elao_

REX : Gen AI et développement logiciel

Récit d'une mutation profonde

Xavier Gorse · Elao

Agile Tour Strasbourg, 29 mai 2026

Hara

Chargemap

agile tour
STRASBOURG

REBOOT
CONSEIL

SHODO

MTG
Strasbourg

Elsass JUG

{EPITECH}

Le contexte

- **Xavier Gorse :** cofondateur Elao et Rix, sur Lyon
- **Elao, 13 personnes, 20 ans :** équipe stable, applications sur mesure
- **Culture craft :** architecture, maintenabilité, tests
- **ADN Agile :** itératif et incrémental
- **Notre posture :** concepteurs avant développeurs

Notre engagement

- **Répondre à un besoin :** trouver des solutions
- **Être responsable :** de ce qu'on livre
- **Prendre du plaisir :** condition sine qua non
- **Construire la confiance :** équipe, client, utilisateur

Donner toutes les informations pour prendre les bonnes décisions.

Le moment où tout ça bascule

> *On change d'OS projet*

L'IA

- **Le constat :** l'IA peut écrire du code qu'on aurait écrit
- **Facilite l'exploration :** technique et fonctionnelle
- ***Harness* :** les outils d'orchestration (Claude Code) sont efficaces
- **Pour une équipe craft :** fascinant et déstabilisant

Si la machine produit ce qu'on aime produire, où est notre place ?

Nos craintes

- **L'organisation :** notre modèle tient-il toujours ?
- **L'équipe projet :** sans code, qu'est-ce qui nous lie ?
- **Perso :** sans code, où est mon plaisir ?

Ce que nous avons construit devait répondre à ces craintes.

Été 2025

- **Mode R&D intense :** Claude Code, Spec Driven Development
- **La question fondatrice :** peut-on garder le contrôle ?
- **À la rentrée :** ça marche, la confiance est là, le cadre aussi
- **Octobre 2025 :** toute l'équipe embarque

Ça fait 8 mois

et personne ne reviendra en arrière.

Diriger, pas subir

> *Ce n'est pas l'IA qui décide. C'est nous qui cadrons.*

Context engineering

- **La seule clé :** la bonne info au bon moment
- **Le prompt seul ne suffit pas :** ajouter du contexte à l'intention
- ***Garbage in, garbage out* :** mauvaise entrée, mauvaise sortie
- **Notre rôle :** ranger, hiérarchiser, choisir

---

*Le prompt définit la **tâche**. Le contexte définit le **cadre**.*

La documentation

Avant

- **Specs périmées** dès le RUN
- **Seul le code** fait foi ensuite
- **Doc isolée** du quotidien projet

Maintenant

- **Vit avec le code :** repo, Markdown
- **Source partagée :** humains et agents
- **Sans doc :** le workflow s'écroule

> *La doc n'est plus juste un **livrable**. C'est un **actif** opérationnel.*

Un seul repo

```
<repo-projet>/
├── docs/                    ← PRÉSENT
│   ├── product/               # fonctionnel
│   ├── implementation/        # lien fonctionnel ↔️ technique
│   ├── decisions/             # ADR, PDR, TDR
│   └── tech/                  # standards transverses
├── work/
│   ├── stories/             ← PASSÉ
│   └── backlog/             ← FUTUR
├── apps/                    # le code
└── ...
```

Notre boucle, par story

- `/stories:brief` : cadrer l'intention
- `/stories:draft` : challenger le besoin

`/stories:draft`

Phase	Expert	Output
1	PO	`context.md`
2	UX	`ux.md`
3	Archi	`architecture.md`
4	QA	`test-strategy.md`
5	Lead dev	`implementation-strategy.md`

> *Futurs experts : a11y, Legal, IA, Secu*

Notre boucle, par story

- `/stories:brief` : cadrer l'intention
- `/stories:draft` : challenger le besoin
- `/stories:plan` : découper, tests inclus
- `/stories:dev` : écrire le code tâche par tâche
- `/stories:done` : capitaliser, doc à jour

Une story, sur disque

```
work/stories/<epic>/<feature>/<YYYY-MM-DD>-<story>/
├── brief.md       ← cadrer
├── draft.md       ← cinq experts
├── plan.md        ← découper
├── dev.md         ← générer
└── done.md        ← capitaliser
```

> *Une fois `done`, le dossier est jetable. La connaissance, elle, reste.*

Human in the loop

- ***By design*** **:** à chaque étape, pas après coup
- **Garder la main :** valider, corriger, refuser
- **Réaligner si dérive :** hallucination, écart, intervention
- **On garde la responsabilité :** ce qu'on valide, on l'assume

*Pas de vibe coding : l'agent **exécute**, l'humain **décide**.*

Ce que ça a changé

> *Concrètement, dans notre quotidien.*

L'organisation, côté Elao

Avant

- **TJM :** on vend du temps
- **Prix au chrono :** estimation en jours
- **Risque côté client :** dérive prix/délai

Maintenant

- **Forfait par story :** chiffré après brief
- **Prix indexé valeur :** plus sur le chrono
- **Coût d'exécution piloté :** variabilité IA transparente

> *On a remis de **l'engagement** avec le client. On y a gagné en **confiance**.*

L'équipe, côté projet

Avant

- **Capacité :** mi-temps fonctionnel, 2 à 3 devs à plein temps
- **Conception :** sujets qui émergent en plein dev/recette
- **Revue de code :** systématique, exhaustive
- **Connaissance :** portée par l'équipe et le code

Maintenant

- **Capacité :** un seul dev à plein temps, calé sur la disponibilité client
- **Conception :** dérisquée en amont, peu d'allers-retours
- **Revue de spec :** on choisit ce qu'on passe
- **Connaissance :** on a confiance car elle est à jour

Chacun, côté perso

- **Notre place évolue :** chacun, pas que les devs
- **Le plaisir reste :** il était dans la solution, pas le code
- **Charge collective nouvelle :** conception à plusieurs, moins seul
- **Le *context switching* :** l'IA en fond, l'esprit dispersé

Reste une zone d'incertitude : l'impact de ce changement sur chacun.

Ce qui s'ouvre

> *L'humain au centre, plus que jamais*

Tous makers

- **Le mouvement de fond :** chacun peut agir sur le produit
- **Capacité d'agir partagée :** fonctionnel, UX, tech, infra
- **Coordination plus implicite :** formaliser, c'est partager
- **Pas demain matin :** mais ça s'amorce déjà

Le maker n'est plus seulement le dev. Qu'est-ce qui nous tient ensemble ?

Et demain ?

- **Organisation :** sortir les outils du CLI, repenser la transmission
- **Équipe :** faire tourner les rôles, protéger les respirations
- **Personnel :** cultiver ce qui ne se délègue pas

L'enjeu n'est plus technique. Il est avant tout humain et organisationnel.

L'IA ne prend pas notre place

Elle nous force à en trouver
une autre.

Merci

Des questions ?

Xavier Gorse · xavier.gorse@elao.com

Pour passer à la pratique : atelier Spec-Driven Development de Salah Fouraq, juste après, à 15h45.

Votre retour sur OpenFeedback