Ingénierie Logicielle

Evolution du Logiciel

Prof. Ousmane SALL, Maître de Conférences CAMES en Informatique

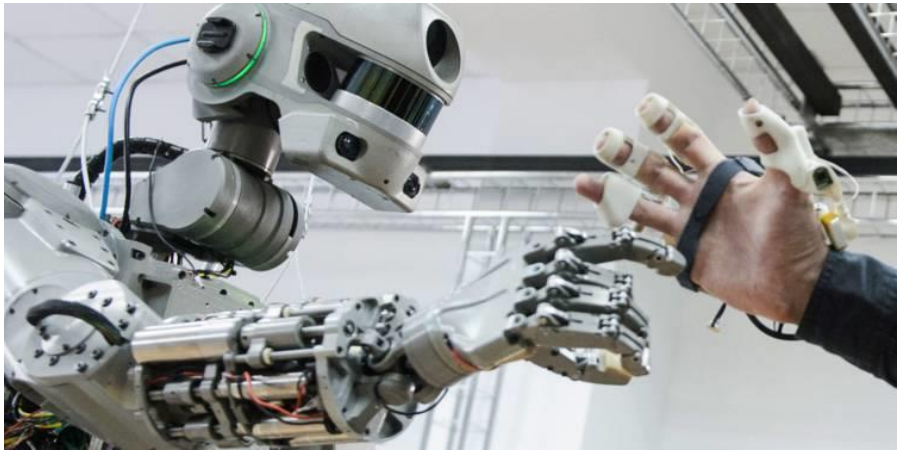
Objectifs de la séquence

- A l'issue de cette séquence vous serez capable d'expliquer pourquoi l'évolution du logiciel est une partie importante du génie logiciel et pour décrire processus d'évolution d'un logiciel. Lorsque vous aurez lu ce chapitre, vous pourrez:
  - comprendre que le changement est inévitable si les systèmes logiciels doivent rester utiles et que le développement et l'évolution des logiciels peuvent être intégrés dans un modèle en spirale;
  - comprendre les processus d'évolution des logiciels et leurs influences sur ces processus;
  - apprendre les différents types de maintenance logicielle et les facteurs qui affectent les coûts de maintenance;
  - comprendre comment les systèmes existants peuvent être évalués pour décider si elles doivent être abandonner, entretenues, remaniées, ou remplacer.

Remarques

- Ce cours utilise comme support le livre de Ian Sommerville *«Software Engineering»*, 9th edition.

Les logiciels et la société



Génie logiciel

Le **génie logiciel** est un domaine des sciences de l'ingénieur dont l'objet d'étude est la conception, la fabrication, et la maintenance des systèmes informatiques complexes. 3 févr. 2011

Génie Logiciel Avancé Cours 1 — Introduction - Stefano Zacchiroli
https://upsilon.cc/~zack/teaching/1011/gla/cours-01.pdf

Le génie logiciel touche au cycle de vie des logiciels. Toutes les phases de la création d'un logiciel informatique y sont enseignées : l'analyse du besoin, l'élaboration des spécifications, la **conceptualisation du mécanisme interne au logiciel** ainsi que les techniques de programmation, le développement, la phase de test et finalement la maintenance.
https://fr.wikipedia.org/wiki/Génie_logiciel

Evolution du logiciel: Evolution

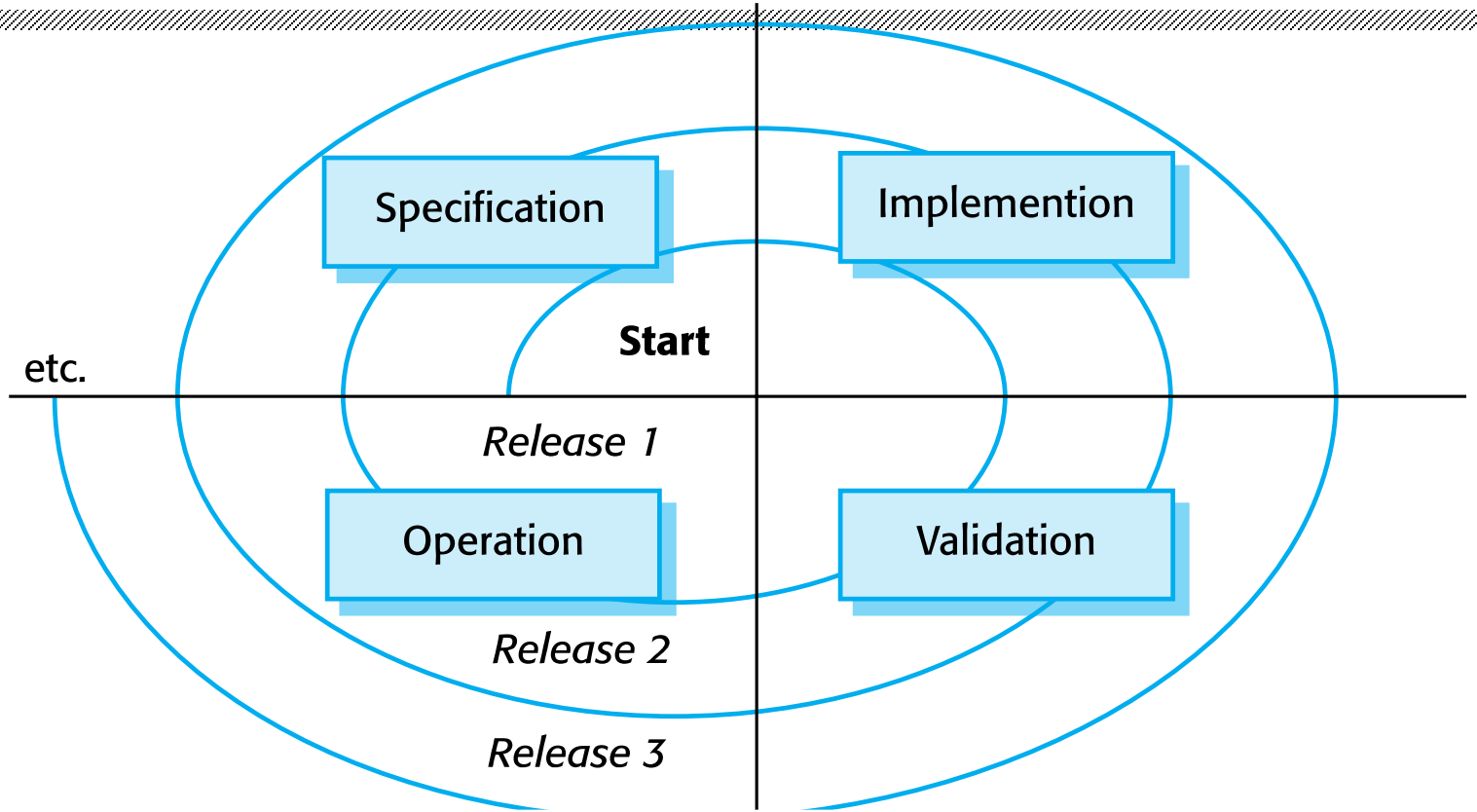
Evolution du logiciel !

- L'évolution des logiciels (software change en anglais) est inévitable si on veut que le logiciel reste utile (utilisable)
  - De nouveaux besoins émergent quand le logiciel est utilisé
  - L'environnement commercial change
  - Des erreurs découvertes et doivent être corrigées
  - De nouveaux ordinateurs et équipements sont ajoutés au système;
  - Il faudra peut-être améliorer les performances ou la fiabilité du système.
  - L'architecture de déploiement (architecture matérielle) a évolué (ajout de suppression d'ordinateurs/serveurs) et besoin d'améliorer les performances et la fiabilité
- Un problème clé pour toutes les organisations est la mise en œuvre et la gestion des modifications apportées à leurs systèmes logiciels existants.

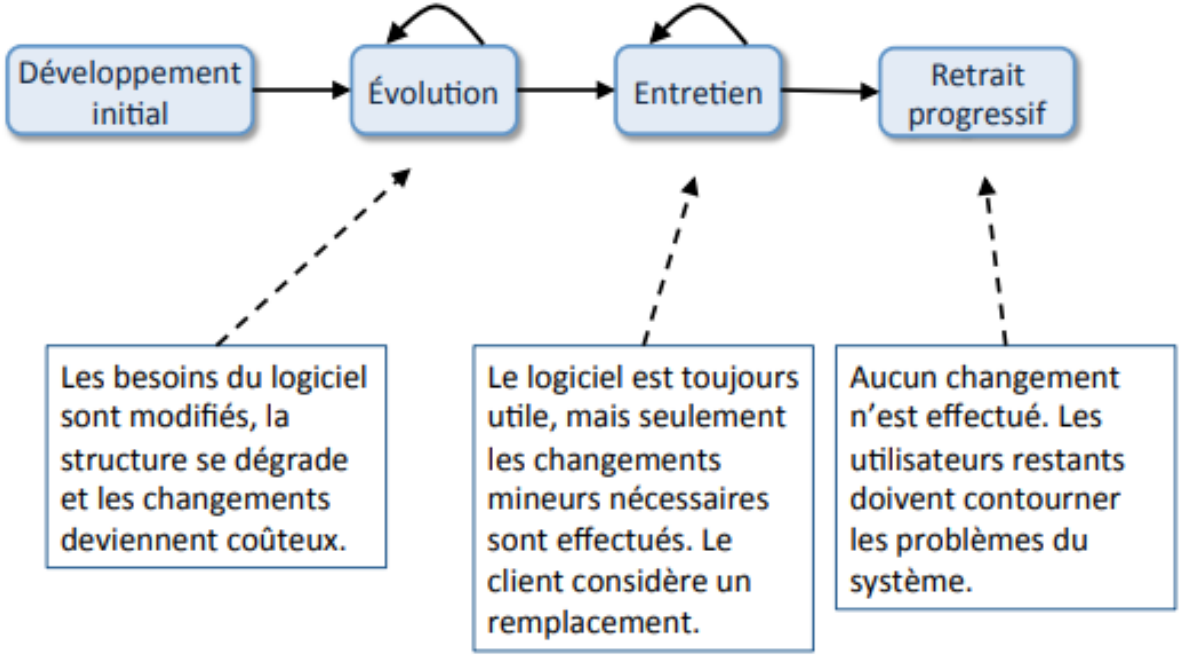
Importance de l'évolution

- Les entreprises ont d'énormes investissements dans leurs systèmes logiciels - ce sont des actifs critiques pour l'entreprise.
- Pour conserver la valeur de ces actifs pour l'entreprise, ils doivent être modifiés et mis à jour.
- Dans les grandes entreprises, la majeure partie du budget consacré aux logiciels est consacrée aux changements et à l'évolution des logiciels existants plutôt qu'à la mise au point de nouveaux logiciels.

Un modèle de développement et d'évolution en spirale



Evolution et Maintenance



Evolution et Maintenance

- Évolution: L'étape du cycle de vie d'un système logiciel est opérationnelle et évolue à mesure que de nouvelles exigences sont proposées et mises en œuvre dans le système.
- Maintenance: À ce stade, le logiciel reste utile, mais les seules modifications apportées sont celles requises pour le maintenir opérationnel, c'est-à-dire des corrections de bugs et des modifications reflétant les modifications apportées à l'environnement du logiciel. Aucune nouvelle fonctionnalité n'est ajoutée.
- Suppression progressive: Le logiciel peut toujours être utilisé mais aucune autre modification n'y est apportée.

Lois de l'évolution logicielle

- Les lois de Lehman – Basé sur des études empiriques par rapport à l'évolution sur une période de 30 ans

Lehman M.M. and Belady L.A. (Eds.), 1985 Software Evolution – Processes of Software Change, Academic Press, London (Free download from wiki.ercim.eu/wg/SoftwareEvolution)

M. M. Lehman. Laws of Software Evolution Revisited. Lecture Notes in Computer Science 1149, pp. 108-124, Springer Verlag, 1997 M. M. Lehman. Laws of Software Evolution Revisited. Lecture Notes in Computer Science 1149, pp. 108-124, Springer Verlag, 1997

Lois de la maintenance logicielle

- **La modification continue** : Un programme utilisé dans un environnement du monde réel doit nécessairement changer sinon il deviendra progressivement de moins en moins utile dans cet environnement.
- **La complexité croissante** : Comme le logiciel est modifié, il devient plus en plus complexe à moins que le travail soit effectué pour réduire la complexité.
- **Évolution des grands programmes** : L'évolution des grands programmes est un processus autorégulateur. Les attributs comme la taille, le temps entre versions et le nombre d'erreurs signalées sont approximativement invariants pour chaque version du programme.

Lois de la maintenance logicielle

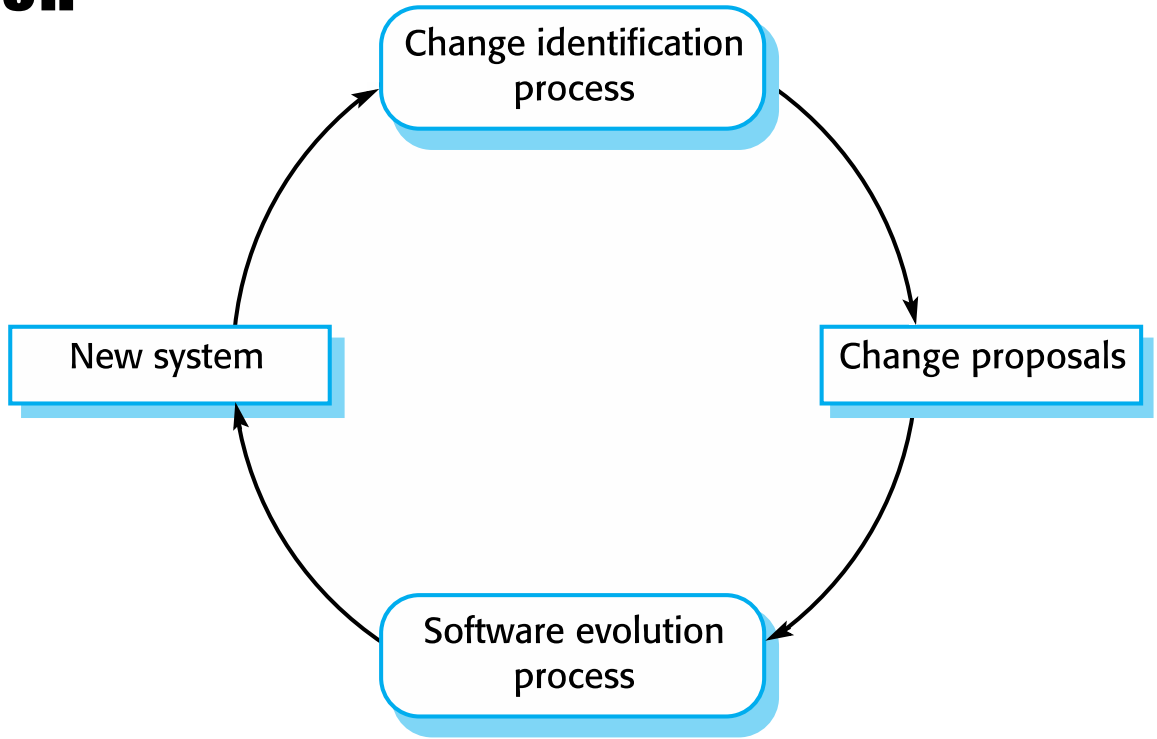
- **La stabilité organisationnelle :** Pendant la vie d'un programme, son taux de développement est approximativement constant et indépendant des ressources qui y sont consacrées.
- **La conservation de la familiarité :** Pendant la vie d'un programme, l'incrément de changement dans chaque version est approximativement constant.

Evolution du logiciel: Processus d'évolution du logiciel

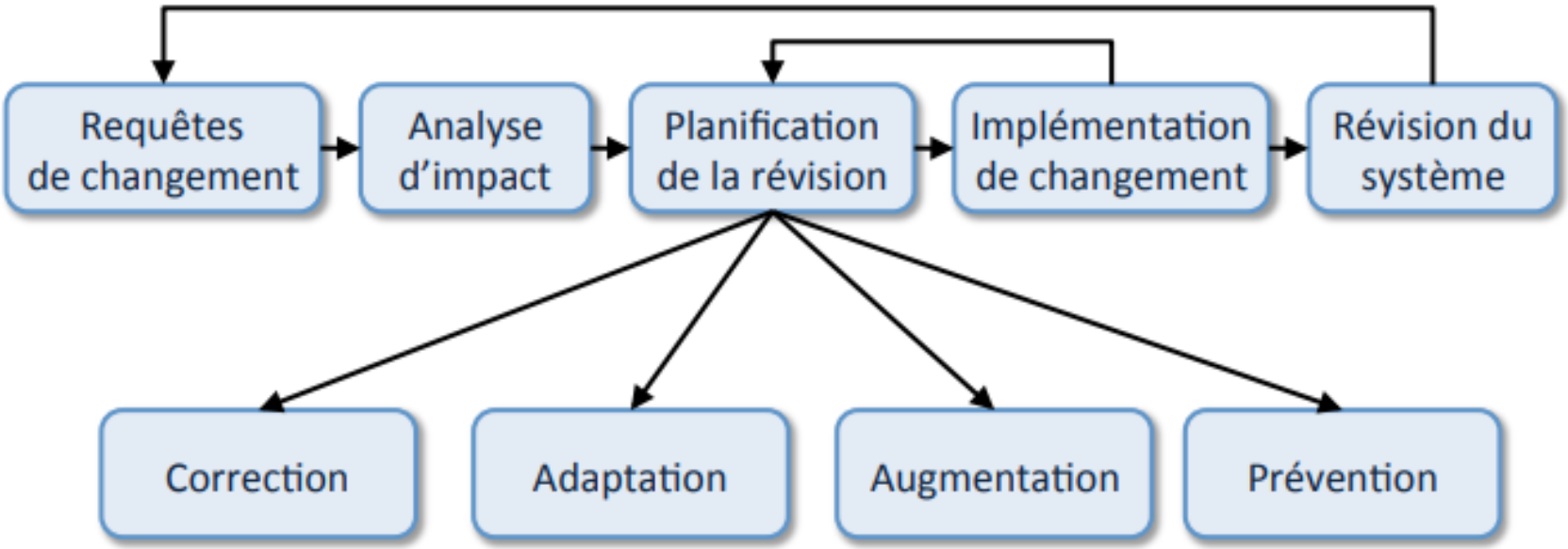
Processus d'évolution

- Les processus d'évolution des logiciels dépendent de
  - Le type de logiciel en cours de maintenance;
  - Les processus de développement utilisés;
  - Les compétences et l'expérience des personnes impliquées.
- Les propositions de changement sont le moteur de l'évolution du système: doivent être liées aux composants affectés par le changement, ce qui permet d'estimer le coût et l'impact du changement.
- L'identification et l'évolution du changement se poursuivent tout au long de la vie du système.

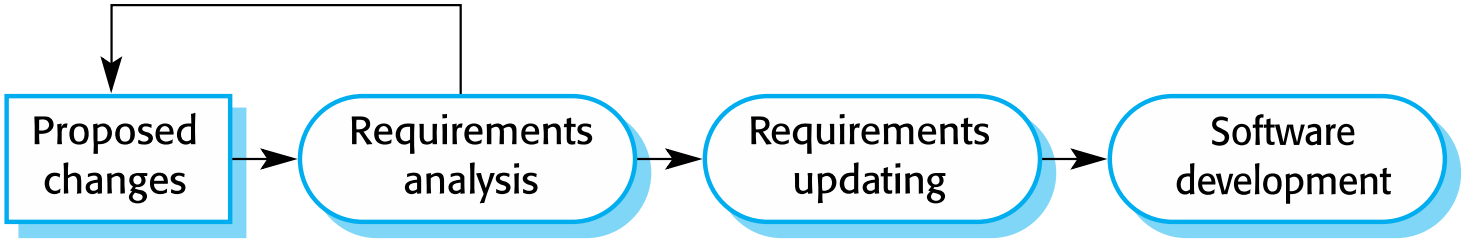
Identification du changement et processus d'évolution



Le processus d'évolution du logiciel



Implémentation du changement



Implémentation du changement

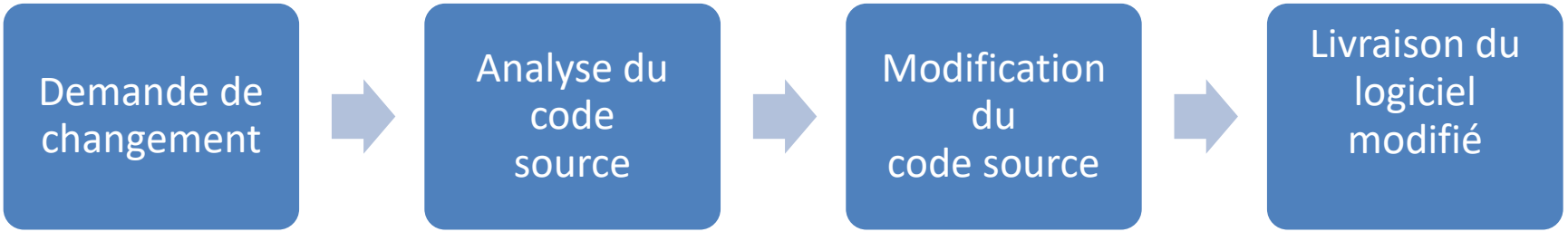
- Itération du processus de développement où les révisions du système sont conçues, mises en œuvre et testées.
- Une différence critique réside dans le fait que la première étape de la mise en œuvre du changement peut impliquer la compréhension du programme, en particulier si les développeurs du système d'origine ne sont pas responsables de la mise en œuvre du changement.
- Pendant la phase de compréhension du programme, vous devez comprendre comment le programme est structuré, comment il offre des fonctionnalités et comment le changement proposé peut affecter le programme.

Les demandes de changement urgentes

Des modifications urgentes peuvent être nécessaires sans passer par toutes les étapes du processus de génie logiciel:

- Si une défaillance grave du système doit être réparée pour permettre le fonctionnement normal;
- Si des modifications apportées à l'environnement du système (par exemple une mise à niveau du système d'exploitation) ont des effets inattendus;
- Si des changements commerciaux nécessitent une réponse très rapide (par exemple, la sortie d'un produit concurrent).

Le processus de maintenance urgente

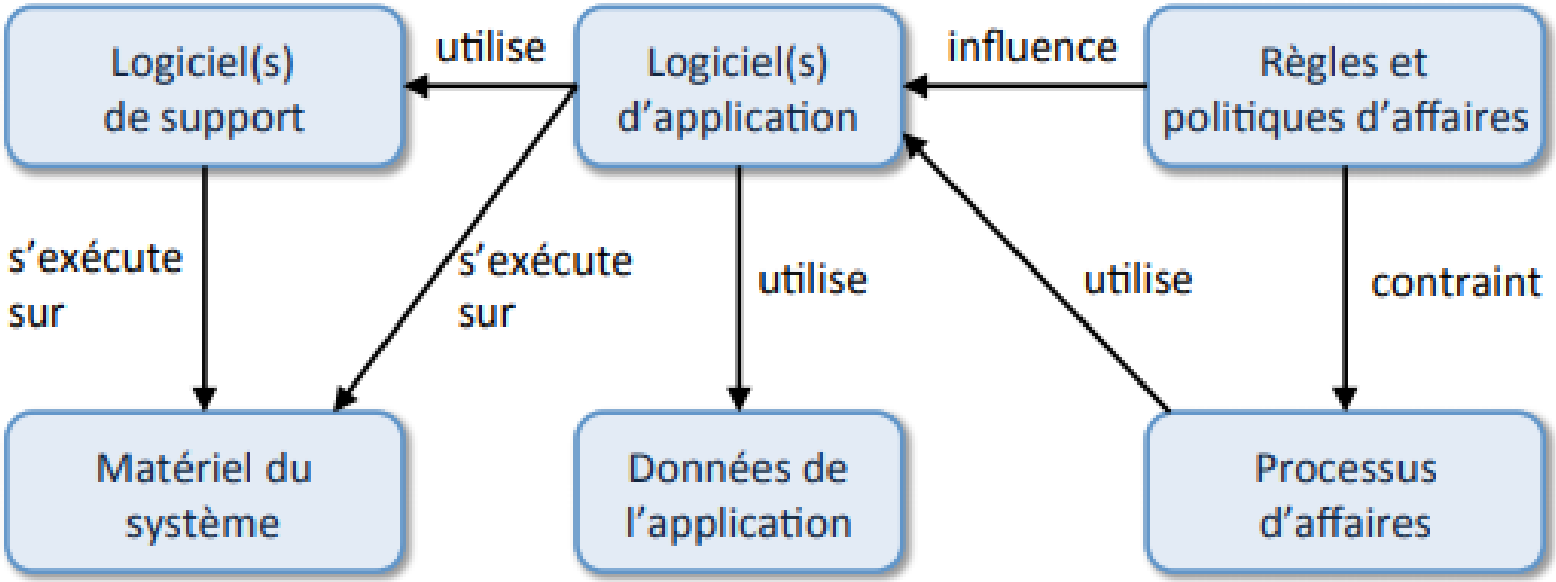


Evolution du logiciel: Les systèmes hérités(Systèmes patrimoniaux ou Legacy systems)

Systèmes hérités

- Les systèmes hérités sont des systèmes plus anciens qui reposent sur des langages et des technologies qui ne sont plus utilisés pour le développement de nouveaux systèmes.
- Les logiciels hérités peuvent dépendre d'anciens matériels, tels que les ordinateurs centraux, et peuvent être associés à des processus et procédures hérités.
- Les systèmes hérités ne sont pas simplement des systèmes logiciels, mais également des systèmes sociotechniques plus vastes comprenant du matériel, des logiciels, des bibliothèques et d'autres logiciels et processus d'appui connexes.

Les éléments d'un Système hérité



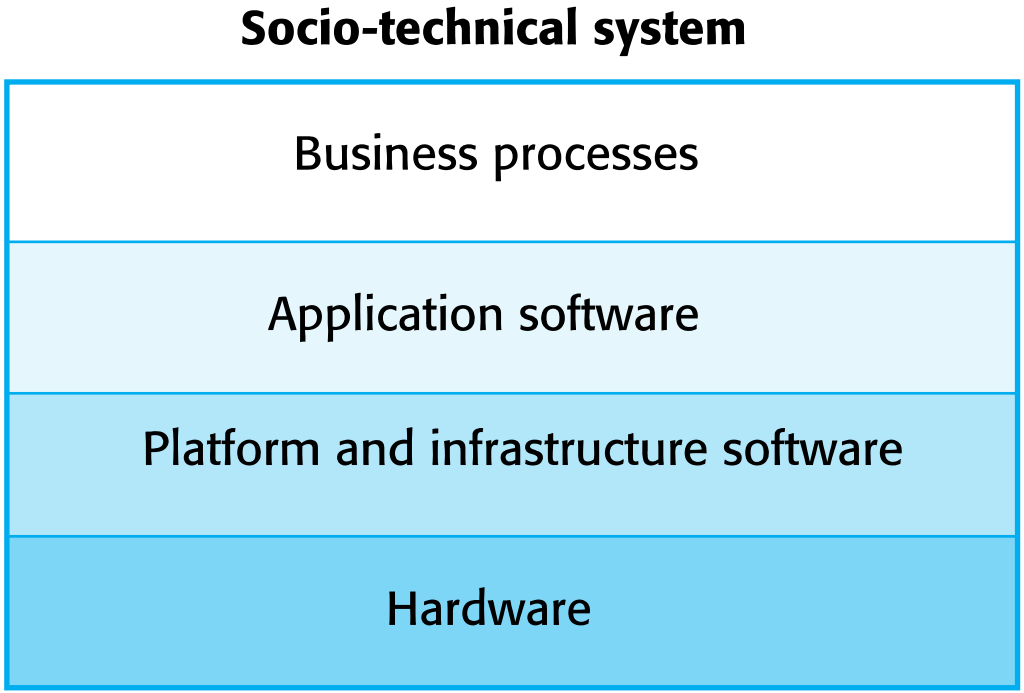
Composants d'un système hérité

- ***Matériel*** Les systèmes hérités ont peut-être été écrits pour du matériel qui n'est plus disponible.
- ***Logiciel Support*** de L'ancien système peut s'appuyer sur une gamme de supports logiciels, qui peuvent être obsolètes ou non pris en charge.
- ***Applications*** Le système d'application qui fournit les services métiers se compose généralement d'un certain nombre de programmes d'application qui peuvent être obsolètes.
- ***Données*** Il s'agit des données traitées par le système d'application. Elles peuvent être incohérentes, dupliquées ou conservées dans différentes bases de données.

Composants d'un système hérité

- ***Processus métiers*** Il s'agit des processus utilisés dans l'entreprise pour atteindre certains objectifs.
- ***Les processus métiers*** peuvent être conçus autour d'un système existant et limités par les fonctionnalités qu'il fournit.
- ***Politiques et règles commerciales*** Il s'agit des définitions de la manière dont les opérations doivent être menées et des contraintes qui lui sont imposées. L'utilisation du système d'application hérité peut être intégrée à ces stratégies et règles.

Couches des systèmes héritées



Remplacement d'un système hérité

- Le remplacement des systèmes hérités est risqué et coûteux, de sorte que les entreprises continuent à utiliser ces systèmes
- Le remplacement du système est risqué pour plusieurs raisons
  - Absence de spécification complète du système en entier sur laquelle se baser pour développer un nouveau système
  - Intégration étroite des processus systèmes et métiers
  - Règles d'affaires non documentées intégrées au système existant
  - Le développement de nouveaux logiciels peut être en retard et / ou dépasser le budget

Opération de changement dans un système hérité

- Les systèmes hérités sont coûteux à changer pour un certain nombre de raisons:
  - Pas de style de programmation cohérent
  - Plusieurs contributeurs, mais personne ne comprend le système en entier
  - Utilisation de langages de programmation désuets
  - Dépendance sur de la technologie ancienne
  - Documentation inadéquate ou dépassée
  - Structure corrompue par plusieurs années de maintenance ad-hoc
  - Optimisation pour l'espace / temps d'exécution et non pas la compréhension du code
  - Les données peuvent être séparées en plusieurs fichiers qui ont des structure incohérentes. Les données elle-même peuvent aussi être incohérentes, dupliquées, désuètes, etc.

Management d'un système hérité

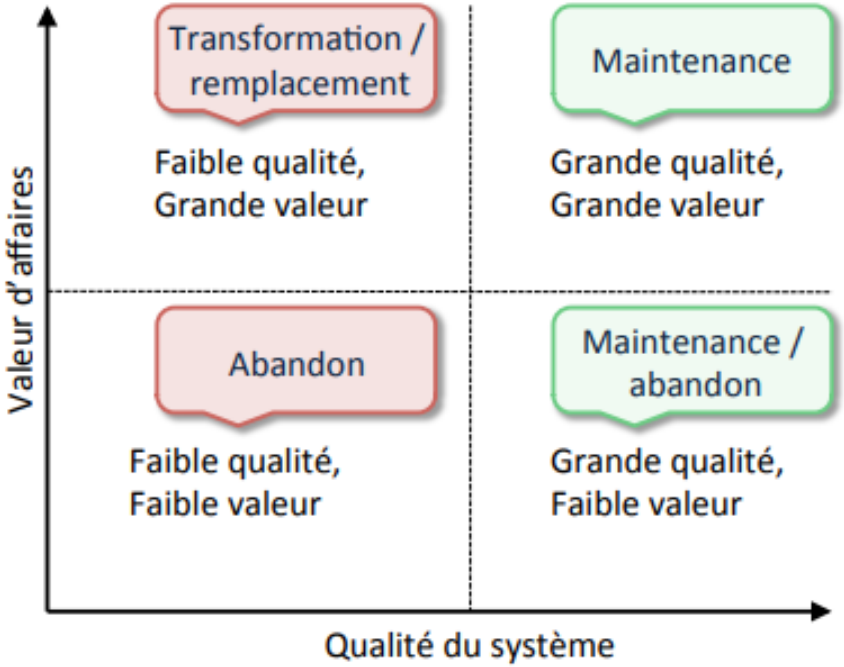
- Les organisations qui s'appuient sur des systèmes hérités doivent choisir une stratégie pour faire évoluer ces systèmes
  - Fermez complètement le système et modifiez les processus d'affaires pour qu'ils ne soient plus nécessaires;
  - Continuer à maintenir le système;
  - Transformer le système en le réorganisant pour améliorer sa maintenabilité;
  - Remplacez le système par un nouveau système.
- La stratégie choisie doit dépendre de la qualité du système et de sa valeur commerciale.

Exemple d'évaluation d'un système existant

4 options possibles:

- Abandonner le système complètement
- Continuer à maintenir le système
- Transformer le système pour faciliter la maintenance
- Remplacer le système hérité par un nouveau système

Les systèmes complexes peuvent utiliser une combinaison de plusieurs options.



Catégories de systèmes hérités

- ***Faible qualité, faible valeur commerciale:*** Ces systèmes devraient être abandonner.
- ***Qualité faible, valeur commerciale élevée:*** Celles-ci apportent une contribution importante à l'entreprise, mais leur entretien est coûteux. Devrait être transformée ou remplacée si un système approprié est disponible.
- ***Grande qualité, faible valeur commerciale:*** abandonnez-le, ou conservez-le complètement en faisant de la maintenance .
- ***Grande qualité, grande valeur commerciale:*** Continuez à l'utiliser en utilisant la maintenance normale du système.

Évaluation de la valeur commerciale

- L'évaluation doit prendre en compte différents points de vue
  - Utilisateurs finaux du système;
  - Clients de l'entreprise;
  - Gestionnaires hiérarchiques;
  - Responsables informatiques;
  - Cadres supérieurs.
- Interrogez différents intervenants et rassemblez les résultats.

Problèmes liés à l'évaluation de la valeur commerciale

- **L'utilisation du système:** Si les systèmes ne sont utilisés qu'occasionnellement ou par un petit nombre de personnes, leur valeur commerciale peut être faible.
- **Les processus métier supportés:** Un système peut avoir une faible valeur commerciale s'il oblige à utiliser des processus métier inefficaces.
- **Fiabilité du système:** Si un système n'est pas fiable et que les problèmes affectent directement les clients professionnels, le système a une faible valeur commerciale.
- **Les sorties du système:** Si l'entreprise dépend des sorties du système, sa valeur commerciale est élevée.

Évaluation de la qualité du système

- **Évaluation des processus métiers:** Dans quelle mesure le processus métier soutient-il les objectifs actuels de l'entreprise?
- **Évaluation de l'environnement:** Quelle est l'efficacité du système et la maintenance est-elle coûteuse?
- **Évaluation de l'application:** Quelle est la qualité du système de logiciel d'application?

Évaluation des processus métiers

- Utiliser une approche axée sur les points de vue et chercher des réponses des parties prenantes du système
  - Existe-t-il un modèle de processus défini et est-il suivi?
  - Est-ce que différentes parties de l'organisation utilisent différents processus pour la même fonction?
  - Comment le processus a-t-il été adapté?
  - Quelles sont les relations avec les autres processus métier et sont-elles nécessaires?
  - Le processus est-il efficacement pris en charge par le logiciel d'application existant?
- Exemple: un système de réservation de de billet pour voyager peut avoir une faible valeur commerciale en raison de l'utilisation répandue de la commande en ligne sur le web.

Facteurs utilisés dans l'évaluation de l'environnement

Facteurs	Questions
Stabilité du fournisseur	Le fournisseur existe-t-il toujours? Le fournisseur est-il financièrement stable et susceptible de continuer d'exister? Si le fournisseur n'est plus en activité, est-ce que quelqu'un d'autre entretient les systèmes?
Taux d'échec	Le nombre de pannes matérielles signalées est-elle élevé? Est-ce que le logiciel de support plante et force le redémarrage du système?
Age	Quel âge a le matériel et le logiciel? Plus le matériel et les logiciels de support sont anciens, plus ils seront obsolètes. Il peut toujours fonctionner correctement, mais le passage à un système plus moderne pourrait présenter des avantages économiques et commerciaux importants.
Performance	La performance du système est-elle adéquate? Les problèmes de performances ont-ils un effet significatif sur les utilisateurs du système?

Facteurs utilisés dans l'évaluation de l'environnement

Facteurs	Questions
Besoins de support	Quel support local est requis par le matériel et les logiciels? Si les coûts associés à cette assistance sont élevés, il peut être intéressant d'envisager le remplacement du système.
Coûts de maintenance	Quels sont les coûts de maintenance du matériel et de licences de logiciels de support? Le matériel ancien peut avoir des coûts de maintenance plus élevés que les systèmes modernes. Les logiciels de support peuvent avoir des coûts de licence annuels élevés..
Interoperabilité	Existe-t-il des problèmes d'interface entre le système et d'autres systèmes? Les compilateurs, par exemple, peuvent-ils être utilisés avec les versions actuelles du système d'exploitation? Une émulation matérielle est-elle requise?

Facteurs utilisés dans l'évaluation de l'environnement

Facteurs	Questions
Compréhensibilité	Est-il difficile de comprendre le code source du système actuel? Quelle est la complexité des structures de contrôle utilisées? Les variables ont-elles des noms significatifs qui reflètent leur fonction?
Documentation	Quelle documentation système est disponible? La documentation est-elle complète, cohérente et à jour?
Données	Existe-t-il un modèle de données explicite pour le système? Dans quelle mesure les données sont-elles dupliquées sur plusieurs fichiers? Les données utilisées par le système sont-elles à jour et cohérentes?
Performance	La performance de l'application est-elle adéquate? Les problèmes de performances ont-ils un effet significatif sur les utilisateurs du système?

Facteurs utilisés dans l'évaluation de l'environnement

Facteurs	Questions
Langage de Programmation	Des compilateurs modernes sont-ils disponibles pour le langage de programmation utilisé pour développer le système? Le langage de programmation est-il toujours utilisé pour le développement de nouveaux systèmes?
Configuration et management	Toutes les versions de toutes les parties du système sont-elles gérées par un système de gestion de la configuration? Existe-t-il une description explicite des versions des composants utilisés dans le système actuel?
Données de test	Existe-t-il des données de test pour le système? Existe-t-il un enregistrement des tests de régression effectués lorsque de nouvelles fonctionnalités ont été ajoutées au système?
Compétences du personnel	Y a-t-il des personnes disponibles qui possèdent les compétences nécessaires pour gérer l'application? Y a-t-il des personnes disponibles qui ont de l'expérience avec le système?

Système de mesure

- Vous pouvez collecter des données quantitatives pour évaluer la qualité du système de candidature.
  - Le nombre de demandes de modification du système; Plus cette valeur accumulée est élevée, plus la qualité du système est médiocre.
  - Le nombre d'interfaces utilisateur différentes utilisées par le système; Plus il y a d'interfaces, plus il est probable qu'il y aura des incohérences et des redondances dans ces interfaces.
  - Le volume de données utilisé par le système. Au fur et à mesure que le volume de données (nombre de fichiers, taille de la base de données, etc.) traitées par le système augmente, les incohérences et les erreurs dans ces données augmentent également.
  - Nettoyer les anciennes données est un processus très coûteux et fastidieux.

Evolution du logiciel: Maintenance logicielle

Maintenance Logicielle

- Modifier un programme après sa mise en service.
- Le terme est principalement utilisé pour changer de logiciel personnalisé. Les produits logiciels génériques évolueraient pour créer de nouvelles versions.
- La maintenance n'implique normalement pas de modifications majeures de l'architecture du système.
- Les modifications sont mises en œuvre en modifiant les composants existants et en ajoutant de nouveaux composants au système.

Types de maintenance

- Corrections de fautes ~ ***Maintenance Corrective***
  - Changer un système pour corriger les bugs / vulnérabilités et corriger les défauts dans la manière dont il répond à ses exigences.
- Adaptation à un nouvel environnement ~ **Maintenance adaptive**
  - Maintenance pour adapter le logiciel à un environnement d'exploitation différent
  - Modification d'un système afin qu'il fonctionne dans un environnement différent (ordinateur, système d'exploitation, etc.) depuis sa mise en œuvre initiale.
- Ajout ou modification de fonctionnalités ~ ***Maintenance Perfective***
  - Modifier le système pour répondre aux nouveaux besoins.
- Amélioration de la qualité du logiciel **~ Maintenance Préventive**
  - Modifications qui visent à rendre les modifications ultérieures plus faciles

Types de maintenance: Le standard ISO/IEC 14764 distingue quatre catégories:

- **Maintenance corrective:** Fixe les différents bogues d'un système. Le système peut ne pas fonctionner comme il devrait le faire selon ses plans de conception. L'objectif de la maintenance corrective est donc de localiser les spécifications originelles afin de déterminer ce que le système était initialement sensé faire.
- **Maintenance adaptative:** Fixe les différents changements qui doivent être faits afin de suivre l'environnement du système en perpétuelle évolution. Par exemple, le système d'exploitation peut être mis à jour et des modifications peuvent être nécessaires afin de s'accommoder à ce nouveau système d'exploitation.

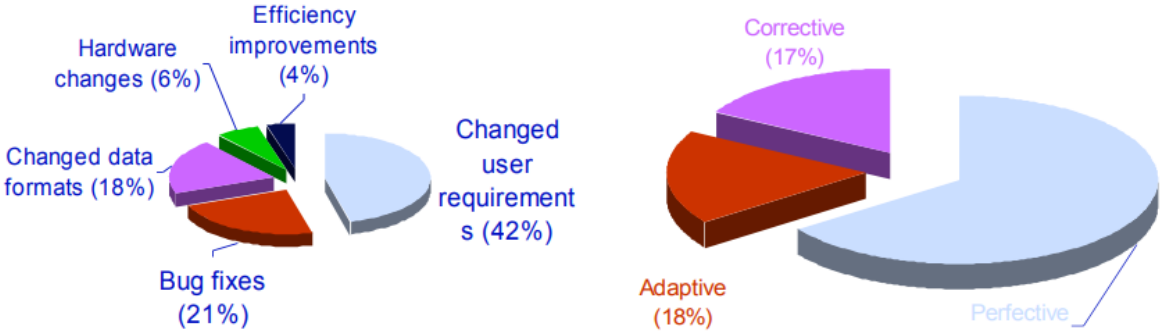
Types de maintenance: Le standard ISO/IEC 14764 distingue quatre catégories:

- **Maintenance perfective**: La maintenance perfective ou d'amélioration comprend tous les changements faits sur un système afin de satisfaire aux besoins de l'utilisateur.
- **Maintenance préventive** : La maintenance exécutée afin d'empêcher des problèmes avant qu'ils surviennent. Ce type de maintenance peut être considéré comme très important dans le cas de systèmes critiques.

Why? / When?	Correction	Enhancement
Proactive	Preventive	Perfective
Reactive	Corrective	Adaptive

Répartition de l'effort de maintenance

- Repérer (détecter) des défauts (des erreurs) (17%)
- Adapter un logiciel à différents environnements (machines, OS, etc) par rapport à son implémentation d'origine (18%)
- Ajouter ou modifier les fonctionnalités du système pour répondre à de nouveaux besoins (65%)



[Lientz&Swanson 1980]

Coûts de la maintenance

- Généralement supérieur aux coûts de développement (2 * à 100 * en fonction de l'application).
- Affecté à la fois par des facteurs techniques et non techniques.
- Augmente à mesure que le logiciel est maintenu. La maintenance corrompt la structure du logiciel, ce qui rend plus difficile toute maintenance ultérieure.
- Les logiciels obsolètes peuvent avoir des coûts de support élevés (par exemple, les anciens langages, les compilateurs, etc.).

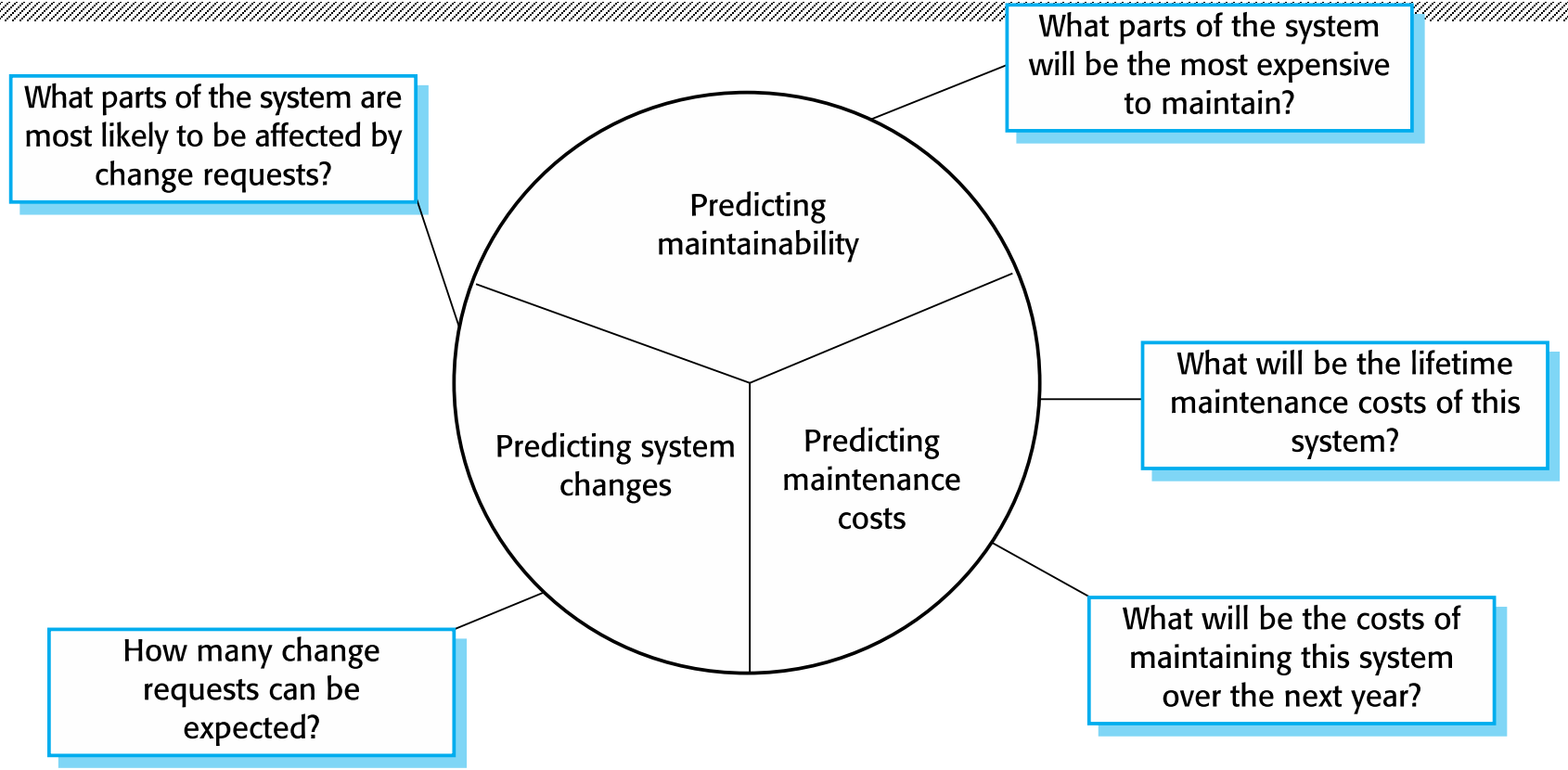
Coûts de la maintenance

- Il est généralement plus coûteux d'ajouter de nouvelles fonctionnalités à un système pendant la maintenance que d'ajouter les mêmes fonctionnalités au cours du développement.
  - Une nouvelle équipe doit comprendre les programmes maintenus
  - Séparer maintenance et développement signifie que l'équipe de développement n'est pas incitée à écrire un logiciel maintenable.
  - Le travail de maintenance du programme est impopulaire
    - Le personnel de maintenance manque souvent d'expérience et possède une connaissance limitée du domaine.
  - À mesure que les programmes vieillissent, leur structure se dégrade et il est de plus en plus difficile de les modifier.

Prévision de maintenance

- La prévision de la maintenance consiste à évaluer quelles parties du système peuvent poser problème et engendrer des coûts de maintenance élevés
  - L'acceptation des modifications dépend de la maintenabilité des composants affectés par la modification;
  - La mise en œuvre des modifications dégrade le système et réduit sa maintenabilité.
  - Les coûts de maintenance dépendent du nombre de modifications et les coûts de modification dépendent de la maintenabilité.

Prévision de maintenance



Prévision du changement

- Pour prévoir le nombre de changements, il faut comprendre les relations entre un système et son environnement.
- Les systèmes étroitement couplés nécessitent des modifications chaque fois que l'environnement est modifié.
- Les facteurs qui influencent cette relation sont
  - Nombre et complexité des interfaces système;
  - Nombre d'exigences système intrinsèquement volatiles;
  - Les processus métier sur lesquels le système est utilisé.

Métriques de la complexité

- On peut prédire la maintenabilité en évaluant la complexité des composants du système.
- Des études ont montré que la plupart des efforts de maintenance sont consacrés à un nombre relativement réduit de composants du système.
- La complexité dépend de
  - Complexité des structures de contrôle;
  - Complexité des structures de données;
  - Objet, méthode (procédure) et taille du module.

Les métriques de processus

- Les métriques de processus peuvent être utilisées pour évaluer la maintenabilité
  - Nombre de demandes de maintenance corrective;
  - Temps moyen requis pour l'analyse d'impact;
  - Temps moyen nécessaire pour mettre en œuvre une demande de changement;
  - Nombre de demandes de changement en attente.
- Si tout ou partie de ces dernières augmente, cela peut indiquer une diminution de la maintenabilité.

Réingénierie logicielle

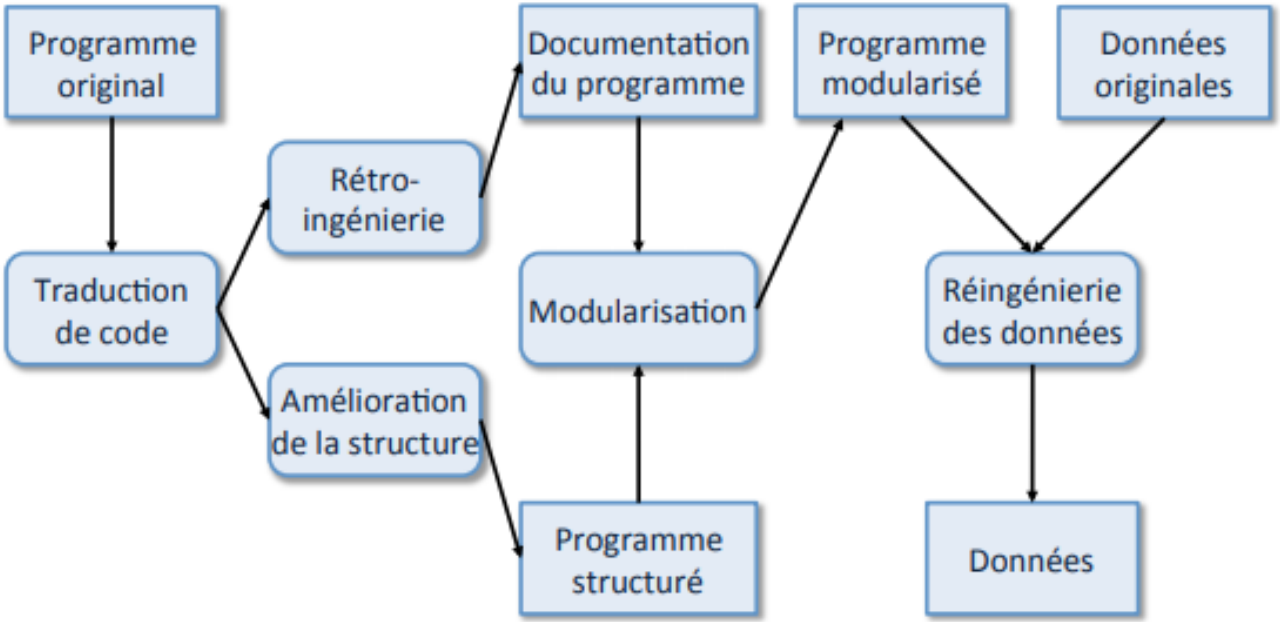
- Restructurer ou réécrire tout ou partie d'un système existant sans en modifier les fonctionnalités.
- Applicable lorsque certains sous-systèmes (mais pas tous) d'un système plus grand nécessitent une maintenance fréquente.
- La réingénierie implique des efforts supplémentaires pour les rendre plus faciles à maintenir. Le système peut être restructuré et documenté.

Avantages de la réingénierie logicielle

- **Risque réduit:** Il y a un risque élevé dans le développement de nouveaux logiciels. Il peut y avoir des problèmes de développement, des problèmes de personnel et des spécifications.

- **Coût réduit:** Le coût de la réingénierie est souvent nettement inférieur au coût de développement d'un nouveau logiciel.

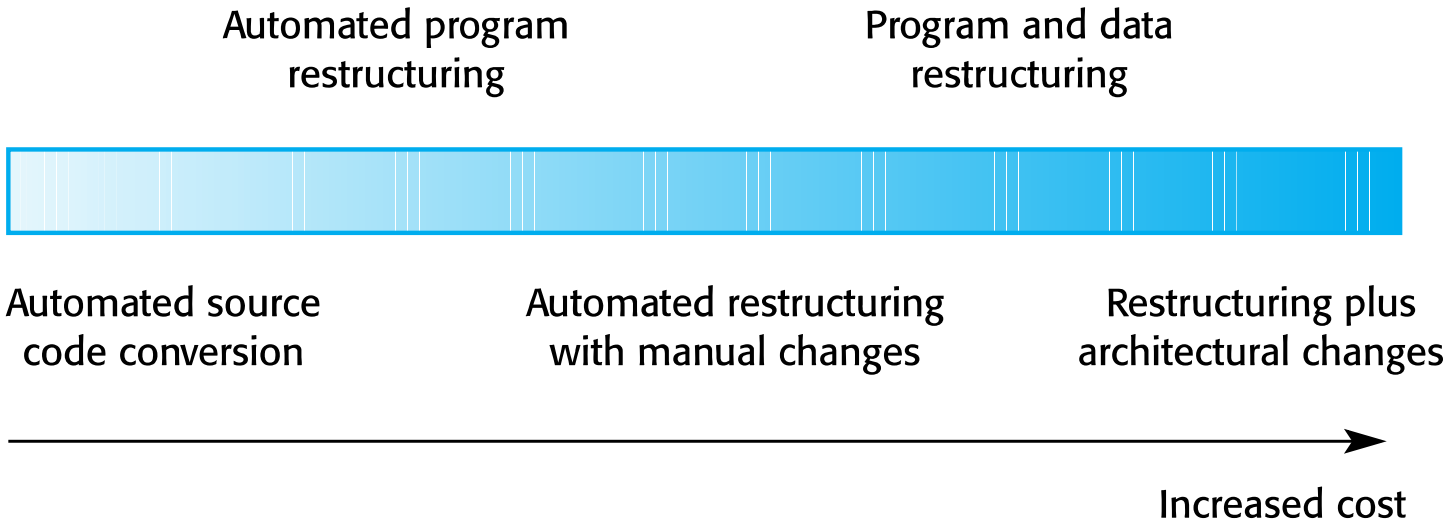
Processus de réingénierie logicielle



Activités du processus de réingénierie

- **Traduction du code source :** Convertir le code dans une nouvelle langue.
- **Ingénierie inverse:** Analyser le programme pour le comprendre;
- **Amélioration de la structure du programme:** Restructurer automatiquement pour la compréhensibilité;
- **Programme de modularisation:** Réorganiser la structure du programme;
- **Réingénierie des données:** Nettoyer et restructurer les données du système.

Approches de réingénierie



Facteurs de coûts de la réingénierie

- La qualité du logiciel.
- Le support d'outil disponible pour la réingénierie.
- L'étendue de la conversion de données requise.
- La disponibilité de personnel expert pour la réingénierie.
- Cela peut être un problème avec les anciens systèmes basés sur une technologie qui n'est plus largement utilisée.

Refactoring

- La refactorisation consiste à apporter des améliorations à un programme visant à ralentir la dégradation par le changement.
- Vous pouvez considérer la refactorisation comme une «maintenance préventive» qui réduit les problèmes de changement futur.
- La refactorisation consiste à modifier un programme pour améliorer sa structure, réduire sa complexité ou le rendre plus facile à comprendre.
- Lorsque vous refactorisez un programme, vous ne devez pas ajouter de fonctionnalités, mais plutôt vous concentrer sur l'amélioration du programme.

Refactoring et réingénierie

- La réingénierie a lieu après la maintenance d'un système pendant un certain temps et l'augmentation des coûts de maintenance. Vous utilisez des outils automatisés pour traiter et reconfigurer un système hérité afin de créer un nouveau système plus facile à gérer.
- Le refactoring est un processus continu d'amélioration tout au long du processus de développement et d'évolution. Il est destiné à éviter la dégradation de la structure et du code qui augmente les coûts et les difficultés de maintenance d'un système.

"Mauvaises pratiques" dans le code du programme

- **Duplication de code:** Le même code ou un code très similaire peut être inclus à différents endroits d'un programme. Cela peut être supprimé et implémenté en tant que méthode ou fonction unique appelée selon les besoins.
- **Méthodes longues:** Si une méthode est trop longue, il convient de la reformuler en un certain nombre de méthodes plus courtes.
- Déclenchement (Switch (case)): Celles-ci impliquent souvent une duplication, où le commutateur dépend du type de valeur. Les instructions switch peuvent être dispersées dans un programme. Dans les langages orientés objet, vous pouvez souvent utiliser le polymorphisme pour obtenir la même chose.

"Mauvaises pratiques" dans le code du programme

- **Regroupement de données:** Les blocs de données se produisent lorsque le même groupe d'éléments de données (champs dans les classes, paramètres dans les méthodes) se reproduit à plusieurs endroits d'un programme. Ceux-ci peuvent souvent être remplacés par un objet qui encapsule toutes les données.
- **Généralité spéculative:** Cela se produit lorsque les développeurs incluent la généralité dans un programme au cas où cela serait nécessaire dans le futur. Cela peut souvent être simplement supprimé.

Evolution du Logiciel:
A Retenir...

A retenir…

- Le développement et l'évolution de logiciels peuvent être considérés comme un processus intégré et itératif pouvant être représenté à l'aide d'un modèle en spirale.
- Pour les systèmes personnalisés, les coûts de maintenance des logiciels dépassent généralement les coûts de développement des logiciels.
- Le processus d'évolution des logiciels est dicté par les demandes de modification et comprend une analyse de l'impact des modifications, la planification des versions et la mise en œuvre des modifications.
- Les systèmes existants sont des systèmes logiciels plus anciens, développés à l'aide de technologies matérielles et logicielles obsolètes, qui restent utiles pour une entreprise.

A retenir…

- Il est souvent moins coûteux et moins risqué de maintenir un système existant que de développer un système de remplacement utilisant une technologie moderne.
- La valeur commerciale d'un système existant et la qualité de l'application doivent être évaluées pour aider à décider si un système doit être remplacé, transformé ou entretenu.
- Il existe 3 types de maintenance logicielle, à savoir la correction de bogues, la modification de logiciels pour fonctionner dans un nouvel environnement et la mise en œuvre de nouvelles exigences ou de modifications.

A retenir…

- La réingénierie logicielle consiste à restructurer et à documenter un logiciel afin de le rendre plus facile à comprendre et à modifier.
- La refactorisation, apportant des modifications de programme préservant les fonctionnalités, est une forme de maintenance préventive.


thank you
danke
謝謝
рахмат
ngiyabonga
teşekkür ederim
gracias
merci
obrigado
dziękuję
bedankt
sukriya
kop khun krap
go raibh maith agat
terima kasih
grazie
arigatô
dakujem
mochchakkeram
спасибо
Баярлалаа
faafetai lava
kiitos
dankie
dhanyavad
nanni
nandri
mauruuru
hvala
köszönöm
enkosi
bayarlalaa
grâcie
dank je
misaotra
matondo
paldies
grazzi
mahalo
tapadh leat
хвала
asante
manana
obrigada
murakoze
tenki
djiere dieuf
tau
дякую
mamnun
chnorakaloutioun
gralias ago
gràcies
sulpáy
akun
dankon
aciù
sobodi
dekuji
sagolun
didi madloba
mèsi
najis tuke
kam sah hamnida
rahmat
ありがとう
tanemirt
rahmet
diolch
dhanyavadagalu
shukriya
merci
мерси
takk
taiku
trugarez
chokrane
তোমাকে ধন্যবাদ
감사합니다
xièxie
ευχαριστώ
barka
kia ora
welalin
tack
spas
mersi
vinaka
спасибі
blagodaram