% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% %

% CDC\_FancyCron.tex for FancyCron Project specification %

% Made by: Eugène NGONTANG <ngonta\_e@epitech.net> %

% engontan@bouyguestelecom.fr %

% %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

\documentclass{bouygues-fr}

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% %

% Prologue %

% %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

\begin{document}

\title{\texttt{FancyCron}}

\subtitle{Cahier de charges Version 1.0.1}

\member{Eugène NGONTANG}{ngonta\_e@epitech.net}

\member{DSI/DOQS/PFI Bougues Telecom}{engontan@bouyguestelecom.fr}

%%\newpage

\summary

{

FancyCron est un logiciel de planification et d'exécution de tâches, pour système Linux et Windows. Le projet est né d'un besoin des administrateurs système à la DSI FAI de Bouygues Telecom. Ces derniers se servent du gestionnaire de tâches Linux Cron, pour l'automatisation des services réguliers sur l'ensemble des serveurs de leurs infrastructures. Ils se trouvent limités dans le suivi de ces tâches, notamment par le manque d'informations pertinentes sur le déroulement de bout en bout de l'exécution des tâches.

FancyCron viendra donc répondre à une problématique de planning production, de reporting et de gestion centralisée des tâches sur nos environnements techniques.

Ce document constitue le cahier des charges destiné à l’équipe de développement du projet, pour prendre en compte les besoins liés à sa réalisation et aux destinataires finaux.

Il décrit également la manière dont l’équipe prendra en charge l’ensemble des phases du projet (analyse, conception, développement, documentation), les paramétrages, la reprise des données ainsi qu’un ensemble de tutoriels et de support d’utilisation du logiciel. Nous aborderons dans un premier temps le contexte général du projet et les parties impliquées. Ensuite nous présenterons le projet d’un point de vue fonctionnel, puis nous décrirons les prestations attendues de l’équipe de développement, et nous terminerons par les modalités de réalisation du projet.

}

\maketitle

\tableofcontents

\renewcommand{\labelitemi}{$\bullet$}

\renewcommand{\labelitemii}{$\circ$}

\newpage

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% %

% Généralités %

% %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

\chapter{Généralités}

\section{Contexte}

Dans l'optique d'augmenter sa productivité, tout en réduisant le temps de reprise de service après incident, l'équipe Socle FAI de la DSI Bouygues Telecom doit faire face à l'augmentation des coûts unitaires de l'ensemble de ses activités. Une remise en cause profonde des méthodes de travail s'impose, parmi lesquelles l'automatisation des services réguliers.

En effet les administrateurs système du SI FAI disposent pour leurs environnements techniques, d'une bonne gamme de serveurs tournant principalement sous système Unix/Linux.

Le nombre d'applications et services installés sur ces serveurs est innombrable, et pour des contraintes SLA, ces services doivent être démarrés à des périodes précises, et fonctionner en permanence pour certains. Cependant il serait pénible pour les ingénieurs système de procéder chaque fois au démarrage et à la mise en service manuels, d'où le besoin crucial d'un outil permettant le paramétrage et lancement automatique de ces taches.

Pour ce faire, nos administrateurs systèmes ont opté pour le logiciel Cron. Cron est le nom d'un programme qui permet aux utilisateurs des systèmes Unix/Linux d'exécuter automatiquement des scripts, des commandes ou des logiciels à une date et une heure spécifiées à l'avance, ou selon un cycle défini à l'avance.

C'est grâce à ce démon que se fait la gestion programmée et continue des services sur l'ensemble de nos serveurs.

\vspace{20pt}

\section{La problématique}

Bien que l'utilisation des crontab soit une pratique assez populaire et efficace pour l'exécution des tâches de fond, cette dernière peut s'avérer limitée dans certaines circonstances. En effet:

\begin{itemize}

\item l'utilisation des crontab nécessite un minimum de connaissances en administration système.

\item avec cron, si le système est éteint au moment où la tâche a été planifiée, elle ne s'effectuera pas cette fois-ci, et il faudra attendre l'occurrence suivante pour voir la tâche s'effectuer. Il n'est donc pas possible de rattraper des tâches avec cron, par exemple vérifier pour chaque tâche si elle a été lancée dans les \texttt{n} derniers jours, \texttt{n} étant la périodicité définie pour cette tâche. Ce qui est particulièrement gênant pour les administrateurs du SI FAI de Bouygues Telecom, qui se trouvent au moins une fois par semaine face à cette situation.

\item une autre difficulté de l'utilisation des crontab demeure d'une part dans la verbosité des fichiers de configuration dans lesquelles sont très souvent renseignés les chemins en dur des binaires, et d'autre part de la non granularité, voire l'absence d'informations sur la supervision des tâches planifiées. Il n'existe aucun moyen prédéfini avec cron, de savoir à quel moment et dans combien de temps doit s'exécuter une tâche.

\item par ailleurs, cron ne permet en rien de savoir si une tâche s'est terminée avec succès ou erreur, et le cas échéant de récupérer la sortie d'erreur générée. C'est à l'administrateur d'user des moyens techniques dont il dispose, pour avoir toutes ces données, ce qui peut s'avérer des fois approximatif et fastidieux. Le fameux /dev/null vers lequel sont souvent redirigés les sorties des scripts exécutés par cron n'est rien d'autre qu'un trash dont on ne peut voir le contenu.

\end{itemize}

Ces quelques points suffisent pour justifier la nécessité de l'optimisation de la gestion des tâches de fond sur nos serveurs.

\vspace{20pt}

\section{Le besoin}

L'équipe socle FAI de la DSI Bouygues Telecom désire aujourd'hui avoir un outil lui permettant de gérer et planifier les tâches et services, et qui sied le mieux aux méthodes de travail internes, aux procédures et contraintes de mise en production au sein de l'équipe. Ainsi les principales fonctionnalités visées dans cette expression de besoin sont les suivantes :

\begin{itemize}

\item la possibilité d'activer/désactiver une crontab, et de savoir quels sont les tâches en cours d'exécution liées à la crontab

\item la possibilité de décaler l'exécution d'une crontab d'un certain temps (1h plus tôt ou tard), sans modifié le cycle normale d'exécution de la tâche (job)

\item pouvoir tracer l'exécution des scripts/services des crontab

\item connaître l'état de l'exécution de ces tâches (planning production)

\item visualiser l'état et les informations sur les crontab à travers une interface unifiée. Ces informations aideraient l'utilisateur par exemple dans le paramétrage lors de la modification ou de la création d'une nouvelle crontab

\item pouvoir trier la liste des tâches par paramètre, en ordre croissant et décroissant. Exemple de paramètres de tri : Statut, Minute, Heure, Jour du mois, Mois, Jour de la semaine, Commande (chemin relatif ou absolu du binaire).

\item gérer les dépendances dans l'exécution des jobs : un script ne lance que si un autre s'est terminé, s'il s'est terminé avec succès ou s'il s'est terminé avec erreur

\end{itemize}

\newpage

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% %

% Présentation du projet %

% %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

\chapter{Présentation du projet}

Pour faire face à la problématique ci-dessus, le SI FAI de Bouygues Telecom a besoin d'un outil avec ces principales caractéristiques : facilité d'utilisation, reporting sur l'exécution des tâches, et planning production.

\section{La solution à développer}

L’application FancyCron n'est pas une reprise du projet Cron, elle s'appuiera sur ce dernier en tant que surcouche et se composera de trois principaux modules :

\begin{itemize}

\item un module client \textcolor{violet}{\texttt{FancyCron-Client}}, en charge de l'exécution des tâches locales à chaque serveur en production. C'est ce module même qui va directement utiliser Cron.

\item un module d'administration appelé \textcolor{violet}{\texttt{FancyCron-Admin}}, qui permettra l'administration et l'utilisation du logiciel à travers une interface web : création, modification des crontab, planification des tâches,...

\item un module monitoring appelé \textcolor{violet}{\texttt{FancyCron-Monitor}}, qui permettra de surveiller l'état des tâches planifiées et de remonter les informations qui seront affichées. Il devra être paramétrable pour ne remonter que certaines informations.

\item un module serveur \textcolor{violet}{\texttt{FancyCron-Server}}, pour la gestion centralisée et le reporting des information de crontab. C'est le seul module à avoir accès à tous les autres et auquel tous les autres ont accès. Il implémentera un patron lui permettant d'assurer ce rôle.

\begin{figure}[H]

\begin{center}

\includegraphics[width=6cm]{Architecture.png}

\end{center}

\caption{\scriptsize \texttt{Mode de fonctionnement FancyCron}}

\end{figure}

\end{itemize}

\subsection{FancyCron-Client}

Ce module va se charger de toute la gestion des crontab sur le serveur où il est installé. Il encapsulera l'exécution des crontab dont il va gérer les entrées sorties. De ce fait les fichiers de configuration usuels des crontab sur les serveurs auront un format unifié, avec en paramètre commun un seul script utilisé par FancyCron-Client.

Désormais les seules différences visibles entre les lignes des de configuration de crontab seront principalement au niveau du cycle d'exécution de la crontab, et l'identifiant de la tâche à exécuter. L'identifiant est une référence sur le chemin et la description du binaire ou la commande à exécuter dans la crontab.

FancyCron-Client devra couvrir les fonctionnalités suivantes de l'application :

\begin{itemize}

\item il utilisera un module de gestion de tâche qui implémentera un conteneur de tâches (jobModule) instancié au démarrage du programme. Ce conteneur encapsulera pour chaque tâche spécifique, les paramètres associés :

\begin{itemize}

\item une valeur indiquant si la tâche est active ou non

\item un paramètre permettant de stocker le calendrier sérialisé d'évènements de la tâche

\item un entier indiquant le cas échéant le temps d'exécution de la tâche

\item une chaîne de caractères représentant le nom de la tâche

\item une chaîne de caractères pour le nom de la commande ou du binaire à lancer quand la tâche doit être exécutée

\item un paramètre permettant de stocker l'historique d'exécution de la tâche (la valeur de rétention sera fixée pour la phase de développement)

\item une valeur/clé permettant de référencer la tâche

\end{itemize}

Tous ces paramètres seront stockés en dur dans une base de données gérée par le serveur. Il disposera donc des méthodes lui permettant de récupérer toutes les tâches planifiées au démarrage, de récupérer les informations lui permettant de modifier la configuration d'une tâche, ou d'en créer une nouvelle, afin de mettre à jour le fichier de configuration crontab local. Il possèdera également un système de gestion d'erreurs pour ces différentes fonctions.

\item le client FancyCron utilisera un module de gestion de dépendances (jobDependenciesModule), définissant pour chaque tâche la liste des tâches dont elle dépend, avec le mode dépendance associé. Ces informations seront stockées dans la même base que pour les paramètres cités plus haut, et le jobDependenciesModule permettra d'accéder à ces valeurs au runtime.

\item c'est au jobModule du client qu'incombe l'ordonnancement des tâches et toutes actions associées. Il devra s'occuper entre autres, de l'état d'une tâche : en cours, terminée, décalée,

ok, ko, ....

\end{itemize}

FancyCron-Client communiquera directement avec FancyCron-Server.

\subsection{FancyCron-Admin}

FancyCron-Admin doit permettre aux administrateurs système (utilisateurs de FancyCron) de créer des crontab via une interface web. Ce module permet l'ajout d'une nouvelle tâche, la modification d'une tâche (renommer, décaler,...), avec des options de recherche de tâches.

L'admin FancyCron devant implémentant l'IHM de l'application qui permettra à l'utilisateur d'accéder aux différentes fonctionnalités telles que :

\begin{itemize}

\item définir l'adresse email à laquelle les messages de crontab doivent être envoyés (logs, fichiers de sorties)

\item planifier de nouvelles tâches (crontab)

\item modifier des tâches

\item supprimer des tâches

\item activer, désactiver l'exécution d'une tâche (activer/désactiver la crontab)

\item décaler l'exécution d'une tâche d'une temps \texttt{t} avant ou après l'heure d'exécution : le décalage prend en compte la période du cycle d'exécution que l'on souhaite décaler, ainsi que le temps de décalage à appliquer, ce qui permettra par exemple le rattrapage des exécutions râtées.

\end{itemize}

\subsection{FancyCron-Monitor}

FancyCron-Monitor permet de surveiller l'état des crontabs et affiche les informations dans l'interface de l'application.

Le monitor et l'admin sont deux applications web.

Ce module permettra de :

\begin{itemize}

\item voir le statut des tâches (en cours, terminée, erreurs, ..), cette visualisation doit se faire entre deux périodes consécutives du cycle d'exécution d'une tâche

\item lister les tâches retardées, en file d'attente (queue), en cours d'exécution, terminée. Ceci permettrait par exemple de diagnostiquer en cas d'incident

\begin{figure}[H]

\begin{center}

\includegraphics[width=10cm]{sam.png}

\end{center}

\caption{\scriptsize \texttt{Exemple d'affichage des tâches planifiées}}

\end{figure}

\item le monitor doit afficher l'heure prévu pour l'exécution de la tâche, l'heure à la quelle la tâche a commencé, le temps prévu pour l'exécution le cas échéant, et l'heure à la quelle elle s'est terminée

\item lister les tâches dont l'exécution n'a pas eu lieu ou a eu lieu à une heure spécifiée, ainsi que les dépendances (tâches requises)

\item avoir un reporting sur l'exécution des tâches, grâce à un module de reporting utilisé dans ce module. On pourra par exemple générer un rapport à l'aide d'un formulaire dans l'interface l'nterface web, suivant les critères proposés par l'application.

\item voir les fichiers de sortie des commandes exécutées dans les crontab, sauvegarder les fichiers, les envoyer en pièces jointes, ou les supprimer

\item avoir un affichage trié des tâches, suivant les paramètres de tri mentionnés ci-dessus.

\item avoir sur une page un historique des tâches, avec leurs états et les éventuelles évènements qui se seraient produits lors de l'exécution d'une tâche.

\begin{figure}[H]

\begin{center}

\includegraphics[width=10cm]{history.png}

\end{center}

\caption{\scriptsize \texttt{Historique des taches}}

\end{figure}

\end{itemize}

Ce module sera développé en priorité avec l'IHM, pour juger et apprécier l'ergonomie du logiciel. Toutes les considérations de développement seront consignées dans le document de conception et spécifications techniques.

Toutes les fonctionnalités du logiciel seront reportées sur l'IHM et exploitables par l'utilisateur via les modules Admin et Monitor

\subsection{FancyCron-Server}

Le serveur FancyCron est au centre du fonctionnement du logiciel. Il ne sera pas visible aux administrateurs/utilisateurs, mais toutes les opérations de distribution et de collecte dans le système doivent passer par lui. C'est en fait ce module qui répond au monitor sur l'état des tâches, et c'est aussi lui qui recueille ces informations auprès des clients que nous considérerons comme des agents FancyCron.

Ce module aura deux principaux rôles : d'une part la gestion centralisée des crontab, car c'est lui qui distribuera les paramètres d'édition de crontab récupérés par l'admin sur les serveurs concernés, et d'autre part le reporting des messages de crontab, qui seront ensuite remontés aux utilisateurs du logiciel par le moniteur.

\begin{figure}[H]

\begin{center}

\includegraphics[width=10cm]{Interaction.png}

\end{center}

\caption{\scriptsize \texttt{Mode de communication FancyCron}}

\end{figure}

Le module couvrira les caractéristiques suivantes :

\begin{itemize}

\item enregistrement des données dans une base de données centralisée grâce à un module de stockage (storageModule) qu'il embarque au démarrage. Le choix du système de gestion de base de données sera déterminé dans la phase de conception, en fonction de la complexité du système à mettre en place

\item transfert de paramètres de configuration au client, récupération des logs, résultats des commandes et autres messages de crontab du client. Il est clair que l'exécution des tâches est entièrement gérée en local par chaque client, et le serveur ne sert que dispatcheur pour la planification de ces tâches.

\item enregistrement/suppression des serveurs et groupe de serveurs utilisant des crontab

\item pour chaque utilisateur cron de chaque serveur, une liste des crontab associées

\item liste des tâches planifiées sur chaque serveur enregistré

\item liste et configuration des contacts pour messages crontab

\item gestion des dépendances non locales à un serveur (une tâche peut dépendre d'une autre non planifiée sur le même serveur)

\end{itemize}

\vspace{20pt}

\section{Contraintes}

\subsection{Environnement technique}

\begin{itemize}

\item La solution doit être portable, sous licence GPL, et fonctionner sur les systèmes Linux/UNIX, Windows, même si les premiers développements seront faits pour linux

\item Vue la modularité du système à développer, le choix d'un langage orienté objet s'impose. Cependant le choix du langage doit aussi tenir compte de l'environnement de naissance du projet, à savoir la DSI Bouygues Telecom, où l'utilisation des langages interprétés est préconisé plutôt que les langages compilés.

Sur la base de ces deux contraintes, nous opterons pour le python qui est le mieux adapté à la situation de par sa conception objet et sa richesse en librairies.

\item La solution devra s'intégrer au logiciel Cron, et ne devra poser aucune contrainte à son utilisation de base.

\item Le logiciel devant s'utiliser via une interface web, un serveur web est requis. Ainsi nous choisirons une des applications suivantes pour notre serveur web : nginx(éventuellement couplé avec passenger), apache, cherrypy(éventuellement couplé à cheetah). Le choix sera fait en tenant compte des contraintes matérielles du serveur web choisi, et de la compatibilité avec python, et de l'évolution du logiciel développé.

Le choix du serveur web sera décidé lors de la phase de conception après analyse fonctionnelle.

\item Enfin un protocole sera défini pour la communication entre le client et le serveur FacncyCron

\end{itemize}

\subsection{Méthode de développement de la solution}

Le développement de la solution devra se faire de manière itérative et chaque itération à partir de la deuxième, devra inclure tous les modules. Ci-dessous une proposition d’itérations :

\begin{itemize}

\item mise en place d’un MCD (Modèle Conceptuel de Données)

\item conception de la solution :

\begin{itemize}

\item production d’une maquette IHM pour l’utilisateur final de la solution

\item définition des données (toutes les informations nécessaires au développement des modules de la solution)

\item Modélisation des traitements (un ou plusieurs diagrammes de séquences)

\item Modélisation des objets nécessaires à la réalisation du projet (un ou plusieurs diagrammes de classes)

\item Modélisation du processus interactif de l'ensemble du système (un ou plusieurs diagrammes d'activités)

\end{itemize}

\item définition et documentation d'une API pour les plugins de l'application(un plugin devra inclure au moins une partie client, une partie serveur, une partie monitor ou une partie admin)

\item développement des fonctionnalités IHM basiques : analyse, conception, développement et tests unitaires

\item développement d'un module Client de base (fonction d'historique des jobs locaux en priorité) : analyse, conception, développement et tests unitaires

\item développement d'un module Server de base (fonction de reporting en priorité) : analyse, conception, développement et tests unitaires

\item développement d'un module Admin de base : analyse, conception, développement et tests unitaires

\item développement d'un module Monitor de base : analyse, conception, développement et tests unitaires

\item développement du premier prototype du logiciel incluant les fonctionnalités d'historique et de reporting : analyse, conception, développement et tests unitaires

\item développement des fonctionnalités de gestion centralisée des crontabs : analyse, conception, développement et tests unitaires

\item développement des fonctionnalités de tri et filtrage : analyse, conception, développement et tests unitaires

\item développement des fonctionnalités de gestion des dépendances et de planning production : analyse, conception, développement et tests unitaires

\item tests d’usines globaux et de vérification d’aptitude au bon fonctionnement (VABF)

\end{itemize}

\newpage

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

% %

% Prestation attendues %

% %

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

\chapter{Prestation attendues}

\section {Approche}

Pour la présentation des besoins, le projet FancyCron est découpé en lots selon une approche fonctionnelle homogène qui est définie ci-après :

\begin{itemize}

\item LOT 1 : MAITRISE D’ŒUVRE

\item LOT 2 : ANALYSE FONCTIONNELLE DETAILLEE

\item LOT 3 : DEVELOPPEMENT

\item LOT 4 : DOCUMENTATION

\item LOT 5 : TUTORIELS ET SUPPORT

\item LOT 6 : RECETTE

\end{itemize}

La maîtrise d’ouvrage est réalisée par l'équipe de la DSI FAI de Bouygues Télécom, qui a la possibilité de commander tout ou partie des lots décrits dans ce document.

\vspace{20pt}

\section{Modalités}

\subsection{Modalités de réalisation et livraison des prestations}

La livraison et la mise en œuvre des prestations pour la première version du logiciel doivent être réalisées suivant le planning suivant :

\begin{itemize}

\item Analyse fonctionnelle détaillée avant le 22/03/2012;

\item Livraison premier prototype avant le 22/05/2012;

\item Documentation avant le 15/07/2012;

\item Tutoriels et support avant le 05/08/2012;

\item Recette le 08/08/2012.

\end{itemize}

Le présent ci-dessus correspond à 3 jours ouvrés par semaine consacrés au projet.

En tout état de cause, la vérification d’aptitude au bon fonctionnement pour la première version de l'application doit être prononcée avant le 13/08/2012.

\subsection{Modalités de réception des prestations}

La réception des produits et services, sujets du présent projet fera l’objet d’une vérification d’aptitude au bon fonctionnement (VABF) et d’une vérification de service régulier (VSR) par la maîtrise d'ouvrage.

La VABF est destinée à vérifier la conformité de l’ensemble des produits et services fournis par l’équipe en charge du développement, conformément aux stipulations du cahier des charges.

Dans l’hypothèse où la procédure de VABF donnerait lieu à des réserves de la DSI FAI Bouygues Télécom, l'équipe projet devra procéder, dans des délais compatibles avec le calendri

er, aux corrections requises et à la livraison correspondante d’une nouvelle version des fournitures.

% \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*%

\end{document}