

Analyse : de l'exigence à la spécification

CM1 — Pourquoi analyser, et comment recueillir un besoin

Pierre Valarcher `pierre.valarcher@u-pec.fr`

BUT2 Informatique

2 septembre 2026

#	Partie	Durée
1	Pourquoi l'analyse coûte moins cher que la correction	20 min
2	La demande n'est pas le besoin	20 min
3	Exigences fonctionnelles et non fonctionnelles	25 min
4	Ce qui vous attend en TD et en TP	10 min

Construire le **bon** produit,
ou construire le produit **correctement** ?

Ce sont deux questions différentes

L'analyse répond à la première. La conception et le développement répondent à la seconde.

Un produit parfaitement construit qui ne résout pas le bon problème est un échec complet.

Construire le **bon** produit,
ou construire le produit **correctement** ?

Ce sont deux questions différentes

L'analyse répond à la première. La conception et le développement répondent à la seconde.

Un produit parfaitement construit qui ne résout pas le bon problème est un échec complet.

Le coût d'un défaut selon le moment où on le trouve

Phase où le défaut est détecté	Coût relatif
Analyse des besoins	× 1
Conception	× 5
Développement	× 10
Tests / recette	× 20 à 50
Production	× 100 et au-delà

Barry Boehm, *Software Engineering Economics*, 1981.

Pourquoi cette explosion ?

Un besoin mal compris se propage *silencieusement* dans tous les artefacts construits ensuite : conception, code, tests, documentation, formation. Le corriger en production, c'est défaire puis refaire toute la chaîne.

Le coût d'un défaut selon le moment où on le trouve

Phase où le défaut est détecté	Coût relatif
Analyse des besoins	× 1
Conception	× 5
Développement	× 10
Tests / recette	× 20 à 50
Production	× 100 et au-delà

Barry Boehm, *Software Engineering Economics*, 1981.

Pourquoi cette explosion ?

Un besoin mal compris se propage *silencieusement* dans tous les artefacts construits ensuite : conception, code, tests, documentation, formation. Le corriger en production, c'est défaire puis refaire toute la chaîne.

Quatre échecs, quatre causes qui sont notre programme

Projet	Cause dominante	Coût
VCF du FBI (2005)	exigences changeantes, jamais figées	170 M\$
NHS (2011)	livraison « big bang », utilisateurs non consultés	10 G£
Aéroport de Denver (1995)	complexité sous-estimée, tests irréalistes	560 M\$
Mariner 1 (1962)	une ligne de code non testée	18,5 M\$

Ce ne sont pas des anecdotes

Ce sont respectivement : l'échec de l'**analyse des exigences**, du **recueil des besoins**, de la **stratégie de test**, et du **test unitaire**. Chaque étape de ce cours répond à l'un de ces risques.

Quatre échecs, quatre causes qui sont notre programme

Projet	Cause dominante	Coût
VCF du FBI (2005)	exigences changeantes, jamais figées	170 M\$
NHS (2011)	livraison « big bang », utilisateurs non consultés	10 G£
Aéroport de Denver (1995)	complexité sous-estimée, tests irréalistes	560 M\$
Mariner 1 (1962)	une ligne de code non testée	18,5 M\$

Ce ne sont pas des anecdotes

Ce sont respectivement : l'échec de l'**analyse des exigences**, du **recueil des besoins**, de la **stratégie de test**, et du **test unitaire**. Chaque étape de ce cours répond à l'un de ces risques.

18 chercheurs, 4 doctorants, des jeux de données génomiques sur une dizaine de postes Linux et macOS. Pas de service informatique : Karim gère les machines « en plus » de son travail.

Sa demande

« Il me faudrait un script qui fasse un `tar.gz` de `/data` toutes les nuits sur le NAS. Ça devrait être vite fait, non ? »

Trois mots qui trahissent

script tar.gz toutes les nuits

Ce sont des éléments de **solution**.
Aucun ne dit quel **problème** est résolu.

18 chercheurs, 4 doctorants, des jeux de données génomiques sur une dizaine de postes Linux et macOS. Pas de service informatique : Karim gère les machines « en plus » de son travail.

Sa demande

« Il me faudrait un script qui fasse un `tar.gz` de `/data` toutes les nuits sur le NAS. Ça devrait être vite fait, non ? »

Trois mots qui trahissent

script tar.gz toutes les nuits

Ce sont des éléments de **solution**.
Aucun ne dit quel **problème** est résolu.

Deux choses à ne jamais confondre

La demande

La solution que le client a *déjà* choisie.

Visible, concrète, formulée spontanément.

Le besoin

Le problème qu'il faut résoudre.

Immergé, souvent jamais formulé.

« Je veux un bouton pour exporter en PDF »



« Je dois transmettre chaque semaine à ma direction un rapport de synthèse non modifiable »

L'analyste n'est pas un scribe qui transcrit une commande : c'est un **enquêteur** qui cherche le *pourquoi* derrière le *quoi*.

Deux choses à ne jamais confondre

La demande

La solution que le client a *déjà* choisie.

Visible, concrète, formulée spontanément.

Le besoin

Le problème qu'il faut résoudre.

Immergé, souvent jamais formulé.

« Je veux un bouton pour exporter en PDF »



« Je dois transmettre chaque semaine à ma direction un rapport de synthèse non modifiable »

L'analyste n'est pas un scribe qui transcrit une commande : c'est un **enquêteur** qui cherche le *pourquoi* derrière le *quoi*.

Deux choses à ne jamais confondre

La demande

La solution que le client a *déjà* choisie.

Visible, concrète, formulée spontanément.

Le besoin

Le problème qu'il faut résoudre.

Immergé, souvent jamais formulé.

« Je veux un bouton pour exporter en PDF »



« Je dois transmettre chaque semaine à ma direction un rapport de synthèse non modifiable »

L'analyste n'est pas un scribe qui transcrit une commande : c'est un **enquêteur** qui cherche le *pourquoi* derrière le *quoi*.

La méthode des « 5 Pourquoi »

Née du *Toyota Production System* (Taiichi Ohno, années 1950). On ne s'arrête pas au premier symptôme : on demande « pourquoi ? » jusqu'à atteindre la cause racine.

⇒ Fil rouge — snapguard : la demande de Karim

- ❶ *Pourquoi un tar.gz nocturne ?* — Pour avoir une copie des données.
- ❷ *Pourquoi une copie ?* — Pour restaurer en cas de perte.
- ❸ *Pourquoi cela n'a-t-il pas marché la dernière fois ?* — La perte a été découverte après l'écrasement des deux seules archives.
- ❹ *Pourquoi n'en garder que deux ?* — Chaque archive fait la taille complète des données.
- ❺ *Pourquoi une archive complète pour des données qui bougent peu ?* — Parce que tar ne sait pas faire autrement.

Cause racine

Le besoin n'est pas « faire une copie ». C'est **pouvoir revenir à l'état de n'importe quel fichier à n'importe quelle date des N derniers jours, avec l'espace disque dont on dispose**. La contrainte d'espace n'est pas un détail : c'est *elle* qui fait échouer la solution demandée.

La méthode des « 5 Pourquoi »

Née du *Toyota Production System* (Taiichi Ohno, années 1950). On ne s'arrête pas au premier symptôme : on demande « pourquoi ? » jusqu'à atteindre la cause racine.

⇒ Fil rouge — snapguard : la demande de Karim

- ❶ *Pourquoi un tar.gz nocturne ?* — Pour avoir une copie des données.
- ❷ *Pourquoi une copie ?* — Pour restaurer en cas de perte.
- ❸ *Pourquoi cela n'a-t-il pas marché la dernière fois ?* — La perte a été découverte après l'écrasement des deux seules archives.
- ❹ *Pourquoi n'en garder que deux ?* — Chaque archive fait la taille complète des données.
- ❺ *Pourquoi une archive complète pour des données qui bougent peu ?* — Parce que tar ne sait pas faire autrement.

Cause racine

Le besoin n'est pas « faire une copie ». C'est **pouvoir revenir à l'état de n'importe quel fichier à n'importe quelle date des N derniers jours, avec l'espace disque dont on dispose**. La contrainte d'espace n'est pas un détail : c'est *elle* qui fait échouer la solution demandée.

L'enquête se mène avec des questions ouvertes

À faire

- « Décrivez-moi une journée type avec le système actuel. »
- « Qu'est-ce qui vous frustre le plus aujourd'hui ? »
- « Racontez-moi la dernière fois où cela a posé un vrai problème. »

Révèlent le contexte, la douleur, la valeur attendue.

À ne pas faire

- « Voulez-vous un bouton bleu ici ? »
- « Donc il vous faut juste un tableau ? »
- « Le système doit-il utiliser SQL ? »

Ferment la conversation et orientent prématurément vers la technique.

La question la plus rentable de l'analyse :
« Racontez-moi la dernière fois où ce problème s'est produit. »

L'enquête se mène avec des questions ouvertes

À faire

- « Décrivez-moi une journée type avec le système actuel. »
- « Qu'est-ce qui vous frustre le plus aujourd'hui ? »
- « Racontez-moi la dernière fois où cela a posé un vrai problème. »

Révèlent le contexte, la douleur, la valeur attendue.

À ne pas faire

- « Voulez-vous un bouton bleu ici ? »
- « Donc il vous faut juste un tableau ? »
- « Le système doit-il utiliser SQL ? »

Ferment la conversation et orientent prématurément vers la technique.

La question la plus rentable de l'analyse :

« Racontez-moi la dernière fois où ce problème s'est produit. »

Ce que l'enquête chez BioSeq a fait apparaître

	Problème métier constaté	Preuve
P1	profondeur d'historique (2 j) < délai de détection d'un incident (4 j)	incident de mars
P2	restaurer un seul fichier impose de décompresser 400 Gio	déclaration de Karim
P3	personne ne vérifie que la sauvegarde de la nuit a réussi	fin d'entretien
P4	le parc est à la fois Linux et macOS	inventaire

Le piège évité

Répondre « d'accord » à la demande initiale aurait produit un script de 15 lignes, livré en une heure, **techniquement irréprochable et parfaitement inutile** : il aurait reproduit la situation qui a causé la perte de données.

Ce que l'enquête chez BioSeq a fait apparaître

	Problème métier constaté	Preuve
P1	profondeur d'historique (2 j) < délai de détection d'un incident (4 j)	incident de mars
P2	restaurer un seul fichier impose de décompresser 400 Gio	déclaration de Karim
P3	personne ne vérifie que la sauvegarde de la nuit a réussi	fin d'entretien
P4	le parc est à la fois Linux et macOS	inventaire

Le piège évité

Répondre « d'accord » à la demande initiale aurait produit un script de 15 lignes, livré en une heure, **techniquement irréprochable et parfaitement inutile** : il aurait reproduit la situation qui a causé la perte de données.

Trois qualités, et une seule qui compte vraiment

Une exigence utile est...

non ambiguë sinon le développeur et le client comprendront deux choses différentes ;

complète sinon l'équipe comblera les trous avec sa propre logique ;

vérifiable sinon on ne pourra jamais prouver que le travail est terminé.

Une exigence non testable n'est pas une spécification :
c'est un vœu.

Trois qualités, et une seule qui compte vraiment

Une exigence utile est...

non ambiguë sinon le développeur et le client comprendront deux choses différentes ;

complète sinon l'équipe comblera les trous avec sa propre logique ;

vérifiable sinon on ne pourra jamais prouver que le travail est terminé.

Une exigence non testable n'est pas une spécification :
c'est un vœu.

Exigences fonctionnelles (EF)

Ce que le système FAIT.

Fonctionnalités, comportements, actions.

« L'utilisateur doit pouvoir se connecter avec un email et un mot de passe. »

Exigences non fonctionnelles (ENF)

COMMENT il le fait.

Qualités, contraintes, caractéristiques.

« Le temps de connexion doit être inférieur à 2 secondes. »

Les formulations qui ne sont pas des exigences

✗ « Le site doit être rapide » — rapide pour qui, dans quelles conditions ? ✗ « Le système doit être sécurisé » — contre quelles menaces ? ✗ « L'interface doit être intuitive » — pour un expert ou un novice ?

Exigences fonctionnelles (EF)

Ce que le système FAIT.

Fonctionnalités, comportements, actions.

« *L'utilisateur doit pouvoir se connecter avec un email et un mot de passe.* »

Exigences non fonctionnelles (ENF)

COMMENT il le fait.

Qualités, contraintes, caractéristiques.

« *Le temps de connexion doit être inférieur à 2 secondes.* »

Les formulations qui ne sont pas des exigences

✗ « Le site doit être rapide » — rapide pour qui, dans quelles conditions ? ✗ « Le système doit être sécurisé » — contre quelles menaces ? ✗ « L'interface doit être intuitive » — pour un expert ou un novice ?

Caractéristique	Question à laquelle elle répond
Adéquation fonctionnelle	fait-il ce qui est attendu, correctement ?
Efficiencce / performance	est-il rapide et économe en ressources ?
Compatibilité	coexiste-t-il bien avec d'autres systèmes ?
Utilisabilité	est-il facile à apprendre et à utiliser ?
Fiabilité	reste-t-il disponible, récupère-t-il bien ?
Sécurité	protège-t-il les données et les accès ?
Maintenabilité	est-il facile à corriger et à faire évoluer ?
Portabilité	peut-on le déployer ailleurs ?

Usage pratique : passez ces huit lignes en revue à voix *haute* avec le client. Beaucoup d'ENF critiques n'existent que parce que quelqu'un a pensé à poser la question.

Caractéristique	Question à laquelle elle répond
Adéquation fonctionnelle	fait-il ce qui est attendu, correctement ?
Efficiencce / performance	est-il rapide et économe en ressources ?
Compatibilité	coexiste-t-il bien avec d'autres systèmes ?
Utilisabilité	est-il facile à apprendre et à utiliser ?
Fiabilité	reste-t-il disponible, récupère-t-il bien ?
Sécurité	protège-t-il les données et les accès ?
Maintenabilité	est-il facile à corriger et à faire évoluer ?
Portabilité	peut-on le déployer ailleurs ?

Usage pratique : passez ces huit lignes en revue à voix *haute* avec le client. Beaucoup d'ENF critiques n'existent que parce que quelqu'un a pensé à poser la question.

⇒ Fil rouge — snapguard : quelques exigences non fonctionnelles

ENF-PERF-01	un instantané incrémental de 500 000 fichiers dont 1 % modifiés : < 30 min
ENF-FIA-01	une interruption ne rend jamais illisible un instantané antérieur
ENF-COMP-01	un instantané reste exploitable avec ls, cp, diff, tar sans notre outil
ENF-PORT-01	compile et passe les tests sur Linux et macOS

ENF-COMP-01 est probablement la plus importante du cahier

Elle dit : *si notre outil disparaît, les données restent récupérables*. Aucun utilisateur ne la formulera jamais — et elle disqualifie d'emblée plusieurs solutions qu'aucune exigence fonctionnelle n'écartait.

⇒ Fil rouge — snapguard : quelques exigences non fonctionnelles

- ENF-PERF-01** un instantané incrémental de 500 000 fichiers dont 1 % modifiés : < 30 min
- ENF-FIA-01** une interruption ne rend jamais illisible un instantané antérieur
- ENF-COMP-01** un instantané reste exploitable avec `ls`, `cp`, `diff`, `tar` **sans notre outil**
- ENF-PORT-01** compile et passe les tests sur Linux et macOS

ENF-COMP-01 est probablement la plus importante du cahier

Elle dit : *si notre outil disparaît, les données restent récupérables*. Aucun utilisateur ne la formulera jamais — et elle disqualifie d'emblée plusieurs solutions qu'aucune exigence fonctionnelle n'écartait.

« *Le système doit analyser un fichier pour détecter la présence de virus.* »

Dans un antivirus

C'est la raison d'être du produit, l'action que l'utilisateur déclenche consciemment.

⇒ **exigence fonctionnelle**

Dans un site de partage de photos

L'utilisateur veut afficher sa photo. L'analyse est une contrainte de qualité qui encadre cette finalité.

⇒ **exigence non fonctionnelle**

La question à se poser : cette action est-elle une *finalité* pour l'utilisateur, ou une *contrainte de qualité* qui encadre une autre finalité ?

« *Le système doit analyser un fichier pour détecter la présence de virus.* »

Dans un antivirus

C'est la raison d'être du produit, l'action que l'utilisateur déclenche consciemment.

⇒ **exigence fonctionnelle**

Dans un site de partage de photos

L'utilisateur veut afficher sa photo. L'analyse est une contrainte de qualité qui encadre cette finalité.

⇒ **exigence non fonctionnelle**

La question à se poser : cette action est-elle une *finalité* pour l'utilisateur, ou une *contrainte de qualité* qui encadre une autre finalité ?

Est-ce une EF ou une ENF ?

« Le système doit créer un lien dur plutôt qu'une copie pour un fichier inchangé. »

Ni l'une ni l'autre.

C'est une solution technique déguisée en exigence

La bonne exigence est : *« le système ne doit recopier que les fichiers modifiés ou nouveaux depuis l'instantané précédent »*.

Le lien dur est **une réponse possible** à cette exigence, pas l'exigence elle-même. Écrite au niveau du besoin, elle reste valide même si l'implémentation change complètement.

Est-ce une EF ou une ENF ?

« Le système doit créer un lien dur plutôt qu'une copie pour un fichier inchangé. »

Ni l'une ni l'autre.

C'est une solution technique déguisée en exigence

La bonne exigence est : *« le système ne doit recopier que les fichiers modifiés ou nouveaux depuis l'instantané précédent »*.

Le lien dur est **une réponse possible** à cette exigence, pas l'exigence elle-même. Écrite au niveau du besoin, elle reste valide même si l'implémentation change complètement.

👉 TD1 — Écrire les exigences de BioSeq

Confrontation de vos lectures de l'entretien, déroulé des 5 Pourquoi, puis **jeu de rôle** : vous interrogez Karim (joué par l'enseignant), qui répond *strictement* à ce qu'on lui demande.

Production : 3 EF vérifiables et 3 ENF chiffrées, couvrant trois familles ISO 25010 différentes.

👉 TP1 — Vos chiffres tiennent-ils debout ?

Vous emportez vos ENF chiffrées au clavier et vous les **mesurez**. Combien de temps prend vraiment la copie de 400 Gio ? Votre seuil était-il réaliste ?

Puis un laboratoire d'observation sur le système de fichiers, qui vous fera découvrir *par vous-mêmes* le mécanisme qui rend l'exigence centrale réalisable.

👉 TD1 — Écrire les exigences de BioSeq

Confrontation de vos lectures de l'entretien, déroulé des 5 Pourquoi, puis **jeu de rôle** : vous interrogez Karim (joué par l'enseignant), qui répond *strictement* à ce qu'on lui demande.

Production : 3 EF vérifiables et 3 ENF chiffrées, couvrant trois familles ISO 25010 différentes.

👉 TP1 — Vos chiffres tiennent-ils debout ?

Vous emportez vos ENF chiffrées au clavier et vous les **mesurez**. Combien de temps prend vraiment la copie de 400 Gio ? Votre seuil était-il réaliste ?

Puis un laboratoire d'observation sur le système de fichiers, qui vous fera découvrir *par vous-mêmes* le mécanisme qui rend l'exigence centrale réalisable.

- ① Lire le dossier BioSeq : contexte, demande de Karim, transcription de l'entretien.
- ② Surligner de deux couleurs : ce que Karim **demande**, ce dont il a **besoin**. Ce ne sont pas les mêmes phrases.
- ③ Relever les **trois faits chiffrés** de l'entretien.
- ④ Arriver avec **une question** que vous auriez posée et qui ne l'a pas été.

C'est *individuel* : la confrontation de vos surlignages ouvre le TD.

- ❶ Un défaut d'analyse coûte cent fois plus cher corrigé en production.
- ❷ La **demande** du client est une solution déjà choisie ; le **besoin** est le problème à résoudre. Creusez avec « pourquoi ? ».
- ❸ Les questions ouvertes révèlent, les questions fermées confirment ce que vous croyiez déjà savoir.
- ❹ Une exigence doit être non ambiguë, complète et surtout **vérifiable**.
- ❺ Les EF disent *ce que* fait le système, les ENF *comment*. La frontière dépend de la finalité du produit.
- ❻ ISO 25010 est votre liste de contrôle : huit questions à poser systématiquement.