

Coding Week

École Centrale
Casablanca
2026

Casablanca, Maroc

CODING WEEK

Programmation structurée



Le coût du changement à A1, Coding Week 1A

Hermann Agossou

Doctorant, UTER MID@S, ECC / UH2C

¹École Centrale Casablanca

Le critère

01

Le critère

Ce que vous savez déjà des objets

Trois questions avant de commencer

Encapsulation

Qu'est-ce qu'elle protège exactement : les données, ou la liberté de les changer ?

Héritage

Une classe fille peut-elle toujours remplacer sa classe mère ?

Méthode privée

Pourquoi la rendre privée ? Contre qui la protège-t-on ?

RÉPONDEZ D'ABORD. ON NE CORRIGE PAS MAINTENANT.

Ces trois réponses fixent le niveau du module suivant. Celles qui sont fausses aujourd'hui reviendront en A2, et c'est voulu.

Deux versions du même programme

Elles font exactement la même chose

VERSION A

'flat_report.py' formate dans 'render_invoice'.

- ▶ Le format monétaire apparaît sur chaque ligne
- ▶ Le total répète la même décision
- ▶ L'appelant fournit aussi 'row_index'

VERSION B

'hidden_report.py' nomme les décisions.

- ▶ 'format_money' possède la présentation
- ▶ 'render_invoice' possède la numérotation
- ▶ L'interface publique est plus étroite

MÊME COMPORTEMENT, MÊMES TESTS AU VERT

Levez la main : laquelle préférez-vous maintenir ? Gardez votre raison en tête.

Le critère

Complexité, dépendance, obscurité

Nommer les choses avant de les mesurer

Complexité

Tout ce qui rend un système difficile à comprendre ou à modifier.

Dépendance

Du code qui ne peut pas être compris ou changé isolément.

Obscurité

Une information nécessaire qui n'est pas visible là où on en a besoin.

Aucun de ces mots n'est un défaut en soi. Ce sont des grandeurs, pas des jugements.

Ousterhout : la complexité se mesure au changement

Une définition opérationnelle, pas une opinion

DÉFINITION

La complexité est tout ce qui, dans la structure d'un système, le rend difficile à comprendre ou à modifier. Elle se juge au coût du changement, pas au nombre de lignes.

Voir: [2] Ousterhout (2018)

Conséquence immédiate : n meilleur z devient une question à laquelle on peut répondre. On choisit un changement, et on compte ce qu'il touche.

D'où vient la complexité

Amplification, charge cognitive, inconnues inconnues

- ▶ Amplification : la version A répète la présentation monétaire.
- ▶ Charge cognitive : elle mélange calcul, numérotation et affichage.
- ▶ Inconnue inconnue : un nouvel appelant doit connaître 'row_index'.

ELLE S'ACCUMULE

Aucun changement isolé ne crée la complexité. C'est pour cela qu'on ne la corrige pas en une fois, et qu'il faut un critère appliqué en continu.

Comparer deux implémentations

Exemple entièrement traité

`'FLAT_REPORT.PY'`

Deux expressions possèdent la décision de formatage.

- ▶ Ligne : devise, puis montant
- ▶ Total : devise, puis montant

`'HIDDEN_REPORT.PY'`

Une seule frontière possède la décision de formatage.

- ▶ `'format_money'` reçoit montant et devise
- ▶ `'render_invoice'` réutilise cette frontière

VÉRIFICATION : A TOUCHE 2 EXPRESSIONS, B TOUCHE 1 FRONTIÈRE

Appliquez le changement dans les deux fichiers, exécutez `'verify.py'` après avoir adapté l'attendu, puis comparez le diff à la prédiction.

Le critère

Terminer l'analyse commencée

La moitié du raisonnement est donnée

Changement demandé : séparer le code et le montant par une espace insécable.

Endroit à modifier	Version A	Version B
Lignes de facture	oui	via 'format_money'
Affichage du total	?	?
Interface publique	?	?
Total des endroits	?	?

À COMPLÉTER, PUIS CONCLURE EN UNE PHRASE

Quelle version choisiriez-vous, et pour quelle raison exprimée en coût de changement ?

Le découpage

02

Parnas : cacher les décisions de conception

Le critère de découpage, énoncé en 1972

LE CRITÈRE

Découper un système autour des décisions de conception que chaque module cache aux autres, et non autour des étapes du traitement. Une décision qui change reste alors confinée dans un seul module.

Voir: [1] Parnas (1972)

Un article sur un critère, pas sur une technologie. C'est pour cela qu'il a survécu à tous ses outils.

Masquer l'information, pas les étapes

Le découpage par flux est le piège courant

DÉCOUPAGE PAR FLUX

Une fonction par étape du traitement.

- ▶ lire calculer formater écrire
- ▶ Le format du fichier est connu de trois modules
- ▶ Changer le format touche trois modules

DÉCOUPAGE PAR DÉCISION

Un module par décision cachée.

- ▶ stockage / calcul / présentation
- ▶ Le format n'est connu que du stockage
- ▶ Changer le format touche un module

Les deux découpages ont le même nombre de fonctions. Un seul confine le changement.

Modules profonds et modules superficiels

Ce que l'interface coûte par rapport à ce qu'elle cache

MODULE PROFOND

Interface étroite, implémentation substantielle.

- ▶ `'store.save(report)'`
- ▶ Cache format, chemin, encodage, erreurs
- ▶ L'appelant n'apprend rien de cela

MODULE SUPERFICIEL

Interface presque aussi complexe que ce qu'elle cache.

- ▶ `'write(path, sep, encoding, mode)'`
- ▶ Cache : un appel à `'open'`
- ▶ L'appelant doit tout savoir quand même

CE N'EST PAS UNE QUESTION DE TAILLE

La taille ne décide de rien. Ce qui compte est le rapport entre ce que l'interface exige et ce qu'elle épargne.

Découper une fonction enchevêtrée

Et nommer ce que chaque partie cache

AVANT : 'MONTHLY_REPORT' POSSÈDE TROIS DÉCISIONS

La même fonction connaît le format CSV, la colonne à sommer et la présentation monétaire en EUR.

01 load(path)

Cache : comment les données sont stockées et lues.

02 total(rows)

Cache : ce que n total z signifie ici.

03 render(total)

Cache : devise, format d'affichage, langue.

VÉRIFICATION

Changez une décision cachée, par exemple le format, et confirmez que la modification reste dans une seule partie. Si elle se répand, le découpage était par flux.

À vous : découper et justifier

Sans aide, puis défense à l'oral

CONSIGNE

Découpez la fonction distribuée en autant de parties que nécessaire, puis écrivez pour chacune la décision de conception qu'elle cache.

- ▶ Une phrase par partie, commençant par *í* cache la décision : *ž*
- ▶ Une partie dont vous ne savez pas dire ce qu'elle cache est mal découpée
- ▶ Défense à l'oral : on vous nommera un changement, vous direz ce qu'il touche

5 minutes. Deux groupes passent au tableau.

Quand l'abstraction fuit

Signature propre, appelants qui savent tout

INTERFACE ANNONCÉE

'store.load()' renvoie des lignes.

- ▶ L'appelant croit ignorer le stockage

OBLIGATION CACHÉE

L'appelant teste encore 'store.kind == csv'.

- ▶ Il retire lui-même l'en-tête du CSV

LE COÛT D'UNE FUITE

Il se mesure à ce que l'appelant est obligé de savoir. Ici, tout appelant doit connaître le format. La décision n'a jamais été cachée, elle a seulement été déplacée.

On y revient en A2 : un contrat rend explicite ce que l'appelant a le droit de supposer.

Ce qui reste quand le langage change

Un critère, pas une liste de règles

- ▶ La complexité se mesure au coût du changement, pas au nombre de lignes.
- ▶ On découpe autour des décisions cachées, pas des étapes du traitement.
- ▶ La profondeur est un rapport : ce que l'interface exige contre ce qu'elle épargne.
- ▶ Nommer, c'est concevoir : un nom est la plus petite interface qu'un lecteur rencontre.

CE QUI ÉTAIT DU PYTHON, ET RIEN D'AUTRE

Les fonctions, les modules, les imports. Les outils changeront ; le critère qui vous a servi à les juger, non.

Références

03