Square des Utilisateurs

Science ou magie ?

Le point sur les estimations des projets logiciels

Le CMSL (Centre de Maîtrise des Systèmes et du logiciel) a organisé, les 12 et 13 juin 2001, dans les locaux du CNAM, un séminaire sur l'estimation des projets logiciels. Ce séminaire était co-animé par Nicolas Trèves, membre d'ADELI et du CMSL.

Dans la continuité des articles précédents[1], publiés dans sa LETTRE, ADELI vous soumet ce bref compte rendu destiné à dresser un panorama des démarches actuelles. Pour plus d'informations, vous pouvez vous reporter au site www.cnam.fr/CMSL qui contient les visuels complets des principales présentations.

Dans la foulée, nous avons le plaisir d'annoncer la parution de l'ouvrage intitulé « Coûts et durée des projets informatiques » de Jacques Printz, Nicolas Trèves, Christiane Deh et Bernard Mesdon.

Première journée : modèles d'estimation et principes de mesures

La première journée enchaîne quatre exposés didactiques.

CQFD des systèmes informatisés – Modèles d'estimation

Jacques Printz pose le problème de l'estimation et recense les solutions actuelles. CQFD est un mnémonique qui nous remémore le célèbre « Ce Qu'il Fallait Démontrer » et aussi les caractéristiques essentielles d'un projet : Coût - Qualité - Fonctionnalités - Délais.

Jacques Printz rappelle quelques évidences dont nous extrayons :

- la forme de la « courbe en S » accentue, à la livraison, le retard pris par la programmation ;
- les tests (vérification, validation) consomment jusqu'à 40 % du coût des projets d'intégration ;
- la productivité de la programmation est un facteur lié aux compétences du programmeur (qui peut varier de 1 à 5) ;
- la qualité du travail d'architecture est un investissement qui réduit le volume des travaux.

De COCOMO 81 à COCOMO II (version 2000)

Nicolas Trèves présente le modèle COCOMO (Constructive Cost Model) élaboré par B.W. Boehm en 1981. Il préconise de « peser » le logiciel à réaliser en Kisl (kilo instructions-source livrées) et d'en estimer la charge globale par la simple formule . Charge = k * Kisl ^{a} où **k** est un coefficient lié au type de logiciel et **a** un exposant légèrement supérieur à 1.

La nouvelle version 2000 apporte des compléments sensibles.

- Elle propose d'affiner progressivement l'estimation aux 3 étapes du projet :
  - Faisabilité - première mesure du nombre d'objets manipulables par l'application ;
  - Définition - après les spécifications fonctionnelles ;
  - Développement - après la conception technique détaillée.

---

[1] *LETTRE n ° 35 – avril 1999 – Que sont les estimations devenues ?*
LETTRE n° 39 – avril 2000 – Estimations de charges - les orientations de la commission
LETTRE n° 41 – oct. 2000 – Estimations de projets informatiques – Traduction d'un article canadien (Kathleen Peters)
LETTRE n° 42 – janvier 2001 – Calibrage et étalonnage – Jean Joskowicz

- Des tableaux indiquent la répartition des charges par phases du projet.
- Elle introduit des facteurs correctifs dans la formule Charge = k * '(Kisl) $^{1+\alpha}$ dans laquelle :
  - k résulte d'une combinaison de facteurs de coût ;
  - α (compris entre 0,05 et 0,2) résulte d'une combinaison de facteurs d'échelle.

Des tables qualitatives donnent des valeurs des facteurs de coût et d'échelle..

Le problème fondamental de COCOMO reste la détermination du nombre d'instructions pour des logiciels novateurs.

Les points de fonctions

Christiane Deh présente ce modèle, proposé par A. Albrecht, au début des années 1980. Ce modèle évalue la taille des applications et des projets, à partir des fonctions attendues du logiciel.

Nous ne reviendrons pas sur les 5 fonctions de base (groupe de données internes, groupe de données externes, entrées, sorties, interrogations). La démarche est fiable : on enregistre des écarts inférieurs à 5 % entre les estimations brutes d'un même logiciel, effectuées par des personnes différentes.

L'estimation brute est pondérée par 14 facteurs d'ajustement qui permettent de moduler la charge brute dans une plage de + ou – 35 % par rapport à l'estimation brute.

Pratique de l'estimation des coûts de projets de systèmes d'information

Bernard Mesdon formule des recommandations d'emploi de méthodes, qui jouent à la fois sur COCOMO et sur les points de fonctions, en s'appuyant sur des exemples concrets.

En conclusion personnelle

Cette journée conforte les recettes traditionnelles (COCOMO et points de fonctions) appliquées, avec plus ou moins de succès, depuis une vingtaine d'années.

Les principales évolutions concernent :

- la progression de l'estimation qui resserre l'incertitude, au fur et à mesure des étapes du projet ;
- la prise en compte des facteurs liés aux conditions de réalisation.

À noter qu'une règle simple apparaît incontournable ; c'est la relation entre la charge de travail et la durée du projet :

Délai, en mois = Racine cubique de la charge, en jours.hommes

Deuxième journée : modèles d'estimation et principes de mesures

La deuxième journée a été consacrée aux témoignages et aux discussions.

En ouverture, Jacques Printz évoque les contraintes liées aux organisations des projets informatiques qui interfèrent avec les structures hiérarchiques des entreprises. Une organisation qui ne saurait ni mesurer ni estimer sa production ne pourrait s'améliorer.

Grand projet EDF

Arnaud Hertz insiste sur l'importance des contraintes de délais. Vouloir diviser la durée du projet par 2, reviendrait théoriquement à multiplier la charge par 16. Sans pousser jusqu'à cet extrême, accepter un léger allongement du délai autour de la valeur critique permet de diminuer sensiblement la charge, donc le coût supporté par le client.

L'analyse, a posteriori, des dérives des projets révèle les causes suivantes :

- gonflement et complexification progressive du cahier des charges pendant les travaux ;
- réalisation de fonctionnalités peu adaptées aux besoins du marché ;
- instabilité d'un système qui sera difficile à maintenir.

Ce qui amène à quelques recommandations :

- tempérer l'optimisme du chef de projet ;
- limiter formellement les desiderata des clients ;
- fractionner les grands projets, en projets à taille humaine.

Maintenance d'un parc applicatif (cas de la TMA - Tierce maintenance applicative)

Jean-François Bailliot (ATOS ORIGIN) appelle l'attention de l'auditoire sur le poids des charges de maintenance d'un système. Échelonnées pendant toute sa durée de vie, elles représentent 150 % des charges de développement.

Il faut bien distinguer les différentes natures de charges de maintenance :

- correction des anomalies ;
- évolution technique ou fonctionnelle (adaptation aux nouveaux environnements) ;
- assistance fonctionnelle aux utilisateurs ;
- gestion de l'activité de maintenance.

Pour chaque application, il faut :

- définir la taille (en points de fonctions) ;
- collecter les charges actuelles de maintenance ;
- analyser les réponses à un questionnaire pour déterminer les paramètres qualitatifs : ancienneté criticité, multiplicité des versions, populations d'utilisateurs, expérience du personnel, qualité de la documentation, utilisation de normes.
- comparer à la base de données qui détermine des valeurs moyennes et des écarts.

En passant, le conférencier souligne que les programmes en exploitation comportent environ 40 % de code inutile, sur lesquels il n'y aura aucune maintenance.

Gestion de projet

Claude Triolaire (Bull) présente Symphony une démarche de gestion de projet, assistée par un ensemble d'outils. Symphony préconise une approche gestion de projet, fondée sur la méthode PMI (Project Management Institute) en s'appuyant sur les concepts de WBS (Work Break down Structure) d'OBS (Organization Break down Structure) et de la courbe en S.

Cette démarche est plus une démarche de suivi de projet que d'estimations. Elle apporte une aide à la gestion du projet, lorsque l'on sait estimer les différentes tâches élémentaires qui constituent le projet.

Les méthodes d'estimation sont-elles pertinentes ?

Rolande Marcinak, secrétaire de l'AFITEP[2], anime une première table ronde.

Les questions abordées par la salle portent sur les thèmes les plus cruciaux.

- Les statistiques prennent-elles en compte les projets qui échouent ?
- La réalisation du logiciel ne représente que 20 % du coût total du projet.
- Comment modéliser l'utilisateur du commerce électronique que l'on ne connaît pas ?
- Comment estimer la réalisation de site web, de places de marché ?
- Comment tenir comptes des additions demandées en cours de projet ?
- Les chefs de projet doivent-ils se préoccuper du marketing de l'analyse de la valeur ?

---

[2] *LETTRE n° 36 – juillet 1999 – ADELI prend un risque*
LETTRE n° 37 – octobre 1999 - Les risques des projets informatiques

Développement d'un projet UML

Philippe Larvet (Alcatel) et Frédérique Vallée (Mathix) préconisent une démarche en 4 étapes :

- définir la caractéristique de coût à évaluer (c'est la variable à expliquer) - Par exemple, on peut estimer le coût total à partir du coût de codage (en s'appuyant sur une proportionnalité statistique) ;
- définir les caractéristiques utiles et connues en amont (ce sont les variables explicatives) ;
- recueillir les données ;
- analyser les données et construire un modèle de coût.

Prenons le cas où l'objectif principal est l'évaluation du nombre de classes métiers à concevoir.

On part des métriques textuelles candidates que l'on affine progressivement :

- nombre total de mots, nombre de mots conservés ;
- nombre de classes candidates automatiquement calculées (par l'analyseur linguistique) ;
- nombre de classes candidates revues manuellement (d'après l'évaluation automatique) ;
- nombre de « vraies classes » (déterminé manuellement).

Le coût de codage (qui ne représente que 17 % du coût du développement) est proportionnel au nombre de classes.

L'objectif secondaire consiste à apprécier l'impact d'autres caractéristiques du projet :

- nombre de fonctions à développer ;
- nombre d'exigences à couvrir ;
- nombre de « use cases » de premier niveau ;
- taux de réutilisation ;
- langage de développement ;
- méthode de développement.

Les premiers résultats sont encourageants. Toutefois, les deux modèles doivent être affinés à l'aide :

- de nouvelles métriques ;
- d'un échantillon plus important ;
- d'autres techniques statistiques de modélisation.

Les modèles, construits à partir d'un retour d'expérience, sont spécifiques de cette expérience.

Estimations de grands projets reposant sur la réutilisation

François de Verdière (IMR Global) n'hésite pas à lancer cette formule brutale : « **L'appel d'offres dans un périmètre peu défini est un appel au meurtre ».**

Il évoque les plus fréquentes causes d'échec des projets :

- périmètre flou ;
- pas de métrique ;
- variation de productivité (peut varier de 1 à 5 d'un contributeur à un autre) ;
- spécificité de l'industrie du logiciel : on met 2 ans pour passer chef de projet, mais celui-ci doit affronter une nouvelle vague technologique tous les … 2 ans.

Les unités d'œuvre en pratique

François Pupier présente la démarche utilisée par STERIA dans ses réponses aux appels d'offres. Une méthode d'estimation pertinente doit être basée sur des mesures ; elle doit être vérifiable, reproductible et perfectible.

On peut combiner trois familles de méthodes :

- analytiques par décomposition en tâches élémentaire ;
- analogiques par comparaison ;
- algorithmiques.

Il faut spécialiser les méthodes par type de projet (ERP, Internent, maintenance applicative).

Il faut distinguer nettement :

- les unités d'œuvre qui mesurent objectivement l'ouvrage à construire,
- des éléments de productivité qui fournissent des prévisions de charges en fonction des contexte de réalisation de l'ouvrage.

Un outil d'estimation paramétrique

Pascal Gendrot présente l'outil PRICE S, illustré par un témoignage d'utilisateur : Christian Gazaube (EADS) qui utilise l'outil pour des estimations de coût en modification.

Débat

Yves Chochon a animé le débat de clôture de ces deux journées, particulièrement riches en présentations et en échanges sur un sujet qui reste en actualité permanente.

Conclusion

Au début du 21ème siècle, les piliers des estimations de charge des projets informatiques restent COCOMO et les Points de Fonctions, nés au début des années 1980.

Ces modèles intègrent des facteurs pragmatiques :

- levée progressive des incertitudes, au fur et à mesure de l'avancement du projet ;
- nécessité de passer de la mesure des dimensions de l'ouvrage aux prévisions de charge en tenant compte des conditions de réalisation ;
- recours à des historiques de production, pour des projets analogues ;
- apport des outils pour automatiser les calculs et gérer des données statistiques.

Au-delà du développement de logiciel, on commence à disposer de méthodes et de techniques pour estimer les charges de maintenance de système.

En revanche, en raison de leur relative nouveauté, les projets d'intégration, de mise en œuvre de progiciels (ERP) ne disposent pas de la même expérience et en sont réduits à utiliser des démarches moins formalisées.

Une famille de projets pose un problème qui reste insoluble ; on ne sait toujours pas estimer les charges d'un projet novateur : réalisation d'un ouvrage original en utilisant des méthodes et des techniques trop nouvelles pour être parfaitement maîtrisées par les acteurs.

Bibliographie

Productivité des programmeurs - Jacques Printz, Éditions Hermès Science Publications (2001)

IFPUG 4.1 Counting Practises Manual, IFPUG, Janvier 1999

B. Boehm, C. Abts, A. Winsor Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, B. Steece : « Software cost estimation with COCOMO II », Prentice Hall, ISBN : 0-13-026692-2 (2000).

Coûts et durée des projets informatiques – pratique des modèles d'estimation

Au cours du séminaire, on a annoncé la prochaine parution de l'ouvrage « Coûts et durée des projets informatiques – pratique des modèles d'estimation » par Jacques Printz, Nicolas Trèves, Christiane Deh, Bernard Mesdon, aux Éditions Hermès Science Publications (2001).

- Jacques Printz est titulaire de la Chaire de Génie Logiciel au Conservatoire National des Arts et Métiers et directeur de l'institut du CNAM CMSL qui a pour mission la collecte et la diffusion des savoir-faire en architecture et méthodologie logicielle.
- Christiane Deh et Bernard Mesdon sont consultants chez Synesys, société de conseil en systèmes d'information.
- Nicolas Trèves, membre d'ADELI depuis 1994, est chargé de cours au CNAM et chargé de mission au CMSL.

En 1995, une étude du Standish Group dressait un tableau accablant de la conduite des projets informatiques. Reposant sur un échantillon représentatif de 365 entreprises, totalisant 8 380 applications, cette étude établissait que :

- 16,2 % seulement des projets étaient conformes aux prévisions initiales ;
- 52,7 % avaient subi des dépassements en coût et délai d'un facteur 2 à 3 avec diminution du nombre de fonctions offertes ;
- 31,1 % ont été purement abandonnés durant leur développement.

Pour les grandes compagnies, le taux de succès tombait à 9 % alors que seulement 42 % des fonctionnalités commandées ont été effectivement livrées.

Ces statistiques peuvent être interprétées de différentes façons selon qu'on les considère du point de vue du maître d'ouvrage ou de celui du maître d'œuvre.

Du point de vue du maître d'ouvrage, elles indiquent une incapacité à sélectionner le maître d'œuvre qui saura réaliser le système souhaité, aux conditions économiques de coût, qualité, fonctionnalité et délai.

Du point de vue du maître d'œuvre, elles indiquent soit une incapacité à faire un devis sérieux des travaux à réaliser pour livrer le système commandé aux conditions fixées par le contrat puis à diriger la réalisation soit une incapacité à dialoguer avec le maître d'ouvrage, ne serait-ce que pour lui expliquer que le système commandé est infaisable aux conditions fixées par le contrat ou, encore, que l'expression de besoin est trop instable ou économiquement mal fondée pour développer quoi que ce soit de solide.

La qualité de la relation entre les maîtrises d'ouvrage et d'œuvre est donc une condition nécessaire au bon déroulement de la transaction entre les différents acteurs qui, le moment venu, permettra d'arrêter le prix du contrat. Ce n'est malheureusement pas une condition suffisante, car les projets informatiques souffrent d'un certain nombre d'impondérables qui rendent leur estimation initiale particulièrement risquée.

Pour un bon déroulement, il faut :

- mettre en place une mécanique de gestion de risque permettant l'identification des incertitudes ;
- organiser le projet pour que - le risque une fois détecté, et bien compris- une solution soit apportée qui ne mette pas en péril l'équilibre économique du projet.

Le chapitre 1 de l'ouvrage introduit cette problématique, en montrant qu'un modèle d'estimation est indissociable des projets eux-mêmes dont il est le centre décisionnel.

Estimer l'effort et le délai nécessaires à la réalisation d'un projet informatique, pour un niveau de qualité donné, reste un exercice hasardeux lorsqu'il est pratiqué de façon naïve et peu rigoureuse.

Estimer un projet informatique, c'est mettre en place un modèle d'estimation qui prend en compte les paramètres techniques du projet tout autant que les paramètres organisationnels et humains.

Au départ de tout projet informatique, il y a des paramètres connus à partir desquels on peut déduire les paramètres inconnus, avec une certaine marge d'erreur qui se réduira avec le temps.

Un modèle comme COCOMO 2000, présenté dans le chapitre 2 de l'ouvrage, met l'accent sur le nombre de lignes de code et le cycle de développement ; il met bien en évidence les conséquences d'un délai trop court.

Un modèle comme les « points de fonctions », décrit dans le chapitre 3, prend comme point de départ la complexité des données, données à partir desquelles on pourra cadrer l'effort total nécessaire à la réalisation des traitements.

Les deux modèles peuvent se combiner pour réduire les fourchettes d'estimation.

Estimer correctement les projets informatiques, suivre les paramètres du modèle d'estimation tout au long de la réalisation, comprendre les écarts entre la réalité et le modèle est un enjeu de management très important pour les maîtres d'ouvrage et les maîtres d'œuvre de projets informatiques quelle qu'en soit la taille.

D'où la nécessité d'illustrer la problématique à l'aide d'études de cas, telles que celles présentées dans le chapitre 4, portant sur trois grandes classes de projets de développement et d'intégration : développement d'applications de gestion, d'infrastructures logicielles et d'applications systèmes, intégration de systèmes.

Nicolas Trèves
Alain Coulon