ ⟨propriété⟩ ⟨architecture⟩.

couverture Pourcentage des lignes de code du programme qui ont été exécutées au moins une fois pendant les tests. Attention : une couverture de 100% ne garantit pas que tous les chemins d'exécution ont été testés (voir complexité cyclomatique). ⟨n1⟩ ⟨propriété⟩ ⟨test⟩.

CRC Méthode (Composant - Responsabilité - Collaboration[5]) pour identifier des composants en cherchant leurs responsabilités et leurs collaborations avec d'autres composants. Les responsabilités permettent de réfléchir à la cohésion et les collaborations au couplage. ⟨n2⟩ ⟨méthode⟩ ⟨analyse⟩ ⟨conception⟩.

cycle de vie Description du déroulement de la vie d'un produit. Précise l'ensemble des phases concernées et de leur enchaînement. Dans un développement logiciel, les phases principales sont : analyse, conception, réalisation, test. On nomme quelques cycles classiques : en V, incrémental, itératif, spirale, en Y, en X. ⟨n1⟩ ⟨ingénierie⟩ ⟨méthode⟩ ⟨gestion de projet⟩.

documentation Ensemble des documents (cahier des charges, plan de management, plan de gestion des risques, plan de test, modèles, etc.) hormis le code, produit pendant un développement. Les états des documents (brouillon, de travail, validé, public, privé, etc) ainsi que les dépendances entre eux sont gérés par un outil de gestion de configuration qui conserve les traçabilités. ⟨n1⟩ ⟨ingénierie⟩ ⟨analyse⟩ ⟨conception⟩ ⟨réalisation⟩ ⟨test⟩ ⟨artefact⟩.

effort Estimation ou mesure de l'effort nécessaire à la réalisation d'un projet. Est le produit d'une durée par une quantité de personnes (Homme.Mois par exemple). ⟨n3⟩ ⟨ingénierie⟩ ⟨gestion de projet⟩.

en V Se dit d'un cycle de vie dont la forme met en évidence une étape de fabrication (branche gauche du V) qui regroupe les phases d'analyse, conception et réalisation et une étape de Vérification & Validation (V&V) (branche droite du V) qui réunit les tests (test unitaire, intégration (phase d'), recette). Met en évidence la vérification et la validation. ⟨n1⟩ ⟨cycle de vie⟩ ⟨ingénierie⟩ ⟨méthode⟩.

en X Se dit d'un cycle de vie dont la forme met en évidence un cycle de vie de réutilisation. La partie haute du X correspond à un cycle de développement classique

5. Le nom réel est *Classe* Responsabilité Collaboration car cette méthode a initialement été développée pour la programmation orientée objet.

en V tandis que la partie basse du X correspond à un cycle en V inversé, dédié à la réutilisation. ⟨n3⟩ ⟨cycle de vie⟩ ⟨méthode⟩.

en Y Se dit d'un cycle de vie dont la forme met en évidence un découplage entre une partie du cycle dédiée à l'application, indépendamment de sa cible de déploiement, la partie gauche du Y et une partie dédiée à la plateforme cible. Les deux parties se réunissent dans le bas du Y pour construire l'application sur sa plateforme cible. Ce cycle de vie est associé à l'Ingénierie dirigée par les modèles (IDM). ⟨n3⟩ ⟨cycle de vie⟩ ⟨méthode⟩.

estimation C'est une activité d'ingénierie qui consiste à imaginer, prédire, le coût, la durée, l'effort de développement d'un système. Elle consiste à compter des éléments (fonctions, classes, IHM, etc) et à s'appuyer sur des connaissances statistiques (productivité par exemple) pour calculer les estimations. ⟨n3⟩ ⟨ingénierie⟩ ⟨gestion de projet⟩.

exigence Forme standard d'une expression du besoin d'un logiciel. Identification lors de l'analyse, traçabilité jusqu'à la recette. Plusieurs catégories : fonctionnelle, non-fonctionnelle, émergente, quantifiable. ⟨n1⟩ ⟨ingénierie⟩ ⟨besoin⟩ ⟨spécification⟩ ⟨artefact⟩.

fonction Calcul réalisé par un élément d'un système. Une fonction répond à un besoin exprimé par un client. Elle peut être spécifiée par un contrat. Elle doit passer des tests de validation et de vérification. ⟨n1⟩ ⟨ingénierie⟩ ⟨besoin⟩.

fonctionnel (point de vue) Point de vue sur un système qui se focalise sur ses fonctions. Permet de répondre aux questions « à quoi ça sert ? », « que fait-il ? », « quel est le service ? » etc. ⟨n1⟩ ⟨ingénierie⟩ ⟨analyse⟩.

génie logiciel Ensemble des processus, méthodes et outils utilisés pour développer des systèmes avec du logiciel (code). ⟨n1⟩ ⟨ingénierie⟩.

gestion de configuration Consiste à garder la mémoire de la fabrication d'un produit. Elle conserve la traçabilité de toutes les dépendances entre les artefacts (code, documents, modèles) pour s'assurer que tel document correspond bien à tel modèle et tel code. Elle gère également les versions de ces artefacts. ⟨n3⟩ ⟨ingénierie⟩ ⟨processus⟩.

incrémental Se dit d'un cycle de vie proche d'un cycle en V mais où, à partir de la conception détaillée et jusqu'à l'intégration (phase d'), le système est décomposé en incréments qui sont conçus, réalisés et testés unitairement indépendamment. ⟨n2⟩ ⟨ingénierie⟩ ⟨cycle de vie⟩.

intégration (phase d') Phase qui consiste à assembler les élément les uns avec les autres. Chaque élément doit avoir passé les tests unitaires. ⟨n1⟩ ⟨ingénierie⟩ ⟨phase⟩.

intégration continue Méthode d'organisation du développement adoptée dans les cycles de vie agile. Consiste à compiler, lancer les tests et des calculs de métrique le plus souvent possible (quotidiennement ou à chaque poussée de modification de code). ⟨n3⟩ ⟨processus⟩ ⟨test⟩ ⟨maintenance⟩.

itératif Se dit d'un cycle de vie dont la forme met en évidence le caractère itératif du développement. Les phases d'analyse, conception, réalisation et V&V s'enchaînent à chaque fois sur un système plus grand, intégrant de nouvelles fonctions ou propriétés. ⟨n2⟩ ⟨ingénierie⟩ ⟨cycle de vie⟩.

langage Un langage est, en général, le moyen utilisé pour la réalisation. On utilise des langages *de programmation* comme Python, Java, C, C++, Scala, Erlang, COBOL pour les calculs. On peut utiliser des langages de *description de données* comme SQL ou des langages de balise comme XML et ses dérivées comme HTML. Une partie du génie logiciel consiste à développer des langages spécialisés. ⟨n1⟩ ⟨réalisation⟩.

ligne de produits Artefact qui décrit plusieurs produits en identifiant leurs points communs et leurs différences afin d'anticiper leurs réalisations. Favorise la réutilisation. ⟨n3⟩ ⟨ingénierie⟩ ⟨artefact⟩ ⟨modèle⟩ ⟨méthode⟩.

maintenance Phase la plus longue de la vie d'un logiciel. Elle regroupe l'ensemble des évolutions d'un produit. On distingue trois types de maintenance : corrective (20%) - suppression des erreurs, adaptative (30%) - adaptation à l'évolution des matériels et logiciels externes, évolutive (50%) - évolution fonctionnelle du logiciel lui-même. ⟨n3⟩ ⟨ingénierie⟩ ⟨phase⟩.

méthode Description d'une démarche pour guider la résolution d'un problème. En GL, une méthode s'appuie en général sur un cycle de vie mais peut aussi décrire les processus à mettre en œuvre, les artefact à produire, les parties prenantes impliquées, etc. Quelques noms : Merise, Jackson System Development, IDEF (SADT), UML component, ... ⟨n1⟩ ⟨ingénierie⟩ ⟨méthode⟩.

métrique Mesure d'une propriété d'un produit ou d'une processus. On peut mesurer la taille, la cohésion, le couplage, le niveau d'imbrication, etc. ⟨n1⟩ ⟨qualité⟩.

modèle Terme général pour nommer une description simplifiée d'un système. Un modèle est réalisé avec un intention (expliquer, communiquer, calculer, construire). En GL, on construit des modèles du produit lors des phases d'analyse, de conception en incluant l'architecture. On distingue souvent plusieurs axes : structurel (point de vue), temporel (point de vue), fonctionnel (point de vue). ⟨n1⟩ ⟨ingénierie⟩ ⟨modèle⟩ ⟨artefact⟩.

module Unité d'organisation du code source. Un module rassemble un ensemble de programmes. On cherche à augmenter la cohésion des modules et à réduire leur couplage. Voir aussi composant. ⟨n2⟩ ⟨architecture⟩ ⟨artefact⟩.

mots techniques algorithme, complexité, langage, syntaxe, sémantique, structure de donnée, type, encapsulation, concurrence, classe, objet, héritage, hiérarchie, polymorphisme, liaison, surcharge, redéfinition, signature.

non régression Pratique s'assurant que la maintenance n'invalide pas les tests déjà effectués. En pratique, les tests sont accumulés et exécutés aussi souvent que possible comme le préconisent les approches agiles. ⟨n2⟩ ⟨test⟩ ⟨maintenance⟩ ⟨qualité⟩ ⟨propriété⟩.

norme Ensemble de documents élaborés par des organismes de standardisation (AFNOR, OMG, ISO, etc.). Permet de capitaliser et partager des connaissances, ainsi que d'assurer l'interopérabilité des outils. Qualité : série ISO 9000 dont la ISO 9126 pour un modèle et la série ISO 25000. Processus logiciel : ISO 12207. Risque : série ISO 31000. ⟨n3⟩ ⟨ingénierie⟩ ⟨risque⟩ ⟨qualité⟩.

oracle Lors de la phase de test, c'est un mécanisme qui permet de prédire le résultat d'une fonction. Si la calcul et l'oracle donnent le même résultat, on dit que le test passe, sinon il échoue. Dans ce cas, il faut remettre en cause la réalisation de la fonction ou, plus rarement, l'oracle lui-même. ⟨n2⟩ ⟨test⟩.

outils Des logiciels ou des services utilisés lors de toutes les phases de développement. Production de documentation, éditeurs de modèles, éditeur de code, débogueur, outils de gestion de configuration, outils de communication (éditeurs partagés, courriel, messagerie instantanée, etc.), outils d'analyse statique (calcul de métriques), site d'intégration continue, etc. Certains outils sont rassemblés dans un Integrated Development Environment (IDE). ⟨n1⟩ ⟨artefact⟩.

partie prenante Regroupe l'ensemble des acteurs d'un projet. On y trouve les acteurs des processus comme le client, l'analyste (analyse), les concepteurs (conception), et des spécialistes (IHM, Base de données, langage de programmation, réseau, tests, etc). ⟨n1⟩ ⟨processus⟩ ⟨rôle⟩.

patron Description d'éléments de solution à des problèmes récurrents. Un patron est en général caractérisé par un nom, un problème, des éléments de solution, des propriétés (avantages, inconvénients) et des exemples d'usage. On trouve des patrons de conception, d'organisation, d'architecture, etc. ⟨n3⟩ ⟨conception⟩.

pattern Voir patron.

phase Période d'un cycle de vie dédié à répondre à certaines questions et à produire des artefacts. Une phase se termine en général par une décision formelle lors d'une réunion où de la validation d'un document. ⟨n1⟩ ⟨ingénierie⟩ ⟨cycle de vie⟩ ⟨méthode⟩.

plan de gestion des risques Document de référence qui décrit les risques identifiés, leur suivi et les actions de réduction ou de maîtrise envisagées. ⟨n3⟩ ⟨ingénierie⟩ ⟨artefact⟩ ⟨risque⟩.

plan de management Document de référence qui décrit l'organisation d'un projet. On y retrouve des éléments concernant (1) le contexte et l'objectif du projet, (2) des éléments du cahier des charges, (3) l'organisation des parties prenantes - Organisation Breakdown Structure (OBS), (4) une décomposition du produit en lots - Product Breakdown Structure (PBS), (5) une décomposition des activités - Work Breakdown Structure (WBS), (6) un planning, (7) une analyse des risques. ⟨n3⟩ ⟨artefact⟩ ⟨ingénierie⟩ ⟨qualité⟩ ⟨gestion de projet⟩.

plan de test Document de référence qui décrit la stratégie de test et les moyens utilisés. Il est produit à partir des exigences lors de l'analyse pour les tests de validation et lors de la conception pour la vérification. ⟨n3⟩ ⟨ingénierie⟩ ⟨test⟩ ⟨artefact⟩ ⟨qualité⟩.

processus Formalisation d'un ensemble d'activités, des acteurs (et leur rôles) et responsabilités, des produits (artefacts). ⟨n2⟩ ⟨ingénierie⟩ ⟨processus⟩ ⟨gestion de projet⟩.

programme Partie du code écrit avec un langage de programmation. On peut distinguer les programmes qui réalisent les fonctions du système, de ceux pour construire ou configurer le système. ⟨n1⟩ ⟨réalisation⟩ ⟨artefact⟩.

propriétés fonctionnelles Ce que fait le système, ses fonctions. Décrites dans le cahier des charges, elles doivent passer la validation pour la recette. ⟨n1⟩ ⟨propriété⟩ ⟨validation⟩.

propriétés non fonctionnelles Propriétés qui caractérise le fonctionnement du système. Le choix de ces propriétés guide les choix d'architecture et de conception. Une liste est présentée dans la norme ISO 9126 (Les 6 principales : Capacité fonctionnelle, Fiabilité, Facilité d'utilisation, Rendement/efficacité, Maintenabilité, Portabilité)). Décrites dans le cahier des charges, elles doivent passer la validation lors des tests systèmes pour la recette. ⟨n1⟩ ⟨qualité⟩ ⟨propriété⟩ ⟨validation⟩.

qualité Mesure le niveau de conformance d'un produit à ses exigences explicites (cahier des charges) et implicites (tête du client). La qualité (standard ISO 9126) est décomposée selon plusieurs facteurs de qualité : Maintenabilité, Portabilité, Rendement/Efficacité, Facilité d'utilisation, Fiabilité et Capacité fonctionnelle. ⟨n1⟩ ⟨qualité⟩ ⟨management⟩ ⟨ingénierie⟩ ⟨gestion de projet⟩.

raffinement Résultat de l'action de raffiner. ⟨n2⟩ ⟨conception⟩.

raffiner C'est décrire avec plus de précision. S'applique en particulier à la phase de conception où les « plans » (modèles) sont décrits avec de plus en plus de détails. ⟨n2⟩ ⟨conception⟩.

réalisation Phase de mise en œuvre de la solutions décrite lors de la conception détaillée. Se traduit par la production d'artefacts comme des programmes, des bases de données, des fichiers de ressources (image, son, vidéo), de configuration, etc. ⟨n1⟩ ⟨ingénierie⟩ ⟨phase⟩.

recette Phase où le client s'assure du respect du cahier des charges en réalisant la validation du produit. ⟨n1⟩ ⟨ingénierie⟩ ⟨phase⟩.

responsabilité Identifier les responsabilités d'un composant permet de réfléchir et justifier la frontière de ce composant et de chercher à améliorer sa cohésion. ⟨n2⟩ ⟨propriété⟩ ⟨architecture⟩.

réutilisation Méthode consistant à développer en réutilisant (*Develop by reuse*) le plus d'artefacts possibles (code, mais aussi besoin ou spécification). En complément, il est nécessaire de mettre en pratique des techniques favorables à la réutilisation (*develop for reuse*) comme l'approche objet. Le cycle de vie en X explicite cette démarche. ⟨n1⟩ ⟨ingénierie⟩ ⟨méthode⟩.

revue Activité d'amélioration de la qualité qui consiste à faire relire un document ou un code. La revue peut être faite par un expert seul ou par un groupe de pairs, lors d'une réunion ou pendant un temps donné. Pour le code cette activité est complémentaire au test. ⟨n1⟩ ⟨qualité⟩.

risque Problème potentiel dont l'occurrence peut mettre en échec un projet. En GL les risques principaux sont (1) le manque de personnel, (2) les plannings sous-estimés, (3) le développement de mauvaises fonctions ou (4) de mauvaises interfaces, (5) vouloir trop bien faire, (6) les demandes permanentes de changement, etc. ⟨n3⟩ ⟨management⟩ ⟨ingénierie⟩ ⟨risque⟩ ⟨gestion de projet⟩.

source Voir code source. ⟨n1⟩ ⟨réalisation⟩ ⟨artefact⟩.

spécification Description la plus précise possible des propriétés d'un système ou d'une fonction. Sert de point de départ à une méthode pour, par raffinements successifs lors de la conception et de la réalisation, réaliser le code. Une spécification peut être une documentation informelle ou un contrat plus formel. ⟨n1⟩ ⟨ingénierie⟩ ⟨méthode⟩.

spirale Cycle itératif piloté par les risques. ⟨n3⟩ ⟨cycle de vie⟩ ⟨ingénierie⟩ ⟨méthode⟩.

structurel (point de vue) Point de vue sur un système qui se focalise sur ses éléments constitutifs. Permet de répondre aux questions « est composé de quoi ? », « contient quoi ? », « où est ? » etc. ⟨n1⟩ ⟨ingénierie⟩ ⟨analyse⟩.

style La plupart des systèmes mélangent plusieurs architectures. Les plus connues sont : client-serveur, pipe&filter, en couches, N-tiers, publish-subscribe, etc. ⟨n3⟩ ⟨architecture⟩.

système Quelque chose auquel on s'intéresse. Par exemple un logiciel, un processus, un équipement, une réaction physique ou chimique, etc. Se décrit au travers de modèles. Selon la nature du système, différentes méthodes peuvent être utilisées. ⟨n1⟩ ⟨ingénierie⟩.

temporel (point de vue) Point de vue sur un système qui se focalise sur sa dynamique (le temps). Permet de répondre aux questions « quand ? », « et après ? ? », « est-ce simultané ? » etc. ⟨n1⟩ ⟨ingénierie⟩ ⟨analyse⟩.

test Phase qui suit la réalisation et qui consiste à s'assurer de la vérification et de la validation de ce qui a été réalisé. On distingue classiquement les tests unitaires des test d'integration. C'est également un morceau de code qui sert à tester un élément logiciel. Il permet de comparer le résultat attendu (oracle) avec le résultat calculé. ⟨n1⟩ ⟨ingénierie⟩ ⟨test⟩ ⟨phase⟩ ⟨validation⟩ ⟨vérification⟩.

test d'integration Test particulier qui consiste à la vérification et la validation de l'assemblage de plusieurs éléments. L'assemblage peut se faire en ajoutant un par un les éléments et en utilisant des morceaux de codes spécifiques pour simuler les éléments manquants. On utilise plutôt des tests boîte noire (test en). ⟨n1⟩ ⟨test⟩ ⟨architecture⟩.

test fonctionnel Test d'un élément du système pour validation de ses exigences fonctionnelles. ⟨n1⟩ ⟨test⟩ ⟨validation⟩.

test système Test de l'ensemble du système pour validation des exigences fonctionnelles ou non-fonctionnelles. ⟨n1⟩ ⟨ingénierie⟩ ⟨test⟩ ⟨validation⟩.

test unitaire Test particulier qui consiste à la vérification et la validation d'un seul élément, en général une fonction ou une classe. Compte tenu de la taille - petite - du système sous test, on peut appliquer des tests boîte blanche (test en) ou boîte noire (test en). ⟨n1⟩ ⟨test⟩ ⟨réalisation⟩.

traçabilité Enregistre les dépendances d'un artefact aux autres artefacts qui ont permis de le produire. ⟨n2⟩ ⟨ingénierie⟩ ⟨maintenance⟩.

validation Action ou processus consistant à s'assurer que quelque chose répond aux besoins. (*To do the right thing.*) On valide en réalisant des tests fonctionnels et des tests systèmes. ⟨n1⟩ ⟨ingénierie⟩ ⟨moa⟩.

vérification Action ou processus consistant à s'assurer que quelque chose est réalisé en suivant les règles de l'art. (*To do the thing right.*) On vérifie en réalisant des tests dits boîte blanche (test en). ⟨n1⟩ ⟨ingénierie⟩.

version Une version d'un artefact (code ou document) permet de conserver la traçabilité de ses modifications et évolutions. Des outils spécialisés de gestion de versions existent comme : mercurial, git, svn ou cvs. ⟨n1⟩ ⟨ingénierie⟩ ⟨artefact⟩.

Acronyms

GL Génie Logiciel.

IDE Integrated Development Environment.

IDM Ingénierie dirigée par les modèles.

MDE Model-Driven Engineering, IDM.

MOA Maître d'ouvrage.

MOE Maître d'œuvre.

OBS Organisation Breakdown Structure.

PBS Product Breakdown Structure.

SWEBOK Guide to the Software Engineering Body of Knowledge, `https://luiscastellanos.files.wordpress.com/2007/03/swebokv3.pdf`.

UML Unified Modeling Language.

V&V Vérification & Validation.

WBS Work Breakdown Structure.

Références

[1] Ian Sommerville (?). Software engineering glossary. `https://ifs.host.cs.st-andrews.ac.uk/Resources/Notes/General%20SE/glossary.pdf`, visité le 10/2/2016. 26 pages.

[2] Université de Mons (Belgique). Software engineering glossary. Avec liens et références. Plus de 200 entrées.

[3] Philippe Deschamp. Le RÉTIF : termes informatiques français. http://deschamp.free.fr/exinria/RETIF/, visité le 10/2/2016. La terminologie informatique officielle en français. Ce n'est pas un glossaire.

[4] IEEE, editor. *610.12-1990 IEEE Standard Glossary of Software Engineering Terminology*. IEEE, September 1990. inclut des références très techniques...datées. 83 pages. http://dis.unal.edu.co/~icasta/ggs/Documentos/Normas/610-12-1990.pdf, visité le 10/2/2016.

[5] Pierre Parrend. Génie logiciel complément au cours 'génie logiciel', mia, sciences-u. http://www.rzo.free.fr/docs_genielog/lexique_genielog.pdf, visité le 10/2/2016, 2005. 5 pages.

[6] REQB. Standard glossary of terms used in requirements engineering. Technical report, Requirements Engineering Qualifications Board, 2011. 24 pages. http://en.gasq.org/fileadmin/user_upload/redaktion/en/Data/REQB_Standard_glossary_of_terms_used_in_Requirements_Engineering_1.0.pdf, visité le 10/2/2016.

[7] techno science.net. Génie logiciel. http://www.techno-science.net/?onglet=glossaire&definition=10830, visité le 10/2/2016. Définition du génie logiciel. Ce n'est pas un glossaire.

[8] C. Lawrence Wenham. Glossary of software engineering terms. http://www.yacoset.com/Home/glossary, visité le 10/2/2016. 1 page, 47 entrées. Pour rire....

Entrées pour ⟨n1⟩

Entrées pour ⟨n2⟩

Entrées pour ⟨n3⟩

Entrées pour ⟨management⟩

Entrées pour ⟨phase⟩

Entrées pour ⟨rôle⟩

Entrées pour ⟨moa⟩

Entrées pour ⟨artefact⟩

Entrées pour ⟨modèle⟩

Entrées pour ⟨cycle de vie⟩

Entrées pour ⟨méthode⟩

Entrées pour ⟨processus⟩

Entrées pour ⟨ingénierie⟩

Entrées pour ⟨risque⟩

Entrées pour ⟨qualité⟩

Entrées pour ⟨gestion de projet⟩

Entrées pour ⟨besoin⟩

Entrées pour ⟨analyse⟩

Entrées pour ⟨spécification⟩

Entrées pour ⟨architecture⟩

Entrées pour ⟨conception⟩

Entrées pour ⟨réalisation⟩

Entrées pour ⟨test⟩

Entrées pour ⟨validation⟩

Entrées pour ⟨vérification⟩

Entrées pour ⟨maintenance⟩

Entrées pour ⟨propriété⟩

Tous les tags

⟨n1⟩ ⟨n2⟩ ⟨n3⟩ ⟨management⟩ ⟨phase⟩ ⟨rôle⟩ ⟨moa⟩ ⟨artefact⟩ ⟨modèle⟩ ⟨cycle de vie⟩ ⟨méthode⟩ ⟨processus⟩ ⟨ingénierie⟩ ⟨risque⟩ ⟨qualité⟩ ⟨gestion de projet⟩ ⟨besoin⟩ ⟨analyse⟩ ⟨spécification⟩ ⟨architecture⟩ ⟨conception⟩ ⟨réalisation⟩ ⟨test⟩ ⟨validation⟩ ⟨vérification⟩ ⟨maintenance⟩ ⟨propriété⟩