(17ème Siecle) 1642 : première ordinateur inventé par Pascal (il a 19ans) Balaise
→ addition, soustraction, multiplication et division.

Hardware : general, concret (côté matériel)
Software : Specifique, abstrait (coté logiciel).
Ce que nous appelons progiramme → architecture de l'ordinateur
computer's architecture

1936-1938 : le premier ordinateur binaire (mécanique)
Build by Konizd Zuse. (Zuse S1)
Destroyed in 1943.
Frequence d'horloge (clock speed) : 1 Hz.
Progiammable Through punch cards (cartes perforées)

Ordinateurs électronique :
(US) 1937-1942 : Atanasoff berry, not progiammable.
(UK) 1943 : Colossus, progiammable, Decrypt Geimen's communicat=
(US) 1945 : ENIAC, progiammable, fast (100kHz)

Ada Lovelace 1842 : écrit un algorithme destiné à exécuté par une machine.
1957 : publie un article → logiciel

21 juin 1948 : First Software by Tom Kilburn. run on Manchester baby
⇒ The highest factor of 262,144

1951 - 1952 : A-0 System by Grace Hopper
1959 : COBOL (Common Business Oriented Language)
1966 : Apollo Guidance Computer, Margaret Hamilton.

1968 : First NATO Conference on Software engineering

	nb ligne
First Software	: 17
Univ v1.0	: 10k
Libreoffice	: 9M
Android	: 15M
Facebook	: 62M
Google	: 2B

Amphi 2:
16/09/2022.

Software engineering (Génie Logiciel):
l'ensemble des activités de conception et de mise en oeuvre des produits et des procédures tendant à rationaliser la production du logiciel et son suivi.
* started in The October 1968 NATO conference on Software.
SWEBOK: Software engineering body of knowledge.
Published by IEEE computer society (initiated in 1998, last version: 2013)

15 domaines:

- Les exigences logiciels
- Conception de Logiciels
- Construction de Logiciels
- Tests logiciels
- Maintenance logiciel
- Gestion de la configuration Logicielle
- Gestion de l'ingénierie Logicielle
- Processus de génie logiciel

Contrôle/planification/Audit

- Modèles et méthodes de génie Logiciel
- Qualité de logiciel
- Pratique professionnelle de génie logiciel
- Economie de génie logiciel
- Fondations informatiques
- Fondements mathématiques
- Fondations d'ingénierie

← Différents types des exigences.
• Analyse des exigences
• Quantification et validation des exigences.

Fermeture révision planification

→ • Architecture logiciel
• Interface utilisateur
→ • Le code / tests unitaires
• Le choix des langages
→ • Différents type de tests
• " niveaux de tests
• Mesures des tests.
→ • Coût de maintenance
• Différents type de maintenance
• technique de maintenance.

→ • Définit° de qualité
• Configurat° qualité
• Coût qualité

"No silver Bullet" by Fred Brooks, 1987.

• Qualité du logiciel:
ISO: degré auquel un 9000/2015 ensemble de caractéristiques inhérents à un objet répond aux exigences

ISO: International Organization for Standardization
System Orgenization.

Exemples de qualité logiciel : Fiabilité / sécurité / performance / Compatibilité / Portabilité / Ergonomie

4 types de coût de qualité : Prévention / Evaluation / Défaillance interne / Défaillance externe.

Listes de Contrôle : liste des éléments à vérifier lors de l'inspection réelle / une liste par langage de programmat°.

Amphi 3 : 23/09/22.

Qualité de logiciel :

Software Quality Assurance (SQA) : Assurance qualité des logiciels

→ Ensemble d'actions visant à garantir que le matériel est de qualité appropriée. Pas de tests, vérifier que 90% du code est testé unitairement.

Standardization bodies : Organismes de normalisation.

IEEE : Institute of Electrical and Electronic Engineers

AFNOR : Association Française de Normalisation.

ANSI : American National Standard Institute.

Quality standards - ISO 25000 series Normes de qualité - Séc ISO25xxx

Publié depuis 2005., 5 partie : 2500x Gestion de la qualité / 2501x Modèle de qualité / 2502x Mesure de la qualité / 2503x Exigences de qualité / 2504x Evaluation de la qualité

Quality standards - ISO 25010 series Série ISO25010

Conforme à la norme ISO 9126.

8 caractéristiques de qualité du produit : Aptitude fonctionnelle / Fiabilité / Efficacité de la performance / Utilisabilité / Sécurité / Compatibilité / Maintenabilité / Portabilité

Formal software inspection Inspect° formelle des logiciels.

Définition de NASA : Processus d'évaluation technique au cours duquel l'évaluation technique est un processus au cours duquel un produit est examiné dans le but de trouver et d'éliminer les défauts et anomalies le plus tôt possible dans le cycle de vie du produit.

Caractéristiques : le contrôle est effectué par des personnes techniquement° compétentes. l'auteur du produit est activement impliqué.

les processus: Préparation / Inspection / Retrouveille / Suivi / (A). Planificat° et présentation par l'auteur.

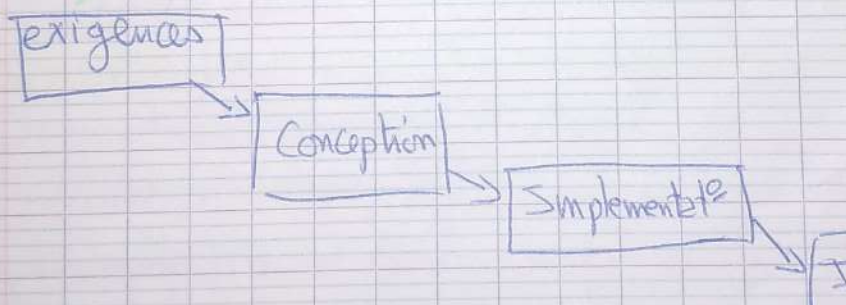
Projet de logiciel: Software project.

=> est l'ensemble de procédure et des activités visant à réaliser un produit logiciel prévu.

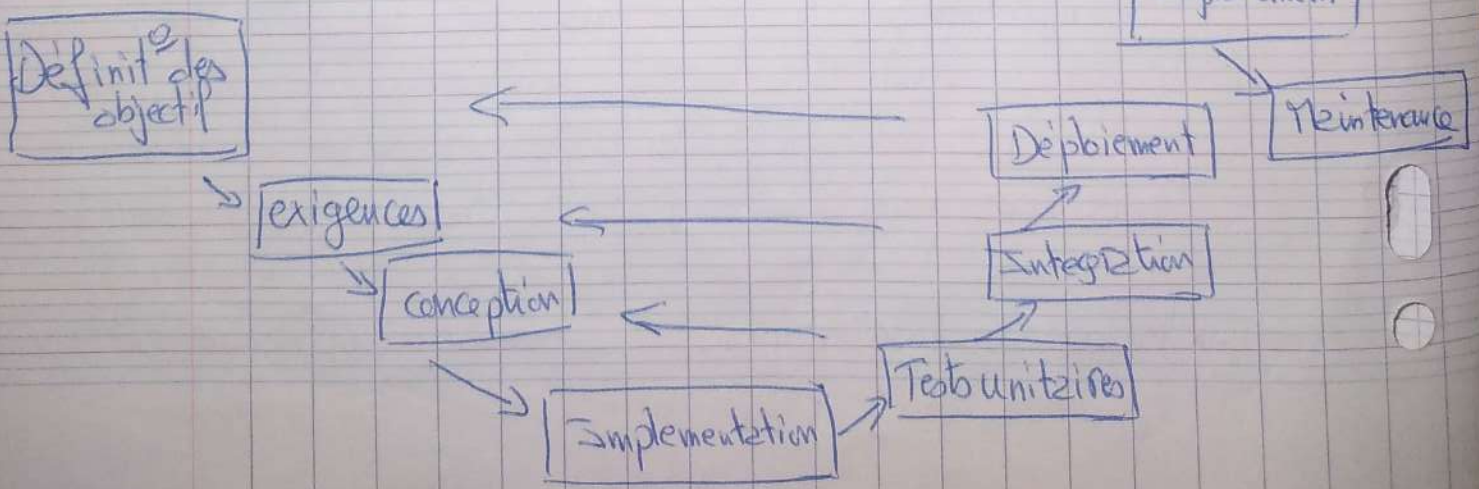
* Some people of a software project: Maître d'ouvrage / Maître d'oeuvre.
* Définition of The Scope: Définition du champ d'application. => mélange temps, coût, qualité
* Processus logiciels: est un ensemble d'activités et de tâches liées entre elles qui transforment des produits d'entrée en produits de sortie.
* Principales activités logicielles: définition des objectifs / Analyse des besoins / Analyse de faisabilité / Spécifications des besoins / Conception / Mise en oeuvre.

Software Development Life Cycle (SDLC): Cycle de vie du développement logiciel.

Waterfall model: Proposed by Royce in 1970.



V- model: extension de la cascade



Incremental model
Spiral model

SDLC ⇒ 4 models.
Different levels of complexity
Solut° depuis 2000s

Amphi 4: 30/09 22.

- structure arborescente avec au moins 3 niveaux : niveau 1 nom de projet / niveau 2 principales activités mises au paravent / niveau 3 et plus : sous tâches.

WBS : Work Breakdown Structure : structure décomposition du travail.

OT : Organigramme des tâches (OT).

Règles de WBS ou OTP : 5 règles :

1/ Il doit s'agir d'une structure arborescente

2/ chaque tâche doit être clairement définie, y compris les produits livrables potentiels.

3/ chaque tâche doit avoir une action finale claire

4/ chaque livrable doit être associé à une tâche.

5/ la réalisation de chaque sous-tâche implique la réalisation de la tâche principale.

PERT : Program Evaluation and Review Technique (Technique d'évaluation et de révision des programmes) : est une méthode d'analyse de différentes tâches du projet. En particulier, permet d'analyser : Dépendances entre les tâches / Durée des tâches / Durée du projet.

- Estimation du temps : $= \frac{o + p + 4m}{6}$

dont o, le temps optimiste (tout se passe parfaitement) poids = 1
p, le temps pessimiste (tout va mal) poids = 1
m, le temps le plus probable, poids = 4.

- chemin critique : l'ensemble des tâches qui permettent d'obtenir le temps le plus court pour terminer le projet.

* Conséquence : Si l'une des tâches du chemin critique prend plus de temps pour être exécutée, le projet prendra plus de temps pour se terminer.

* Algorithme du chemin critique :

1/ Sélectionner les tâches dont la date de fin est la plus tardive.
2/ Placer la tâche sélectionné dans le chemin critique.
3/ Sélectionner le prédécesseur de la tâche sélectionné avec la dernière date de fin.
4/ Répétez les étapes 2-3 jusqu'à ce que vous atteignez le noeud de départ.

diagramme de Gantt: Introduced by Henry Gantt around 1910

Estimation des coûts:

Méthode 1: COCOMO (COnstructive COstMOdel) (1981)

$coût = \alpha \times kLOC^{\beta} + \gamma$

α : coût marginal par 1000 lignes code.
γ : coût fixe d'un projet
β : facteur d'échelle.

Méthode 2: personnes/hommes.

* Types de risques:

Classification 1: Humain / Gestion / Technique.
" 2: Processus / Qualité / Viabilité
" 3: Impact uniquement sur un projet donné ou sur tous les projets.

Valeur attendue = probabilité d'un risque x coût.

Eléments d'un projet logiciel:
- Nombreuses activités pour un projet de logiciel.
- La planification doit être l'une des premières choses à faire.
- Doit correspondre à la portée du projet.

07/10 22

UML = Unified Modeling Language

Il provient de 3 notations graphique (OMT from James Rumbaugh, Booch method from Grady Booch, OOSE from Ivar Jacobson).

Publish UML 1.0 in 1997, From 1.1, développé et normalisé by The Object Management Group (OMG)

UML (2.0): (2005) évolut° majeure avec de nouveaux types de diagrammes. / UML 2.5.1 (2017) dernière version à partir de 2021.

Langage de modélisation graphique.

objectifs: Fournir une descript° d'un logiciel / Permettre la visualisation des différents aspects d'un logiciel / Analyse du logiciel.

→ indépendant du processus

→ Langage graphique permettant de modéliser des systèmes et des processus.

les Vues:

Un modèle est composé de plusieurs vues.

Une décrit un système sous différent angles.

Exp: Structurel view / Behavioral view / Interaction view

donne des infos sur le comportement du modèle.

donne des infos sur la façon dont les différentes parties du modèle se comportent les uns par rapport aux autres.

Types de diagrammes:

UML définit 13 diagrammes en 3 catégories.

Diagrammes de structure:

Diagr- de classe, d'objets, de composants

Diagrammes de comportement:

Dig- des activités, diagramme de la machine

Diagrammes d'interaction:

Dig de séquence, dig de communication.