

MASTER 2 – ISIDIS

Projet de synthèse

V.A.L.

**Spécifications Techniques**

Sommaire

[Sommaire 2](#_Toc347279358)

[I. Objet du document 3](#_Toc347279359)

[II. Définitions, Sigles et Abréviations 3](#_Toc347279360)

[III. Présentation fonctionnelle générale 4](#_Toc347279361)

[1) Objectif de l’application 4](#_Toc347279362)

[2) Diagramme de cas d’utilisation 4](#_Toc347279363)

[3) Diagramme de classes métier 5](#_Toc347279364)

[4) Diagrammes de la conception détaillée 5](#_Toc347279365)

[A. Diagramme de composants 5](#_Toc347279366)

[B. Diagramme de déploiement 6](#_Toc347279367)

[C. Use Case 1 : Collecter les informations terrains 6](#_Toc347279368)

[D. Use Case 2 : Prioriser message 10](#_Toc347279369)

1. Objet du document

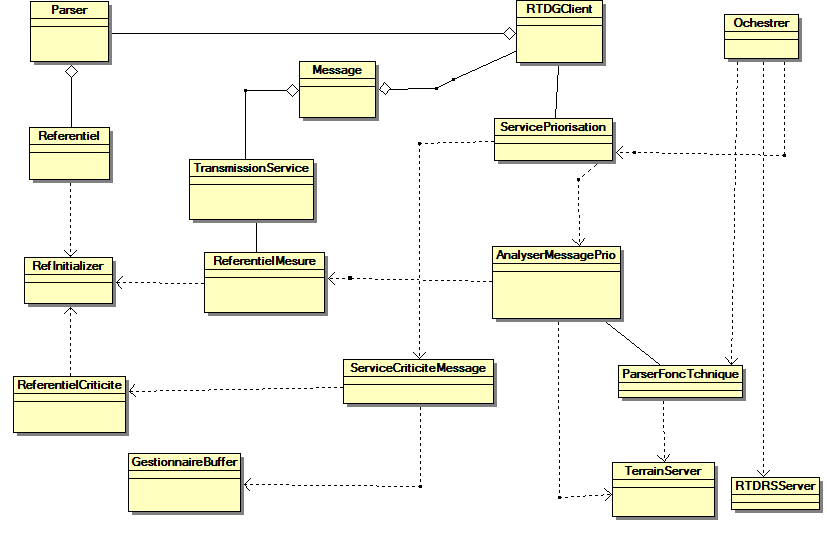
La spécification technique de besoin est un document destiné aux développeurs et qui permet de spécifier en détail les exigences fonctionnelles déjà décrites sur le document des spécifications fonctionnelles, il présente la conception détaillée réalisée dans le cadre du projet VAL.

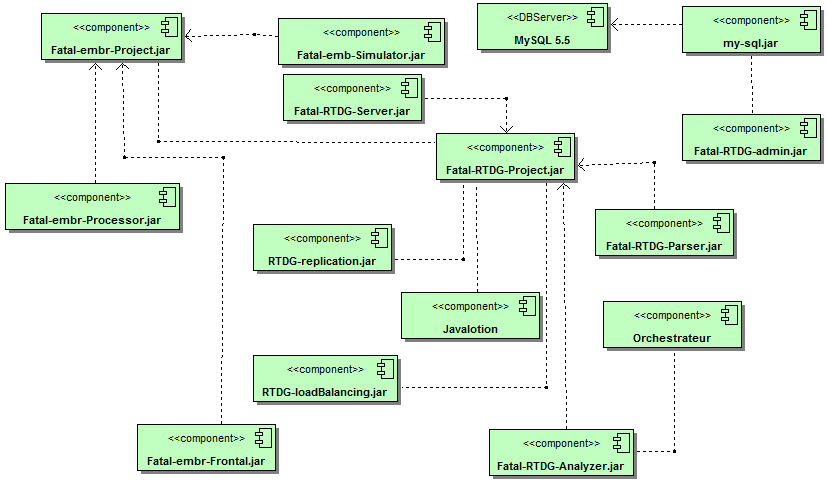
1. Définitions, Sigles et Abréviations

|  |  |
| --- | --- |
| Nom | Description |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

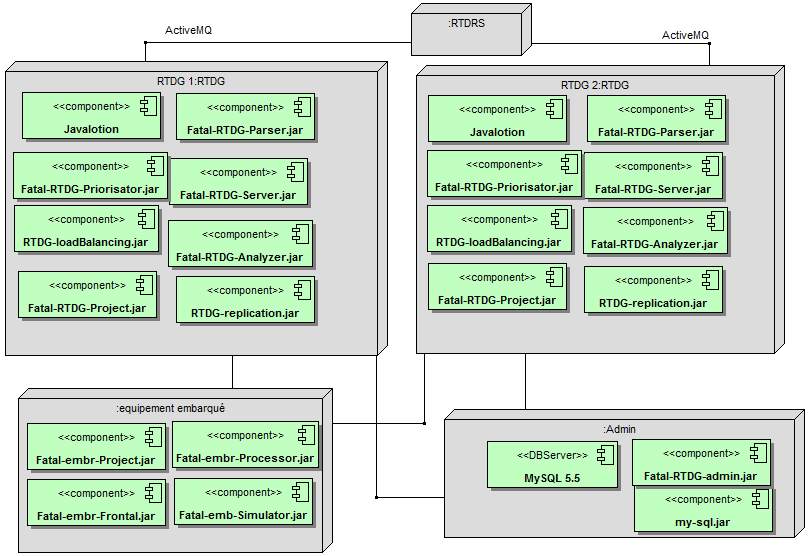
1. Présentation fonctionnelle générale
2. Objectif de l’application

Le projet a pour objectif la réalisation d’une application destinée à une entreprise de transports en commun exploitant un réseau ferré et des équipements automatiques assurant le transport de passagers sur un secteur géographique à définir, en collaboration avec l’équipe du centre de contrôle, notre choix s’est porté sur le réseau de la ville de Clermont-Ferrand, ainsi que l’étendue de son infrastructure (quais, couloirs, tunnels, rames, appareils de détection .. etc). La solution logicielle doit être sûre et ce en l’utilisant via un environnement de simulation dynamique, tout en étant ouverte et évolutive pour être implantée dans toute entreprise du secteur.

1. Diagramme de cas d’utilisation
2. Diagramme de classes métier
3. Diagrammes de la conception détaillée
4. Diagramme de composants



1. Diagramme de déploiement



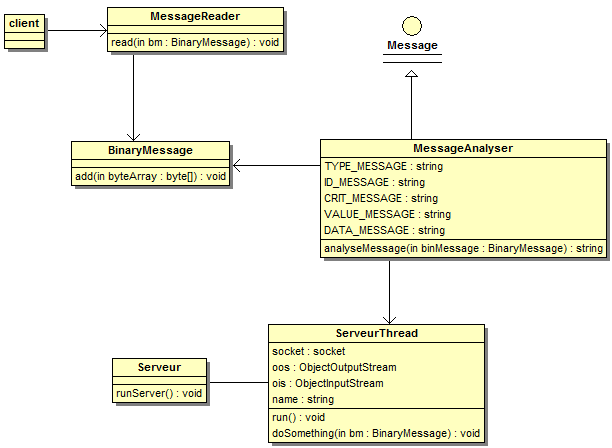
1. Use Case 1 : Collecter les informations terrains

* US : Définir la structure des messages

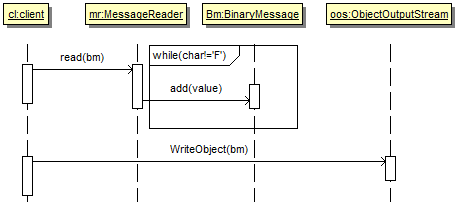
Le message qui est envoyé du composant embarqué vers le composant RTDG est sous format binaire et où sont incluses toutes les informations nécessaires. Quand le RTDG reçoit cette trame, il la décortique pour en sortir les informations et les traduire en chaines de caractères pour pouvoir y appliquer les contre-mesures adéquates, si besoin est, ou encore détecter la position du train.

Cette trame englobe un certain nombre d’informations détaillés comme suit :

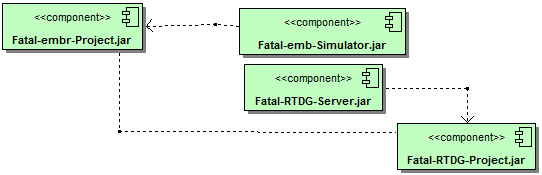
* Source : 2 bits qui nous disent si le message vient d’une station, d’une rame, de rails ou d’un tunnel.
* Id\_source : 8 bits représentant l’id du composant qui nous envoie l’information. Chaque rail, rame, station ou tunnel possède un id qui lui correspond.
* Sensor Identification : 10 bits pour représenter le capteur qui envoie l’information. Les 4 premiers bits représentent le type du capteur, par exemple si c’est un capteur de température, d’oxygène ou d’ouverture d’une porte. Les 6 bits qui suivent représentent l’id de ce capteur qui envoie l’information.
* Type : 2 bits pour représenter le type de l’information qu’on va recevoir, par exemple pour une porte, la donnée peut être booléenne pour dire que la porte est soit fermée soit ouverte, ou bien un capteur qui envoie des informations numériques tel le capteur de température.
* Intelligent sensor : 2 bits servant à dire si le capteur est intelligent ou pas, les capteurs intelligents sont ceux qui ne nous envoient que les messages critiques, les autres sont neutres et c’est à nous de définir si le message est critique ou pas.
* Data : 10 bits de données qui représentent la donnée que nous envoie le capteur.
* US : Création du protocole de communication
* Diagramme de classes de la conception détaillée

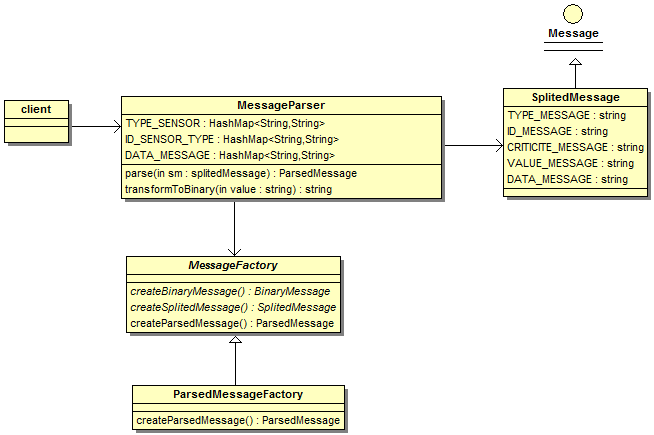
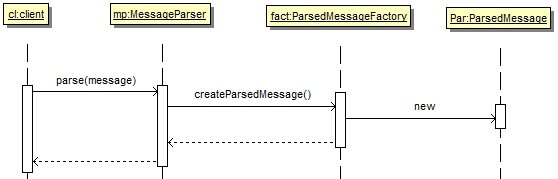


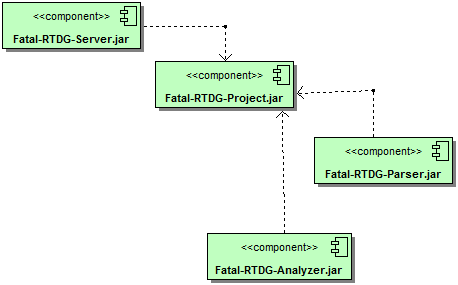
* Diagramme de séquences de la conception détaillée



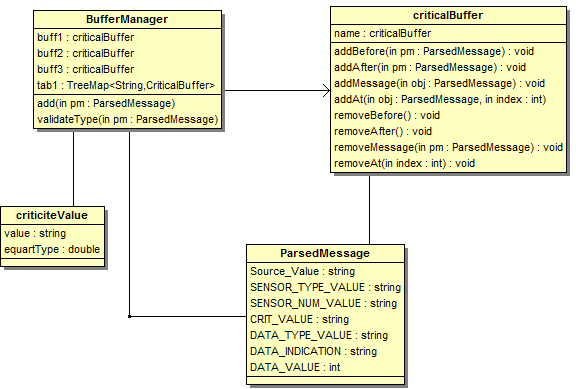
* Diagramme de composants de la conception détaillée

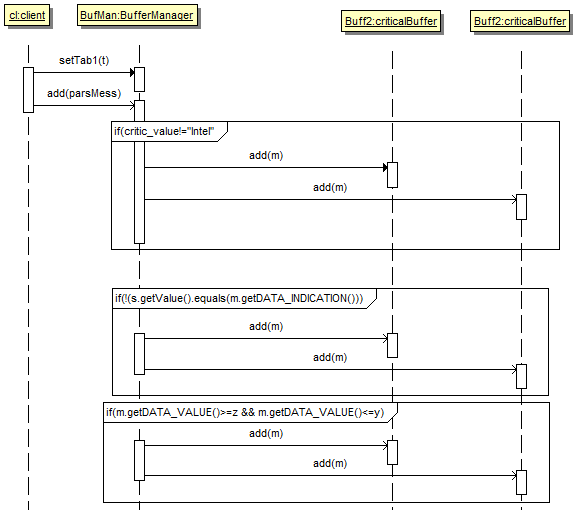


* US : Interpréter message reçu depuis un composant embarqué (Parser)
* Diagramme de classes de la conception détaillée
* Diagramme de séquences de la conception détaillée
* Diagramme de composants de la conception détaillée

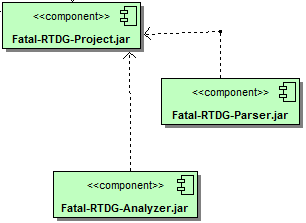


1. Use Case 2 : Prioriser message

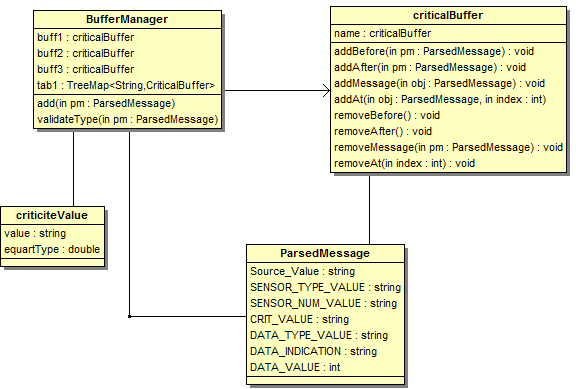
* US : Créer les différentes structures selon la criticité du message
* US : Affecter les messages non critiques
* Diagramme de classes de la conception détaillée
* Diagramme de séquences de la conception détaillée



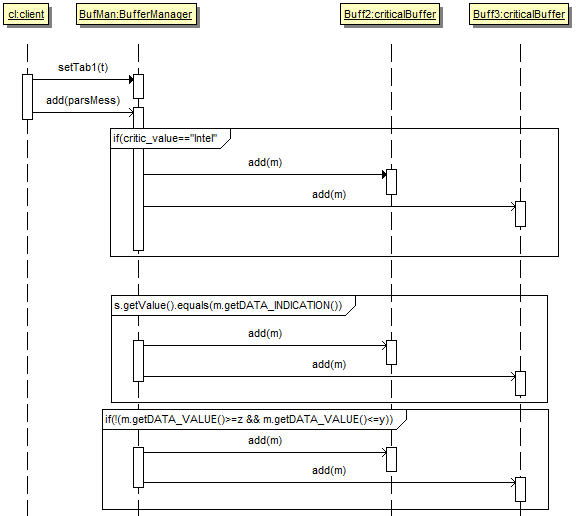
* Diagramme de composants de la conception détaillée



* US : Création des référentiels
* US : Affecter les messages critiques
* Diagramme de classes de la conception détaillée



* Diagramme de séquences de la conception détaillée



* Diagramme de composants de la conception détaillée

