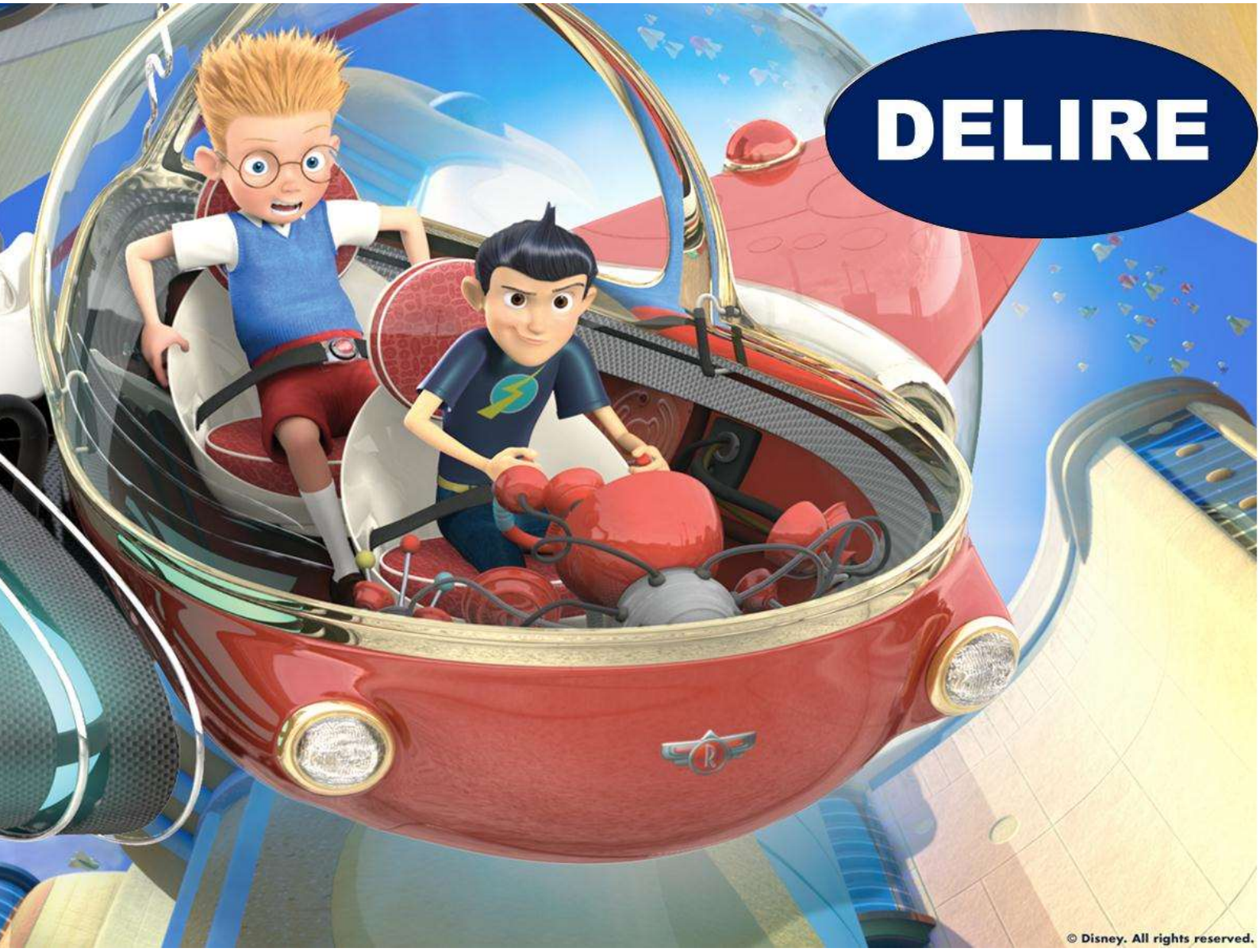

Le projet **DELIRE**

Développement par **E**quipe de **L**ivrables **I**nformatiques et **R**éalisation **E**ncadrée

PR5 – La fiche de livraison

Les différents composants de votre architecture

Nous allons commencer par une vision réductrice de votre application. Celle-ci est composée de 3 composants

1. DATA
2. TRAITEMENT
3. DIALOGUE

DATA est le composant qui encapsule la base de données. Faisons l'hypothèse qu'il existe 3 types de données : D1, D2 et D3 gérée dans la base. Alors, de façon très grossière et simplificatrice, pour chaque type de base, DATA exporte un service pour identifier le nombre d'objet, un service pour lire les données d'un objet, un service pour modifier ledit objet. Soit, au total 9 services

1. NumberD1, ReadD1, WriteD1
2. NumberD2, ReadD2, WriteD2
3. NumberD3, ReadD3, WriteD3

Ces services sont de services qui ne seront pas offerts à l'extérieur.

TRAITEMENT est le composant qui offre les services de manipulation des objets en mémoire. Pour ce faire chaque service doit aller chercher les données d'un objet, les transformer en mémoire et les re-stocker en base. Il s'appuie donc sur les services de gestion de données NumberDi, ReadDi, WriteDi. On peut faire l'hypothèse que TRAITEMENT exporte 4 services de traitement : TraitT1, TraitT2, TraitT3, TraitT4. Ces services pourront potentiellement être commercialisés

DIALOGUE est le composant qui offre les commandes interactives de dialogue. Par exemple :.

1. Une commande pour se connecter : Connect
2. Une commande pour afficher le contenu de la base visible à l'utilisateur : Display
3. Une commande de création et une commande de modification de chaque objet
    a. CreateD1, ModifyD1
    b. CreateD2, ModifyD2
    c. CreateD3, ModifyD3
4. Une commande pour se déconnecter : Exit

Comment, quand et que livre-t-on

En règle générale, dans les grandes entreprises éditrices de logiciels, les équipes de développement et les équipes de test fonctionnels dont distincts. A titre d'exemple, DS fait faire ces tests fonctionnels par des équipes en Inde, où le salaire est nettement plus faible qu'en France.

L'objectif va être de fournir els équipes de tests en composants pour pouvoir commencer à avancer dans les tests fonctionnels. Dans ce cadre, les composants d'infrastructure sont livrés très tôt dans le cycle de vie de la Release, les composants applicatifs étant livrés au fur et à mesure de leur disponibilité.

Bien entendu, cette règle ne s'applique pas pour le projet GALERE. On peut néanmoins tenter de le simuler.

La priorité est d'avoir le composant d'encapsulation de la base de données DATA. Et là il n'est pas question de livrer un composant pour lequel les services relatifs à un type d'objet ne soient pas disponibles.
Presque : on peut livrer le composant même si des règles d'intégrité n'ont pas encore été développés : ainsi, si vous avez prévu que deux objets ne peuvent avoir le même identifiant user, vous pouvez néanmoins livrer le composant même si cette vérification n'est pas encore implémentée. Il faudra simplement le prendre en compte dans l'ordonnancement des tests fonctionnels.
N'oubliez pas, il est nécessaire d'accompagner le composant DATA lors de sa première livraison d'un script qui va initier la base de données :

1. Créer les tables
2. Initier les compteurs des objets à 0
3. Créer un rôle bootstrap ADMIN-ADMIN qui va permettre de créer les autres utilisateurs dans la base.

Une commande permettant de modifier le password ne peut nuire.

Ensuite, on attend la livraison du composant TRAITEMENT
Il n'est pas nécessaire que l'ensemble des services aient été implémentés. Par contre, il faudra livrer une implémentation par défaut, un beau carton avec un superbe message « SERVICE TraitT4 ISN'T AVAILABLE YET » si d'aventure le service TraitT4 était appelé trop tôt par un composant de dialogue.
On peut également envisager des étapes où des services n'ont une implémentation que simplifiée, avec tous les cas non encore pris en compte.
Là aussi, dans les deux cas, il faudra simplement le prendre en compte dans l'ordonnancement des tests fonctionnels.

Enfin, on peut permettre la livraison du composant DIALOGUE, même si toutes les commandes ne sont pas encore disponibles. Bien entendu, il faudra le prendre en compte dans l'ordonnancement des tests fonctionnels.

La fiche de livraison d'un composant
Lorsque le développeur fait la promotion d'un de ses composants, celui-ci est accompagné par une fiche de livraison

Nom du composant
Responsable du développement
Date de début du développement
Le composant est-il finalisé ou non. Si non, quels sont les services incomplets, non implémentés
Quel est, du point de vue du responsable, le %age de développement réalisé
Date de fin du développement du composant

Des modifications d'interfaces publiques (modifications de signatures, ajout de services publics) ont-ils été nécessaires
Des modifications d'architecture ont-elles été nécessaires
Des services supplémentaires ont-ils été nécessaires

Temps passé cumulé en développement proprement dit (c'est-à-dire avant d'exécuter les Unit tests
Temps passé cumulé en exécution des Unit tests

Temps passé cumulé en correction d'erreurs identifiées en Unit tests ou en relecture de code
Le cout total (compte tenu du %age identifié) est- il conforme aux prévisions

Nombre de lignes de code
Nombre de lignes de commentaire

Les Unit test ont-ils tous été exécutés
Les résultats obtenus sont-ils conformes aux prévisions
Nombre d'erreurs trouvées en Unit tests ou en relecture de code.
Toutes les erreurs trouvées en Unit tests ont-elles été corrigées

Les temps d'exécution ont-ils été vérifiés
Sont-ils conformes aux attentes définies dans le Plan Qualité

Plus toutes les informations qui vous sembleront pertinentes…

L'objectif de cette fiche de livraison est double

1. D'un part permettre à ceux dont c'est la mission (chef de projet, responsable Qualité, responsable Planning) de pouvoir faire le suivi du projet (voir document PR6 – Métriques et suivi). Chaque information doit donc être mesurée (parce que mesurable) et comparée avec la prévision. Sinon, ce ne serait pas un indicateur.,
2. D'autre part accumuler des informations chiffrées sur la phase de réalisation qui vont vous permettre ensuite d'en dégager des ordres de grandeur qui enrichiront vos savoir faire

Surtout pas d'informations fausses ou incomplètes. De façon à ne pas avoir à réinventer ces informations, pensez à les noter chaque jour en fin de travail sur votre composant. .

Refaire les tableaux de bord

Nous allons traiter ici la remise à jour du planning, et du budget

Lorsque vous avez conçu le planning de la phase de réalisation, vous avez fait des estimations pour la charge (nombre d'heures*ingénieur) de chaque tâche. En général en vous appuyant sur votre expérience, de temps en temps au petit bonheur la chance.

Maintenant que vous êtes en phase de réalisation, vous commencez à avoir des informations plus précises. Dans ce qui suit, je suppose que les tâches qui seront réalisées sont celles qui ont été planifiées. .

Une tâche peut avoir 3 statuts :

1. Réalisée
2. En cours de réalisation
3. No encore démarrée

Si la tâche est terminée, vous avez exactement combien elle vous a couté, à quel moment elle a démarré et quand elle a fini. Informations que vous pouvez donc d'ores et déjà mettre à jour dans le planning

Si la tâche n'a pas encore commencé, le mieux est de conserver les hypothèses qui ont permis de construire le planning initial

Si la tâche est en cours de réalisation, vous connaissez combien elle a déjà couté, quel pourcentage des objectifs a été atteint : une règle de 3 basique vous permet d'extrapoler ce qu'elle va couter. Vous connaissez également à quel moment elle a commencé, vous pouvez donc affiner votre planning.

Bref, vous pouvez par petites touches voir se dessiner le délai et le coût que prendra à terme la phase de réalisation.