Document d'architecture

Date : 27/11/2005

Version : 1.0

Responsable du document : Kamil Guenatri

Historique des révisions

Date	Version	Description	Auteur
27/11/2005	1.0	Création du document	Kamil Guenatri

Table des matières :

1. Introduction

1.1 Objectif

Le document d'architecture du logiciel fournit une vue de haut niveau de la structure technique du logiciel 2DB. Il sert de moyen de communication entre l'architecte et les autres membres du projet sur les choix conceptuels et techniques.

1.2 Portée

Ce document est destiné aux membres de l'équipe de développement, et plus particulièrement aux architectes, aux développeurs et aux testeurs.

1.3 Références

- Document Vision
- Modèle des cas d'utilisation
- Glossaire

2. Objectifs du logiciel

2.1 Contexte

Le logiciel 2DB doit permettre au chef de projet de consulter, évaluer et d'alerter sur l'avancement des activités d'un projet de développement logiciel à partir de mesures effectués . L'interprétation de ces mesures peu conduire a l'amélioration du processus, la prise de décision et la réalisation de bilans de projet

2.2 Besoins fonctionnels

Les cas d'utilisations retenus comme étant les plus importants et/ou les plus critiques pour l'architecture sont :

- CU01 : importer les données d'un projet.
- CU08 : mettre en forme l'information
- CU09 : consolider le projet
- CU11 : archiver les données d'un projet

2.3 Besoins non fonctionnels

Les qualités et contraintes retenus comme étant les plus importantes à l'impact sur l'architecture sont :

- Langues : internationalisation du logiciel
- Performance : l'outil doit avoir des temps de réponse raisonnable (pas plus de 3 secondes)

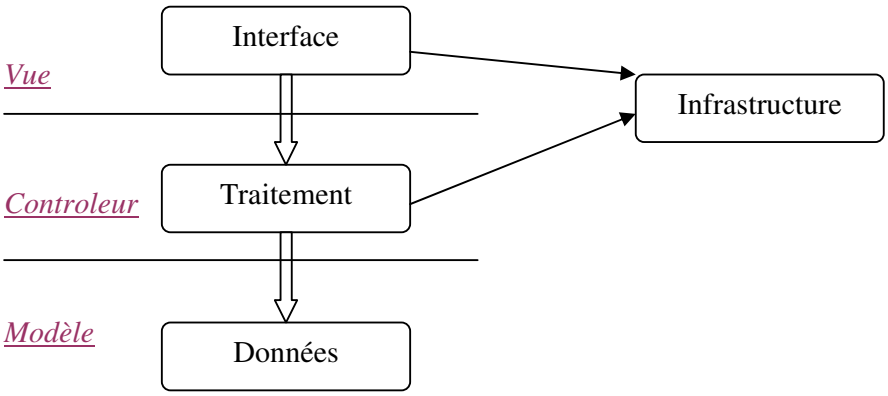
3. Structure de l'architecture

3.1 Vue des couches

Le modèle *Model-View-Controller (MVC)* sépare la modélisation du domaine, la présentation et les actions reposant sur l'entrée utilisateur en trois classes distinctes :

- **Modèle**. Le modèle gère le comportement et les données du domaine d'application, répond aux demandes d'informations sur son état (souvent issues de la vue) ainsi qu'aux instructions de changement d'état (souvent issues du contrôleur).
- **Vue**. La vue gère l'affichage des informations.
- **Contrôleur**. Le contrôleur interprète les entrées clavier et souris de l'utilisateur et informe le modèle et/ou la vue des changements nécessaires.

La figure 1 présente les relations structurelles entre les objets qui constituent l'organisation des couches du logiciel .



3.1.1.Couche Interface

Cette couche a pour rôle de fournir une interface au chef de projet pour qu'il puisse utiliser le logiciel. Elle va interagir avec la couche CONTROLEUR. Cette interaction va se traduire par un échange de données en fonction des évènements déclenchés par le collaborateur.

La couche INTERFACE gère les aspects graphiques du logiciel, c'est le niveau le plus haut de l'architecture et c'est la seule qui va interagir avec l'utilisateur.

3.1.2.Couche Traitement

Cette couche a pour rôle de piloter l'application en fonction des cas d'utilisation tout en respectant les règles de cohérence de l'outil, elle permet aussi le chargement des données XML . Cette couche agit avec la couche MODELE, afin d'effectuer les traitements sur les objets métier de l'application.

3.1.3.Couche Données

Cette couche a pour rôle de gérer les objets métiers qui sont les composants élémentaires du logiciel. Elle enregistre les modifications des objets métiers demandés par la couche CONTROLE. Cette couche est régie par les mesures que nous fournit l'outil de workflow PSI .A noter que les données seront stockées dans des fichiers ce qui permet de ne pas utiliser de base de données

3.1.4.Couche Infrastructure

Cette couche a pour rôle de gérer deux types de composants :

- Les composants utiles à l'application et éventuellement réutilisables pour le développement d'une autre application.
- Les composants réutilisés par notre application.

Les composants appartenant à la couche INFRASTRUCTURE seront utilisés par la couche INTERFACE et par la couche CONTROLE. Les composants de la couche MODELE étant spécifiques au logiciel 2DB, ils n'auront aucun lien avec la couche INFRASTRUCTURE.

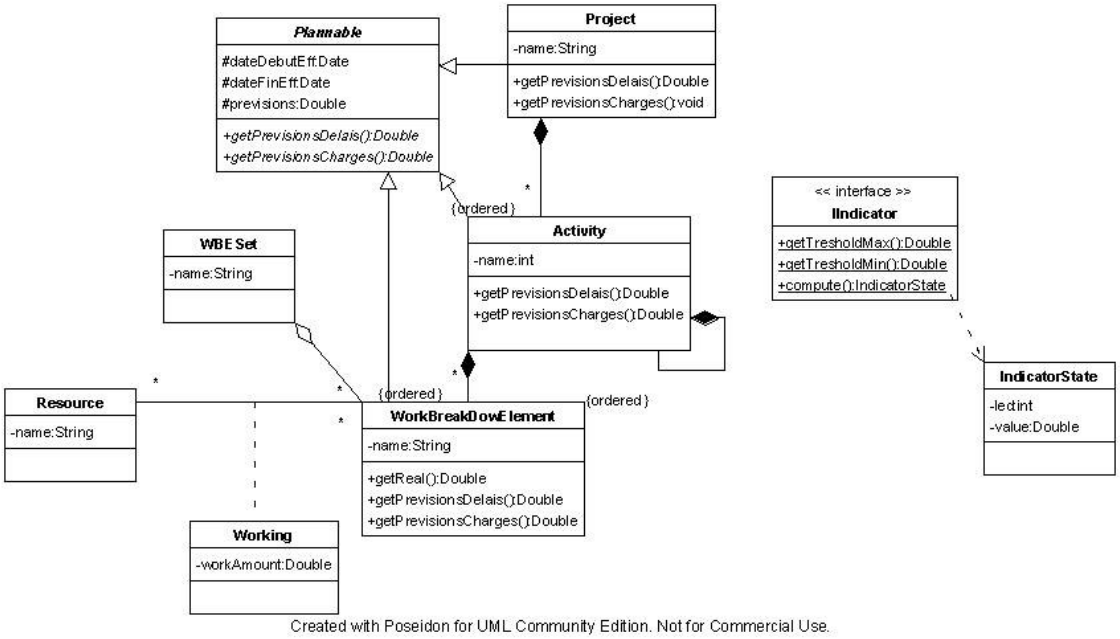
3.2 Sous-systèmes et paquetages

A definir

3.3 Interfaces

L'interface graphique a été fixée à l'itération2 via la maquette éxécutable, la réalisation de cette dernière nous a permis dans un premier temps de modeliser graphiquement les fonctionalités de l'outil et de revoir la structuration des principaux menus avant l'implémentation de l'IHM

4. Modèle des classes d'analyse



5. Caractéristiques de l'architecture

5.1 Avantages

- L'utilisation d'une architecture en couche de type INTERFACE / CONTROLEUR / MODELE permet de décomposer de façon claire la réalisation d'un cas d'utilisation. Ce système permet de déléguer à chaque couche une responsabilité lors de l'utilisation d'une fonctionnalité du logiciel. Chacune d'elles communiquera avec une ou plusieurs couches sous-jacentes.
- Eliminer le risque du risque R08 : «Performance et qualité prévue non atteinte » qui aurait été important si l'application était orienté client-serveur
- Facilité d'utilisation et d'installation du logiciel du fait d'utiliser des fichiers XML pour les données

5.2 Inconvénients

- Les développeurs peuvent avoir des problèmes concernant la localisation de certaines structures de données, ou les accès entre composants de couches différentes.