PROJET DE DOMINANTE [A4 – SPE.INF]

DEVELOPPEMENT

27/02/2020

1
CARACTERISTIQUES GLOBALES DU PROJET

Ce projet doit vous amener à identifier un fichier parmi un ensemble de fichiers et d'en extraire des informations. Les informations, une fois extraites, devront être alors remises à votre tuteur qui validera avec vous leurs authenticités.

Ces fichiers sont cryptés. Il vous faut donc les décrypter.

« Crypter, verbe transitif : Coder un message afin de le protéger et de le rendre incompréhensible à ceux qui ne disposent pas du code. »

Source : L' Internaute https://www.linternaute.fr/dictionnaire/fr/definition/crypter/

Vision rapide du projet

Nom du Projet					
Date début du projet :	19/06/2020	**Date fin du projet :**	03/07/2020	**Date de soutenance :**	03/07/2020
Nombre d'étudiants par groupe :	3	**Mode de constitution des groupes :**	Libre	**Durée de la soutenance par groupe :**	20 mn
Bloc d'appartenance :	Dominante logicielle A4 SPECIALITE INFO			**Durée des questions réponses :**	20 mn

Achats à prévoir			
Matériel par groupe projet			
Nom	**Quantité**	**Prix**	**Lien Web**
RAS	RAS	RAS	RAS
Matériel à partager			
RAS	RAS	RAS	RAS

Logiciels à télécharger avant le démarrage du projet			
Nom	**Quantité**	**Prix**	**Lien Web**
Visual Studio	1 par étudiant	0	Sur l'E.N.T.
SQL Server	1 par groupe	0	Sur l'E.N.T.
Eclipse ou NetBeans	1 par étudiant	0	Sur l'E.N.T.
Oracle	1 par groupe	0	Sur l'E.N.T.

Livrables à produire par les étudiants et à vérifier			
Nom	**Type**	**Date**	**Eléments à vérifier**
Livrable 1	Modélisation UML	24/06/2020	Architecture logicielle - Il s'agit de guider, conseiller les étudiants sur l'architecture logicielles des plateformes (non soumis à évaluation ce jour-là) – **Nécessite profil modélisation UML – Env. 10/15 mn par groupe**
Livrable 2	Dossier écrit	03/07/2020	Rapport - Le dossier des groupes (soumis à évaluation) – **Nécessite un profil Pilote/Tuteur**
	Oral de présentation de projet	03/07/2020	Oral de soutenance - Soutenance (soumis à évaluation) - **Nécessite un profil Pilote/Tuteur**
	Oral technique	03/07/2020	Technique individuelle - Soutenance (soumis à évaluation) - **Nécessite profil dev java .net oracle**
	Modélisation UML	03/07/2020	Architecture logicielle - Soutenance (soumis à évaluation) - **Nécessite un expert modélisation UML**
	Plateforme logicielle	03/07/2020	Client + middlewares + bases de données - Soutenance (soumis à évaluation) **Nécessite un expert Java/JEE et C#**
	Fichier	03/07/2020	Le fichier cible décrypté + la clé - Soutenance (soumis à évaluation) **Nécessite un profil Pilote/Tuteur**

Vision rapide des phases du projet

- Phase 1 – Lancement de projet
  - Période phase : du 19/06/2020 au 22/06/2020
  - Type étape : Lancement de projet – Réalisation étudiants – Suivi de projet (pilote) – Suivi de projet (Intervenant) – Livrable - Soutenance
  - But : Présenter aux étudiants les grandes lignes du projet et l'organisation
  - Porteur : Tuteur
  - Données d'entrée
    - Le cahier des charges du projet (ce document)
  - Données de sortie :
    - Composition des groupes
    - Questions éventuelles à poser au concepteur du projet
  - Description rapide de l'étape : Les étudiants lisent à haute voix le projet et commencent son appropriation. Ils peuvent avoir des questions. Le premier niveau de réponses est apporté par le Tuteur. Si le Tuteur n'a pas la réponse, il charge un intendant de contacter le concepteur pour avoir des informations complémentaires
  - Etape évaluée : oui/non
  - Item(s) Scholaris : RAS

- Phase 2 – Réalisation de la modélisation
  - Période phase : du 19/06/2020 au 24/06/2020
  - Type étape : Lancement de projet – Réalisation étudiants – Suivi de projet (pilote) – Suivi de projet (Intervenant) – Livrable - Soutenance
  - But : Construire la modélisation UML des plateformes .Net et Java
  - Porteur : Les étudiants
  - Données d'entrée
    - Cahier des charges du projet
  - Données de sortie :
    - Diagramme de classes – Diagramme de séquences – Cas d'utilisation
  - Description rapide de l'étape : Il s'agit dans cette phase de réaliser la modélisation des plateformes avant de passer au codage.
  - Etape évaluée : oui/non
  - Item(s) Scholaris : RAS

- Phase 3 – Guidance sur l'architecture
  - Période phase : du 24/06/2020 au 24/06/2020
  - Type étape : Lancement de projet – Réalisation étudiants – Suivi de projet (pilote) – Suivi de projet (Intervenant ou tuteur expert en UML) – Livrable - Soutenance
  - But : Conseiller les étudiants sur leurs choix d'architecture
  - Porteur : Pilote ou Intervenant
  - Données d'entrée
    - Diagramme de classes – Diagramme de séquences – Cas d'utilisation
  - Données de sortie :
    - Les diagrammes pseudo validés
  - Description rapide de l'étape : Les étudiants présentent les diagrammes UML du projet. Le Tuteur ou l'intervenant peut alors les guider dans leurs choix
  - Etape évaluée : oui/non
  - Item(s) Scholaris : RAS

- Phase 4 – Réalisation informatique
  - Période phase : du 25/06/2020 au 02/07/2020
  - Type étape : Lancement de projet – Réalisation étudiants – Suivi de projet (pilote) – Suivi de projet (Intervenant) – Livrable - Soutenance
  - But : Construire les plateformes
  - Porteur : Les étudiants
  - Données d'entrée
    - Diagrammes pseudo validés
    - Spécifications fonctionnelles du projet
    - Spécifications techniques du projet
    - Consignes de soutenances
  - Données de sortie :
    - RAS
  - Description rapide de l'étape : Les étudiants dans cette phase devront construire les plates-formes .Net et Java, Implémenter des instances d'SQL Server et Oracle, décrypter les fichiers, identifier l'information secrète et préparer la soutenance
  - Etape évaluée : oui/non
  - Item(s) Scholaris : RAS

- Phase 5 – Soutenances
  - Période phase : du 03/07/2020 au 03/07/2020
  - Type étape : Lancement de projet – Réalisation étudiants – Suivi de projet (pilote) – Suivi de projet (Intervenant) – Livrable - Soutenance
  - But : Evaluer le travail des étudiants
  - Porteur : Jury de soutenance

- Données d'entrée
  - Rapport de projet
  - Ppt de présentation
  - Livrables techniques (Plateforme Java et Oracle – Plateforme .Net et SQL Server – Client .net)
- Données de sortie :
  - Evaluations
- Description rapide de l'étape : Les étudiants présentent leurs projets. Le Jury évalue.
- Etape évaluée : oui/non
- Item(s) Scholaris :
  - Voir grille d'évaluation

2
CAHIER DES CHARGES DU PROJET

Dans cette section, vous trouverez les spécifications fonctionnelles et techniques du projet

Partie 1 CDC – Les Spécifications Fonctionnelles

Vous devez construire une plateforme qui doit permettre à l'utilisateur :

- De s'authentifier
- De lancer le déchiffrement
- D'obtenir le fichier cible en clair ainsi que sa clé

Partie 2 CDC – Les Spécifications Techniques

Votre équipe se compose d'experts .NET (C# /WCF/SQL Server) et d'experts Java EE (EJB /JMS/web services/Oracle).

L'application .NET se charge de déchiffrer les fichiers codés en appliquant sur chaque fichier une série de clés. La plateforme java EE reçoit tout flux sur lequel une clé a été appliquée. La plateforme .Net envoie donc à la plateforme Java EE : un flux résultant de l'application d'une clé, le nom de fichier associé et la clé utilisée. La plateforme Java doit s'assurer que le flux est déchiffré correctement (qu'il s'agit d'un texte en français). Si c'est le cas, elle doit rechercher les informations cachées. Si l'information secrète est trouvée, la plateforme Java EE doit fournir à la plateforme .NET les informations suivantes : les informations cachées, le nom du fichier contenant l'information et la clé de déchiffrement.

En résumé, la plateforme .NET se charge du déchiffrement et la plateforme Java EE se charge de valider le déchiffrement et de retrouver les informations cachées.

Votre plateforme distribuée doit remplir au mieux les exigences non fonctionnelles (scalabilité, disponibilité, modularité, sécurité…) liées à ce type de projet.

SPF1	**Authentification utilisateur**
In : login, password, tokenApp	L'utilisateur doit fournir son login et son mot de passe. La platefome.Net se charge de l'authentification. Le client lourd doit fournir son token de sécurité spécifique à l'application. La plateforme ne donne accès à ces services qu'en cas de token valide. En cas de succès la plateforme fournit un token qui permettra à l'utilisateur de bénéficier des services de la plateforme.
Out : bool, tokenUser	

SPF2	Déchiffrer
In : tokenUser, tokenApp, txt files	A l'aide du client, l'utilisateur indique quels fichiers il souhaite déchiffrer. Plusieurs fichiers (à la limite du nombre de cœurs du processeur) seront traités en parallèles. Le client communique son token d'application et son token d'utilisateur. Si la plateforme accepte la requête, elle reçoit les fichiers à déchiffrer. Elle tente de déchiffrer avec un premier code les fichiers reçus. Une fois ce travail effectué elle envoie les fichiers à la plateforme jee. Elle continue à déchiffrer ces mêmes fichiers avec un autre code. Elle réitère cette opération jusqu'à ce que la plateforme jee lui indique qu'elle (la plateforme .Net) a réussi à trouver le bon code. Elle arrête alors de vouloir identifier le code et fournit au client le code, le fichier en clair et le rapport pdf de taux de confiance du document. Pour savoir si un document a bel et bien été déchiffré par la plateforme .Net, pour chaque 'document' la plateforme jee constitue un échantillon de mots présents dans le document. Au plus l'échantillon est grand, au plus le taux de confiance du document sera important, mais au plus le temps de traitement sera long. Elle compare les échantillons avec un dictionnaire de mots présent en base (à trouver sur internet par exemple). Si elle estime que suffisamment de mots présents dans le document sont en correspondance avec son dictionnaire de mots, alors elle conclura que le 'document' a été déchiffré. Elle recherche alors l'information secrète cachée dans le 'document'. Elle enverra alors à la plateforme .net, le triplet {information secrète/nom du document/ code de déchiffrement}. La plateforme .NET informera à son tour le client lourd. Un email d'avertissement sera envoyé à l'utilisateur pour le prévenir de la réussite de l'opération cet email peut être envoyé par la plateforme Java EE ou .NET.
Out : bool, file, key, pdf file	

Pour la plateforme .Net, ci-après, la liste des spécifications techniques à respecter fournissant ainsi les critères d'acceptations techniques :

- Stx 1 - Architecture : 3 tiers - SOA
- Stx 2 - Framework : Microsoft Framework 4.0.
- Stx 3 - Langage : C# 4.0.
- Stx 4 - Solution : 9 couches.
- Stx 5 - Client :
  - A déployer sur les postes clients.
  - Lourd / asynchrone.
  - 3 couches (1 composant physique).
  - Parallélisassions des traitements issus du middleware (traitements lourds). Client non-bloqué lorsqu'il attend les réponses de la plateforme.
  - WPF possible, donc pas obligatoire. En effet un design riche n'est pas nécessaire (même s'il doit être soigné)
- Stx 6 - Middleware :
  - A déployer sur un serveur.
  - Architecture de type service.
  - Point d'entrée unique.
  - Communication sécurisée basée sur le chiffrement de bout en bout (sécurité du message)

  - Structure des messages : MSG [bool statutOp – string info – object[] data – string operationName – string tokenAppl – string tokenUser – string appVersion – string operationVersion]
  - Une seule méthode exposée sur le point d'entrée unique de signature : STG m_service(STG msg)
  - Distribué : oui.
  - Journalisation de l'activité du CAM.
  - Les contrôleurs de workflow et les composants métiers doivent supporter des charges de travail importantes.
  - 5 couches.
  - 1 composant physique pour les couches 5 – 4 – 3.
  - 1 composant physique pour la couche 2.
  - 1 composant physique pour la couche 1.
- Stx 7 - Data :
  - SQL Server STD Ed 2008.
  - Mode d'authentification mixte.
  - Mise en place de l'agent SQL Server dans un processus Windows distinct.
  - Mise en place du moteur SQL Server dans un processus Windows distinct.
  - Sauvegarde automatique journalière (sauvegarde complète).
- Stx 8 – Communication
  - Liaison de type netTcp entre le client lourd et la plateforme.
  - Communication sécurisée avec un chiffrement à 128b.

Précisions sur la structure des messages (struct MSG) échangés entre les méthodes (toutes couches) :

- bool statutOp : Le statut de l'opération en cours (true/flase)
- string info : Un message d'information sur l'opération en cours
- object[] data : Les données utiles à l'exécution de l'opération en cours
- string operationName : Le nom du service invoqué
- string tokenApp : Le code de sécurité de l'application
- string tokenUser : Le code de sécurité après authentification de l'utilisateur
- string appVersion : Le numéro de version de l'application
- string operationVersion : le numéro de version de l'opération demandée

Pour Java, la réception des messages générés par la plateforme .NET (contenu des documents décodés par .NET) et la logique de traitement des messages pour repérer les informations secrètes doivent s'exécuter dans deux domaines différents. Ce sera un plus si l'application Java EE est répartie sur des machines distinctes. Les machines distinctes peuvent être des machines virtuelles.

L'intégration du tiers 'réception des messages' et du tiers 'traitement des messages' se fait au moyen de l'API JMS. Un middleware orienté message s'interpose donc entre la logique d'acquisition et la logique de traitement. Par logique de traitement du message, nous entendons opérations sur le texte décodé par l'application .NET

L'API JAX-WS est utilisée pour exposer la plateforme Java EE à la plateforme client .NET. Cette application Java EE ne requiert aucune interface homme machine. Les messages reçus sont envoyés dans une destination JMS de type queue. Le traitement (qui peut être long) des messages est asynchrone. Les messages de la destination sont pris en charge par des composants de type Message-Driven Bean. Le choix du type de message au sens JMS est à votre discrétion : TextMessage, ByteMessage…

Rappelons que tout message JMS est composé d'une enveloppe contenant un header, une section optionnelle de propriétés personnalisables et d'un corps de message qui contient l'information à transférer. La section de propriétés permet de transporter / transférer des informations complémentaires qui peuvent être entre autre de type entier ou chaîne de caractère. Ces propriétés définies par le développeur sont des paires clé/valeur.

Pour des raisons de scalabilité la plateforme Java EE doit être essentiellement (voire totalement) « stateless ». Dans un environnement orienté EJB, les API de threading ne doivent pas être utilisées par les développeurs. La gestion des threads est de la responsabilité du container. Il est important de rappeler que l'architecture EJB est adaptée aux environnements extrêmement concurrents.

La plateforme valide le déchiffrement en comparant un échantillon de mots /motifs prélevés dans le 'document' décodé avec une base de mots (français). La base est de type Oracle. Internet propose un grand nombre de dictionnaires de mots. Le choix du dictionnaire vous incombe. Vous pourrez par exemple utiliser le dictionnaire au format csv pour alimenter votre « base » MySQL de mots.

Si l'information secrète a été trouvée dans le ' document', la plateforme Java EE s'appuiera sur un service WCF pour communiquer les informations à la plateforme .NET.

Partie 3 – Les Livrables

Vous trouverez ci-après la liste des livrables à fournir dans ce projet :

- Livrable 1 – 24/06/2020 – Première version de l'architecture – Non soumis à évaluation
- Livrable 2 – 03/07/2020 – Présentation de projet – Soumis à évaluation
    - Rapport projet.
        - Page de garde
        - Sommaire
        - Introduction.
        - Rappel du besoin.
        - Découpage du projet.
        - Planification initiale.
        - Répartition des tâches.
        - Modélisation de l'application distribuée (architecture logicielle globale – architecture physique globale – modélisation diagramme de classes par couche et par plateforme – diagramme d'activité – diagramme de séquence par couche et par plateforme).
        - Analyse des écarts.
        - Analyse des compétences acquises par étudiants
        - Bilan.
    - La plateforme distribuée basée sur les technologies .NET WCF et java EE.
    - Le client.
    - Les fichiers déchiffrés et leurs rapports de probabilité.
    - L'information secrète, la clé et le numéro de fichier.

Partie 4 – La composition des groupes projet

Les groupes de projet sont constitués de 3 étudiants (4 si le contexte fait en sorte qu'avec des groupes de 3 étudiants, un groupe se retrouve composé de 2 membres).

Partie 5 – Spécifications et organisation pour le déchiffrement des fichiers.

Vous trouverez en temps et en heure l'ensemble des fichiers à décrypter. Ils ne vous seront remis qu'à partir de la deuxième semaine de projet. Les fichiers sont nombreux. Vos compositions logicielles doivent tenir compte en amont de cette contrainte.

Nous vous donnons dans le cahier des charges de ce projet, le descriptif et les informations nécessaires pour développer l'architecture logicielle et l'algorithme à mettre en œuvre pour déchiffrer les fichiers (voir spécifications).

Vous devez fournir à votre tuteur en fin de processus :

- Le numéro de fichier comportant l'information secrète
- La clé qui a permis de le décrypter
- L'information secrète.

Pour identifier l'information secrète, c'est facile, elle sera identifiée comme étant « l'information secrète ».

Pour le lot de fichier les fichiers ont été cryptés de la manière suivante :

- La clé à une longueur de 4 caractères
- Les caractères sont uniquement en majuscule
- Les caractères sont uniquement issus de l'alphabet (France)
- La clé ne comporte aucun caractère accentué
- L'opérateur utilisé est XOR
- La texte source est codé caractère après caractère à l'aide de la clé.

Exemple : le texte source est « bonjour à tous ». La clé est « clé ». Alors l'algorithme va créer le texte crypté de la manière suivante :

b	o	n	j	O	u	r		à		t	o	u	S
c	l	e	c	L	e	c	l	e	c	l	e	c	l

3
EVALUATIONS

Vous trouverez sur Moodle la grille d'évaluation de ce projet

CAMPUS
D'ENSEIGNEMENT SUPÉRIEUR
ET DE FORMATION PROFESSIONNELLE