MÉTHODE SAKITI · ARCHITECTE DU SUCCÈS NUMÉRIQUE

CURSUS 2 · AXE A — DIRECTION TECHNIQUE

DU CODE À LA DIRECTION

Du développeur au DSI stratège : quatre jalons, une passerelle, quand l'expertise logicielle conduit à la direction du système d'information

27 → 40 → 92 → 91 · F2 → F2 → F6 → F6

MONOGRAPHIE APPROFONDIE · CURSUS & PLAN DE SUCCESSION

ANCRAGE UEMOA · GÉNIE LOGICIEL · CLOUD NATIVE

SOMMAIRE

01 Le cursus en un coup d'œil

02 Pourquoi cette voie — le contexte UEMOA

03 Jalon 1 · Ingénieur Logiciel Cloud-Native (27)

04 Jalon 2 · Architecte Logiciel Global (40)

05 La passerelle F2 → F6 — le franchissement décisif

06 Jalon 3 · Engineering Manager (92)

07 Jalon 4 · DSI Stratège (91)

08 Plan de mise en œuvre sur 30 mois

09 Étude de cas — la trajectoire de Moussa

10 Activer ce cursus — lecture RH & DSI

11 Conclusion

A Annexe A — récapitulatif des quatre jalons

B Annexe B — compétences & certifications

C Annexe C — glossaire & repères

D Annexe D — variantes, bifurcations & cursus voisins

E Annexe E — indicateurs de progression par jalon

F Annexe F — ressources de montée en compétences

G Annexe G — anti-modèles à éviter dans ce cursus

H Annexe H — les deux voies vers la direction : DSI ou CTO

I Annexe I — questions fréquentes

Cette monographie approfondit un unique cursus — le deuxième de la collection — jalon par jalon, de l'entrée au sommet, avec son plan de mise en œuvre et son étude de cas.

RÉSUMÉ EXÉCUTIF

Quatre jalons prouvés, une passerelle décisive : comment le génie logiciel mène, en quelques années, à la direction du système d'information.

Le cursus **« Du code à la direction »** conduit un développeur jusqu'au poste de DSI Stratège. Il enchaîne deux trajectoires internes à la Famille 2 — Ingénieur Logiciel Cloud-Native, puis Architecte Logiciel Global — qui bâtissent une expertise de conception, puis une unique passerelle vers la Famille 6 — Engineering Manager, puis DSI — qui transforme cette expertise en leadership.

Son enjeu se concentre sur un seul mouvement : le franchissement de l'ingénierie au management. Deux jalons relèvent de l'approfondissement ; le troisième, du changement de nature. C'est sur ce franchissement, sa compétence-pont et ses conditions de réussite, que cette monographie insiste le plus, car c'est lui qui fait — ou défait — les dirigeants issus du logiciel.

Pour les organisations de la zone UEMOA, ce cursus est un instrument de maîtrise applicative et de succession : former en interne le dirigeant qui connaît la chaîne de valeur logicielle, plutôt que de dépendre d'un marché de profils rares et chers. Ce document en détaille chaque jalon, le plan de mise en œuvre sur trente mois, une étude de cas chiffrée et la lecture RH & DSI pour l'activer.

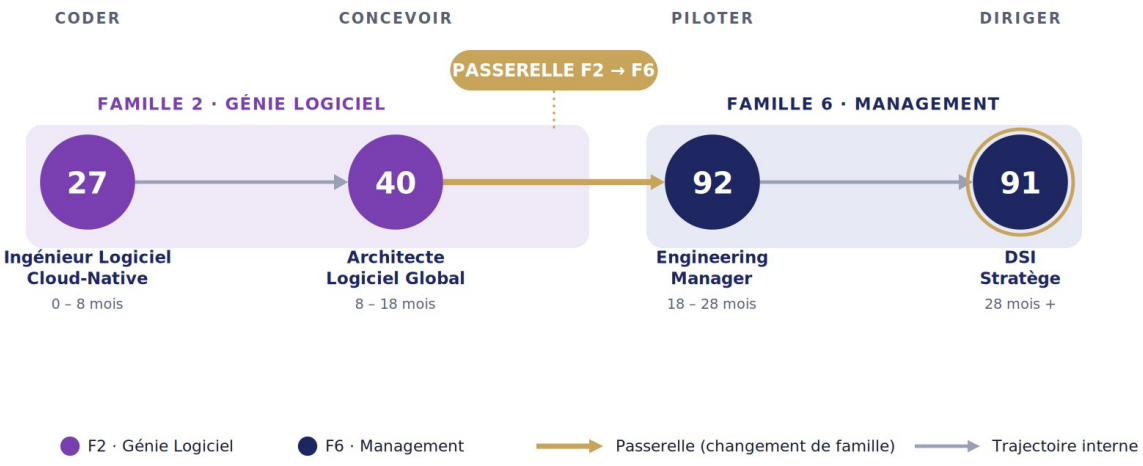
> **LE FIL CONDUCTEUR**
>
> Un principe traverse ce cursus : **la preuve, pas la promesse**. Chaque jalon se valide par un livrable déployable — un service en production, une architecture adoptée, un delivery prévisible, un schéma directeur porté au COMEX. C'est cette discipline qui sécurise la progression, du premier rôle technique jusqu'au sommet.

01 LE CURSUS EN UN COUP D'ŒIL

Du code au pilotage stratégique : quatre jalons, une seule passerelle, la voie qui transforme l'expertise logicielle en direction du système d'information.

Le cursus **« Du code à la direction »** relève de l'axe « Direction technique ». Il part d'un poste accessible à un développeur — l'**Ingénieur Logiciel Cloud-Native** — et conduit, en quatre jalons prouvés, jusqu'au poste de **DSI Stratège**. Sa logique : deux trajectoires internes à la Famille 2 (Génie Logiciel) qui bâtissent une expertise de conception, puis une unique passerelle vers la Famille 6 (Management) qui transforme cette expertise en autorité d'organisation.

Trois des quatre mouvements relèvent de la même logique : approfondir, prouver, capitaliser. Le mouvement central — le franchissement de l'ingénierie au management, de l'Architecte à l'Engineering Manager — est le moment décisif du cursus. C'est lui qui distingue un excellent architecte d'un dirigeant, et c'est sur lui que ce document insiste le plus.



Les quatre jalons

JALON	RÔLE (N°)	FAMILLE	NIVEAU	DURÉE
1	Ingénieur Logiciel Cloud-Native (27)	F2 · Génie Logiciel	Coder	0 – 8 mois
2	Architecte Logiciel Global (40)	F2 · Génie Logiciel	Concevoir	8 – 18 mois
3	Engineering Manager (92)	F6 · Management	Piloter	18 – 28 mois
4	DSI Stratège (91)	F6 · Management	Diriger	28 mois +

Profil d'entrée, profil de sortie

À l'entrée, un développeur qui écrit du code propre et testé. À la sortie, un dirigeant qui aligne le système d'information sur la stratégie d'une banque ou d'une fintech. Entre les deux, une transformation continue — **du coder au concevoir, puis du concevoir au diriger**.

> **LE MOUVEMENT DÉCISIF**
>
> La passerelle **F2 → F6** (jalon 2 → jalon 3) est le cœur de ce cursus. Elle s'appuie sur une compétence-pont précise : la coordination transverse que l'architecte exerce déjà entre les équipes, et l'autorité technique qu'il a gagnée. On ne franchit pas ce pas en quittant la conception, mais en l'élargissant aux personnes et au budget.

02 POURQUOI CETTE VOIE — LE CONTEXTE UEMOA

L'économie des API, la digitalisation bancaire et l'essor des fintechs font de l'ingénierie logicielle un actif stratégique — et de ce cursus une réponse à la pénurie de dirigeants issus du produit.

Dans la zone UEMOA, la valeur se déplace vers la couche applicative. Les banques digitalisent leurs parcours clients, les fintechs construisent des produits de paiement et de crédit, les agrégateurs interconnectent les opérateurs de mobile money. Tout cela repose sur du logiciel : des API robustes, des architectures qui tiennent la montée en charge, des équipes qui livrent vite et bien. Or les profils capables de concevoir ces systèmes — puis de diriger les organisations qui les produisent — sont rares.

Ce cursus répond à trois besoins structurels du marché régional. D'abord la **maîtrise de la chaîne applicative** : concevoir en interne ses produits plutôt que de dépendre de prestataires. Ensuite la **capacité à livrer à l'échelle** : structurer des équipes d'ingénierie productives et fiables. Enfin la **montée en valeur des talents internes**, des développeurs vers la direction, plus rapide et moins coûteuse qu'un recrutement externe.

Trois moteurs de la demande

MOTEUR	CE QU'IL IMPOSE	RÔLE DU CURSUS
Économie des API & mobile money	Des produits logiciels interopérables, robustes, conformes.	Forme le développeur, puis l'architecte qui structure ces produits.
Digitalisation bancaire	Des équipes qui livrent vite, à l'échelle, avec qualité.	Construit la culture d'ingénierie, du code au delivery management.
Pénurie de dirigeants produit	Des DSI et managers crédibles, issus du logiciel.	Trace une voie interne vers la direction, ancrée sur la preuve.

Pour une direction des systèmes d'information de la zone, ce cursus est un instrument de maîtrise et de souveraineté applicative. Le dirigeant issu du génie logiciel connaît intimement la chaîne de valeur qu'il pilote : il sait ce que coûte une dette technique, ce qu'exige une montée en charge, ce que vaut une équipe qui livre bien.

À qui ce cursus profite dans la zone

EMPLOYEUR	POSTE-CIBLE	POURQUOI CE CURSUS
Banque / établissement financier	DSI, Engineering Manager, architecte SI	Digitalisation, maîtrise applicative, alignement métier.
FinTech / agrégateur de paiement	Architecte, Head of Engineering	Produits d'API, scalabilité, time-to-market.
ESN / éditeur de logiciels	Architecte logiciel, directeur technique	Qualité, dette maîtrisée, structuration des équipes.
Grande entreprise / télécom	DSI, manager d'ingénierie	Transformation numérique, gouvernance du SI.

FIL ROUGE — LA BANQUE WAARIPAY

Tout au long de ce document, nous illustrerons le cursus par le cas d'une banque régionale fictive mais représentative, **WaariPay**, dont la plateforme de paiement est structurée en six domaines métier (bounded contexts), et qui cherche à former en interne le dirigeant capable d'en piloter l'évolution.

LE DIFFÉRENCIATEUR DU PROFIL

Un DSI issu de ce cursus possède un atout que peu partagent : il a **écrit le code, puis conçu l'architecture, puis dirigé les équipes** qu'il pilote désormais. Il ne demande jamais l'impossible à ses développeurs, estime les délais avec justesse et repère une fausse bonne idée technique avant qu'elle ne coûte cher. Dans un secteur où la crédibilité se gagne sur le terrain, cette continuité vaut de l'or.

03 JALON 1 · INGÉNIEUR LOGICIEL CLOUD-NATIVE (27)

Famille : F2 · Génie Logiciel **Mouvement :** Départ **Niveau :** Coder **Durée :** 0 – 8 mois

Le point d'entrée du cursus : passer du code qui fonctionne au code qui dure, déployable et observable.

Le rôle

L'Ingénieur Logiciel Cloud-Native conçoit et écrit des applications pensées pour le cloud : microservices, API, conteneurs, tests automatisés, intégration continue. Il ne se contente pas de produire des fonctionnalités : il livre du logiciel déployable, testé et observable. C'est un rôle de « coder », mais de coder en ingénieur — en laissant derrière lui des services robustes plutôt que des prototypes fragiles.

Beaucoup de développeurs entrent dans la filière par ce poste, souvent dans une fintech ou une banque qui construit ses produits de paiement. Il ne demande pas une expertise rare, mais une discipline : tout code part avec ses tests, toute API avec son contrat, tout service avec son observabilité. C'est là que se prennent les habitudes — bonnes ou mauvaises — qui détermineront la suite de la carrière.

Missions principales

MISSION	EN QUOI ELLE CONSISTE
Développer des services	Concevoir et coder des microservices et des API (REST, gRPC).
Tester	Écrire des tests unitaires et d'intégration, viser une couverture utile.
Conteneuriser & livrer	Empaqueter avec Docker, intégrer dans une chaîne CI/CD.
Observer	Instrumenter le code (logs, métriques, traces) pour le diagnostic.
Collaborer	Participer aux revues de code et aux décisions d'équipe.

Compétences à acquérir

- Un langage solide (Java, Go, Python ou Node) et ses bonnes pratiques.
- Conception d'API : REST et gRPC, versionnement, contrats clairs.
- Microservices et principes « 12-factor » du logiciel cloud-native.
- Tests automatisés : unitaires, intégration, intégration continue.
- Conteneurs : Docker, images, exécution sur orchestrateur.
- Git et observabilité applicative : logs structurés, métriques, traces.

> **LA PREUVE ATTENDUE**
>
> Livrer un microservice en production : API documentée, tests automatisés, pipeline CI/CD et observabilité. Le critère : un autre développeur peut comprendre, tester et faire évoluer le service à partir du dépôt et de sa documentation, sans solliciter son auteur.

Le mouvement

C'est le **départ** du cursus. Le professionnel capitalise un socle de génie logiciel cloud-native — conception, tests, livraison, observabilité — qui servira à chacun des jalons suivants. La qualité de ce socle conditionne la suite : un développeur qui pense déjà « contrats d'API » et « testabilité » deviendra un bien meilleur architecte que celui qui empile des fonctionnalités.

> **ANCRAGE UEMOA — WAARIPAY**
>
> Chez WaariPay, ce poste est ouvert aux développeurs qui construisent les API de paiement et les services exposés aux partenaires. C'est le vivier du cursus : la banque y repère les ingénieurs qui conçoivent proprement et les

> oriente vers la voie architecture.

Une semaine-type

ACTIVITÉ	PART DU TEMPS
Conception et écriture de code (services, API)	≈ 50 %
Tests, revues de code, intégration continue	≈ 25 %
Diagnostic, observabilité, correction d'incidents	≈ 15 %
Rituels d'équipe, documentation, apprentissage	≈ 10 %

Pièges à éviter

- Coder sans tests : accumuler une dette qui rendra le système impossible à faire évoluer.
- Négliger les contrats d'API : casser les intégrations des partenaires sans le savoir.
- Ignorer l'observabilité : livrer un service que l'on ne sait pas diagnostiquer en production.

04 JALON 2 · ARCHITECTE LOGICIEL GLOBAL (40)

Famille : F2 · Génie Logiciel **Mouvement :** Trajectoire interne (F2 → F2) **Niveau :** Concevoir **Durée :** 8 – 18 mois

Le sommet technique de la Famille 2 : passer du composant au système, et concevoir l'architecture dont dépendent toutes les équipes.

Le rôle

L'Architecte Logiciel Global conçoit l'architecture d'ensemble du système : découpage en domaines métier (bounded contexts), choix des patterns, contrats d'API, modes de communication (synchrone ou événementiel), cohérence transverse, maîtrise de la dette technique et standards de qualité. Il ne code plus seulement ses propres services : il définit les règles du jeu pour toutes les équipes, et arbitre les décisions structurantes.

C'est le passage du **coder** au **concevoir**. L'architecte raisonne à l'échelle du système et dans la durée : il pense évolutivité, couplage, robustesse, coût total. Surtout, il exerce déjà une **coordination transverse** — il aligne plusieurs équipes sur une vision commune — qui préfigure le management et rendra naturelle la passerelle suivante.

Le principe directeur : le monolithe modulaire

Un principe guide l'architecte de ce cursus : **déployer comme un, concevoir comme plusieurs**. Le monolithe modulaire est le choix par défaut recommandé — un déploiement unique, mais une conception découpée en domaines clairs et faiblement couplés. Les microservices ne sont extraits qu'en cas de besoin réel et démontré : montée en charge isolée, équipe dédiée, cycle de vie distinct. Cette discipline évite la complexité prématurée qui ruine tant de projets — distribuer un système avant d'en maîtriser les frontières.

Application : les six domaines de WaariPay

Concrètement, l'architecte de WaariPay a découpé la plateforme de paiement en six domaines métier cohérents, déployés en monolithe modulaire. Chaque domaine possède son modèle, son langage et ses frontières ; les échanges passent par des contrats explicites.

DOMAINE (BOUNDED CONTEXT)	RESPONSABILITÉ
Comptes	Identité des clients, soldes, cycle de vie des comptes.
Paiements	Initiation, exécution et statut des transactions.
Partenaires	Intégration des opérateurs et agrégateurs (mobile money, banques).
Conformité	Contrôles LCB-FT, KYC, limites réglementaires.
Notifications	Messages transactionnels et alertes aux clients.
Réconciliation	Rapprochement des flux, détection des écarts, clôtures.

Missions principales

MISSION	EN QUOI ELLE CONSISTE
Découper le système	Définir les domaines métier (bounded contexts) et leurs frontières.
Gouverner les API	Établir les contrats, le versionnement et les règles d'intégration.
Arbitrer l'architecture	Choisir patterns, communication, monolithe modulaire ou microservices.
Maîtriser la dette	Suivre la qualité, prioriser le remboursement de la dette technique.
Documenter les décisions	Tenir des ADR (architecture decision records) et des standards.
Mentorat technique	Faire monter les développeurs, diffuser les bonnes pratiques.

Compétences à acquérir

- Architecture logicielle : DDD, bounded contexts, patterns d'intégration.
- Monolithe modulaire vs microservices : savoir choisir et justifier.
- Conception d'API et communication événementielle (messaging, event-driven).
- Qualité, dette technique, performance et robustesse à l'échelle.
- Sécurité applicative et conformité intégrées dès la conception.
- Leadership technique : ADR, standards, mentorat, coordination transverse.

> **LA PREUVE ATTENDUE**
>
> Livrer une architecture de référence réellement adoptée — par exemple le découpage d'un système de paiement en domaines clairs — assortie d'ADR justifiant les choix et de standards suivis par les équipes. Le critère : les équipes construisent sur cette architecture sans la contourner, et les décisions sont tracées et défendables.

Le mouvement

Trajectoire interne **F2 → F2** (27 → 40), et point culminant de la Famille 2. Le développeur qui maîtrisait ses services conçoit désormais le système entier. Ce jalon est décisif pour la suite : c'est en architecturant que l'ingénieur acquiert la **vision systémique et la pratique de la coordination** qui rendront crédible son passage au management. On ne dirige bien que ce que l'on a su concevoir et faire converger.

> **ANCRAGE UEMOA — WAARIPAY**
>
> La plateforme de WaariPay est structurée en six domaines métier — comptes, paiements, partenaires, conformité, notifications, réconciliation. L'architecte qui a posé ce découpage en monolithe modulaire en est le référent : il connaît les frontières, les contrats et les compromis. C'est le candidat naturel à la direction de l'ingénierie.

Une semaine-type

ACTIVITÉ	PART DU TEMPS
Conception, ADR, revues d'architecture	≈ 35 %
Coordination transverse entre équipes	≈ 25 %
Code de référence, preuves de concept	≈ 20 %
Mentorat technique et diffusion des standards	≈ 20 %

Pièges à éviter

- Microservices prématurés : distribuer avant de maîtriser les frontières, et payer la complexité.
- Architecture « tour d'ivoire » : concevoir loin du code, sans validation par la réalité.
- Décisions non tracées : choisir sans ADR, et perdre la mémoire des compromis.

05 LA PASSERELLE F2 → F6 — LE FRANCHISSEMENT DÉCISIF

Le moment où l'architecte devient dirigeant. Ce qui change, ce qui se conserve, et la compétence-pont qui rend le saut possible.

Les deux premiers mouvements du cursus relèvent d'une même logique d'approfondissement technique. Le troisième — le passage de l'Architecte Logiciel (40) à l'Engineering Manager (92) — est d'une autre nature. C'est le **franchissement de famille** : de la Famille 2 (Génie Logiciel) à la Famille 6 (Management). C'est lui qui transforme un parcours d'expert en cursus de leadership, et le moment le plus délicat de toute la trajectoire.

La compétence-pont

Une passerelle ne se franchit jamais à vide : elle s'appuie sur une compétence commune aux deux familles. Ici, cette compétence-pont est **la coordination transverse que l'architecte exerce déjà, couplée à l'autorité technique qu'il a gagnée**. L'architecte aligne plusieurs équipes, arbitre des priorités, défend des choix devant des pairs et des décideurs. Le management ne lui demande pas d'oublier cela : il lui demande de l'exercer sur des personnes, un budget et un delivery plutôt que sur des composants.

Ce qui change, ce qui se conserve

DIMENSION	AVANT (ARCHITECTE LOGICIEL)	APRÈS (ENGINEERING MANAGER)
Objet du travail	Des systèmes, des contrats, des patterns	Des personnes, un budget, un delivery
Mesure du succès	Qualité et cohérence de l'architecture	Performance, prévisibilité et croissance de l'équipe
Compétence centrale	Conception technique	Leadership, en conservant la crédibilité technique
Horizon	La cohérence du système	La trajectoire des équipes et du produit

Ce qui se conserve est essentiel : la **crédibilité technique**. Un manager d'ingénierie qui ne comprendrait plus l'architecture qu'il pilote perdrait son autorité auprès de ses ingénieurs. Ce qui s'ajoute l'est tout autant : faire grandir les autres, arbitrer un budget, garantir un delivery prévisible, dialoguer avec le produit et la direction. Le saut n'est pas un abandon de la technique, c'est son élargissement.

Conditions de réussite

- Avoir réellement coordonné des équipes : la légitimité du manager vient de cette pratique.
- Accepter de ne plus concevoir soi-même : déléguer l'architecture, faire grandir les architectes.
- Acquérir vite les fondamentaux du management : 1:1, feedback, gestion de la performance.
- Conserver un lien avec la technique sans y retomber : arbitrer, pas re-concevoir à la place.

> **LE RISQUE MAJEUR**
>
> Le piège le plus fréquent de ce franchissement est le **retour au rôle d'architecte**. Sous la pression, le nouveau manager reprend les décisions de conception qu'il maîtrise au lieu de développer son équipe et son delivery. Il se rassure, mais il échoue à son nouveau rôle. Réussir la passerelle, c'est accepter d'être évalué non plus sur ses propres choix techniques, mais sur la capacité de son équipe à livrer.

Signes qu'un architecte est prêt à franchir

Tous les excellents architectes ne deviennent pas — et ne souhaitent pas devenir — managers. Avant d'engager la passerelle, quelques signaux distinguent ceux pour qui le franchissement sera naturel de ceux qu'il mettrait en difficulté.

SIGNE DE MATURITÉ	CE QU'IL RÉVÈLE
Il fait grandir les développeurs spontanément	Il tire satisfaction du succès collectif, pas seulement de ses choix.
Il pense « équipe » et « delivery »	Il raisonne déjà au-delà de la cohérence technique.
Il dialogue avec le produit et le business	Il sait traduire l'architecture en valeur et en délais.
Il accepte de déléguer la conception	Il est prêt à lâcher l'architecture pour développer son équipe.

Un programme de préparation en trois mois

Le franchissement se prépare comme un jalon à part entière. Sur trois mois, en parallèle de la fin du jalon d'architecte, le futur manager amorce sa transition sans quitter brutalement la technique.

MOIS	ACTION DE PRÉPARATION	RÉSULTAT VISÉ
Mois 1	Encadrer un ou deux développeurs, mener les premiers 1:1	Premier contact avec la posture de manager
Mois 2	Co-porter le delivery d'une équipe avec le manager en place	Compréhension concrète du pilotage de livraison
Mois 3	Prendre une responsabilité budgétaire ou de roadmap limitée	Validation de l'appétence avant la bascule complète

06 JALON 3 · ENGINEERING MANAGER (92)

Famille : F6 · Management **Mouvement :** Passerelle franchie (F2 → F6) **Niveau :** Piloter **Durée :** 18 – 28 mois

La première marche du leadership : diriger les équipes d'ingénierie, leur delivery et leur budget, en restant techniquement crédible.

Le rôle

L'Engineering Manager dirige une ou plusieurs équipes de développement. Il recrute, fait monter en compétences, garantit un delivery prévisible et de qualité, définit la roadmap technique avec le produit, gère un budget et arbitre entre nouvelles fonctionnalités et dette. Il reste techniquement crédible — il comprend l'architecture et challenge les choix — mais son temps se déplace vers les personnes, la planification et les parties prenantes. C'est le premier rôle où l'on est jugé sur un collectif.

Ce poste est le garant de la capacité de livraison de l'organisation : c'est lui qui transforme des intentions produit en logiciel livré, à un rythme soutenable et avec une qualité maîtrisée. Dans une fintech ou une banque qui scale ses équipes, l'Engineering Manager est un acteur central, en dialogue direct avec la direction et le produit.

Du contributeur au manager : ce qui change au quotidien

AU QUOTIDIEN	CONTRIBUTEUR (ARCHITECTE)	MANAGER (EM)
Emploi du temps	Conception, code, revues	Entretiens, planification, déblocage de l'équipe
Source de satisfaction	Un système bien conçu	Une équipe qui livre et progresse
Unité de mesure	Sa propre production	La production du collectif
Réflexe sous pression	Reprendre la main techniquement	Faire monter et déléguer

Missions principales

MISSION	EN QUOI ELLE CONSISTE
Management d'équipe	Recruter, mener les 1:1, évaluer la performance, faire grandir les ingénieurs.
Delivery & roadmap	Garantir un delivery prévisible, prioriser avec le produit.
Qualité & vélocité	Tenir un équilibre durable entre rythme de livraison et qualité.
Budget	Construire et tenir le budget des équipes d'ingénierie.
Parties prenantes	Aligner produit, direction et équipes sur les priorités.

Compétences à acquérir

- Leadership : donner une direction, motiver, faire grandir une équipe.
- Gestion de la performance : objectifs, feedback, entretiens individuels.
- Delivery management : prévisibilité, agilité à l'échelle, gestion du flux.
- Budgétisation : construire et défendre un budget d'équipe.
- Communication : dialoguer avec le produit et la direction en langage de valeur.
- Socle technique conservé : comprendre et challenger l'architecture.

LA PREUVE ATTENDUE

Livrer de façon prévisible et de qualité, faire grandir l'équipe (rétention, montées en compétences mesurables) et tenir un budget. Le critère : l'équipe livre mieux et se renforce, sans que le manager ait à « reprendre la main » sur

la conception ou le code.

Le mouvement

C'est la **passerelle F2 → F6 franchie** (40 → 92). Le saut de famille est accompli : de l'ingénierie au management. Il repose sur la compétence-pont « coordination transverse + autorité technique » et sur la conservation de la crédibilité technique. Ce jalon est la première marche du leadership ; il ouvre la dernière, celle de la direction du système d'information.

> **ANCRAGE UEMOA — WAARIPAY**
>
> Pour scaler ses équipes produit, WaariPay a besoin de managers d'ingénierie issus du terrain. L'ancien architecte des six domaines, devenu Engineering Manager, sait exactement ce qu'il demande à ses équipes — et où sont les compromis. La banque forme ainsi sa relève de direction.

Pièges à éviter

- Redevenir architecte : reprendre les décisions de conception au lieu de faire grandir l'équipe.
- Micro-manager : contrôler chaque détail au lieu de fixer un cap et de faire confiance.
- Négliger le delivery au profit du code : oublier que le rôle se mesure au collectif.

07 JALON 4 · DSI STRATÈGE (91)

Famille : F6 · Management **Mouvement :** Trajectoire interne (F6 → F6) **Niveau :** Diriger **Durée :** 28 mois +

Le sommet du cursus : porter la stratégie du système d'information, aligner la technologie sur les métiers et dialoguer avec le COMEX.

Le rôle

Le DSI Stratège porte la stratégie du système d'information de l'organisation. Il définit le schéma directeur, gouverne l'architecture d'entreprise, aligne la technologie sur les besoins des métiers, gère un budget global et un portefeuille de projets, garantit la sécurité et la conformité, et conduit la transformation numérique. Il dialogue avec la direction générale, les métiers et le COMEX. Là où l'Engineering Manager pilote des équipes, le DSI pilote une fonction et une trajectoire d'entreprise.

Issu du génie logiciel, le DSI de ce cursus possède un atout décisif : il connaît la chaîne de valeur applicative de bout en bout. Là où d'autres dirigeants raisonnent en abstractions, lui sait ce que coûte une dette technique, ce qu'exige une montée en charge, ce que vaut une équipe qui livre. Dans une banque soumise à la régulation BCEAO, cette maîtrise concrète est un avantage stratégique pour arbitrer build-vs-buy et conduire la transformation.

Missions principales

MISSION	EN QUOI ELLE CONSISTE
Stratégie & schéma directeur	Définir la trajectoire SI alignée sur la stratégie d'entreprise.
Architecture d'entreprise	Gouverner la cohérence du SI et les grands choix structurants.
Alignement métier	Traduire les besoins des métiers en capacités du SI.
Sécurité & conformité	Répondre des exigences réglementaires (BCEAO) au plus haut niveau.
Budget & portefeuille	Allouer les ressources, arbitrer le portefeuille de projets.
Relation COMEX & conduite du changement	Porter la transformation numérique auprès de la direction.

Compétences à acquérir

- Vision stratégique : relier système d'information et objectifs d'entreprise.
- Architecture d'entreprise : gouverner la cohérence du SI dans la durée.
- Alignement métier & conduite du changement : embarquer les directions.
- Finance : construire et défendre un budget global, raisonner en investissement.
- Gouvernance & conformité : maîtriser les cadres réglementaires de la zone.
- Communication COMEX : convaincre la direction générale, porter le risque.

> **LA PREUVE ATTENDUE**
>
> Définir et exécuter un schéma directeur du système d'information aligné sur la stratégie de l'organisation, avec une gouvernance d'architecture et des indicateurs présentés au COMEX. Le critère : le SI est devenu un levier stratégique reconnu, porté et arbitré au niveau de la direction générale.

DSI Stratège ou CTO ?

Le **DSI Stratège** et le CTO sont deux aboutissements voisins mais distincts. Le DSI oriente l'ensemble du système d'information de l'entreprise — applications, données, processus, alignement métier, conformité ; son horizon est la transformation de l'organisation. Le CTO, aboutissement du cursus voisin « Du socle au sommet », porte davantage la

technologie produit et l'infrastructure. Issu du génie logiciel et de l'architecture applicative, le profil de ce cursus se dirige naturellement vers la DSI, au carrefour de la technologie et des métiers.

Les chantiers d'un DSI dans la zone

CHANTIER	ENJEU RÉGIONAL
Transformation numérique	Digitaliser les parcours clients et internes face à des concurrents agiles.
Conformité & reporting BCEAO	Industrialiser la production réglementaire et la maîtrise du risque.
Souveraineté des données	Garder la maîtrise des données sensibles et des dépendances technologiques.
Maîtrise des coûts SI	Optimiser un budget contraint tout en soutenant la croissance.
Talents & rétention	Attirer et retenir des compétences rares sur un marché tendu.

Le mouvement

Trajectoire interne **F6 → F6** (92 → 91). Du management d'équipes à la direction de la fonction SI entière. L'Engineering Manager élargit son périmètre : il ne pilote plus des équipes mais une stratégie, et passe d'un horizon de delivery à un horizon de transformation. C'est l'aboutissement du cursus — qui ouvre lui-même de nouveaux futurs : direction générale, conseil, entrepreneuriat.

> **ANCRAGE UEMOA — WAARIPAY**
>
> Le DSI de WaariPay, issu du code puis de l'architecture des six domaines, apporte au COMEX une crédibilité rare sur la maîtrise applicative, les délais réalistes et la conformité. La banque a formé en interne un dirigeant qui parle aussi bien aux développeurs qu'aux métiers.

Pièges à éviter

- Se couper du delivery : perdre la connaissance concrète qui faisait sa force.
- Sous-estimer la conduite du changement : imposer le SI sans embarquer les métiers.
- Négliger la conformité régionale : traiter la régulation comme une contrainte secondaire.

08 PLAN DE MISE EN ŒUVRE SUR 30 MOIS

Un jalon à la fois, une preuve à chaque étape : la cadence qui sécurise la progression du développeur au dirigeant.

Le cursus complet — jusqu'au poste de DSI — se déploie sur plusieurs années. Les trente premiers mois, décrits ici, mènent du développeur à l'Engineering Manager confirmé ; l'accession à la direction du système d'information relève d'un horizon plus long, fonction des opportunités d'organisation. La règle est constante : chaque jalon mobilise plusieurs mois, le temps de produire une preuve déployable avant d'ouvrir le suivant.

PHASE	MOIS	OBJECTIF	PREUVE ATTENDUE
Jalon 1	M1 – M8	Maîtrise du génie logiciel cloud-native	Microservice en production : API, tests, CI/CD, observabilité
Jalon 2	M9 – M18	Conception de l'architecture d'ensemble	Architecture de référence adoptée + ADR + standards
Passerelle	M15 – M20	Acquérir les fondamentaux du management	Coordination d'équipe, premiers 1:1, lead technique
Jalon 3	M19 – M28	Diriger les équipes d'ingénierie	Delivery prévisible + équipe renforcée + budget tenu
Au-delà	M28 +	Accéder à la direction du SI (DSI)	Schéma directeur SI porté au COMEX

La cadence recommandée

Le rythme prime sur la vitesse. Mener deux jalons de front dilue l'effort et fragilise la preuve. Un jalon tenu, validé par un livrable réel, vaut mieux que plusieurs jalons entamés et abandonnés. La phase de passerelle chevauche volontairement les jalons 2 et 3 : on commence à coordonner et à exercer des responsabilités d'équipe avant d'en faire son métier principal, pour éprouver l'appétence au management.

Le détail mois par mois

PÉRIODE	FOCUS	LIVRABLE DE LA PÉRIODE
M1 – M4	Fondamentaux : langage, tests, conteneurs, CI/CD	Premier service conteneurisé, testé et intégré
M5 – M8	API, microservices, observabilité	Microservice en production, API documentée
M9 – M12	DDD, découpage en domaines, contrats	Première architecture de domaine + ADR
M13 – M18	Architecture globale, dette, standards	Architecture de référence adoptée par les équipes
M15 – M20	Coordination, lead technique, 1:1	Prise de responsabilité partielle sur une équipe
M21 – M24	Management : performance, delivery	Delivery prévisible sur une équipe
M25 – M28	Budget, parties prenantes, scale	Deux équipes pilotées, budget tenu, promotions

GOUVERNANCE DU PARCOURS

Chaque jalon est encadré par un **sponsor** (un dirigeant qui porte le cursus) et un **mentor** (un pair plus avancé). La preuve de fin de jalon est revue formellement avant d'ouvrir le suivant. Cette gouvernance légère transforme une intention de carrière en trajectoire pilotée.

09 ÉTUDE DE CAS — LA TRAJECTOIRE DE MOUSSA

Comment WaariPay forme en interne, en trente mois, le futur dirigeant de son système d'information — sans recrutement externe.

Le point de départ

Moussa est développeur chez WaariPay : cinq ans d'expérience, une réelle aisance sur les API de paiement, mais l'envie d'avoir un impact plus large. Il documente ses services, soigne ses contrats d'API, et propose spontanément des améliorations d'architecture. La DSI, qui veut maîtriser sa chaîne applicative et préparer sa relève, identifie en lui un potentiel et active le cursus « Du code à la direction ».

La trajectoire, trimestre par trimestre

TRIMESTRE	JALON	PREUVE PRODUITE
T1 – T2	Ingénieur Logiciel Cloud-Native (27)	Microservice de paiement en production : API documentée, tests, CI/CD, observabilité.
T3 – T6	Architecte Logiciel Global (40)	Découpage de la plateforme en six domaines (monolithe modulaire), ADR à l'appui.
T7	Passerelle (amorce)	Coordination de deux équipes, lead technique, premiers entretiens individuels.
T8 – T10	Engineering Manager (92)	Delivery prévisible sur deux équipes, budget tenu, deux développeurs promus.

Le bilan

En moins de trente mois, Moussa est passé d'un rôle de développeur au management des équipes d'ingénierie de la banque. Chaque étape s'est validée par une preuve déployable, ce qui a permis à la DSI de sécuriser sa progression sans pari. Le poste de DSI reste un horizon — quelques années encore — mais la trajectoire est tracée et la relève de direction de WaariPay est désormais interne.

Les trois décisions-clés de la DSI

La réussite de Moussa ne tient pas au hasard mais à trois décisions de la DSI. **D'abord, choisir le bon profil** : non le plus brillant techniquement, mais celui qui pensait déjà « système » et « équipe ». **Ensuite, garantir le temps et l'accompagnement** : un mentor architecte puis un mentor manager, et un temps protégé à chaque jalon. **Enfin, exiger la preuve** : aucune étape validée sans livrable déployable, ce qui a permis d'avancer sans pari.

Le calcul économique

	RECRUTEMENT EXTERNE	CURSUS INTERNE (MOUSSA)
Délai	6 – 12 mois de recherche, profil rare	Progression planifiée, disponibilité immédiate
Coût	Prime de rareté, package élevé, risque d'échec	Accompagnement (mentors, temps, formation) bien moindre
Risque	Intégration incertaine, départ possible	Profil connu, fidélisé, ancré dans l'organisation
Connaissance métier	À reconstruire	Maîtrise native de la plateforme et de ses six domaines

CE QUE L'EXEMPLE DÉMONTRE

Le cursus n'est pas un plan de formation théorique : c'est un instrument de pilotage. Il transforme un besoin de poste critique en trajectoire individuelle prouvée, jalon après jalon — au bénéfice conjoint du collaborateur et de

RECRUTEMENT EXTERNE	CURSUS INTERNE (MOUSSA)
l'organisation.	

10 ACTIVER CE CURSUS — LECTURE RH & DSI

De l'intention au pilotage : comment une direction RH ou DSI met concrètement ce cursus en mouvement.

Quatre usages pour une direction

USAGE	CE QUE LE CURSUS PERMET
Mobilité interne	Pourvoir un poste rare (manager d'ingénierie, futur DSI) sans recrutement externe.
Plan de succession	Préparer la relève de direction en traçant le chemin depuis le développement.
Fidélisation	Offrir aux meilleurs développeurs une perspective lisible jusqu'au leadership.
Maîtrise applicative	Garder en interne la connaissance de la chaîne de valeur logicielle.

Une feuille de route en 90 jours

À **J+30**, repérer les développeurs à fort potentiel (ceux qui pensent système et contrats, pas seulement fonctionnalités) et le poste-cible critique (manager d'ingénierie, relève DSI). À **J+60**, valider l'appétence des candidats au parcours et désigner sponsor et mentors (architecte, puis manager). À **J+90**, lancer le jalon 1 avec une preuve attendue clairement définie et une revue de fin de jalon planifiée.

Facteurs de réussite & risques

FACTEUR DE RÉUSSITE	RISQUE SYMÉTRIQUE À ÉVITER
Sponsor et mentors désignés pour le cursus	Lancer le parcours sans accompagnement ni temps dédié
Preuve déployable validée à chaque jalon	Valider la progression sur la seule bonne volonté
Soin particulier au franchissement F2 → F6	Promouvoir manager sans préparer au management
Rythme soutenable, un jalon à la fois	Accélérer et fragiliser les preuves

RECOMMANDATION

Documenter ce cursus pour votre organisation, identifier un à trois candidats par vivier de développeurs et en activer le parcours dès cette année. Sur un marché régional où les dirigeants issus du produit sont rares et chers, la trajectoire interne planifiée est l'option la plus rapide, la moins coûteuse et la moins risquée.

11 CONCLUSION

Une voie directe vers la direction du SI, à condition d'en respecter la discipline : la preuve à chaque jalon, et un franchissement préparé.

« Du code à la direction » est l'un des cursus les plus naturels vers la direction du système d'information. Il a pour lui la cohérence : deux trajectoires internes qui bâtissent une expertise de conception, une passerelle unique qui la transforme en leadership. Il a contre lui une exigence : ce franchissement ne s'improvise pas. Promouvoir un excellent architecte au management sans l'y préparer, c'est risquer de perdre un excellent architecte **et** de rater un manager.

Pour le professionnel, ce cursus offre une perspective rare : commencer comme développeur et viser, en quelques années, la direction du système d'information d'une organisation. Pour l'employeur, il offre un instrument de maîtrise et de succession : former en interne le dirigeant qui connaît la chaîne applicative, plutôt que de dépendre d'un marché tendu. Pour la zone UEMOA, il répond à un besoin structurel : faire émerger des dirigeants issus du logiciel, capables d'aligner la technologie sur les métiers.

Trois principes à retenir

- La preuve, pas la promesse : chaque jalon se valide par un livrable déployable.
- Le franchissement se prépare : la passerelle F2 → F6 est le vrai sujet du cursus.
- La crédibilité technique se conserve : on dirige d'autant mieux que l'on a conçu et coordonné.

Pour aller plus loin

Ce cursus dialogue avec ses voisins de l'axe « Direction technique ». **« Du socle au sommet »** (cursus 1) mène à la direction par l'infrastructure ; **« Du test à l'architecture »** (cursus 4) construit l'expertise logicielle par la qualité. Une organisation peut activer plusieurs de ces voies en parallèle, selon ses viviers et ses postes critiques.

> **ENGAGEMENT**
>
> Du développeur au dirigeant, un chemin lisible et pilotable, prouvé à chaque étape. C'est ainsi que le génie logiciel devient une voie royale du leadership — et que les organisations de la zone forment, en interne, les DSI dont elles ont besoin.

A ANNEXE A — RÉCAPITULATIF DES QUATRE JALONS

JALON	RÔLE (N°)	FAMILLE	MOUVEMENT	DURÉE
1	Ingénieur Logiciel Cloud-Native (27)	F2	Départ	0 – 8 mois
2	Architecte Logiciel Global (40)	F2	Interne	8 – 18 mois
3	Engineering Manager (92)	F6	Passerelle F2 → F6	18 – 28 mois
4	DSI Stratège (91)	F6	Interne	28 mois +

Enchaînement complet : 27 → 40 → 92 → 91 • **Familles :** F2 → F2 → F6 → F6 • **Aboutissement :** DSI Stratège (91).

B ANNEXE B — COMPÉTENCES & CERTIFICATIONS PAR JALON

JALON	COMPÉTENCES CLÉS	CERTIFICATION / PREUVE CIBLE
Ingénieur Logiciel (27)	Langage, API (REST/gRPC), microservices, tests, Docker, CI/CD	Service en production, certifs cloud dev
Architecte Logiciel (40)	DDD, monolithe modulaire, patterns, API, dette, ADR	Architecture adoptée, certifs architecture
Engineering Manager (92)	Leadership, performance, delivery, budget, communication	Delivery prévisible, formation management
DSI Stratège (91)	Stratégie SI, architecture d'entreprise, finance, gouvernance	Schéma directeur, programmes exécutifs

NOTE SUR LES CERTIFICATIONS

Les certifications matérialisent une compétence mais ne la remplacent pas. Aux niveaux développeur et architecte, la preuve réelle est le système livré et adopté. Aux niveaux management et direction, elle se déplace vers le résultat démontré : équipe qui livre, schéma directeur exécuté.

C ANNEXE C — GLOSSAIRE & REPÈRES

TERME	DÉFINITION
API	Interface applicative permettant à des systèmes de communiquer (REST, gRPC).
Microservice	Service autonome, déployable indépendamment, responsable d'une capacité métier.
DDD	Domain-Driven Design : conception guidée par les domaines métier (bounded contexts).
Monolithe modulaire	Déploiement unique mais conception découpée en domaines faiblement couplés.
Bounded context	Frontière d'un domaine métier cohérent, avec son modèle et son langage propres.
ADR	Architecture Decision Record : trace écrite et justifiée d'une décision d'architecture.
CI/CD	Intégration et livraison continues : automatisation du build, des tests et du déploiement.
Delivery management	Discipline de livraison prévisible et de qualité par les équipes.
Passerelle	Franchissement d'une famille de métiers à une autre, porté par une compétence-pont.
Compétence-pont	Compétence commune à deux familles, qui rend possible le passage de l'une à l'autre.

D ANNEXE D — VARIANTES, BIFURCATIONS & CURSUS VOISINS

« Du code à la direction » n'est pas l'unique destin d'un ingénieur logiciel. Plusieurs bifurcations s'ouvrent en chemin, et s'arrêter en route reste une carrière pleine et entière.

Ce cursus décrit une voie directe, mais le graphe des compétences en autorise d'autres. L'**Architecte Logiciel (40)** est en particulier un **rôle-charnière** : depuis ce point, plusieurs trajectoires divergent vers d'autres familles ou d'autres formes de leadership. Connaître ces bifurcations aide une direction à orienter chaque talent vers la voie qui lui convient.

Trois bifurcations fréquentes

BIFURCATION	POINT DE DÉPART	VERS	CURSUS VOISIN
Vers le produit & le paiement	Ingénieur API (proche de 27)	Intégration paiement, architecture mobile money, Product Manager	B4 · Du développeur au paiement
Vers la sécurité applicative	CI/CD (proche de 27)	DevSecOps, AppSec, Zero Trust	C3 · Du CI/CD au DevSecOps
Vers la direction technique produit	Architecte (40)	Manager Platform, CTO	A1 · Du socle au sommet

S'arrêter en chemin

Tous les professionnels n'iront pas jusqu'au sommet, et c'est légitime. S'arrêter au jalon 2 — Architecte Logiciel Global — constitue une carrière technique complète, respectée et bien rémunérée, sans jamais franchir vers le management. La profondeur d'expertise vaut le leadership : une organisation a besoin d'architectes seniors autant que de dirigeants. Le cursus n'impose pas le sommet ; il rend la voie lisible pour ceux qui le visent.

> **LECTURE**
> Présenter ces bifurcations dès l'entrée du cursus est un atout RH : le collaborateur comprend qu'il garde la main sur sa trajectoire, ce qui renforce l'engagement et réduit le risque de départ.

E ANNEXE E — INDICATEURS DE PROGRESSION PAR JALON

Comment objectiver l'avancement : pour chaque jalon, des indicateurs de maîtrise et leur cible, à revoir avant d'ouvrir le suivant.

JALON	INDICATEUR DE MAÎTRISE	CIBLE DE FIN DE JALON
1 · Ing. Logiciel (27)	Qualité du service livré	Service en prod : tests, CI/CD, observabilité, API documentée
1 · Ing. Logiciel (27)	Autonomie	Évolution d'un service sans assistance de son auteur initial
2 · Architecte (40)	Adoption de l'architecture	Équipes construisant sur l'architecture, sans la contourner
2 · Architecte (40)	Traçabilité des décisions	Décisions structurantes couvertes par des ADR
3 · Eng. Manager (92)	Prévisibilité du delivery	Engagements de livraison tenus, qualité maîtrisée
3 · Eng. Manager (92)	Santé de l'équipe	Rétention et montées en compétences mesurables
4 · DSI Stratège (91)	Alignement stratégique	Schéma directeur SI adopté au COMEX
4 · DSI Stratège (91)	Conduite du changement	Métiers embarqués, transformation engagée

PRINCIPE

Un indicateur n'a de valeur que s'il déclenche une décision : ouvrir le jalon suivant, prolonger le jalon en cours, ou réorienter vers une bifurcation. La revue de fin de jalon est le moment où cette décision se prend, sponsor et mentor réunis.

F ANNEXE F — RESSOURCES DE MONTÉE EN COMPÉTENCES

Pour chaque jalon, les modalités d'apprentissage les plus efficaces : la pratique d'abord, la preuve ensuite, le mentorat tout du long.

JALON	MODALITÉS DE MONTÉE EN COMPÉTENCES
Ingénieur Logiciel Cloud-Native (27)	Projets réels avec tests et CI/CD, revues de code par des pairs seniors, katas et bonnes pratiques ; mentorat d'un ingénieur confirmé.
Architecte Logiciel Global (40)	Conception d'architectures de bout en bout, étude de systèmes de référence, lecture de DDD et patterns, rédaction d'ADR ; mentorat d'un architecte.
Engineering Manager (92)	Formation au management (1:1, feedback, performance), accompagnement par un manager expérimenté, lectures de leadership ; coaching.
DSI Stratège (91)	Programmes exécutifs (stratégie, finance, gouvernance), mentorat d'un DSI confirmé, exposition COMEX et métiers progressive.

LA RÈGLE CONSTANTE

À tous les jalons, l'ordre est le même : **pratiquer, prouver, puis formaliser**. La formation accompagne une compétence construite par la pratique ; elle ne la remplace pas. Et à mesure que l'on monte vers le management et la direction, la preuve se déplace du livrable technique vers le résultat d'organisation.

G ANNEXE G — ANTI-MODÈLES À ÉVITER DANS CE CURSUS

Les erreurs récurrentes qui font dérailler le parcours — et l'antidote correspondant, à garder en tête à chaque jalon.

ANTI-MODÈLE	POURQUOI IL NUIT	ANTIDOTE
Le héros indispensable	Un ingénieur que personne ne peut remplacer fragilise l'équipe et se bloque lui-même.	Documenter, partager, faire monter les autres dès le jalon 1.
Les microservices prématurés	Distribuer avant de maîtriser les frontières ajoute une complexité coûteuse.	Monolithe modulaire par défaut, extraction sur besoin réel.
L'architecte hors-sol	Concevoir loin du code produit des architectures que la réalité dément.	Rester au contact du code et valider par l'usage.
Le manager-pompier	Reprendre les tâches techniques sous pression fait échouer le rôle de manager.	Déléguer, faire grandir, accepter d'être jugé sur le collectif.
Le dirigeant déconnecté	Un DSI coupé du delivery perd la crédibilité qui faisait sa force.	Conserver un lien régulier avec le terrain et les équipes.
La promotion sans préparation	Promouvoir au mérite technique sans former au management mène à l'échec.	Préparer explicitement le franchissement F2 → F6.

À RETENIR

La plupart des échecs de ce cursus ne sont pas techniques : ils tiennent à un franchissement mal préparé ou à un réflexe d'expert conservé trop longtemps. Nommer ces anti-modèles, c'est déjà commencer à les éviter.

H ANNEXE H — LES DEUX VOIES VERS LA DIRECTION : DSI OU CTO

Deux aboutissements voisins de l'axe « Direction technique » : comprendre leur différence aide à orienter chaque cursus vers le bon sommet.

L'axe « Direction technique » offre deux sommets distincts mais proches : le **DSI Stratège**, aboutissement de ce cursus, et le **CTO**, aboutissement du cursus « Du socle au sommet ». Tous deux dirigent la technologie, mais avec un centre de gravité différent. Le choix dépend du profil d'origine, de la culture de l'organisation et de la nature de son activité.

DIMENSION	DSI STRATÈGE (CE CURSUS)	CTO (CURSUS VOISIN A1)
Origine typique	Génie logiciel, architecture applicative	Infrastructure, plateforme Cloud Native
Centre de gravité	Système d'information, alignement métier	Technologie produit, infrastructure
Périmètre	Applications, données, processus, conformité	Plateforme, fiabilité, technologie cœur
Interlocuteurs	Métiers, COMEX, conformité	Produit, ingénierie, sécurité
Question centrale	Le SI sert-il la stratégie de l'entreprise ?	La technologie est-elle un avantage compétitif ?

Les deux rôles ne s'excluent pas : dans les grandes organisations, ils coexistent et se complètent — le DSI orientant le système d'information de l'entreprise, le CTO la technologie produit. Dans une structure plus petite, une même personne peut porter les deux casquettes. Ce qui compte, pour une direction RH, est d'orienter chaque talent vers la voie qui épouse son profil : le constructeur d'applications vers la DSI, le bâtisseur de plateformes vers la fonction de CTO.

> **LECTURE**
> Présenter ces deux voies côte à côte clarifie les plans de succession : on ne prépare pas un DSI comme un CTO, et savoir lequel viser dès l'entrée du cursus évite des trajectoires mal ajustées.

I ANNEXE I — QUESTIONS FRÉQUENTES

Les interrogations les plus courantes des directions et des candidats au moment d'engager ce cursus.

Faut-il être un développeur d'exception pour réussir ?

Non. Le meilleur prédicteur de réussite n'est pas la virtuosité technique pure, mais la rigueur d'ingénieur associée à une capacité à penser système et équipe. Un très bon développeur replié sur son code fera souvent un moins bon architecte et un moins bon manager qu'un ingénieur solide, ouvert et coopératif.

Combien de temps faut-il réellement ?

Environ trente mois pour atteindre le poste d'Engineering Manager confirmé. L'accession à la direction du système d'information relève d'un horizon plus long — plusieurs années supplémentaires — qui dépend des opportunités d'organisation autant que du candidat. Le cursus trace le chemin ; il ne garantit pas le calendrier.

Peut-on sauter le jalon d'architecte ?

C'est fortement déconseillé. L'architecture apporte deux acquis indispensables au management : la vision systémique et la pratique concrète de la coordination transverse. Promouvoir un développeur directement manager, sans ce passage, le prive de la légitimité technique et de l'habitude de l'arbitrage qui font les bons Engineering Managers.

Et si le candidat ne veut pas devenir manager ?

C'est un choix légitime, et fréquent. S'arrêter au jalon d'architecte constitue une carrière complète et valorisée. Le rôle de la direction est alors d'ouvrir une bifurcation technique — vers le produit, la sécurité ou la direction technique produit (voir l'annexe D) — plutôt que de forcer un franchissement non désiré, qui se solderait par un échec.

Comment éviter de perdre un bon architecte en le promouvant mal ?

En préparant le franchissement comme un jalon à part entière (section 05) : programme de transition sur trois mois, mentorat d'un manager expérimenté, et droit au retour arrière si l'appétence n'est pas au rendez-vous. Un franchissement réversible et accompagné protège à la fois le talent et l'organisation.

Ce cursus convient-il aux petites structures ?

Oui, en adaptant l'échelle. Dans une petite fintech, une même personne peut cumuler les rôles d'architecte et d'Engineering Manager avant d'évoluer vers la direction du système d'information. La logique des jalons et des preuves reste la même ; seules la taille des équipes et la séparation des rôles varient.

> **EN RÉSUMÉ**
>
> Ce cursus n'est ni automatique ni universel : c'est un chemin lisible, à activer avec discernement, pour les profils qui en ont l'appétence et les organisations qui en ont le besoin.

*Ce document constitue l'approfondissement du **Cursus 2 · Du code à la direction**, deuxième des treize cursus de la collection « Les cursus intra & inter familles » de la Méthode SAKITI.*