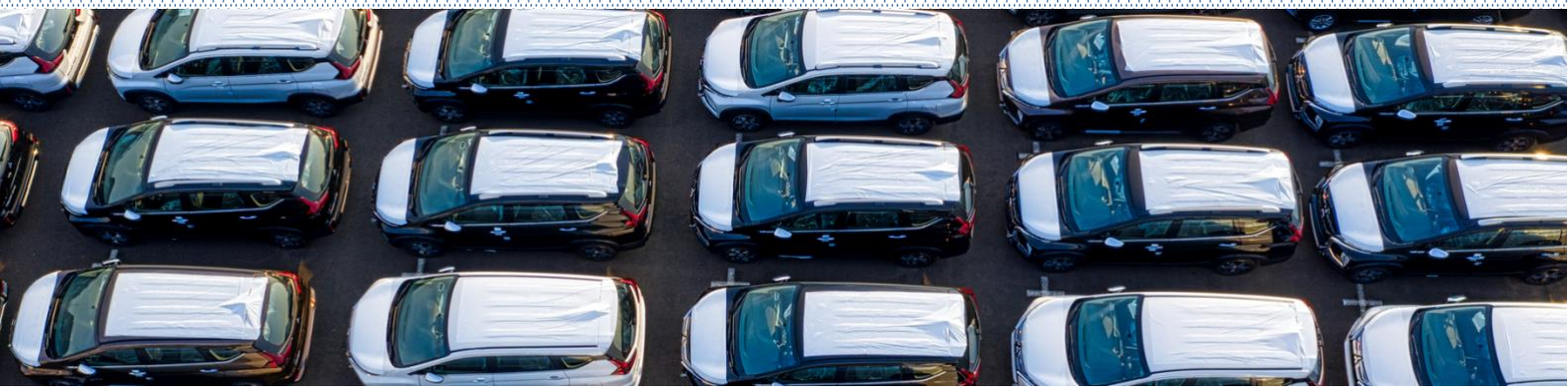
LIVRE BLANC

Le rôle de l'architecte

Les missions de l'architecte



Le Cercle CESAM

Juillet 2024

V1.0

CESAM COMMUNITY

SOMMAIRE

INTRODUCTION

Dans un contexte de globalisation et de transformation numérique rapide, les entreprises sont confrontées à des systèmes de plus en plus complexes nécessitant une coordination et une intégration méticuleuses. Le rôle de l'architecte s'avère alors indispensable pour garantir la cohérence et l'efficacité de ces systèmes. Que ce soit dans des secteurs variés tels que l'aéronautique, l'automobile, l'énergie ou le ferroviaire, l'architecte doit maîtriser les interfaces et assurer la convergence des différents acteurs impliqués.

Le rôle d'un architecte ne se limite pas à une seule personne mais peut être porté par une équipe d'architecture composée de divers spécialistes, notamment des architectes de sous-systèmes, des gestionnaires d'interfaces, des responsables des exigences et des modeleurs. Cette équipe collabore pour assurer la conformité technique et la satisfaction des besoins des clients tout au long du cycle de vie du système, de sa conception à sa maintenance en passant par son déploiement.

Ce livre blanc vise à détailler les missions et les responsabilités de l'architecte en soulignant l'importance de la collaboration et de la coordination dans la réussite des projets complexes. Il explore les défis auxquels les architectes sont confrontés et les compétences nécessaires pour surmonter ces obstacles, tout en mettant en lumière l'impact crucial de l'architecte sur la performance et la compétitivité des entreprises modernes. Il est le reflet des expériences partagées des membres du Cercle CESAM.

LES MISSIONS DE L'ARCHITECTE

Partout où les problématiques sont complexes et interconnectées et où leur résolution demande une maîtrise des interfaces et la convergence des acteurs, l'architecte a un rôle à jouer. Voyons à présent quelles sont les missions que doit porter l'architecte.

On parle souvent de l'architecte, laissant entendre qu'il s'agit d'un homme ou d'une femme seule alors qu'il fait en général plutôt référence à un rôle qui peut (et parfois doit) être porté par une équipe alors nommée équipe d'architecture. On pourra par exemple trouver au sein d'une équipe d'architecture :

- Les architectes des différents sous-systèmes et composants
- Gestionnaire des interfaces
- Requirement manager
- Modeleur

Attention à ne pas confondre rôle et fiche de poste, en particulier sur les phases de vie aval à la conception. Les activités de l'architecte doivent être poursuivies sur l'ensemble du cycle de vie, même si ce n'est pas par une personne nommée architecte ou une équipe d'architecture.

1. GERER LE CYCLE DE VIE DE L'ARCHITECTURE

Les activités d'un architecte système sont définies – et ce de manière similaire – dans les standards industriels et les référentiels professionnels qui décrivent les processus de développement de systèmes complexes, comme la norme ARP 4754 dans l'aéronautique, la norme ISO 26262 dans l'automobile, la norme IEC 61513 dans l'énergie, la norme EN 50126 dans le ferroviaire ou le référentiel d'ingénierie système de l'International Council on Systems Engineering (INCOSE).

De manière générale, ces normes font un focus sur la gestion par l'architecte (ou l'équipe d'architecture) des étapes du cycle de vie d'un système complexe, depuis la conception jusqu'à la maintenance, en passant par le déploiement.

2. ARCHITECTURE BOITE NOIRE

2.1. CAPTURER LES BESOINS CLIENTS INTERNES / EXTERNES ET LES CONSOLIDER

Capturer les besoins clients internes et externes, et les consolider est une activité qui se décompose en 3 étapes :

1. Identifier et organiser les parties prenantes du système sous la forme de diagramme(s) hiérarchique(s) et d'un diagramme d'environnement qui précise les interfaces/interactions de ces parties prenantes avec le système considéré ;
2. Cadrer la demande des parties prenantes du système en capturant l'objectif qui motive le développement du système donné, et décrivant de manière succincte le périmètre opérationnel, fonctionnel et technique du système cible et des principaux indicateurs de performance à atteindre ;
3. Capturer les besoins des parties prenantes du système en commençant par la capture des besoins « bruts » des parties prenantes, tels que ces dernières les expriment, à l'aide des sources d'information disponibles (interviews, documents, etc.), puis en formalisant ces besoins bruts à l'aide d'un cadre standard d'énoncé de besoin et en hiérarchisant enfin les besoins formalisés.

L'essentiel

L'architecte **identifie le périmètre** externe à prendre en compte et **clarifie les missions** du système ainsi que les **objectifs à atteindre**. Il formalise et hiérarchise tous les **besoins des parties prenantes** en garantissant leur **non-ambiguïté** et leur **exhaustivité**.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Que des problématiques d'organisation peuvent surgir lorsque la capture des besoins menée par l'architecte entre en conflit avec les activités business (BU ou Business Analyst) ;
- Vouloir aller trop loin dans la modélisation de l'environnement. Pour éviter cela, il faut sans cesse raisonner en termes de valeur et donc voir ce qui a un impact substantiel sur le système d'intérêt. Connaître les processus métiers de l'entreprise permet de mieux évaluer la valeur ;
- Omettre de consulter les parties prenantes de certaines phases de vie (développement, maintenance, etc.) ;
- Les acteurs principaux ne sont pas alignés sur le périmètre défini du système (cela arrive souvent car on ne prend pas le temps de faire cet effort).

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Ne pas laisser l'aspect organisationnel devenir le moteur principal dans la définition du périmètre. Cela permet d'éviter de biaiser les choix d'architecture dès cette étape ;
- Faire comprendre la valeur : l'architecte apporte un aspect analyse de risque et valeur, là où les business analysts sont dans une optique de couverture et de consistance. On peut procéder soit par un partage des peines (analyse des irritants, des problèmes, de tout ce qui empêche d'atteindre les objectifs) soit par une démonstration de l'apport d'une méthodologie ;
- S'assurer du bon niveau de connaissance des acteurs impliqués dans la validation des cas d'utilisation et du bon niveau de responsabilité (au sens décisionnel) vis- à- vis des impacts potentiels. Ceci est vrai pour l'ensemble des activités d'architecture ;
- Ne pas freiner les gens dans l'expression du besoin : s'ils expriment plutôt des solutions il faut les noter et remonter au besoin (par la méthode des 5 pourquoi par exemple) ;
- Il faut veiller à intégrer les contraintes/attentes des équipes safety, cyber sécurité...

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase de captation des besoins :

> “ Nous avons fait challenger les livrables existants par les équipes. Cela a naturellement fait émerger le besoin de mettre en place une nouvelle méthodologie sur la capture des besoins, ce que nous avons fait en accompagnant les collaborateurs via des formations.

> “ Nous faisons des boucles très courtes entre boîte noire et boîte blanche afin de mettre en évidence les implications des besoins sur les contraintes induites dans l'architecture.

> “ Nous préparons les deux visions boîtes noires et boîtes blanches, et mettons les éléments qui sont plutôt de l'ordre de la solution possible dans la partie boîte blanche avant de remonter au besoin.

2.2. ANALYSER LES BESOINS DU CLIENT ET LES DECLINER EN EXIGENCES

Analyser les besoins du client et les décliner en exigences se décompose en 2 étapes principales :

1. Prioriser les besoins des parties prenantes du système en définissant au préalable, en étroite interaction avec les sponsors du projet de développement du système concerné, des critères explicites de valeur et de risque, pour en faire émerger les plus prioritaires qu'il convient de prendre en compte impérativement,
2. Décliner les besoins en exigences du système en formalisant d'abord les exigences fonctionnelles et techniques du système en respectant un cadre standard d'énoncé d'exigences, puis en les organisant sous la forme de hiérarchies d'exigences fonctionnelles et d'exigences techniques en garantissant leur traçabilité avec les besoins.

L'essentiel

L'architecte définit une liste priorisée de besoins afin de maximiser le rapport valeur / risque du système. C'est la base d'une déclinaison hiérarchisée des besoins en exigences fonctionnelles et techniques, qui s'accompagne d'un maintien de la traçabilité à la fois interne et externe à partir des besoins.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Fonctionner en mode tout ou rien sans priorisation de valeur et sans aucun recul,
- Vouloir aller trop loin dans l'écriture et le découpage des exigences. L'architecture système est alors perçue comme un processus long, laborieux et sans valeur ajoutée,
- Couvrir le champ des parties prenantes sans aller les consulter,
- L'absence de décideur qui porte la responsabilité de décider de la valeur. Ce n'est pas le rôle de l'architecte système de le faire seul.

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Former les équipes et communiquer à chaque début de projet sur le caractère pragmatique du déploiement des méthodes d'architecture et la recherche de valeur.
- Inclure du « tayloring » assisté dans le processus d'architecture en permettant aux projets d'adapter de manière guidée la démarche à leurs contextes. Il est intéressant par exemple de distinguer le travail d'architecture type « on part d'une feuille blanche » de celui qui vise plutôt à modifier un existant. Ce tayloring doit s'accompagner de la mise en place d'un ensemble de critères (ex : re-use, innovation, niveau de complexité, impact sur l'architecture existante, maturité des équipes, des clients, de l'organisation…)
- Avoir un pool d'architectes expérimentés (à la fois sur les aspects méthodes et métier) qui supportent les équipes. Cela nécessite la mise en place de parcours construits, basés sur une évaluation du profil et des appétences et qui crante progressivement la connaissance métier (ex : stage d'immersion dans les métiers). Il faut donc bien identifier quels sont les métiers les plus critiques. Il faut surtout une politique RH forte mise en œuvre.

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase de captation des besoins :

- “ Nous avons mis en place des contraintes de temps pour terminer l'architecture boîte noire ce qui a naturellement forcé les équipes à adapter le niveau de grain des livrables.
- “ Prendre le temps de communiquer sur des « success stories » en choisissant des exemples qui parlent à tout le monde a été un vrai plus.
- “ Nous avons organisé des comités de décision pour cranter l'analyse risque/valeur. Nous avons fait challenger les livrables existants par les équipes. Cela a naturellement fait émerger le besoin de mettre en place une nouvelle méthodologie sur la capture des besoins, ce que nous avons fait en accompagnant les collaborateurs via des formations.

2.3. DEFINIR LES USAGES

Cette activité vise à définir les usages du système. Ceci peut se faire en 2 étapes :

1. On synthétise d'abord toutes les phases de vie du système complexe dans un diagramme de cycle de vie,

2. On identifie ensuite les cas d'utilisation possibles du système, associés aux différentes phases de vie, puis on décrit leur dynamique à l'aide de scénarios opérationnels. On fait apparaître le concept de flux d'échange, ce qui est très important. Cela fait émerger les comportements attendus du système, vus des parties prenantes, dans chacun des cas d'usage décrits.

L'essentiel

L'architecte mène une **analyse boîte noire** à la fois statique et dynamique de son système d'intérêt afin de définir la manière dont le système complexe est utilisé par ses parties prenantes et les comportements qu'il doit réaliser vus des parties prenantes.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Chercher à aller trop loin dans l'analyse (viser l'impossible exhaustivité) au lieu de se focaliser sur les cas d'utilisation essentiels
- Se focaliser sur les cas d'usage où finalement les équipes sont les plus à l'aise (cœur de métier)
- Oublier les cas d'usage dysfonctionnels (c'est à dire ne traiter que les cas nominaux) ou également trop détailler ceux-ci (au risque de noyer les cas qui portent l'essentiel de la valeur)

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Choisir un outil adapté à la sémantique des parties prenantes et la maturité de la phase projet,
- Viser l'exhaustivité dans la liste des cas d'usage et des interactions mais ne décrire sous forme de scénarios que les cas d'usage identifiés comme “essentiels” selon des critères valeurs (d'un point de vue client) versus risques (en interne),
- Se mettre d'accord avec les parties prenantes sur les périmètres de responsabilités des cas d'usage dysfonctionnels estimés “essentiels”,

- Se time-boxer et bien penser à la cohérence globale de l'architecture via une pratique itérative,
- Trier & factoriser les scénarios en fonction des comportements attendus du système.

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

“ Nous classons les cas d'usage en 4 catégories :
  - Forte valeur et faible risque : nous faisons attention à ne pas passer trop de temps là-dessus car ils sont bien connus et maîtrisés
  - Faible valeur et faible risque : il peut y avoir discussion avec le client mais on ne rentre pas dans le détail de ces cas d'utilisation
  - Forte valeur et fort risque : c'est là où nous ou nos clients sommes le moins confortables mais c'est justement là où nous devons mettre l'effort
  - Faible valeur et fort risque : le but est d'aller négocier ou questionner les besoins clients associés

“ Nous appliquons des principes de frugalité (en limitant volontairement le temps alloué à l'activité par rapport au temps estimé) pour stimuler la productivité : cela oblige à aller naturellement à l'essentiel,

“ Nous avons fait des BD pour aider le client à bien visualiser les usages proposés sur certains scénarios

3. ARCHITECTURE BOITE BLANCHE

3.1. CONCEVOIR UN SYSTEME QUI REPOND AUX BESOINS / CONTRAINTES DES PARTIES PRENANTES AVEC LES PERFORMANCES ATTENDUES, JUSTIFIER LES CHOIX D'ARCHITECTURES, PROPOSER DES ALTERNATIVES ET FAIRE CONVERGER LES SOUS-

Cette activité vise à :

1. Spécifier les comportements du système sous la forme d'une architecture fonctionnelle en identifiant et hiérarchisant les fonctions internes du système, en définissant et en synthétisant les interfaces fonctionnelles existantes entre ces fonctions par le biais à la fois d'un diagramme d'interactions fonctionnelles et d'un arbre de décomposition fonctionnelle (également appelé Function Breakdown Structure ou FBS). On définit enfin les modes de fonctionnement du système.

2. Allouer les fonctions internes du système aux composants par le biais d'une architecture logique puis physique en identifiant d'abord les composants qui implémentent les fonctions internes à l'aide d'une matrice d'allocation, puis en hiérarchisant ces composants et en définissant enfin leurs interfaces via un diagramme d'interactions.

3. Mettre en œuvre un protocole de convergence et de justification itératif vis-à-vis des parties prenantes et des équipes sous-systèmes en construisant et en faisant évoluer des vues d'architecture. L'objectif est de faire adhérer toutes les parties prenantes à une même vision de l'architecture du système afin d'en sécuriser la définition et faire valider les compromis éventuels. L'architecte système est responsable d'arbitrer et doit s'assurer de la convergence mais ce n'est pas forcément lui qui fait les choix d'implémentation.

Ces activités ne doivent pas être vues comme séquentielles mais doivent être menées en parallèle. Tous les aspects dysfonctionnels doivent être pris en compte.

L'essentiel

L'architecte mène une **analyse boîte blanche** qui lui permet d'identifier **les fonctions couvrant les besoins et usages** identifiés et priorisés tout en **appréhendant les couplages** à venir entre les composants. Il commence par mener une réflexion autant que possible **indépendante de la technologie utilisée** au niveau des composants puis il pilote **les choix d'implémentation**. Il est garant de la **convergence multi-métier sur les nécessaires compromis** à trouver.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Empiéter sur les domaines métiers. L'architecte doit être le lien entre les différents métiers et disciplines et doit permettre les trade-off grâce à une vision d'ensemble. La responsabilité technique doit être portée par l'architecte mais cela est parfois compliqué à être accepté par les métiers (ex : les métiers du logiciel qui travaillent à partir de vues dédiées qui prennent le dessus),
- Ne pas réussir à mener à bien les itérations avec les sous-systèmes en phase d'architecture. Ce qui peut générer des surcoûts et des reprises en aval. Cela est souvent par faute de temps (ils voient l'intérêt mais n'ont pas automatisé l'activité dans leur processus) ou par manque de valeur ajoutée visible de leur part (surtout quand les problématiques d'intégration ne sont pas historiques sur ces couches),
- Croire et laisser croire que la mise en place des pratiques d'architecture est une démarche tout ou rien, sans latitude possible sur son implémentation,
- Confondre logique/physique avec logiciel/matériel,
- Voir l'architecte comme le garant de l'architecture fonctionnelle et pas dysfonctionnelle.

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Intégrer les pratiques d'architecture étape par étape et monter progressivement en ambition. Faire un premier projet en mode POC et ensuite généraliser.
- Organiser les niveaux d'abstraction des diagrammes en respectant la règle des 7x7x7
- Adapter ses pratiques à la complexité du produit et à la maturité de l'entreprise. Il faut toujours garder en tête que la mise en place de l'architecture doit créer de la valeur.
- Prendre en compte à la fois la dimension statique et temporelle de l'architecture
- Démontrer la valeur strate par strate. On commence par faire monter en maturité le niveau système avant de s'attaquer au niveau sous-système et ainsi de suite. Démontrer la valeur par l'exemple aux strates du dessous est un facilitateur et met en avant plus tôt les changements nécessaires.
- Mapper les activités d'architecture sur les livrables afin de donner le lien entre activités et livrables multi-métier
- Pratiquer l'écoute active via l'organisation régulière (fréquence à adapter au contexte) de réunions multi métiers d'échange autour des architectures. Les livrables d'architecture pouvant être utilisés comme support de discussion et de compte rendu de réunion. Cela peut également être fait de manière informelle.
- Si le mot fonction est employé il doit être associé à l'adjectif "interne" ou "externe" afin de bien clarifier de quoi on parle
- Il faut analyser les variantes dysfonctionnelles des cas d'usage.

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

- Nous avons mis en place un ensemble de revues systèmes qui clarifient les niveaux de maturité des architectures attendus aux jalons et articulent le travail entre système et sous-systèmes. L'architecte participe également aux revues des sous-systèmes (qui doivent prendre en compte les exigences système)
- Nous avons instauré des rituels de travail de 2 ou 3h hebdomadaires pour partager les travaux d'architecture entre les strates et faciliter la convergence multi métier.
- Les équipes logicielles se sont d'abord senties perturbées et inquiétées par la mise en place d'une architecture fonctionnelle. Nous les avons intégrés plus tôt dans la définition des fonctions "boites blanche" pour qu'ils se sentent plus à l'aise et moins en danger.

3.2. INTERACTIONS ENTRE L'ARCHITECTE ET LES ETUDES DYSFONCTIONNELLES

Il s'agit d'analyser les dysfonctionnements du système complexe pouvant conduire à des conséquences critiques en termes de fiabilité, disponibilité, sûreté et sécurité. On identifie les risques de dysfonctionnement du système puis on itère l'architecture et les exigences. Le risque résiduel doit être dans une zone acceptable en mitigeant son occurrence et sa criticité.

L'essentiel

L'architecte contribue à l'analyse de sûreté du système en permettant, grâce à son architecture, d'identifier et analyser tous les risques possibles afin de garantir l'atteinte des objectifs de sûreté.

4. MODELISATION DU SYSTEME ET DES CHAINES DE VALEUR DANS L'ARCHITECTURE

Sur base des éléments collectés auprès des différentes parties prenantes, l'architecte formalise des modèles logiques du système, organisés selon un cadre d'architecture système. Il est le garant du respect de la syntaxe – plus ou moins formelle – donnée par le langage de modélisation choisi et des activités de modélisations à effectuer. La signification précise de chaque composant d'un modèle systémique doit avoir été définie de manière explicite et non ambigüe.

L'essentiel

Les travaux de l'architecte reposent sur la construction de modèles qui permettent de produire (de manière automatisée dans une vision idéale) une documentation d'architecture système exhaustive, cohérente et non ambiguë.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Voir les modèles uniquement comme des outils de communication et d'alignement et ne pas baser ses décisions d'architecture sur l'exploitation des modèles,
- Ne pas mettre en place de liens de traçabilité entre les modèles,
- Ne pas expliciter les liens entre modèles d'ingénierie et modèles d'architecture
- Ne pas assez se focaliser sur la forme : un diagramme illisible n'est pas communicable et est donc difficilement exploitable.
- Ne pas se servir des modèles pour mettre en avant les chaînes de valeur et les communiquer auprès des parties prenantes.

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Procéder par étape dans la mise en place d'une ingénierie tirée par les modèles : d'abord on formalise, puis on partage les modèles, on construit des architectures, on ré-utilise des architectures, on s'en sert pour de la simulation, puis pour spécifier etc.
- Mettre en place un plan d'architecture par projet qui instancie la boîte à outil et le niveau de profondeur attendu pour chaque case (il faut cependant donner des critères par exemple, modéliser la nouveauté)
- Attacher l'aspect risque et valeur sur les modèles afin d'en avoir une vue globale partagée et se servir des modèles pour mettre en avant les périmètres sur lesquels il faut travailler (particulièrement pertinent dans une entreprise avec un fort historique)
- Avoir une démarche de modélisation de bout en bout - du développement aux tests (ex : formaliser des scenarios avec une optique test (traçabilité)), tout en tenant compte du temps alloué
- Communiquer en utilisant toujours les mêmes modèles (pour ne pas perdre les gens) et adapter le niveau de détails à l'audience (il faut donc plusieurs niveaux pour chaque modèle)
- Ne pas hésiter à ajouter des images ou autre pour rendre la modélisation plus facile à comprendre et mieux communicable
- Voir un objectif clair par modèle et un seul message par diagramme : on doit savoir pourquoi on fait de la modélisation. Ne pas hésiter à créer un second diagramme si on doit passer un second message
- L'information doit être accessible sans que les outils soient un frein
- Il faut bien penser au cycle de vie des modèles créés et en particulier bien définir la manière de les gérer en configuration
- Jeter les diagrammes quand ils ne sont plus utiles (arrêter de les maintenir)

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

> “ Nous utilisons nos modèles d'architecture pour valider des analyses safety,
>
> “ Nous avons mis en place un modèle 150% qui est instancié par projet en fonction du contexte et des objectifs à atteindre
>
> “ Nos outils de modélisation : Visio, Draw IO, Enterprise Architect, Capella, Rhapsody, Tom Sawyer ("simple" mise en page automatique à partir de connexion à d'autres outils).

5. PROPOSITION, JUSTIFICATION ET CHOIX DES ARCHITECTURES CONCURRENTES

L'architecte identifie les solutions candidates issues de l'analyse boîte blanche pouvant répondre aux besoins selon les principes de l'ingénierie collaborative. Avec les bonnes parties prenantes, il définit les critères d'évaluation des architectures (ex : niveau de couverture des besoins, QCD, Risk, Re-use, Quantification de la nouveauté…) puis documente également leurs points forts et leurs faiblesses en capitalisant les justifications associées.

L'essentiel

La sélection d'une architecture parmi plusieurs architectures concurrentes se fait selon un processus de sélection qui doit être justifié sur la base de critères, sélectionnés selon les principes de l'ingénierie système, critères selon lesquels l'architecture peut être évaluée selon ses forces et faiblesses.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Il n'y a pas assez de solutions candidates. C'est souvent induit par trop de contraintes : manque de temps, de moyens et parfois d'imagination (on est déjà content d'avoir une solution répondant aux besoins)
- Dans les processus itératifs rapides (y compris les processus AGILES), il est parfois délicat de positionner dans le cycle, l'identification d'architectures candidates et les justifications. On a tendance à présenter la solution obtenue
- Pas de justification capitalisée au moment de la conception, ce qui entraîne du rétro-engineering après coup, ou de la perte d'informations et des difficultés à apporter les justifications lors d'un audit, de l'analyse d'impacts ou d'évolutions ultérieures
- Analyse boîte noire insuffisante, ce qui entraîne un biais dans la sélection des critères, des parties prenantes…
- Ne pas anticiper la variabilité des besoins dans l'évaluation des architectures candidates
- Mauvais choix des critères permettant de choisir entre les architectures concurrentes : mauvaise prise en compte du non-fonctionnel (évolutivité, résilience…), critères de différences…
- Ne pas prendre en compte l'architecture du modèle de données qui est clef sur certains types de systèmes
- Implication de trop peu de parties prenantes ou surreprésentation de certaines parties prenantes (marketing, client…) : on reste dans son quant-à-soi sans ouvrir la réflexion
- Pas de factualisation des critères : l'évaluation n'est plus objective

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Prévoir cette phase dans le processus de développement : imposer par exemple une revue technique régulièrement dédiée à ce sujet (1 fois par mois par exemple), ou un critère aux jalons projet, ou encore un questionnement tous les x sprints
- Ajuster l'effort au niveau du risque et de l'enjeu associé en fonction des sujets
- Faire émerger des architectures en rupture en organisant des ateliers créativité et innovation
- Mettre en place une bonne gestion de la connaissance (dont systèmes passés, veille techno et brevets…)
- Modéliser et décrire les différentes alternatives pour aider à les présenter, les valider, et faire les choix en connaissance de cause
- Communiquer au travers des diagrammes de l'architecture afin de s'en servir de supports au discours
- Utiliser des processus de positionnement des architectures les unes par rapport aux autres : la matrice de Pugh, analyse risque/valeur, effort/bénéfice, matrice DSM…
- Utilisation d'outils logiciels permettant d'explorer rapidement un grand nombre d'architectures
- Réaliser un prototype pour les cas où le risque en jeu le justifie (c'est le principe du POC)
- Pour des évolutions, il est intéressant de comparer avec le statu quo pour challenger l'apport de valeur
- Pour le modèle de données (des systèmes d'information), distinguer et relier le modèle conceptuel au modèle d'implémentation (ex : un modèle conceptuel complet - au sens mathématique - peut donner un modèle d'implémentation complexe et coûteux)

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

- “ Nous inscrivons l'identification d'options et non pas une seule architecture dans le processus de conception
- “ Nous nous posons systématiquement la question de l'enjeu associé à la conception. C'est ce qui drive nos choix d'alternatives.
- “ Nous utilisons un Novelty Ranking qui évalue le risque qu'on prend en fonction du nombre de fois où la solution a été déployée

5.1. VALIDER LES CHOIX TECHNIQUES

L'architecte identifie les activités nécessaires à la validation des choix ainsi que les moyens nécessaires à la validation (essais, modèles, simulations, revues…). Il réalise les activités de validation nécessaire (modèles, simulations, analyses, etc… par exemple ARP4754A pour les systèmes aéronautique) et capture les résultats de validation en les reliant aux exigences concernées.

Sur les axes d'architecture côtés HIGH, il identifie les activités de justification (au sens IADT du terme, pas forcément TEST !) et dans le cas de type test les moyens afin d'alimenter le plan d'expérience. Il prévoit les revues nécessaires pour valider les choix et capitalise les éléments qui ont permis de faire les choix (dossier de justification de la solution).

Il convient de bien définir les critères de choix.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Mauvaise capitalisation des activités au fil de l'eau, difficultés à ressortir les preuves lorsque nécessaire et difficulté à démontrer la bonne couverture de la validation
- Mauvaise gestion en configuration des moyens de validation (modèles…)
- Ne pas impliquer les personnes des moyens d'essai très tôt dans le cycle de dev.
- Mauvaise définition des critères de choix.
- Manque de traçabilité des choix

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Définir tôt dans le projet la stratégie de validation dans un plan (indispensable par exemple en cas de certif ARP dans l'aéronautique)
- Identifier et gérer les hypothèses projet pour aider à la validation
- Gérer les moyens de validation comme les autres données d'ingénierie en configuration
- Aligner les moyens de validation aux objectifs de validation (recalage, validation de modèles, de leur finesse par rapport aux objectifs…)
- Avoir la capacite de tracer les résultats servant à la justification/choix dans la spécification
- Gérer les évolutions pour garantir que le dossier de justification de la solution est toujours cohérent

6. ÉVALUER DE L'ARCHITECTURE

6.1. ÉVALUER LA MATURITE DE LA DEFINITION DE L'ARCHITECTURE

Tout au long du cycle de conception de l'architecture, il convient de suivre la confiance qu'on peut avoir dans la définition de l'architecture sur base d'indicateurs que l'architecte définit lui-même ou qui sont imposés par le projet/organisation. Ces indicateurs sont à la fois utiles au projet de conception et aux parties prenantes pour suivre l'avancement.

L'essentiel

Il convient ici de bien définir les objectifs en début de projet vis-à-vis des indicateurs, de mettre en place des indicateurs de maturité en début de projet et de faire des revues (de pairs) de manière régulière (à la fois d'audit et ingénierie collaborative).

Les écueils principaux

Parmi les principaux écueils, on notera :

- Ne pas piloter la montée en maturité de la définition
- Avoir des indicateurs qui ne sont que sur la production d'éléments et pas sur leur qualité
- Lier l'avancement contractuel à des indicateurs uniquement de production de documents ce qui peut conduire à maintenir deux jeux de documentations ou pire que la documentation ne soit que pour le reporting
- Réutiliser des architectures existantes dont la maturité est avérée peut empêcher d'identifier des axes d'amélioration
- Une hypothèse de réutilisation stricte mal analysée peut ne pas s'accorder avec le contexte d'utilisation réel et générer des réajustements potentiellement coûteux
- Difficulté d'identifier la maturité de sujets en innovation ou de domaines inconnus
- La maturité d'un système n'est pas la somme de la maturité de ses constituants (propriétés émergentes, intégration…)
- Ne pas monitorer la maturité de la définition sur l'ensemble du cycle de vie (en particulier sur les phases aval à la conception)
- Oublier la maturité des interfaces dans l'évaluation de la maturité de la définition de la solution
- Oublier la maturité du modèle de données dans l'évaluation de la maturité de la définition de la solution
- Ne pas prendre en compte l'évolution des enjeux dans le suivi de l'évolution de la maturité
- Faire de l'évaluation de la maturité de la définition de la solution un travail en chambre

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Mettre en place quelques critères d'évaluation standard même de haut niveau, instanciables par les projets, et structurant pour donner une vision transverse aux projets au management et aux architectes. Voici quelques idées :
    - Évaluer la maturité par rapport à l'ensemble du cycle de vie, des use cases, des fonctions et scénarios opérationnels

- Fonctionnalité : Est-ce que les Use Cases sont identifiés et correspondent aux besoins des stakeholders. Est-ce que les fonctions, la décomposition et les dépendances fonctionnelles sont identifiées ?
- Structuration : Est-ce que les sous-systèmes, composants sont clairement définis et organisés ?
- Organisation : Quel est le niveau de correspondance entre l'organisation (incarnation & communication) et la structuration de la solution
- Interfaces : Identification des interfaces externes et internes (fonctionnelles/physiques) et leur nombre
- Allocation : Est-ce que les fonctions sont allouées aux composants ? Est-ce que les interfaces fonctionnelles sont allouées aux interfaces Physiques ?
- Taux de réutilisation (de composants, de patterns, d'interfaces …)

- Dissocier la documentation/livrable (ex : fichier word) des éléments d'architecture qui le constituent (ex : modèles, données …)
- Faire des revues d'évaluation régulières de la maturité de la définition de la solution avec les bonnes parties prenantes (l'équipe technique, les experts sur l'ensemble du cycle de vie)
- Prendre en compte l'avis des experts et responsables des lots
- Prévoir dans le développement des boucles de maturité progressives

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

> Nous avons mis en place un fichier de complétude des architectures qui suit chaque attendu des analyses d'architectures en termes de livrables et d'activités (ex : taux d'étude des phases de vie, nombre de cas d'utilisation, de scénarios…) ce qui permet de définir l'avancement sur l'architecture.

> Nous suivons l'avancement de la définition de l'architecture sur deux axes : le nombre de macro-uses cases étudiés (défini dès le départ de la conception) d'une part et la qualité de l'analyse (basée sur le nombre de peer reviews) réalisée sur ces cas d'utilisation d'autre part.

> Outils intéressants disponibles à l'INCOSE (SRL System Readiness Level) & NASA (ARL Application Readiness level) pour instanciation

6.2. ÉVALUER LA CONFORMITE DE L'ARCHITECTURE AUX BESOINS PRIORITAIRES OU A VALEUR

L'architecte évalue la conformité de l'architecture aux besoins prioritaires qu'il doit dans un premier temps identifier en tant que tels. La conformité est ensuite établie, dans un premier temps, au regard de l'état de l'art puis vis à vis des constituants (2ème temps une fois les besoins déclinés via les premières ébauches d'architecture).

Les résultats de cette évaluation et les premiers retours de conformité sont consolidés via l'établissement d'une matrice de conformité permettant une vision globale sur ce qui est conforme, non conforme ou partiellement conforme.

L'essentiel

L'évaluation de la conformité d'une architecture aux besoins prioritaires ou à valeur est un processus qui doit être initié très tôt dans le cycle de développement avec l'identification des besoins qui portent la valeur pour le client et la mise en place de la matrice de conformité définissant les éléments attendus aux différentes phases (justification, essais élémentaires, essais d'ensemble.).

Les écueils principaux

Parmi les principaux écueils, on notera :

- Mauvais échanges entre strates d'ingénierie et mauvaise prise en compte des non-conformités potentiellement remontées
- Politique de poker menteur vs contrats et contraintes financières (ou pour avoir le contrat)
- L'équipe amont n'utilise pas l'ingénierie minimale des exigences (pas de balise amont) afin de donner une matrice factuelle.
- Tous les besoins sont prioritaires
- Absence d'exigences clairement définies

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Identifier les besoins prioritaires (KDD, key design drivers) avec les parties prenantes émettrices
- Capitaliser l'état de l'art et avoir une gestion de cet état de l'art (mis à jour par la R&T notamment) pour savoir à tout instant de quoi on est capable
- Disposer d'une bonne gestion des marges multi-strates d'ingénierie
- Avoir un état factuel de la couverture de la proposition technique qui soit pris en compte avec le niveau de performance associé
- Élément contribuant au T du QCDT lors du passage de phase
- Très tôt dans le cycle de développement avoir une cartographie préliminaire des Key Driver Parameters (métriques principales) sur les différentes couches d'analyse (besoin, système, sous-système…) permettant d'évaluer de manière préliminaire la faisabilité et la conformité aux besoins. Établir un lien de traçabilité entre ces KDP.

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

> “ Chez nous la priorisation de la valeur est faite par les équipes qui ont en charge l'écriture de la boite noire. La réponse se fait au travers l'architecture sous forme de lot avec un découpage logique qui représente la chaine d'intégration

> “ Contraintes contractuelles plus fortes que les alertes de non-conformité des sous-systémiers et client prévenu trop tard des non-conformités, programme annulé tardivement et pertes financières et d'image à la fin

> “ Élaboration d'un modèle de prédimensionnement permettant de faire le lien entre les KDP opérationnels, les KDP systèmes et KDP sous-systèmes. Élaboration d'une matrice DSM détournée -> "KDP Structure Matrix" permettant d'assurer un lien de traçabilité des KDP et d'analyser les impacts sur la non tenue des certains paramètres

> “ Tous les besoins sont prioritaires. Chez nous, il est très difficile d'avoir l'analyse de la valeur comme justification. La phrase magique " Le Marché n'acceptera jamais " est très souvent employée

6.3. ÉVALUER LA MATURITE TECHNIQUE DES CHOIX DE LA SOLUTION

Cela nécessite de prévoir un cycle de développement intégrant des boucles permettant une montée en maturité progressive.

Les choix d'architecture sont soumis à une évaluation du risque propre à chacun d'eux permettant de se positionner sur leur acceptation selon des ratios bénéfices/risques. Les bénéfices restent centrés sur la satisfaction des parties prenantes (client, business, industrialisation…) via des critères de choix type coût, masse, design drivers.... On doit identifier les nouveautés de l'architecture par rapport aux programmes passés et à l'expérience de la société.

Chacun des choix d'architecture doit avoir une évaluation de son risque inhérent, pour lequel des plans de levées de risques sont proposés.

7. INTERFACES

7.1. GERER LES INTERFACES FONCTIONNELLES ET PHYSIQUES, INTERNES ET EXTERNES

En parallèle de la définition des architectures boîte noire et boîte blanche, l'architecte doit identifier les interfaces, externes et internes (avec les systèmes extérieurs, entre les fonctions du système et entre ses composants).

Tout comme il doit s'assurer que chaque fonction et chaque composant est porté en responsabilité par un rôle, de même il doit mettre sous contrôle l'ensemble des interfaces identifiées.

L'essentiel

La complexité d'un système est en autre, liée au nombre de ses interfaces qui peut conditionner le nombre de parties prenantes à impliquer. La gestion des interfaces (humaines, physiques, logicielles…) est une activité clef de la maîtrise des risques sur un projet. Dans la partie boîte noire, la définition des interfaces est un élément majeur dans la formalisation du périmètre d'étude et l'intégration du système dans son environnement.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Identifier des interfaces externes sans se poser la question du besoin qu'on doit couvrir et aller trop vite dans le détail technique (ex : vouloir définir finement les types de données à faire transiter). C'est souvent amplifié par une volonté de re-use mal cadrée par rapport au nouveau contexte/besoin
- Ne pas suivre en maturité les interfaces, c'est à dire ne pas gérer une interface car encore peu mature (cela met les gens en attente et génère des retards potentiels) et/ou ne pas anticiper le fait que la définition de certaines interfaces va évoluer (ce qui génère un rework perçu comme inutile)
- Mettre à jour les interfaces sans informer les parties prenantes impactées par cette modification
- Ne pas identifier de responsabilité pour une interface
- Mal gérer en configuration les deux organes de part et d'autre lors de l'intégration des interfaces
- Interfaces bien définies sur des aspects précis (ex : fonctionnement statique mécanique) qui permettent à priori l'intégration mais qui sont mal définies au final sur d'autres aspects (ex : fonctionnement en dynamique)
- Détourner des interfaces (dans le cadre d'un re-use) et risquer de sortir du nominal de design de l'interface (avec des problématiques associées, ce qui reviendrait plus cher que de définir la bonne interface)

- Une mauvaise définition des interfaces externes conduisant à une ambiguïté sur le périmètre (ex : l'interface pour authentifier l'utilisateur : le client considère que c'est dedans et le concepteur du logiciel considère que c'est hors de son périmètre)

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Utiliser l'architecture pour identifier les interfaces
- S'assurer que chaque interface est bien portée (en responsabilité)
- Mettre en place un processus de gestion de maturité des interfaces
- Dans le cadre de re-use se reposer la question du besoin auquel on répond avec chaque interface
- Mettre des détrompeurs (Poka-Yoke) sur des interfaces qui sont physiquement identiques mais fonctionnellement différentes (ex : détrompeur sur les prises murales vides et oxygène dans les chambres d'hôpital)
- Définir toutes les caractéristiques de l'interface (le nominal et la robustesse lorsqu'on sort du nominal)
- Utiliser du MBSE pour définir de façon la plus exhaustive possible toutes les interfaces
- Standardiser la manière de définir les interfaces
- S'appuyer sur la maîtrise des interfaces pour optimiser les interactions avec les parties prenantes (ex : la gestion de plusieurs interfaces avec une même partie prenante peut permettre d'augmenter la communalisation entre ces interfaces)

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

> “ Nous avons mis en place un processus de gestion de maturité des interfaces, en particulier sur le suivi des interfaces mécaniques. Chaque document d'interface a un niveau de maturité défini et affiché, ce qui permet de partager les risques associés à chaque interface.

> “ Sur des systèmes logiciels, nous procédons à de l'early-integration plutôt qu'à une montée progressive en maturité des interfaces. (Attention néanmoins à ce que cette pratique ne se couple pas à l'écueil mentionné au-dessus sur le questionnement du besoin à couvrir).

> “ Pour le hardware : on est plutôt sur la mise en place d'une étape d'intégration virtuelle (intégration des modèles numériques métier - 3D, hydrauliques, électriques…) pour anticiper les problèmes d'intégration

> “ Mise en place d'une gestion outillée des interfaces en utilisant le cadre CESAM dans Xatis (Safran)

8. ROLE DE L'ARCHITECTE DANS LE CADRE D'UNE LIGNE DE PRODUITS

Par convention on parlera ici de gammes et de lignes de produits en prenant la définition suivante :

- Une gamme de produits est un ensemble de produits qui partagent des communalités fonctionnelles et organiques (en termes de composants) pour répondre à une multitude d'usages mais sans optimisation de la variabilité (ex : la gamme des moyens courriers)
- Une ligne de produits est une gamme de produits qui est architecturée et dont la variabilité est maîtrisée avec une architecture flexible et des composants standards / scalables / customisés. L'objectif est de réduire les coûts récurrents et non-récurrents, d'améliorer le temps de mise sur marché et de limiter le coût de la gestion de la variabilité (manuf, SAV, achats…) (Ex : châssis entre les Golf et les Leon)

Un produit standard : c'est l'ensemble des composants permettant de construire les différents produits de la ligne de produits (formalisé dans la PBS 150%) sous contrôle d'une gouvernance.

Attention : la gestion d'un catalogue de composants réutilisables n'est pas considérée ici comme une stratégie de ligne de produit puisque cela n'adresse pas le niveau système. Ce qui ne veut pas dire que ce n'est pas une stratégie intéressante à envisager. Il peut aussi y avoir une différence entre ce qui est marketé/affiché vis à vis du client et ce qui est architecturé/réalisé en interne.

8.1. POUR UN ARCHITECTE DE LIGNE DE PRODUIT, GERER LE CYCLE DE VIE : CREATION D'UNE LIGNE DE PRODUIT

Deux cas sont possibles pour la création :

- Des clients demandant des choses similaires aboutissant à une opportunité de création d'une ligne de produit ;
- Le marketing/la stratégie fait une étude d'opportunité identifiant le fait de mettre en place une stratégie de ligne de produit. L'étude de marché peut englober plusieurs niveaux de produits et de services. Il faut également bien prendre en compte la récurrence des demandes et l'évolutivité du produit (liées aux facteurs légaux, technologiques...). L'architecte doit avoir en entrée ces données pour construire dans le temps sa ligne de produit.

Dans le cadre d'un projet interne (pas du développement d'un produit pour un client), il s'agit de définir l'architecture standard en utilisant toutes les techniques de l'architecte plus les techniques de maîtrise de la variabilité.

L'essentiel

Bien définir une ligne de produit commence par bien délimiter le périmètre du marché qu'on cherche à adresser et à récolter l'ensemble des besoins qu'elle doit couvrir ainsi que le niveau de maturité attendu par ce marché. Cela doit se faire en collaboration entre les gens qui représentent les clients et ceux qui portent la technique (y compris les aspects fabrication, logistique…). Garder sous contrôle ce périmètre est un élément clef de la réussite d'une ligne de produit : on a en effet tendance à l'agrandir par opportunisme ou le réduire par volonté d'optimisation locale. La manière d'adresser ce périmètre (au bon niveau de grain, dans la bonne temporalité…) est également un facteur de succès. La mise en place d'une ligne de produit induit forcément un changement de paradigme à prendre en compte.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Identifier des interfaces externes sans se poser la question du besoin qu'on doit couvrir et aller trop vite dans le détail technique (ex : vouloir définir finement les types de données à faire transiter). C'est souvent amplifié par une volonté de re-use mal cadrée par rapport au nouveau contexte/besoin
- Pas de gouvernance dès la création de la ligne de produit
- Écueil récurrent vis à vis des demandes clients : créer une ligne de produit uniquement sur opportunités projets sans mise en place d'une stratégie basée sur une étude de marché (même très light ou en récupérant les informations des commerciaux/tickets SAV etc.), d'une gouvernance associée ou de processus off-cycle qui demandent un investissement hors projet
- Phasage temporel : développement pas en phase avec les besoins projet
- Non prise en compte du niveau de récurrence et de l'évolutivité (produit, du marché au sens large etc.) dans la stratégie de mise en place d'une ligne et du scope

- Gestion du changement : la rationalisation des coûts par la mise en place d'une ligne de produit peut créer des peurs au sein des équipes. Accompagner le changement peut être un coût non négligeable de la mise en place d'une ligne de produit

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Se donner une vision cible du produit 150% mais le développer (en fonction des opportunités) étape par étape afin de se laisser la liberté dans la roadmap, ce qui permet d'être plus efficace pour l'adéquation avec le besoin client
- Mettre en place deux feature models : un feature model boîte noire qui permet d'appréhender l'ensemble des possibilités qu'offre la ligne de produit vue du client et un feature model boîte blanche qui relie les composants nécessaires en regard des options choisies
- Ne pas négliger les aspects communication entre projets et ligne de produit (via une gouvernance). Les projets doivent comprendre qu'on cherche un optimum global au niveau de l'entreprise. La ligne de produit doit intégrer les besoins des projets pour ne pas être hors sol.
- L'outil de production doit être associé aux réflexions de la ligne de produit
- Conscientiser les choix stratégiques qui sont fait autour de la ligne de produit
- Inciter les projets à la réutilisation
- Clairement définir les limites de la ligne de produit, c'est à dire le type de systèmes qu'elle ne couvrira pas
- Mailler l'architecture documentaire (de la spécification jusqu'aux tests) sur l'architecture de la ligne de produit afin de favoriser la capitalisation sur projet et la gestion de la configuration
- Sur des produits très complexes, ne pas hésiter à développer la ligne de produit au niveau des sous-systèmes (sous réserve d'avoir pensé globalement l'approche modulaire)
- Mettre en place une montée en compétence des collaborateurs sur les principes des lignes de produits
- Découpler les interfaces externes des fonctionnalités coeur afin de les rendre plus facilement intégrables (et ne pas à avoir à re-designer les fonctions coeur dès que les interfaces externes changent)

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

> Nous avons mis en place une stratégie de type "briques & modules" c'est-à-dire que nous nous sommes dotés d'une vision d'architecture modulaire cible en travaillant sur notre diversité existante et nous développons les briques technologiques de cette architecture par étape.

> Nous commençons à envisager chaque élément de nos systèmes comme une brique de légo réutilisable et les incluons dans une démarche plus globale de "platforming" jusqu'aux couches les plus hautes de nos systèmes

> Nous avons synchronisé une démarche de ligne de produit calculateur avec une ligne de produit logiciel et ainsi les interfaces logiciel/calculateur sont stabilisées

> Nous avons mis en place notre ligne de produit sur le logiciel afin de mieux maîtriser la montée en cadence des développements

8.2. POUR UN ARCHITECTE DE LIGNE DE PRODUIT, GERER LE CYCLE DE VIE : MAINTENANCE ET EVOLUTION

L'évolution d'une gamme de produits est dictée par l'évolution des besoins. Ces changements de besoins peuvent provenir d'un certain nombre de sources, telles que le marché, les besoins futurs de l'entreprise ou le désir d'introduire de nouveaux produits dans la ligne de produits.

L'essentiel

Garder le contrôle du périmètre et de la dette technique (gestion de l'obsolescence et des choix techniques d'implémentation) de la ligne de produit via une gouvernance forte est un enjeu majeur dans sa maintenance et son évolution.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Il n'existe pas de gouvernance ou pas de gouvernance solide (c'est à dire qui n'est pas reconnue dans l'entreprise ou outrepassée par d'autres processus - par les projets par exemple).
- Une gouvernance trop rigide ou décorrélée des besoins opérationnels qui ne permet plus de s'adapter aux changements du marché
- Le projet qui vient forcer la ligne de produit à faire une modification et in-fine dénaturer la ligne de produit
- Gestion de l'obsolescence :
    - On s'accroche à toutes les fonctionnalités, même à celles qui ne sont plus d'actualité, ce qui conduit à une charge de maintenance trop élevée
    - L'obsolescence des composants non prise en compte dans la création conduit à un coût de maintenance trop élevé de la ligne de produit, voire la met en péril
    - La création et la maintenance sont découplées et n'ont pas de règles communes

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Mettre en place une rétroaction des projets vers la ligne de produit au sein de la gouvernance pour réinjecter ce qui pourrait être réutilisé et comprendre ce qui n'est pas utilisé
- Il faut faire en sorte que la maintenance et l'évolution de la ligne de produit soit faite au bon moment c'est-à-dire au moment où on a besoin de faire une instanciation. Cela implique un pilotage de la ligne de produit en fonction des temporalités projet
- Faire bouger les architectes entre la maintenance de la ligne de produit et les projets pour améliorer la pertinence des architectures vis à vis des besoins projets
- Avoir maillé lors de la création de la ligne de produit l'architecture documentaire (de la spécification jusqu'aux tests) sur l'architecture de la ligne de produit afin de favoriser la capitalisation sur projet et la gestion de la configuration
- Ne pas hésiter à élaguer certains éléments/features voire faire une nouvelle ligne de produit si on se rend compte que la maintenance ou l'évolution de la ligne de produit ne fait plus sens opérationnellement ou économiquement

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette phase :

- “ Aujourd'hui il nous arrive de passer beaucoup de temps à retrouver des rapports de tests, les justifications des choix de design etc... par manque de maillage entre la documentation et l'architecture, et de partage entre les différentes entités
- “ On travaille surtout en extension de lignes de produit (en ajoutant de nouveaux artefacts) pour gagner en efficacité voire adresser de nouveaux marchés plus rapidement
- “ Les environnements de développement varient énormément sur des séries très longues : nous sommes obligés de garder de vieux PC uniquement pour faire des modifications en cas de soucis sur ces anciens produits

8.3. POUR UN ARCHITECTE SYSTEME SUR PROJET : ASSURER LA COHERENCE DE L'ARCHITECTURE DU PROJET AVEC LE PRODUIT STANDARD (QUAND IL EXISTE)

L'essentiel

Plusieurs cas peuvent être considérés :

- Instanciation d'un produit de la ligne de produit sur projet (100%, 80%-20%, 20%-80%)
- Un système qui utilise des composants de la ligne de produit

Nous faisons la différence entre les développements spécifiques ou ad-hoc (pas issu de l'instanciation d'un produit) et les produits qui découlent de l'instanciation de tout ou partie de la ligne de produit.

La clef = les interfaces. Il faut être attentif aux interfaces. Si on maîtrise les interfaces ça permet d'intégrer beaucoup plus facilement du spécifique.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Le choix de la capitalisation des configurations employées par les projets (à mettre dans la partie produit ou pas ; avec éventuellement un paramètre qui permet de l'activer ou non)
- 20% de ligne de produit + 80% de développement spécifique. Se pose la question de se surcontraindre sur les 80% et de faire une usine à gaz pour être capable d'utiliser les 20%
- Les différences de temporalité entre la ligne de produit et les projets pour le build
- S'assurer que la temporalité du projet est en ligne avec la temporalité du maintien de la ligne de produit. Si la gouvernance décide de ne plus supporter la feature qui m'est utile pour mes projets alors le projet se trouve à supporter le maintien du produit
- Si la gouvernance n'est pas assez forte, le projet doit faire attention à ce qu'il demande au produit
- La gestion de l'obsolescence
- La gestion du change : une nouvelle fonctionnalité va arriver dans la ligne de produit alors doit se poser la question de l'impact dans les projets (ex : augmentation du coût d'infrastructure alors qu'on n'a pas besoin des nouvelles fonctionnalités embarquées)

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Le commercial doit travailler en collaboration avec la ligne de produit afin d'influencer le client
- Vérifier la compatibilité des temporalités / durées de vie du produit avec ce qui est demandé dans le projet quand on utilise une ligne de produit
- Accepter les compromis globaux et accepter conséquemment de dégrader de manière acceptable les performances de son projet (tout en gérant le bien de son projet)
- Savoir dire non ! Non au client, non à la ligne de produit… pour éviter l'usine à gaz sans suffisamment de valeur ajoutée
- Ne pas avoir peur des couches intermédiaires / adaptateurs qui permettent de découpler les fonctionnalités cœurs des demandes clients. Cette couche intermédiaire augmente la complexité et peut dégrader la performance mais est clef pour éviter certains écueils cités ci-dessus

Témoignages

> " On a prévu de déployer une application dans un device avec mémoire limitée, le produit va faire une évolution qui va faire augmenter la mémoire nécessaire et ça ne passe plus. On peut passer sur une gestion en branche et payer les coûts de conception et maintien ou demander une évolution (via un paramètre) de la ligne de produit (et donc augmenter la complexité). Il n'y a pas de bon compromis.

9. ANALYSE D'IMPACT DES DEMANDES DE CHANGEMENT (MODIFICATIONS) ET EVOLUTION

L'essentiel

L'analyse d'impact s'inscrit dans le processus de gestion des modifications. L'analyse d'impact s'appuie sur :

1. Les données et leur traçabilité
2. La gestion de configuration (sur l'ensemble du cycle de vie du produit, de la conception (maturité de la conception) jusqu'à la vie opérationnelle)

Le maintien de la traçabilité de l'architecture conduit à avoir la capacité d'identifier "rapidement" les impacts d'un changement sur l'architecture. La traçabilité s'entend sur l'ensemble des items de développement et pas uniquement sur les exigences (justifications, risques…).

Une analyse d'impact s'appuie sur un existant et dépend donc de la qualité de l'existant.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Ne pas capitaliser les choix de conception : en cas de modification on doit faire du rétro-engineering car on n'a plus la capacité de comprendre ce qui a conduit aux choix faits, avec des implicits technologiques
- Le manque de rigueur : on peut avoir les bonnes personnes ou les bons outils mais si on manque de rigueur dans la capture et la structuration des informations, alors cela n'est pas efficace (les processus incitent à cette rigueur). Cela peut être dû à un manque de temps, à la culture d'entreprise, à une peur liée à la responsabilité associée à l'information, à une rétention d'information (manière de se rendre nécessaire)
- Aller trop dans le détail de la traçabilité : cela devient ingérable. Il faut trouver le bon équilibre. Cela demande de l'expérience pour trouver ce curseur gain/effort.

- Que l'architecte soit le seul à porter les analyses d'impact : cela doit être fait collectivement entre l'ensemble des parties prenantes (le projet, les métiers, les disciplines, la maintenance, le manufacturing…) qui doivent valider
- Avoir une traçabilité déconnectée des configurations du produit et sans référencer les versions précises des éléments qu'elle met en œuvre

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Pour faire une bonne d'analyse d'impact il faut avoir la capacité de lier les items de développement et donc avoir une traçabilité de qualité :
    - Bien identifier où il est pertinent de faire de la traçabilité formelle (car cela a un coût)
    - Choisir ses formats : Les diagrammes sont les formats qui résistent le mieux à l'épreuve du temps et aux changements de culture des différents intervenants sur le projet. L'aspect visuel permet d'identifier graphiquement les effets dominos.
    - Savoir-faire « chanter la data » (avoir la capacité d'interroger la data) : une fois qu'on a pu capturer la data de manière "intelligente" c'est à dire structurée (architecturée !), il devient plus facile de venir interroger la data et de se faire rapidement une idée des impacts en lien avec cette donnée. Le fait de respecter un cadre d'architecture de manière stable dans le temps est un enabler fort.
    - En plus de la traçabilité formelle (qui a un coût) on peut mettre en place dans les livrables (plans, codes etc.) des commentaires qui permettent grâce à des moteurs de recherche de compléter les analyses
    - Gérer la traçabilité transverse aux gestions de configuration (ex : entre soi et son fournisseur)
    - Faire de la traçabilité entre les modèles de simulation et les éléments de design
- On peut tout à fait vérifier ou compléter une analyse d'impact faite sur modèles en conduisant des simulations et des tests
- Faire un plan détaillé du dossier de justification (pour vérifier que toutes les informations nécessaires y entrent et se retrouvent facilement) aligné sur la structuration du dossier d'architecture. Cela peut éviter une traçabilité formelle trop conséquente. Cela s'applique certes sur le dossier de justification mais bien entendu cela peut s'étendre sur l'ensemble du corpus documentaire.
- Avoir des processus outillés de gestion des modifications - dans lesquels s'inscrit le processus d'analyse d'impact - définis et formalisés, partagés et suivis en application (utiles, utilisés, utilisables)
- Faire des analyses itératives en largeur d'abord puis en descendant dans les niveaux de détail des modèles et dans les classes de modèle (mécanique, physique, logicielle…)

Témoignages

> Nous faisons des commentaires dans le code (pour expliquer par exemple qu'on se base sur telle ou telle norme)

> Chez nous, ce sont les développeurs qui doivent faire de l'archéologie dans des architectures existantes qui justifient le mieux leur design. Il y a intérêt à faire circuler les personnes entre le développement des nouveaux produits et le maintien opérationnel des anciens produits

> Le moteur du Rafale qui a été fait dans les années 80, dont le dossier a été fait avec une bonne rigueur, est plus facilement interrogeable que certains dossiers rédigés plus récemment (et pour lesquels on compte peut-être trop sur les moteurs de recherche et les nouveaux outils)

> On a éprouvé le template du dossier de justification auprès de plusieurs personnes pour valider sa pertinence (complétude, niveau de grain, facilité à retrouver l'information etc.)

- On utilise un outil qui gère les demandes de modifications (ECR (Engineering Change Request)) et les analyses d'impact réalisées par les différentes parties prenantes. La preuve formelle des analyses d'impact sont des ECN (Engineering Change Notification). La décision d'implémentation de la modification est prise par un Change Control Board (CCB).
- Nous utilisons les modèles dynamiques dans Catia pour réaliser certaines de nos analyses d'impact
- Lors d'une analyse d'impact nous sommes allés en profondeur sur les aspects mécaniques avant d'identifier tous les aspects physiques concernés et on a raté un impact thermique

10. INTEGRATION, VERIFICATION, VALIDATION & QUALIFICATION (IVVQ)

En préambule, il convient de clarifier la définition du sens donné aux termes d'intégration de vérification, de validation et de qualification dans ce document.

Le processus d'intégration correspond à l'ensemble des activités permettant d'assembler les sous-systèmes pour constituer le système en regard des interfaces définies.

Le processus de vérification regroupe l'ensemble des pratiques qui garantissent que le système est cohérent sur le plan fonctionnel et organique et qu'il prend bien en compte ses exigences. La vérification répond à la question : est-ce que le système est correctement conçu et implémenté du point de vue de l'ingénieur ? (Are we doing the system right?)

Le processus de validation regroupe l'ensemble des pratiques qui garantissent que le système est cohérent sur le plan opérationnel et qu'il prend bien en compte les usages et les besoins de ses parties prenantes. La validation répond à la question : est-ce que le système est correctement conçu et implémenté du point de vue de ses parties prenantes ? (Are we doing the right system?)

La qualification est un processus de validation spécifique qui vise à obtenir une décision positive du client acquéreur vis-à-vis de la satisfaction de certains besoins.

La certification est un processus de validation spécifique qui vise à obtenir une décision positive d'autorités actant que le système répond aux exigences règlementaires et est acceptable pour l'utilisation opérationnelle.

Les activités IVVQ sont généralement réalisées de manière itérative : au début à partir des modèles virtuels du système, puis progressivement avec le système réel. Les activités IVVQ doivent être planifiées afin d'anticiper les moyens nécessaires à leurs réalisations, tant en termes de moyens matériels (pouvant obéir à des contraintes réglementaires, sécurité et environnement) que de ressources humaines (compétence, expérience et éventuellement habilitation/qualification des personnes).

L'objectif est d'identifier les interfaces/fonctions à risque…

L'architecte est responsable de s'assurer que les preuves issues des activités d'IVVQ permettent d'atteindre le niveau de conformité exigé à chaque étape.

10.1. INTEGRER LE SYSTEME

L'essentiel

Les activités d'intégration varient en fonction de la nature et de la maturité de développement du produit et sont conduites tout au long du cycle de conception. L'intégration du système peut ainsi se faire de manière virtuelle (on n'attend pas d'avoir des pièces) afin de lever les risques au plus tôt dans le projet, ou de manière réelle.

Les modèles numériques sont donc des systèmes contributeurs forts de cette activité.

Le focus principal de l'intégration est la mise en place des interfaces.

Un des enjeux de l'intégration est lié aux problématiques de paramétrage et de configuration (essentiellement lié aux problématiques de systèmes à forte dominante logicielle, dû à la combinatoire et à l'interdépendance des paramètres).

Les écueils principaux

Parmi les principaux écueils, on notera :

- Ne pas avoir de gestion de la configuration des différents composants à intégrer et notamment la représentativité et les limitations des composants qu'on intègre
- Le manque de compétence et d'expérience des personnes impliquées dans l'intégration
- Commencer trop tardivement et en particulier ce qui est le plus risqué, ne pas faire d'intégration virtuelle
- Ne pas planifier l'intégration
- Attendre la conformité complète de l'environnement d'intégration alors qu'une partie des risques peut être levé avec un environnement partiellement conforme (cependant cela implique qu'il faudra refaire les activités une fois l'environnement conforme obtenu ce qui peut être jugé non efficient)
- N'avoir une stratégie d'intégration que fonctionnelle/logique versus une stratégie basée sur la levée des risques (safety, résilience, maintenabilité, performance…)

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- S'assurer de la disponibilité des "bonnes" ressources : moyens et personnes (compétences, expérience)
- S'aider des modèles d'architecture pour identifier les interfaces sensibles et sur lesquelles il faut se focaliser
- Décrire et séquencer de manière précise les activités d'intégration en particulier dans le cadre d'innovations. Cela permet en outre d'anticiper les ressources/moyens nécessaires et le coût de la séquence.

Témoignages

> Nos activités d'intégration sont conduites par le chef de projet système, interfaces par interfaces, en fonction d'une stratégie préalablement établie par l'équipe d'intégration de vérification et de validation, selon une logique risque/valeur. L'architecte système contribue à ces activités en particulier sur la clarification des exigences, la mise en place de la stratégie. Le découpage des tâches d'intégration privilégié est celui du scénario opérationnel.

> Notre stratégie d'intégration vise à intégrer le plus rapidement et de la manière la plus efficiente possible (le moins d'effort pour le plus de couverture possible). Pour cela nous menons une analyse de dépendance par composant. Cela nous permet de définir le plus petit scénario nous permettant de minimiser les interfaces à traiter. On grossit jusqu'à obtenir le périmètre global. On évite le bigbang !

10.2. VALIDER ET VERIFIER LE SYSTEME

L'essentiel

Tout comme pour les activités d'intégration, les activités de validation/vérification varient en fonction de la nature et de la maturité de développement du produit et sont conduites tout au long du cycle de conception. Par exemple, une revue de la satisfaction des besoins avec les parties prenantes constitue une activité de validation qui peut se conduire très tôt dans le cycle de développement.

La validation/vérification du système peut ainsi se faire de manière virtuelle à travers des simulations ou physique via des tests. La validation/vérification peut aussi se mener au travers d'activités de natures diverses : vérification par les pairs, traçabilité & revue de spécification, inspection… Certaines industries parlent d'activités IADT : Inspection, Analyse, Démonstration, Test.

Il est important de mener les activités de manière progressive pour éviter de lever les risques trop tardivement.

Le focus principal de la validation concerne la satisfaction des besoins.

Le focus principal de la vérification se porte sur le respect des exigences (fonctionnelles, organiques) et du design.

Un des enjeux de la validation/vérification est la traçabilité, du point de vue du client tout comme de l'organisation qui porte la conception et l'implémentation (pour être capable par exemple de retrouver des éléments plusieurs années après le projet).

Une validation/vérification efficiente minimise le nombre d'itérations sur la conception ou sur les activités de validation/vérification en elles-mêmes.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Le manque d'anticipation des activités de validation/vérification qui ne permet pas d'avoir les bons moyens, ou les bonnes ressources dans les temps, ou qui conduit à des changements en phase avancée du projet
- La validation finale/vérification incomplète : on ne fait pas une couverture complète des besoins/exigences soit pour des arbitrages planning/budget soit par manque de maîtrise. (Nota : ce qui est différent d'une validation/vérification étagée)
- Mener des activités de validation/vérification sans maîtriser la configuration du système validé/vérifié
- Une logique "happy path" (on ne teste que les cas nominaux et pas les cas problématiques) est une fausse économie. Réduire le scope de validation/vérification conduit souvent à des conséquences dramatiques
- On cherche parfois trop à automatiser tous les tests ce qui est coûteux et dur à maintenir. A l'inverse, ne rien automatiser induit des coûts récurrents qui deviennent conséquent sur la durée. C'est un optimum à trouver.

Les bonnes pratiques

Voici quelques bonnes pratiques à prendre en compte :

- Décrire, au plus tôt, la stratégie de validation/vérification (qu'est-ce que je vais valider/vérifier, avec quel moyen, selon quelle responsabilité… pour minimiser les risques)
- Mener les activités de validation/vérification de manière progressive
- Impliquer toutes les parties prenantes le plus tôt possible dans les actions de validation/vérification pour minimiser les risques (et notamment s'assurer que les critères de satisfaction des besoins/exigences sont définis, et que les exigences sont validables/vérifiables)
- Intégrer les activités de validation/vérification dans le planning et dans la charge
- S'assurer de la disponibilité des "bonnes" ressources au plus tôt : moyens et personnes (compétences, expérience)
- S'aider des modèles d'architecture pour identifier au plus tôt les risques à lever
- Décrire et séquencer de manière précise les activités de validation/vérification
- Anticiper certains tests de performances (même sur des environnements partiellement représentatifs) pour dérisquer au maximum
- Capitaliser les plans de test d'un projet à l'autre (sans oublier bien sûr de réfléchir lors de l'adaptation) car in fine ce sont des assets qui se prêtent bien à la capitalisation

Témoignages

- “ Nous sommes organisés de manière historique avec des moyens de validation physiques et une culture essai-erreur qui peut être coûteuse (également en temps investi dans ces activités et l’impact sur le planning) et nous tentons de nous transformer vers des moyens plus numériques
- “ Sur nos API logiciel nous ne spécifions pas forcément bien les options à embarquer (uniquement de manière implicite) ce qui nous pose des soucis tardifs (avec notamment des plages de fonctionnement non prévues) alors que nous aurions pu anticiper via des actions de validation
- “ Nous faisons des choix de design qui ne se traduisent pas en exigence et que nous devons néanmoins vérifier. On utilise Polarion dans lequel nous mettons des éléments de design et qui nous servent de support aux activités de vérification.
- “ Nous avons un jeu de procédures qui correspond à notre état de l’art en interne et nous permet par expérience de compléter la couverture des tests (en complément de la spécification)
- “ On a pour projet d’utiliser l’IA pour maintenir les tests automatiques. Les produits évoluent et il faut faire évoluer les tests en parallèle ce qui est coûteux.
- “ Nous faisons des tests sur un nombre de cycles prédéfini avec des conditions externes spécifiques et cela prend des années ce qui demande une forte anticipation
- “ Pour nous, il est intéressant d’investir dans les tests car leur durée de vie est souvent plus longue que la génération de produit. Regarder le cycle de vie des moyens de vérification/validation nous permet d’optimiser les investissements. Quand on investit X millions dans un banc de test, il y a toujours quelqu’un pour vous demander le ROI.
- “ On fait du “chaos testing”[1] (manœuvres de singe) afin de couvrir des cas qui ne sont pas forcément envisagés

10.3. QUALIFIER LE SYSTEME

L’essentiel

La qualification (acceptance en anglais) est un processus de validation spécifique qui vise à obtenir une décision positive du client acquéreur vis-à-vis de la satisfaction de certains besoins. Elle se fait généralement dans l’environnement du client.

Ce qui est défini dans le chapitre précédent est applicable avec quelques écueils et bonnes pratiques supplémentaires.

Les écueils principaux

Parmi les écueils supplémentaires, on notera :

- Ne pas négocier en avance les critères de satisfaction relatifs à la qualification
- Attendre d’avoir complètement finalisé le travail avant de montrer des résultats partiels au client. Mieux vaut avoir les retours client au plus tôt

[1] https://netflixtechblog.com/the-netflix-simian-army-16e57fbab116

Les bonnes pratiques

Voici quelques bonnes pratiques supplémentaires à envisager :

- Dérisquer la qualification par une validation préalable
- Faire de la qualification par lot qui s'étale dans le temps pour détecter au plus tôt une insatisfaction éventuelle du client (et associer ces qualifications à des paiements plus régulier)

10.4. CERTIFIER LE SYSTEME

L'essentiel

La certification est un processus de validation spécifique qui vise à obtenir une décision positive d'autorités actant que le système répond aux exigences règlementaires et est acceptable pour l'utilisation opérationnelle.

Dans l'esprit ce sont les mêmes activités que précédemment mais avec une partie prenante différente et des conséquences importantes (refus d'autorisation de mise sur le marché, de vol, de mise en service…)

Les écueils principaux

Parmi les écueils supplémentaires, on notera :

- Évolution du référentiel de certification non anticipé au lancement du projet/programme

Les bonnes pratiques

Voici quelques bonnes pratiques supplémentaires à envisager :

- Faire une veille régulière des évolutions normatives en participant aux différentes instances pertinentes
- Avoir des compétences en interne sur la déclinaison des réglementations/normes dans le design

Témoignages

> “ Nous avons des délégations de certification (via des habilitations par un organisme externe) qui nous permettent de jouer certains tests de la certification

11. CONTRIBUTION A LA GESTION DE PROJET

11.1. PARTAGE DE RESPONSABILITE ENTRE L'ARCHITECTE ET LE CHEF DE PROJET

Avant de définir la contribution de l'architecte à la gestion du projet, il convient tout d'abord de clarifier le partage de responsabilités entre le rôle de l'architecte et le rôle du chef de projet, les deux rôles pouvant être incarnées par la même personne ou des personnes distinctes.

Il existe différents cas de figure et ces différences, en fonction des entreprises, voire des différents projets au sein d'une même entreprise, sont génératrices de nombreuses confusions avec des recouvrements potentiels d'activités et de responsabilités ou à l'inverse des responsabilités non incarnées ou portées.

Pour définir les responsabilités des différents acteurs, nous emploierons ici les acronymes du RACI (Responsible, Accountable, Consulted et Informed) auquels nous donnerons le sens R = Réalisateur, A = Approbateur ou Responsable, C = Consulté, I = Informé.

A noter que ce partage de responsabilité entre l'architecte et le chef de projet doit être défini sur toute la durée du projet (et pas uniquement en phase de conception) et peut évoluer selon la phase du projet. **Il est ainsi important de rappeler que les activités et les responsabilités de l'architecte doivent être poursuivies sur l'ensemble du cycle de vie.**

A noter enfin, que sur les phases de projet ou l'activité de l'architecte est réduite, le rôle de l'architecte peut être attribué à un autre membre de l'équipe.

L'essentiel

Une distribution claire du périmètre entre le chef de projet et l'architecte est indispensable pour assurer le succès du projet et mérite de ne pas se satisfaire de l'implicite.

L'architecte est en charge à minima de définir l'architecture du système, c'est-à-dire de réaliser l'architecture boîte noire et boîte blanche de son système, de modéliser le système et les chaînes de valeur dans l'architecture, de proposer, justifier et choisir des architectures concurrentes, d'évaluer les architectures en maturité et en conformité aux besoins prioritaires, de gérer les interfaces, de conduire des analyses d'impacts, de valider et vérifier le système.

Le chef de projet quant à lui, à minima, pilote les coûts, risques et délais du projet.

En matière d'atteinte des objectifs Qualité, Performance, Coût et Délai du projet, l'architecte est responsable (« Accountable » au sens du RACI) des objectifs Qualité et Performance qui lui sont alloués (en prenant en compte les contraintes Coût et Délai), tandis que le chef de projet est responsable (« Accountable » au sens du RACI) des objectifs Coût et Délai qui lui sont alloués (en prenant en compte les contraintes Qualité et Performance).

En cas d'impossibilité d'atteindre les objectifs Qualité et Performance avec les contraintes Coûts et Délai, se pose alors la question de savoir qui arbitre.

Le chef de projet étant responsable (possiblement en délégation) du QCDP global du projet, c'est soit lui qui possède ce pouvoir d'arbitrage, soit une autorité supérieure.

Dans tous les cas il est important de bien identifier cette autorité dès le lancement du projet.

Nota :

Nous parlons ici bien de rôle et pas d'acteur. Il faut garder en tête qu'il peut y avoir des cas où :

- La personne qui porte le rôle d'architecte porte aussi les objectifs Coût et Délai (sur un sous périmètre par exemple). Cet acteur joue alors également un rôle de chef de projet sur ce sous-périmètre.
- La personne qui porte le rôle du chef de projet porte aussi les aspects Qualité et Performance (sur une phase spécifique du projet ou sur un projet de reconduction ou dit « business as usual») Cet acteur joue alors également un rôle d'architecte.

Les écueils principaux

Parmi les principaux écueils, on notera :

- Les rôles non clairement explicités : qui est responsable de quelle activité et de quel objectif ? Par qui et comment sont fait les arbitrages…
- La problématique de bande passante, c'est-à-dire de charge des acteurs qui ne peuvent pas réaliser l'ensemble des activités et responsabilités qui leur incombent ;
- La non-explicitation des résultats d'arbitrages ;

- La prise de décisions qui vont au-delà du périmètre de délégation du projet (ex : dégradation de la qualité du produit qui a un impact sur l'image de l'entreprise et/ou son business) ;
- Les confusions entre rôle et titre qui peuvent générer des ambiguïtés (ex : un architecte qui aurait un rôle de développeur sur un projet) ;
- Des chefs de projet portant historiquement la responsabilité technique des projets ce qui a peut complexifier la mise en place d'un nouveau rôle d'architecte. Souvent les chefs de projets système (même programme parfois) font des choix d'architecture sans en avoir conscience ce qui induit des risques/couts non étudiés et anticipés (par exemple : choix de sous-traitants sur des aspects financiers uniquement, sans prendre en compte des évaluations techniques).

Les bonnes pratiques

Sans exclusif, voici quelques bonnes pratiques à prendre en compte :

- Les rôles doivent être explicités au début du projet ainsi qu'à tout changement d'organisation ;
- Les arbitrages doivent être explicités sur les différentes phases projet (cela peut être juste un compte rendu ou de façon plus détaillée, un dossier de justification des choix) ;
- Il convient d'adapter l'organisation (quelle somme de responsabilité doit porter l'architecte) en fonction de la complexité du composant/sous-système ainsi que la complexité du reporting (KPI & modèles). Souvent le contexte sectoriel ou organisationnel conditionne le choix du partage de responsabilité entre l'architecte et le chef de projet ;
- En plus de la clarification des rôles, il est recommandé de mettre en place une formation sur le métier d'architecte pour les acteurs projet.

Témoignages

Nous avons compilé ici un certain nombre de verbatims de chef de projet ou d'architecte système de différentes entreprises, et qui font écho à cette distinction nécessaire des rôles :

> Pour assurer un rôle d'architecte tout au long de la vie du projet nous avons nommé un responsable d'intégration et test qui prend de facto le rôle d'architecte après la phase d'étude du projet ;

> Dans le cadre de gros projets, nous avons des responsables de lot qui sont des chefs de projet et des architectes qui ont délégations des objectifs QCDP sur leur périmètre et sont pilotés par le nœud du dessus ;

> Sur un petit projet, nous avons un responsable technique qui joue un rôle d'architecte ;

> Nous optons pour une répartition des rôles en fonction des projets et du profil des ressources disponibles au moment du lancement du projet, avec 3 rôles clefs identifiés et incarnés :
>
> - Le chef de projet système (en charge de la gestion du coût, du délai, du planning)
> - L'architecte système (en charge de la conception, définition de la solution et choix techniques répondant au besoin du client)
> - Un responsable IVVQ (concentré sur la partie aval du cycle en V). Il s'assure que ce qui est livré est en phase avec ce qui a été spécifié

> Dans les petits projets, le chef de projet peut avoir également le rôle d'architecte : il s'occupe des spécifications avec un ingénieur système, qui est alors le responsable technique. Dans de plus gros projets, un rôle de responsable technique vis-à-vis du client peut également être défini : il s'agit alors d'un architecte impliqué dans la remontée du cycle en V.

> Nous avons une organisation où les rôles sont beaucoup plus compartimentés avec :
>
> - Un chef de programme qui porte le QCDP du programme
> - Un project design authority, responsable technique sur le projet vis à vis du client
> - Un architecte système, responsable technique en interne lors des phases de conception

- Un responsable ingénierie système, responsable de conduire les activités d'ingénierie système sur le projet
- Un responsable intégration vérification et validation en charge des activités d'intégration et de V&V

“ Nous avons une organisation où la taille et le délai important des projets nécessitent un partage de responsabilité technique entre des responsables techniques et un architecte. Les responsables techniques sont chargés de lots (correspondant à des lots d'études métier sur un périmètre produit). Le responsable de lot a la charge de la réalisation des études techniques sur son lot (ex : notes de dimensionnement mécanique) et réalise le suivi budgétaire de ses activités. Ce sont des appuis pour l'architecte. L'architecte, lui, porte la responsabilité de la performance du produit, et contrairement au responsable technique, il n'a pas d'aspect budgétaire à gérer.

A noter que cette répartition a été pensée pour pallier des problématiques historiques où les responsabilités solution, budget et planning étaient portées par un même rôle sans que cela ne soit tenable : il y avait soit un traitement des problématiques techniques au détriment des aspects coût et délai, soit au contraire un focus coût et délai au détriment de la technique. La mise en place d'une responsabilité séparée sur la performance du produit a permis de mettre en place un équilibre plus sain entre ces différents aspects à prendre en compte sur les projets et un renforcement de la préoccupation de la conformité du produit final.

11.2. CONTRIBUTION DE L'ARCHITECTE AUX ACTIVITES PORTEES PAR LE CHEF DE PROJET

En support du chef de projet, l'architecte est amené (en fonction du split de responsabilité) à :

- Identifier et quantifier les risques techniques pour le chef de projet
- Valider le séquencement des activités et la faisabilité du planning
- Aider le chef de projet au jalonnement du projet
- Apporter un support dans les échanges avec les autorités réglementaires / organismes de certification et de manière plus générale supporter toutes discussions techniques (avec les sous-traitants,
- Contribuer à la rédaction des plans (plan de gestion des exigences…)

Les écueils principaux

- Manque de collaboration : le chef de projet ne fait pas contribuer l'architecte aux activités projet ayant une dimension technique
- Trop solliciter l'architecte sur la contribution des activités projets et ne plus lui laisser de temps pour réaliser ses missions principales
- Mauvaise compréhension du chef de projet de la mission et du rôle de l'architecte
- Confondre la fiche de poste avec le ou les rôles portés sur un projet donné : certains chefs de projet ayant un background technique

Les bonnes pratiques

- Bien réussir à contextualiser les rôles au sein d'un projet donné

12. ASSURER LA COORDINATION TECHNIQUE DU PROJET

L'architecte assure, en délégation du chef de projet, la coordination technique du projet de conception de son système, c'est à dire qu'il :

- Anime l'équipe d'architecture telle que définie en introduction du chapitre 2 (architectures des différents sous-systèmes et composants, gestionnaire des interfaces, requirement manager, Modeleur…)
- Organise et cadence les revues systèmes du projet nécessaire au dérisquage des passages de jalon projet
- Partage à tous les acteurs du projet la même vision de l'architecture (client / projet / systèmes)

CONCLUSION

Le rôle de l'architecte est essentiel pour la gestion de la complexité des systèmes modernes. En tant que garant de la cohérence et de l'intégrité des systèmes, l'architecte doit naviguer à travers de nombreuses responsabilités. Cela inclut la capture des besoins des clients, la coordination technique des projets, la validation des choix techniques et la gestion des interfaces. Ces défis nécessitent une expertise approfondie et une vision globale.

La collaboration avec les chefs de projet, les équipes techniques et les parties prenantes est cruciale pour assurer le succès des projets. L'architecte doit posséder non seulement des compétences techniques mais aussi des compétences relationnelles, telles que la facilitation, le leadership et une communication claire et accessible. Ces qualités permettent de créer des consensus et d'assurer l'adhésion de toutes les parties prenantes au projet.

En conclusion, l'architecte n'est pas seulement un expert technique mais aussi un facilitateur de collaboration, un gestionnaire de complexité et un acteur clé dans la transformation digitale des organisations. Son rôle est fondamental pour assurer la conformité technique, la satisfaction des besoins des parties prenantes et, in fine, la compétitivité des entreprises dans un environnement toujours plus complexe et exigeant.

A propos du Cercle CESAM

CESAM Community est développée par l'Association CESAMES depuis 2010. Son but est de partager les bonnes pratiques d'Architecture d'Entreprise et d'Architecture Système. À travers la certification CESAM, elle atteste la capacité des acteurs à mettre en œuvre ces bonnes pratiques. L'association CESAMES a ainsi construit la plus grande communauté autour du MBSE (aujourd'hui, plus de 8500 Professionnels sont formés ou certifiés à la méthode CESAM). Elle a le soutien de grands partenaires qu'ils soient académiques, institutionnels et professionnels.

Le Cercle CESAM est un groupe de travail qui a pour but de développer et de partager un standard international pragmatique d'architecture système et de le décliner par grands domaines industriels. Pour le bénéfice business de ses membres.

Aujourd'hui le Cercle compte une quinzaine de membres dont ITER, Sagemcom, Safran (SHE, SAE, SED), Dassault Systèmes, Idemia, Airbus, Somfy.

Les 2 axes de travail du Cercle sont : Méthode et outils (formaliser et partager des applications de la méthode CESAM par grands domaines sectoriels (études de cas, bonnes pratiques, modalités d'outillage…)) et Professionnalisation (contribuer à la professionnalisation du métier d'architecte système pour valoriser les architectes au sein de leurs organisations.)

Le Cercle travaille actuellement sur le livre blanc « le rôle de l'architecte » qui sera publié courant 2023.

Membres du Cercle qui ont contribué à cette publication

Anthony Ferrer, System Architect (MBSE), SAGEMCOM

Cécile Beyssac, Architecte système principal & Responsable de l'ACADEMY, CESAMES

Jean-Marc Cherel, Chief Engineer, IDEMIA

Nicolas Gueit, Model-Based Systems Engineering Framework Referent, SAFRAN LANDING SYSTEMS

Pierre Colin, Physical and Functional Integration division Head, ITER

Rahid Djafri, System Architecte (MBSE), SAGEMCOM

Regis Vincent, Systems Engineering Senior Expert / Lean Sigma Manager, SAFRAN HELICOPTER ENGINES

Chief System Architect, SOMFY

Copyright

Ce travail est soumis au droit d'auteur. Tous les droits sont réservés à C.E.S.A.M.E.S., qu'il s'agisse de tout ou partie du matériel, notamment les droits de traduction, de réimpression, de réutilisation des illustrations, de récitation, de diffusion, de reproduction sur microfilms ou de toute autre manière matérielle, de transmission ou de stockage et récupération, adaptation électronique, logiciel informatique, ou par une méthodologie similaire ou différente actuellement connue ou développée ultérieurement.

L'utilisation de noms descriptifs généraux, de noms déposés, de marques de commerce, de marques de service, etc. dans cette publication n'implique pas, même en l'absence d'une mention spécifique, que ces noms sont exemptés des lois et règlements de protection pertinents et donc libres d'utilisation générale.

Les autorisations peuvent être demandées directement auprès de CESAM Community.

Publisher

CESAM Community est gérée C.E.S.A.M.E.S

71 rue de Mirosmenil – 75008 Paris – France

email: contact@cesam.community

Website: https://cesam.community/fr/

Photo credit: Fauxels (PEXELS)