

**RESPONSABLE LICENCE 3 : RESPONSABLE DE STAGE :**

**M. MM.**

**Emmanuel JEANDEL Bruno GUILLAUME**

**Yves LEPAGE**

**Jean LIEBER**

**Emmanuel NAUER**

**Hue-Nam LY**

**Rapport De Stage**

**Développer une infrastructure Web pour un système de correction de phrases en français à partir de cas**

**Du 9 avril au 6 juin 2018**

**Année Universitaire 2017-2018**



**Licence 3 informatique**

Sommaire

1. **BASE DE DONEES**
2. Présentation de la base de donnes
3. Génération du script de la création et de l’insertion
4. **CONCEPTION**
5. Diagramme contexte statique
6. Diagramme de cas d’utilisation
7. Diagramme d’activité
8. Diagramme de séquences
9. **ENVIRONNEMENT DE DEVELOPPEMENT**
10. **INTERFACE WEB**

(\*\*numerotation de pages sera indiqué à la version finale du rapport\*\*)

**REMERCIEMENT**

Je tiens à présenter mes sincères remerciements à :

Monsieur Jean LIEBER :

Monsieur Emmanuelle NAUER :

Monsieur

Introduction :

De nos jours, il existe plusieurs systèmes qui corrigent des fautes d’orthographe ou grammaire automatique. Mais que se passe-t-il si on rentrait une phrase syntaxiquement incorrecte ? Est-ce que ces systèmes nous donnent une bonne ou mauvaise solution ?

Dans notre cas, on cherche à corriger ce genre de problème et fournir à l’utilisateur une correction acceptable à partir du raisonnement à partir de cas.

Le raisonnement à partir de cas (RAPC) : (\*\*pour expliquer le RAPC est ce que je peux utiliser l’explication du PDF du sujet ? \*\*) consiste donc à s’appuyer sur des couples (problème, solution) ou problème est une phrase incorrecte qui est corrigée en solution pour proposer une correction à une nouvelle phrase cible. Par exemple :

*Problème : Je aimer des pommes.*

*Solution : J’aime des pommes.*

La phrase cible sera proposée par les utilisateurs via une interface web. Cela déclenchera le système de remémoration puis d’adaptation et la solution sera affichée sur l’interface web. Par exemple :

*Cible : Je manger des tomates.*

*Solution : Je mange des tomates.*

Si le résultat est accepté par l’utilisateur il forme alors un couple (cible, solution) qui sera insérée dans la base de cas pour être utiliser plus tard en tant que (problème, solution).

La base de cas minimal est alimentée de manière manuelle par M. Levy. Ensuite elle sera complétée de manière semi-automatique à partir du corpus WiKoPaCo qui est un historique de modification de pages sous Wikipédia. Puis, elle sera améliorée au fur et à mesure d’utilisation. Le remémoration et l’adaptation sont des algorithmes développés par M. Giang en Python. En s’appuyant sur le RAPC pour corriger un problème et donner une solution optimale.

Dans ce stage, je dois créer une base de données qui stocke les cas et développer une interface web multilingue qui donnera la possibilité aux utilisateurs de rentrer une phrase incorrecte en français ou anglais et affichera la solution aux utilisateurs.

Tout d’abord, je vais présenter de la création d’une base de données, la conception, suivi l’environnement de développement, et enfin le développement de l’interface web.

1. **BASE DE DONEES**
2. Présentation de la base de données

Pour ce stage, le type de base de donnes n’est pas imposé. Donc je choisie le mySQL, car c’est le type que j’utilise pendant mon année universitaire.

Le seul inconvénient ici c’est que cette base devrait adapte aussi au travaille de Mr. Levy. Ainsi, Apres plusieurs réunions, on a décidé que cette base est de la forme ci-dessous (fig.1).

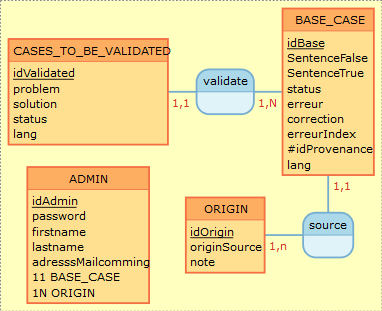


Fig.1 : Base de donnees.

Notre base de donne contient trois tables :

La table Case\_BASE est composée une clé primaire idBase, ainsi deux colonnes « problem » et « solution », ces deux dernières forment alors la couple (problème, solution) de RAPC. La colonne « status » est de forme « bon, mauvais, en attente » pour indique si la solution est correct rapport au problem. Les deux colonnes suivantes sont « erreur et correction » elles indiquent le(s) mot(s) a été corrigé. « erreurIndex » donne la position du mots erroné dans la phrase. « lang » pour l’instant notre projet ne comporte seulement du français et d’anglais mais elle peut être évoluer dans le futur.

La table ORIGIN (ou provenance) stocke les informations de la provenance de la couple (problème, solution), c’est-à-dire de quelle source Mr Levy a trouvé ce genre de couple. La clé primaire est « idOrigin » et « originSource » est le nom de source (Par exemple : WiKoPaCo).

La table ADMIN contient les informations nécessaires de l’administrateur.

1. Génération du script de la création et de l’insertion
2. Création

Le code de création d’une base est très standard (Fig.2):

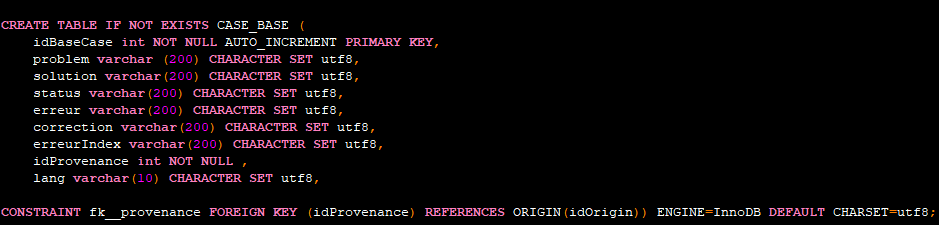


Fig.2 Code création une table

Le seul problème que j’ai rencontré ici est la partie « ENGINE=InnoDB ». Il faut précise le type ENGINE si non le système va prendre celui par défaut : MYISM. Cette dernière ne prendre pas en charge les clés étranges.

1. Insertion

Les donnes du projet sont collectées par Mr Levy et stockées dans un fichier.csv. A partir de ce fichier on a deux choix :

1. On peut l’import directement sur phpMyAdmin avec le système import fichier que fournir le site.
2. On développe un script php pour importer les données et les insérer dans la base.

J’utilise la deuxième méthode. Je développe deux fonctions : une qui stock les informations du fichier.csv dans un « array>(Fig.3). Et la deuxième fonction insere les contenues de cette « array » dans la base de donnes(Fig.4).



Fig3. importCSVFile Fig.4 insertData

1. **LA CONCEPTION**
2. Diagramme contexte statique

Le diagramme de contexte permet de visionner les interfaces entre le système sous enquêtes et le système des entités externes avec lesquels il est en interaction (Fig.6).

Dans notre cas, des entités externes sont des utilisateurs (USER) et le système sous enquêtes est le moteur de correction.

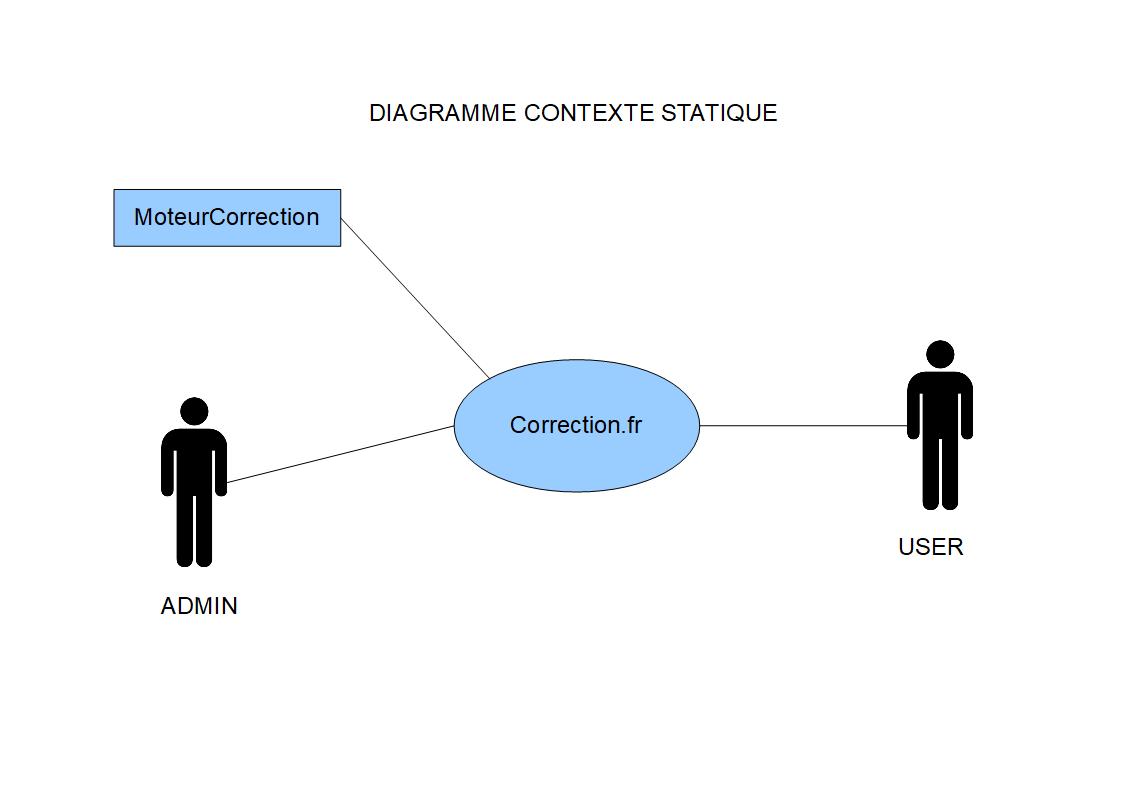
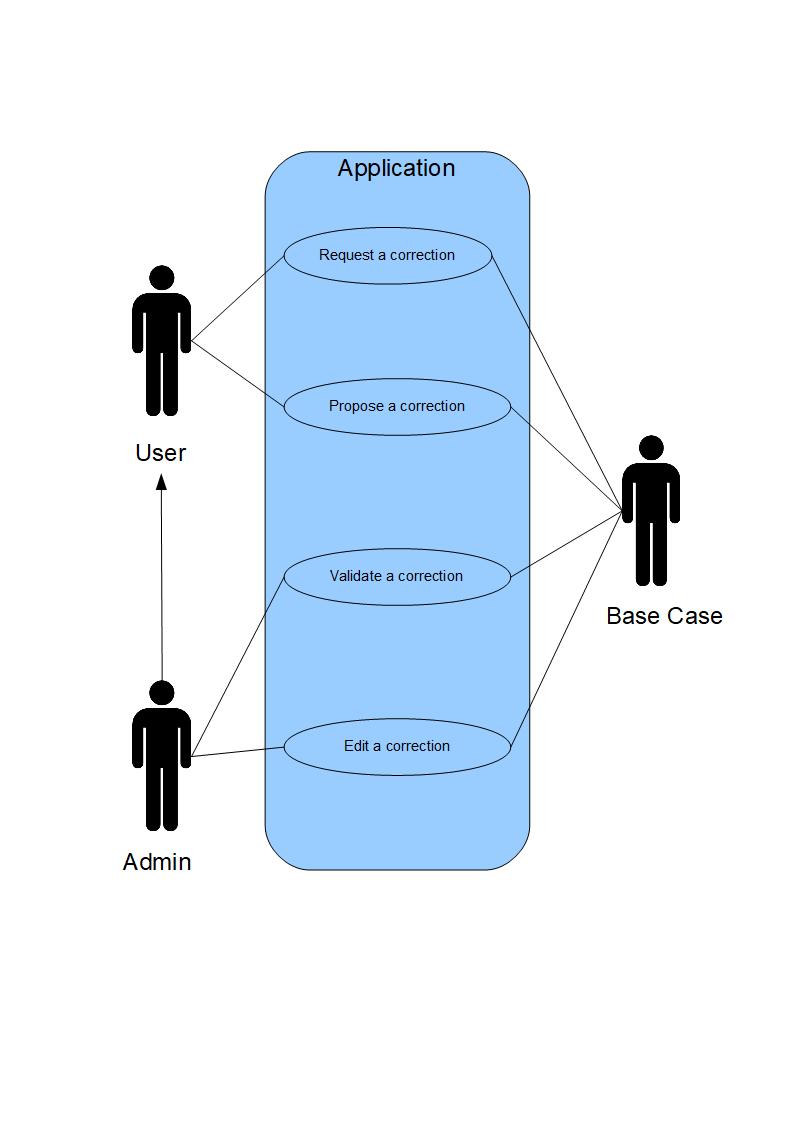


Fig.6 Diagramme contexte statique

1. Diagramme Cas Utilisation

Le diagramme de cas d’utilisation représente les fonctionnalités nécessaires pour les utilisateurs. Après les études aux besoins des utilisateurs, on a trouvé quatre types de cas :

* Demande de correction (*request correction*).
* Proposer une correction (*propose correction*).
* Valider une correction (*validate a correction*).
* Modifier une base de cas (*Edit base case*).

Les deux derniers types sont réservées aux utilisateurs avec le droit administrateur.

Fig.7 Diagramme cas utilisation

1. Diagramme Activité

Le diagramme d’activité représente le déroulement des actions, il est utilisé pour détailler les spécifications d’un cas d’utilisation et le déclenchement d’événements en fonction des états du système et modéliser ses comportements. Ainsi après, le diagramme de cas d’utilisation, on possède alors quatre diagrammes d’activité.

1. Demander une correction

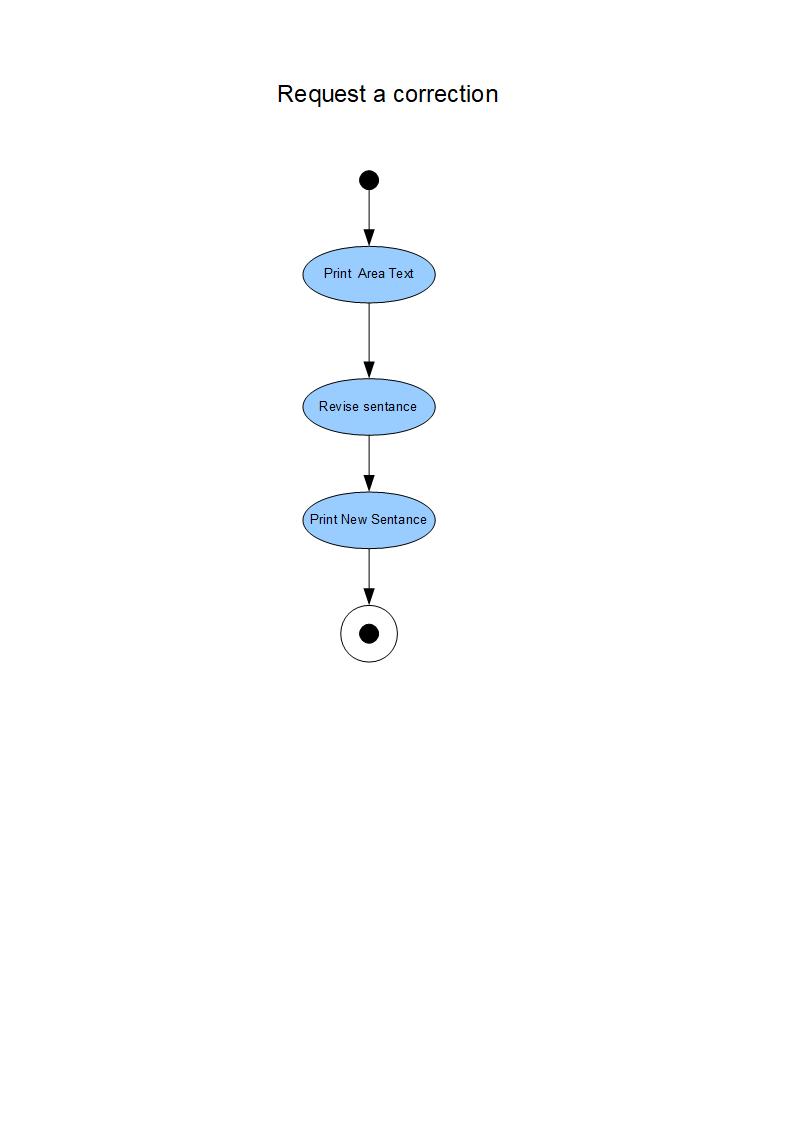
 Le système affiche une zone de texte ou un utilisateur peut entrer la phrase incorrecte. Ensuite il récupère la phrase et déclenche le moteur de correction (Revise Sentence), puis, elle affiche le résultat à l’utilisateur.

Fig.8 Demande une correction

1. Proposer une correction

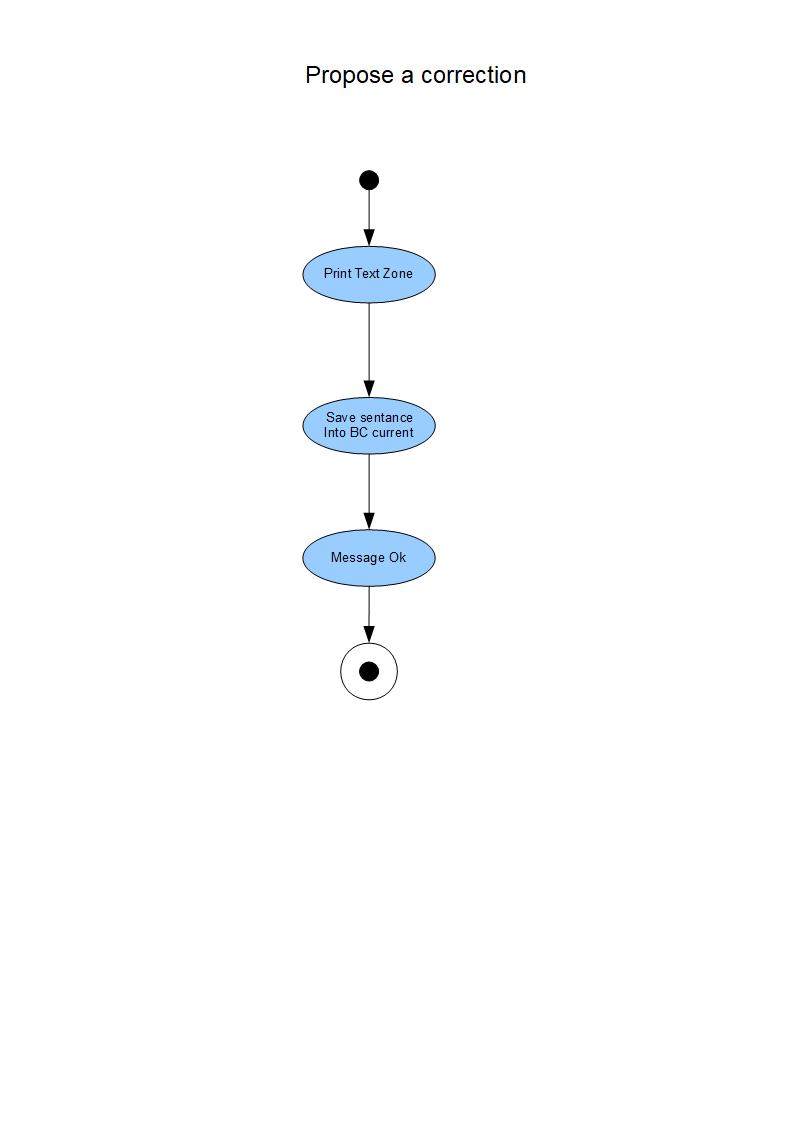
 Le système a le même comportement que ‘’Demander une correction ‘’, mais au lieu de déclencher le moteur de correction, il sauvegarde la phrase proposée par utilisateur dans la base courante.

Fig.9 Proposer une correction

1. Valider la phrase

Cette activité est réservée pour les administrateurs. Le système affiche la phrase proposée par un utilisateur, puis elle déclenche le moteur de corrections et affiche le résultat. Il attend alors une confirmation de l’administrateur sur ce résultat pour savoir s’il droit stocker la phrase proposée dans la base de données ou l’enlever de la base de cas courant.

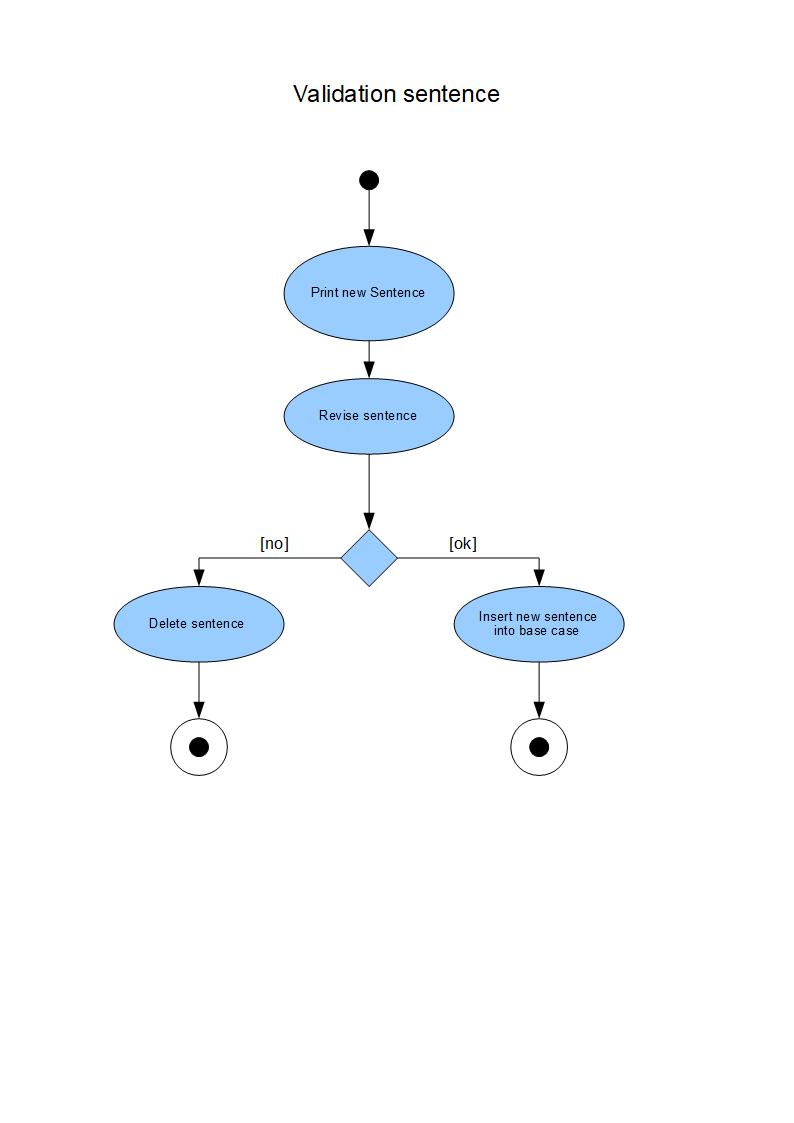


Fig.10 Validation une phrase par Admin

1. Modifications une phrase

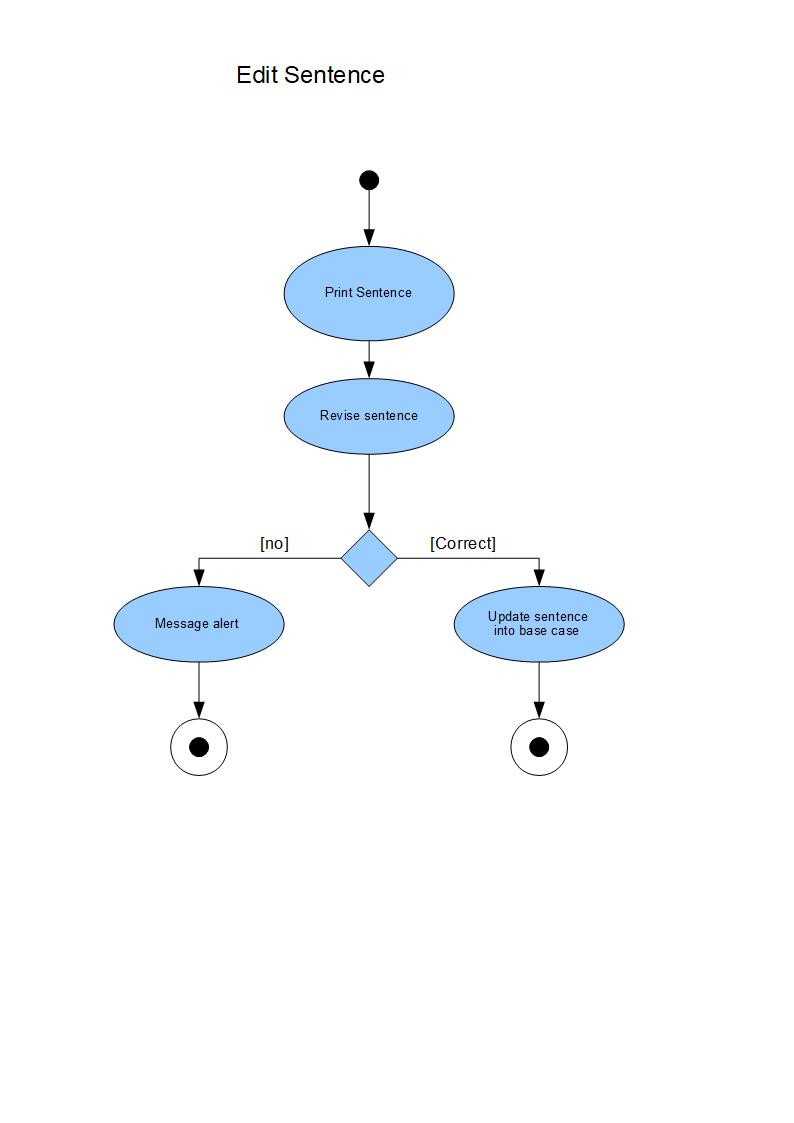
Cette activité est identique que celle précédemment. Mais au lieu d’insérer dans la base courante, le système va modifier la phrase que l’administrateur pense que c’est incorrect.

Fig.11 Modifications une phrase par Admin

1. Diagramme séquence

Ce type de diagramme permet de décrire les différents scenarios d’utilisation du système. Il représente des interactions entre les acteurs et le système dans l’ordre chronologique. Pour générer ce genre de diagramme je me sers d’une site web : websequencediagrams.

1. Demander une correction

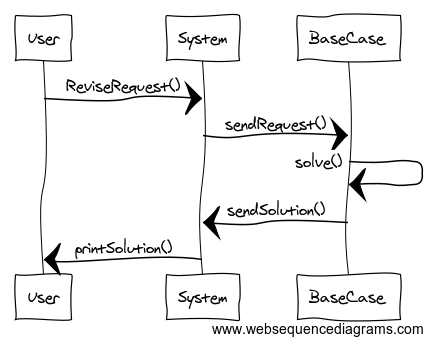
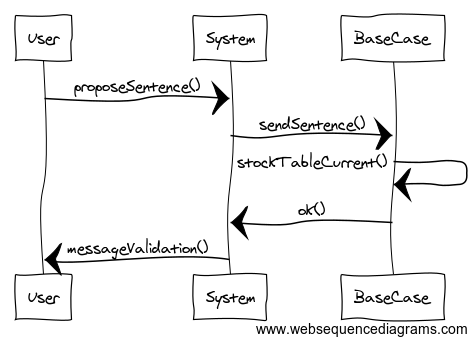


Fig.12 Demander une correction

L’utilisateur demande une correction au système, le système envoie la requête au moteur de correction, qui va résoudre le problème et donne une solution au système, le système va afficher le résultat aux utilisateurs.

1. Proposer une correction



L’utilisateur propose une phrase, le système l’envoie à la base de cas, qui le stock dans une table courant.

Fig.13 Proposer une Correction

1. Validation une phrase

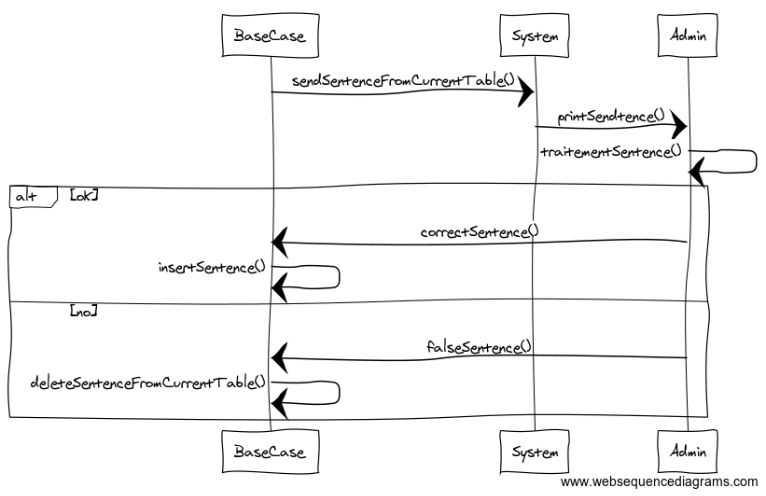


Fig.14 Validation une phrase

Le système récupère la phrase stockée dans la table courante, et l’affiche aux administrateurs. Ce dernier valide la phrase, si c’est correct, elle sera insérée dans la base de cas, sinon elle sera enlevée de la table courante.

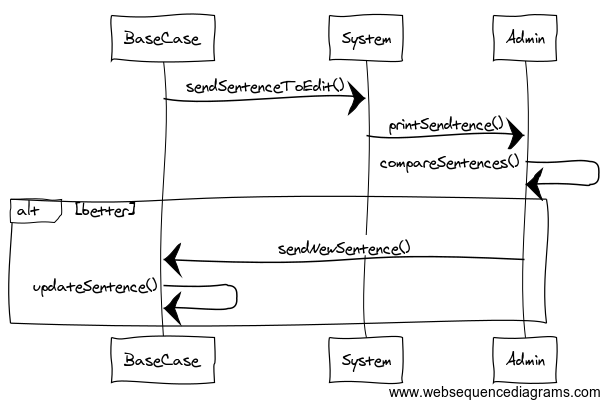
1.  Modifications une phrase

Fig.15 Modifications une phrase

Le système affiche la phrase a modifié, l’administrateur compare la nouvelle phrase par rapport à l’ancienne. Si elle est meilleur, la nouvelle phrase remplacera alors l’ancienne dans la base de case.

(\*\*Environnement Développement\*\*)

(\*\*GIT\*\*)

(\*\*base de donnes MySQL\*\*)

(\*\*interface web sublimetext\*\*)

(\*\*serveru wamp\*\*)

(\*\*Interface web\*\*)

(\*\*multiling DONE\*\*)

(\*\*création de la maquette via mybalsamiq DONE\*\*)

(\*\*en cours php javascript \*\*)