# REMERCIEMENT

Je voudrais remercier plusieurs personnes sans lesquelles ce travail n’aurait pas pu  
voir le jour.

* Tout d’abord un grand merci à mon encadreur professionnel M. Tiemi Noupoué Thibault grâce à qui j’ai pu réaliser mon  
  stage dans mon lieu de travail, ses conseils et son encadrement qui m’ont aidé à achever ce mémoire.
* Ensuite, je voudrais aussi remercier le Dr Noumedem mon superviseur et M. Tchuenkam mon encadreur académique qui m’ont encadré de la meilleure façon qui soit.
* Mes collègues de bureau qui m’ont donné de leur temps et compétence pour la mise en place du système
* Mes parents qui n’ont jamais cessé de croire en moi et m’encouragé.
* Mes amis Evangelista imbotshi esli, Tondji lee, Sani marcel qui ont toujours été la -là pour me conseillér et me guidér.
* Mon frère Tchouala igor et mes sœurs Njiki michelle et Saclélé falonne
* Mes camarades de classe de promotion pour la grande famille que nous formons, ainsi que mes enseignants pour leur enseignement fructueux ~~pour nous~~.(problème de phrasage peut mieux dire)

# SOMMAIRE

[Remerciement 1](#_Toc482561627)

[Sommaire 1](#_Toc482561628)

[Liste des tableaux 2](#_Toc482561629)

[Liste des figures 2](#_Toc482561630)

[Glossaire 2](#_Toc482561631)

[Résumé 2](#_Toc482561632)

[Abstract 2](#_Toc482561633)

[Introduction 2](#_Toc482561634)

[Partie 1 : Etat de l’art 2](#_Toc482561635)

[Chapitre 1 : Etat de l’art 2](#_Toc482561636)

[I- Etude de l’existant 2](#_Toc482561637)

[II- Présentation de la solution 2](#_Toc482561638)

[III- Cahier de charge 2](#_Toc482561639)

[Chapitre 2 : Méthodologie et technologie 5](#_Toc482561640)

[I- Présentation des méthodes de gestion de projet 5](#_Toc482561641)

[II- Choix de la méthode Agile 9](#_Toc482561642)

[III- Choix de la méthode d’analyse 13](#_Toc482561643)

[Partie 2 : Conception et implémentation de la solution 15](#_Toc482561644)

[Chapitre 3 : Conception de la solution 15](#_Toc482561645)

[I- Module de compte 15](#_Toc482561646)

~~[I](#_Toc482561647)~~[- Module de histoire 15](#_Toc482561647)

[II- Module formation 15](#_Toc482561648)

[a. Module quiz 15](#_Toc482561649)

[III- Module contribution 15](#_Toc482561650)

[IV- Module de map 15](#_Toc482561651)

[V- Module fil d’actualité 15](#_Toc482561652)

[VI- Module communauté 15](#_Toc482561653)

[VII- Module traducteur 15](#_Toc482561654)

[VIII- Module de notification 15](#_Toc482561655)

[IX- Module Tchat 15](#_Toc482561656)

[X- Module paramètre 15](#_Toc482561657)

[Chapitre 4 : Implémentation de la solution 16](#_Toc482561658)

[I- Outil de développement et langage utilisé](#_Toc482561659)~~[e](#_Toc482561659)~~ [16](#_Toc482561659)

[II- Architecture 16](#_Toc482561660)

[III- Algorithme développé 16](#_Toc482561661)

[IV- Interface graphique 16](#_Toc482561662)

[Conclusion 16](#_Toc482561663)

[Bibliographie 16](#_Toc482561664)

[Annexes 16](#_Toc482561665)

# LISTE DES TABLEAUX

# LISTE DES FIGURES

# GLOSSAIRE ET ABREVIATIONS

UML : Unified Modeling Language

XP : eXtreme Programming

# RESUME

Si définir les concepts de culture et de compétence culturelle n'est pas aisé, définir celui de compétence interculturelle l'est encore plus. Pourtant, en tant qu'ensemble d'aspects intellectuels et de formes de comportements qui caractérisent une civilisation tout en permettant de développer le sens critique, le goût et le jugement, la culture devrait avoir un impact sérieux sur les autres activités de la vie quotidienne. La langue véhicule la culture d'un peuple, au sens large du terme, c'est-à-dire, l'ensemble des connaissances (historiques, littéraires, religieuses, populaires...) partagées par le plus grand nombre. Sans une langue propre a un peuple on ne pourrait connaitre son identité c’est pour cela que dans la plupart des pays nous avons des langues maternelles en plus des langues internationales. Le présent projet vise a donné la possibilité a -à toute~~s~~ personne~~s~~ désireuse d’apprendre, de connaitre, de faire des traductions en une langue étrangère, de comprendre le sens des expressions populaires (argot) difficiles à comprendre pour des étrangers, de vulgarisér l’utilisation de ces expressions, de permettre a-a nos langues de traversér les frontières afin d’être parlées par d’autre et enfin de permettre a-à des utilisateurs parlant des langues différentes de pouvoir communiquer (se comprendre) en brisant la barrière de la langue. Afin d’assurer la réalisation de ce projet nous utiliserons comme méthodes de développement la méthode agiles eXtreme Programming (XP) car elle est axée sur le développement en module et adéquate pour les petites équipes de développement ~~petites~~ (3 à 4 développeurs par module). Comme langage de modélisation nous avons opté pour UML car celui-ci nous permet de faire un développement objet et ces différents diagrammes nous permettent d’avoir une vision plus claire du projet grâce aux différents scéenarios que nous mettronst en place et les différentes interactions qu’un utilisateur peut (pourra) avoir avec le système.

# ABSTRACT

If defining the concepts of culture and cultural competence is not easy, defining that of intercultural competence is even more. Yet, as a set of intellectual aspects and forms of behaviour that characterise a civilization while allowing to develop critical sense, taste and judgement, culture should have a serious impact on the other activities of daily life. The language conveys the culture of a people, in the broad sense of the word, that is, the whole of the knowledge (historical, literary, religious, popular ...) shared by the greatest number. Without a language of its own people one could not know its identity that is why in most countries we have mother tongues in addition to international languages. The aim of this project has been to give all persons who wish to learn, to know, to make translations into a foreign language, to understand the meaning of popular expressions which are difficult to understand for foreigners, of popularizing the use of these expressions, allow our languages to cross borders in order to be spoken by others and finally allow users speaking different languages to be able to communicate by breaking the language barrier. In order to ensure the realization of this project we will use as methods of development the Agile Method eXtreme Programming (XP) as it is focused on module development and suitable for small development teams (3 to 4 developers per module). As modeling language we opted for UML because it allows us to make an object development and these different diagrams allow us to have a clearer vision of the project thanks to the different scenarios that we will put in place and the different interactions that a user can have with the system.

# INTRODUCTION

Face à un monde de plus en plus métissé pour ce qui est des modes de vie, des us et coutumes des peuples, face à la propension égoïste et naturelle des hommes à afficher leurs identités spécifiques au détriment des idéaux de paix qui devraient animer l'humanité, tout un chacun devrait s'interroger car actuellement, la maitrise d’une langue étrangère (maternelle pas sûr qu’une langue étrangère soit maternelle) est une compétence capitale. D’une part, parce que le temps d’aujourd’hui est (celui de )à la mondialisation, et d’autre part, l’aptitude à parler une langue étrangère est le gage d’une accession rapide dans le domaine professionnel et scientifique (tu risque beaucoup être interrogé sur cette phrase soulignée essaye de refaire cela.besoin que tu m donnes des explications dessus ). La langue maternelle ou première (langue apprise) a un rôle primordial dans la construction de l'identité car elle est indissociable de la pensée, comme l'affirme Henri Delacroix : **"la pensée fait le langage en se faisant par le langage"** , Le fait de parler un dialecte définit plus précisément l'identité du locuteur, puisque sa langue trahit sa provenance régionale. Triste est de constatér qu’il existe une barrière entre les jeunes et les parents car ceux-ci ne parlent pas soit la même langue ou n’utilise pas le même langage, de ce fait les jeunes perdent leurs valeurs culturelles et ~~et les parents~~ délaissent leurs langues maternelles au profit des langues coloniales pour pouvoir communiquer avec leurs ~~petit enfant~~ (parents). Plus le temps passe, plus nos langues meurent, et un peuple sans langue maternelle est un peuple sans identité.

Que serait le devenir de nos cultures, langues maternelles, traditions ou coutumes si nos propres parents ne nous l’enseignent ou encore si les jeunes ne cherchent pas à les connaitre ? Comment comprendre l’argot des jeunes si on ne les côtoie pas au quotidien ?comment apprendre une langue maternelle (étrangère)~~qu’on ne connait pas~~ sans personne pour nous l’apprendre ? Comment former nos enfants (à parler nos langues) ~~en utilisant nos langues sans aller dans des centres de formations~~ ? Comment dialaoguer avec une personne parlant une langue étrangère à la nôtre ? comment prononcer (apprentissage de la phonétique) les expressions dans une langue ? Comment briser les barrières dans l’apprentissage de la langue ? Comment impliquer les personnes (enfants et parents) dans l’apprentissage de la langue ? Comment mettre en pratique l’utilisation d’une langue ?

Face à toutes ces interrogations, nous avons décidé de mettre sur pied un portail de langue qui va permettre de valoriser le patrimoine culturel~~le~~, la diversité des cultures, le multiculturalisme par le monde. Cette plateforme aura pour but de mettre en avant la culture, promouvoir les langues maternelles, réécrire la genèse des langues aux travers des histoires,s’auto-former sur une langue grâce à des cours, avoir un traducteur en toutes les langues (maternelles ou internationales) en fonction des pays et comprendre le sens de l’argot d’un pays (expressions tendances de la rue ou des jeunes). Dans le but de mener à bien la réalisation des différents modules de ce projet nous avons eu à subdiviser notre travail en quatre (04) phases. Tout d’abord nous avons réalisé une étude de l’existant et de collection des besoins; c’est durant cette phase que nous avons eu une idée claire du projet. Ensuite nous avons réalisé une analyse qui nous a permis de prendre en main notre projet, une conception qui nous a permis de ressortir tous les diagrammes, les IHM (Interface homme machine) qui nous seront utiles ainsi que le guide utilisateur. Enfin une phase d’implémentation et de test qui va nous permettre de voir le comportement des utilisateurs face au système, relevér les difficultés rencontrées afin d’apporter des améliorations.

# Partie 1 : Etat de l’art

## Chapitre 1 : Etat de l’art

### Etude de l’existant

### Présentation de la solution

### Cahier de charge

#### Contexte du projet

#### Objectifs

##### Objectif principal

##### Le projet zarb a pour objectif principal **de ressusciter nos langues maternelles et permettre de comprendre l’argot (les expressions tendnaces des jeunes). d’un pays** Objectifs spécifiques

* Apprendre nos dialectes
* Créer un réseau de mise en pratique d’une langue
* Permettre aux utilisateurs de langues ou d’origines diverses de se comprendre (communiquer)
* Vulgariser l’argot d’un pays
* Comprendre le sens des expressions utilisées dans l’argot
* Etendre l’utilisation de nos langues maternelles
* Donner la possibilité aux utilisateurs d’avoir un traducteur automatique de langue ~~maternelle~~
* ~~Communiquer sur tout ce qui gravite autour d’une langue (villages,histoires,,etc…)~~

##### Objectifs de coûts

Hébergement

Communication

#### Fonctionnalités

Afin d’être suûr d’acheminer ce projet jusqu'à un résultat rentable nous avons divisé notre travail en module

* **Module compte** : Un utilisateur doit renseigner son email ou numéro de téléphone et son mot de passe afin d’avoir accès à la plateforme. Dans le cas où il ne possède pas ces informations, nous lui donnons la possibilité de créer un compte.
  + Se connecter
  + Créer un compte
* **Module histoire** : Récit de l’origine des langues, l’histoire et l’évolution d’ une langue, une expression, un village, une coutume et une tradition
  + Ajouter des histoires
  + Modifier
  + Supprimer
  + Commenter
  + Liker
  + Rechercher
* **Module formation** : Un utilisateur peut voir la liste des cours et y adhéré. Ces cours sont publiés par des personnes qui ont le statut de formateur
  + Devenir formateur
  + Ajouter, modifier, supprimer des cours (payant, gratuit)
  + Adhérer à des cours
  + Recommander un formateur
  + Faire des quiz
  + Commenter
  + Rechercher
* **Module contribution**: Puisque nous ne possédons pas l’intégralité ~~n’avons pas toutes l~~des expressions, dles langues, dles villages et dles proverbes du monde, nous donnons la possibilité aux utilisateurs de nous faire des suggestions ou des propositions et ceux-ci sont affichés suite à une validation de l’administrateur.
  + Apporter des contributions (expressions, langues, villages, proverbes, traduction)
* **Module map**: Une carte qui présente tous les ~~les différents~~ pays du monde oùu on trouve les différentes langues, expressions, proverbes et villages ~~que nous avons~~.
* **Module fil d’actualité globale**: Ce module pemet de suivre l’actualité globale des utilisateurs de la plateforme.
  + Faire une publication
  + Commenter, partager, aimer, traduire une publication
  + Bloquer un membre
* **Module communauté**: Un regroupement d’utilisateur ayant pour but d’apprendre une langue bien précise. Les utilisateurs y accèdent après avoir adhéré. Le groupe est l’ensemble d’utilisateur qui a pour but d’apprendre une langue dont l’accès est fonction du type de groupe.
  + Adhérer à une communauté (langue)
  + Gérer des groupes (sous communautés) : créer, modifier, supprimer
  + Faire des publications
  + Commenter, partager, aimer, traduire une publication
* **Module traducteur** : Traduire un texte ou un mot en toutes les langues (maternelles ou internationales) disponibles en fonction d’un pays et en langues des signes.
* **Module notification**: Permet de notifier l’utilisateur lorsqu’un certain nombre d’action ont été faites sur la plateforme.
* **Module Chat**: Permet aux utilisateurs ayant une affinité (amis, membres~~appartenance à~~ de la même communauté et ou du même groupe, participants~~ayant adhérer~~ au même cours) d’échanger sans se soucier de la langue que parle l’autre car le texte s’affiche toujours en la langue défini par l’utilisateur.
* **Module paramètre**: Permet à l’utilisateur de paramètrer (changer la langue de traduction, changer la langue du système, changer le statut de blocage d’un membre)

#### Etude de la faisabilité

#### Ressources

##### Humains

Chef de projet pour la gestion et l’affectation des tâches allouées à toute l’équipe

Un infographe pour le design de la plateforme

Un éditeur de texte pour la correction des textes

6 traducteurs pour la vérification des traductions faites sur le site web

6 ingénieurs pour le développement web et mobile de la plateforme

##### Matérielles

* Ordinateur portable pour chaque membre de l’équipe pour pouvoir travailler même en cas de coupure de lumière.
* Un modem wifi pour une connexion constante et un travail en réseau
* Connexion internet pou pouvoir collecter les données

##### Logicielles

* EasyPHP et Wamp server : serveur web
* MySQL : SGBD
* Eclipse : IDE pour le mobile

#### Planification

La planification des tâches nous permettra de présenter la répartition des tâches sur des intervalles de temps bien délimité.

#### Communication

* Internes
  + Entre les membres de l’équipe de Management : Téléphone, Email, réunion.
* Externes
  + Auprès de tous ceux qui ont connaissance du projet : Email, Oral, exposée.

## Chapitre 2 : Méthodologie et technologie

### Présentation des méthodes de gestion de projet

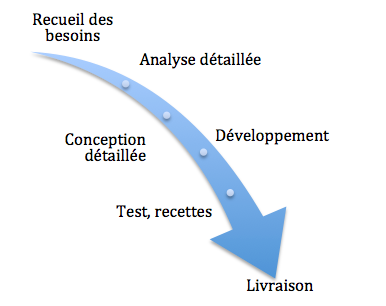
L’objectif du chef de projet est de pouvoir mener son projet à terme en respectant les délais et le budget alloué. Pour atteindre cet objectif, il doit prendre en compte **les 3C qui sont les trois contraintes que constitue le projet.**

* **Contenu**
* **Calendrier**
* **Coût**

Pour atteindre son objectif, le chef de projet peut avoir recours à plusieurs méthodes de gestion de projet. On distingue alors deux types de méthodes de gestion de projet :

* La méthode classique
* La méthode agile

#### La méthode classique

  Elle se caractérise par le recueil des besoins, définition du produit, le développer et le tester avant de le livrer. On parle alors ici d’une approche prédictive « cycle en **cascade** »

Comme son nom l’indique, il s’agit ici de prévoir des phases séquentielles où il faut valider l’étape précédente pour passer à la suivante. Le chef de projet doit alors s’engager sur un planning précis de réalisation du projet en prévoyant des jalons de débuts et fins de phases ainsi que les tâches à effectuer.

Il faut tout faire bien du premier coup car elle ne peut pas permettre de retours en arrière. Une décision ou un problème rencontré dans une phase peuvent remettre en cause partiellement ou totalement les phases précédentes validées.

Dans un cycle « en cascade » les risques sont détectés tardivement puisqu’il faut attendre la fin du développement pour effectuer la phase de test. Plus le projet avance, plus l’impact des risques augmente : il sera toujours plus difficile et coûteux de revenir en arrière lorsqu’on découvre une anomalie tardivement.

Afin d’anticiper au mieux ces risques il est nécessaire de produire des documents très détaillés en amont (recueil des besoins, cahier des charges, zoning, wireframe etc…) qui seront validés par le client. Néanmoins, ces documents restent théoriques et conceptuels jusqu’à ce que le dispositif soit testé dans des conditions réelles ; le client validera le contenu papier (conception, maquette, développement fonctionnalités etc…) mais sera toujours plus sensible à ce qu’il verra sur son écran.

Au final, du point de vue du client, c’est le chef de projet qui aurait dû anticiper ce problème alors qu’il est impossible de tout prévoir à l’avance surtout dans un environnement instable qui évolue constamment où il y a sans cesse de nouvelles technologies qui font leur apparition.

#### La méthode agile

Les méthodes agiles utilisent un principe de **développement itératif** qui consiste à découper le projet en plusieurs étapes qu’on appelle **« itérations »**.

Ces itérations sont en fait des mini-projets définis avec le client en détaillant les différentes fonctionnalités qui seront développées en fonction de leur priorité. Le chef de projet établi alors un macro-planning correspondant aux tâches nécessaires pour le développement de ces fonctionnalités.

Le but est d’assumer le fait que l’on ne peut pas tout connaître et anticiper quel que soit notre expérience. On découpe alors le projet en itérations plutôt que de tout prévoir et planifier en sachant que des imprévus arriveront en cours de route. Elle privilégie les 4 valeurs suivantes :

* Les individus et leurs interactions plus que les processus et les outils
* Des logiciels opérationnels plus qu’une documentation exhaustive
* La collaboration entre les développeurs et les clients plus que la négociation contractuelle
* L’adaptation au changement plus que le suivi d’un planning.

Les  valeurs agiles se déclinent en 12 principes communs aux **méthodes agiles.**On retrouve ces principes dans le fameux manifeste du développement agile:

1 – La priorité numéro 1 est la **satisfaction du client**. Des versions fonctionnelles du logiciel doivent être régulièrement livrées au client, de manière rapprochée dans le temps. L’agilité vient de ce jeu de ping-pong entre le développeur et le client jusqu’à ce que le client décide qu’il est satisfait avec la version de test livrée.

2 – Les développeurs doivent bien accueillir et de manière réactive des**demandes d’évolution** même s’ils sont déjà bien avancés dans le développement. L’objectif est de permettre l’acquisition d’un réel **avantage concurrentiel**pour le client.

3 -Les développeurs doivent **réduire les cycles de développement** et **livrer des versions opérationnelles** aux clients dans un laps de temps **le plus court possible**, entre 2 semaines et 2 mois.

4 – Les utilisateurs métier (clients du logiciel) et les développeurs doivent**travailler ensemble quotidiennement**. Le manifeste utilise la terminologie de “business people” pour parler des utilisateurs du logiciel.

5 – Construisez votre projet informatique avec des **individus motivés,** donnez-leur **l’environnement de travail** et le**support** dont ils ont besoin et surtout **faites-leur confiance** pour que le travail soit fait

6 – La méthode la plus efficace pour communiquer des informations à une équipe projet et au sein d’une équipe projet est la **conversation en face à face**

1. – Le bon fonctionnement du logiciel est la première mesure de progression

8 – Le processus de développement agile s’inscrit dans un processus de développement durable. C’est à dire que les développeurs et les sponsors du projet doivent pouvoir maintenir indéfiniment un rythme de travail soutenable. Il faut donc trouver le bon rythme de travail et s’y tenir.

9 – Il faut porter une attention particulière à l’excellence technique et à une bonne conception, ce qui accroît l’agilité.

10 – La simplicité est primordiale, il s’agit de l’art d’éliminer le travail inutile et superflu. Il faut savoir répondre au besoin exprimé par le client de manière informatiquement simple, avec un développement qui peut facilement évoluer.

11 Les meilleures architectures, spécifications et conceptions émanent des équipes qui s’auto-organisent. Les responsabilités ne sont donc pas l’apanage d’un chef de projet mais sont partagées par chaque membre de l’équipe agile.

12 – A intervalles réguliers, l’équipe agile s’interroge sur la manière d’améliorer encore son**efficacité**, puis règle et ajuste son comportement en conséquence.

Voici les avantages du développement itératif :

* **Meilleure qualité de la communication :**L’utilisateur à la possibilité de clarifier ses exigences au fur et à mesure
* **Meilleure visibilité :** Le client a eu meilleure visibilité sur l’avancement des travaux
* **Meilleur contrôle de la qualité :** les tests sont effectués en continu
* **Meilleure détection des risques :** Les risques sont détectés plus tôt
* **Motivation et confiance de l’équipe :** satisfaction d’atteindre un objectif fixé
* **Contrôle des coûts :** le projet peut être arrêté s’il n’y a plus de budget

#### Tableaux comparatif méthodes classique vs méthodes agiles

Nous constatons suite à ce tableau comparatif que les méthodes agiles sont les mieux adaptés pour notre projet en ce sens qu’elles :

* Nous permettrons de découper le projet principal en module (sous projet), ensuite de s’assurer que chaque module fonctionne correctement avant de les assembler pour avoir notre projet final fonctionnel.
* Offrent une meilleure adaptabilité, visibilité et gestion des risques
* Donne la possibilité au client de voir l’évolution du projet sans besoin de documentation

### Choix de la méthode Agile

Les **méthodes agiles** sont une manière de réduire le cycle de développement des projets informatiques, de répondre plus rapidement aux évolutions des demandes de l’utilisateur final versatile. Les projets informatiques agiles sont gérés de manière adaptative, incrémentale et itérative. On distingue alors plusieurs méthodes agiles, nous vous présenterons quelques une :

* **ASD (Adaptive software Development)**

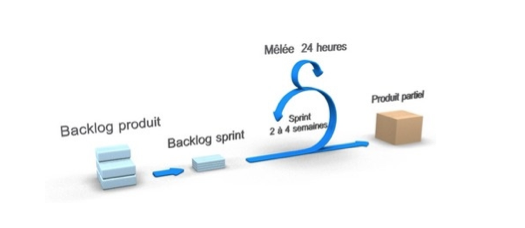
Créée par **Jim Highsmith** (signataire du Manifeste) en 2000 en publiant l’ouvrage Adaptative Software Development, a collaborative approach to managingcomplexsystems. Ses caractéristiques principales sont :

* + Focaliser sur l’objectif (mission focused)
  + Se baser sur des composants (component-based)
  + Itérer
  + Découper le temps et fixer des deadlines (timeboxing)
  + Piloter le projet par les risques (risk-drivendevelopment )
  + Accepter le changement
* [**Crystal**](http://www.access-dev.com/tag/crystal/)

Cette méthode a été mise au point par **Alistair Cockburn** (signataire du Manifeste) en 1997. Elle consiste à sélectionner la méthode applicable en fonction du nombre de personnes à coordonnées. Ses caractéristiques principales sont :

* + Des livraisons fréquentes
  + Des aménagements permanents
  + Une bonne communication interpersonnelle
  + Confiance, liberté d’expression et sécurité personnelle
  + Focus sur l’objectif et disponibilité
  + Un contact permanent avec les utilisateurs
  + Un environnement de travail approprié pour l’automatisation des tests, la gestion de configuration et les intégrations fréquentes
  + Une collaboration étroite entre toutes les parties prenantes, y compris en dehors de l’équipe
  + Une réflexion constante sur ces propriétés
* **Scrum**

Le Scrum ou « mêlée », créée par **Ken Schwaber et Jeff Sutherland (signataires du Manifeste)** en 1993, est un terme emprunté au rugby qui désigne la solidarité et la force qui lient les membres de l’équipe au succès de l’itération.Le cycle de vie de Scrum est rythmé par des itérations de quatre semaines qu’on appelle sprints.



Avant chaque [sprint](http://www.access-dev.com/tag/sprint/), on effectue une réunion de planification appelée le sprint planning meeting qui consiste à sélectionner les exigences prioritaires pour le client dans le produit [backlog](http://www.access-dev.com/tag/backlog/" \o "Voir les articles classés avec Backlog) qui seront développées, testées et livrées au client : le backlog sprint (sous-ensemble du produit backlog).

Des mêlée sont organisées quotidiennement (mêlée) durant le sprint afin de contrôler l’avancement pour s’assurer les objectifs sont tenus. A la fin du sprint, une démonstration des derniers développements est faite au client qui donnera lieu à un bilan qualitatif sur le fonctionnement de l’équipe.

Les valeurs mises en avant par cette méthode sont les suivantes :

* + Visibilité : Avoir une vision réelle sur le résultat
  + Inspection : Vérifier l’écart par rapport à l’objectif initial
  + Adaptation : S’adapter en fonction des écarts constatés afin de les ajuster. Scrum est favorable à des petits ajustements fréquents
* [**XP**](http://www.access-dev.com/tag/xp/)**(eXtremeProgramming)**

Créée en 1999par **Kent Beck, Ward Cunningham, Ron Jeffries et Palleja Xavier** adapté aux équipes réduites avec des besoins changeants. Son but principal est de réduire les coûts du changement. La première des pratiques de l’eXtremeProgramming, la plus fondamentale également, est le test automatisé. XP inclut la systématisation des tests automatisés, unitaires, fonctionnels, d’intégration, de performance, le plus est le mieux. Poussée à l’extrême, cette pratique a donné naissance au principe du TDD (Test DrivenDevelopment). Le TDD modifie sensiblement la pratique du développement. Son principe est d’une redoutable simplicité, mais sa mise en oeuvre est un véritable défi. Voici comment Henrik Kniberg résume le TDD : **‘le développeur écrit en premier lieu un test automatisé de la fonctionnalité qu’il souhaite implémenter, développe ensuite juste assez de code pour satisfaire le test et finalement remanie le code pour améliorer le design et supprimer la duplication.’**. Outre les bénéfices classiques des tests unitaires, le TDD possède deux vertus cardinales : il permet de maintenir le focus du développeur sur les fonctionnalités réellement utiles ; il a des effets spontanés et profonds sur la qualité du design.Au-delà du TDD, XP comprend un certain nombre de pratiques dont la mise en place est souvent associée à l’adoption des méthodes agiles – sans pour autant sue cette mise en place soit obligatoire.  
Voici, sans exhaustive, les plus importantes de ces pratiques :

* + le **Refactoring** ; nous l’avons déjà rapidement abordé, c’est une pratique intrinsèque du développement incrémental et qui mériterait un billet à part entière. Pour une bonne introduction aux techniques de refactoring objet.
  + l’**Intégration Continue** ; cette pratique consiste à déclencher de façon systématique le système de build – qui comprend notamment la compilation et l’exécution des tests automatisés – chaque fois qu’un membre de l’équipe valide des sources dans le référentiel projet ; c’est une pratique essentielle pour détecter et corriger au plus tôt les problèmes d’intégration des développements (principe de *failfast*)
  + la **Propriété Collective** ; la propriété collective du code consiste à ne pas spécialiser certains membres de l’équipe sur certains composants – et donc à favoriser la polyvalence au sein de l’équipe. Elle est évidemment favorisée par des pratiques telles que la programmation en binôme qui favorise l’appropriation d’une base commune de code. Les équipes avec un haut niveau d’appropriation collective de code sont réputées pour être plus robustes : un sprint ne sera par exemple pas forcément menacé par l’absence ponctuelle d’un membre de l’équipe.
  + les **Normes de développement** ; sans être une particularité de l’eXtremeProgramming, les normes de développement en sont un élément clé, facilitant les tests, l’intégration continue et l’appropriation collective de la base de code. Elles renforcent également la consistance des APIs.
  + la Programmation en binôme (**Pair Programming**), qui consiste à développer à deux sur un même poste ; cette pratique, souvent mal comprise, n’est que rarement systématisée. Il ne faut par contre pas s’interdire son utilisation ponctuelle, par exemple sur une nouvelle technologie, une problématique pointue, un fonctionnel plus complexe, ou plus simplement, si l’un des membres de l’équipe demande de l’assistance. C’est aussi une pratique utile lors des montées en charge de l’équipe : elle permet d’intégrer plus rapidement les nouveaux développeurs et de leur communiquer la culture de l’équipe.
  + un **Rythme soutenable** ; ce principe est commun à toutes les méthodes agiles, et directement issu du *heijunka* Lean ; il part du principe qu’il n’y a rien de plus contre-productif à moyen terme que de maintenir une équipe de développement sous la pression d’une charge de travail supérieure à sa capacité de production – cela revient à étrangler la poule aux œufs d’or et se traduit généralement par une baisse massive de la qualité et de la motivation.

Cette méthode s'appuie sur :

* + une forte réactivité au changement des besoins du client ;
  + un travail d'équipe ;
  + la qualité du code ;
  + la qualité des tests effectués au plus tôt.

Parmi les méthodes agiles citer plus haut nous avons choisis de travailler avec la méthode XP car :

* Elle est efficace pour les équipes de développement donc le nombre n’est pas important (4 ou 5 développeurs)
* Elle prône la communication entre les développeurs
* Elle met en avant Le retour d'information qui est primordial pour le programmeur et le client
* Elle demande du courage car Le courage permet de sortir d'une situation inadaptée
* Elle prône l’intégration continue

### Choix de la méthode d’analyse

Pour programmer une application, il ne convient pas de se lancer tête baissée dans l’écriture du code : il faut d’abord organiser ses idées, les documenter, puis organiser la réalisation en définissant les modules et étapes de la réalisation. Cette démarche antérieure à l’écriture que l’on appelle modélisation a pour produit un modèle.

Il existe plusieurs méthodes d’analyse parmi lesquelles :

* **Merise (Méthode d’Etude et de Réalisation Informatique par les Systèmes d’Entreprise).** Elle repose sur 5 principes fondamentaux qui ont précédé à son élaboration (l’approche systémique, les cycles de construction des systèmes d’information, l’approche fonctionnelle, la vision duale des données-traitement et l’approche du général au particulier).
* **UnifiedProcess utilisant le langage de notation UML (UnifiedModelingLanguage)** est un langage graphique de modélisation de données et de traitements.
* **SADT (StructuredAnalysis and Design Technique)** est une démarche systémique de modélisation d’un système complexe ou d’un processus opératoire.
* **NIAM** est une méthode d’analyse et de conception pour les systèmes d’information.
* **OMT (Object Modeling Technique)** est une méthode qui permet de couvrir l’ensemble des processus d’analyse et de conception en utilisant le même formalisme. L’analyse repose sur les trois points de vue : statique, dynamique, fonctionnel. Donnant lieu à trois sous-modèles.
* **Booch** est une méthode qui permet de faciliter l’implémentation de programmes dans des langages de programmation orientée objet, ainsi que de représenter les différentes phases du développement d’un projet.
* **OOSE** est une méthode de développement créée par Ivar Jacobson, caractérisée par la définition des « use cases » (cas d’utilisation). Elle a été intégrée dans UML à partir de 1995.
* **2TUP** est un processus de développement logiciel qui bénéficie de la maturité de nombreuses méthodes telles qu’OOSE, BOOSH, OMT.

Parmi les méthodes d’analyse orienté objet (UML et Merise 2.0), nous avons été amené à choisir **UML** pour éprouver nos connaissances acquises durant notre formation.

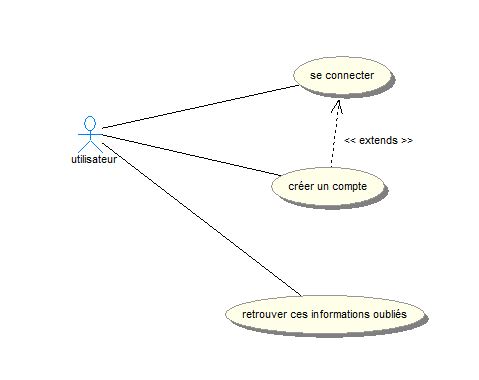
UML est un métalangage car il fournit les éléments permettant de construire le modèle qui, lui, sera le « *langage*» du projet. Il est impossible de donner une représentation graphique complète d’un logiciel, ou de tout autre système complexe, de même qu’il est impossible de représenter entièrement une statue (à trois dimensions) par des photographies. Mais il est possible de donner sur un tel système des vues partielles, analogues chacune à une photographie d’une statue, et dont la juxtaposition donnera une idée utilisable en pratique sans risque d’erreur grave. UML 2.0 comporte ainsi treize types de diagrammes dont quelques-uns sont représentés dans la suite. Ils représentent autant de vues distinctes pour représenter des concepts particuliers du système d’information.

# Partie 2 : Analyse, conception et implémentation de la solution

## Chapitre 3 : Analyse et conception de la solution

### Module de compte (inscription, connexion, mot de passe oubliée)

#### Diagramme de cas d’utilisation



#### Diagramme de séquence

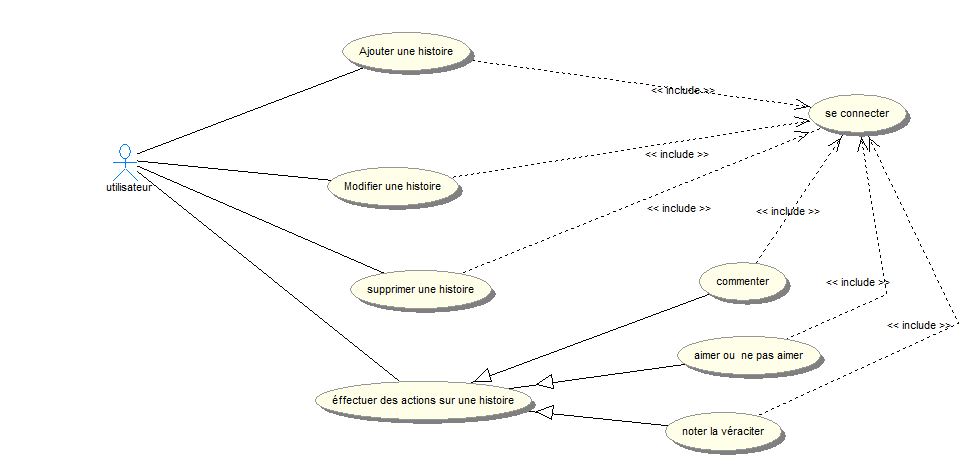
#### Diagramme de classe

#### Diagramme de déploiement

### Module de histoire

Récit de l’origine, l’histoire et l’évolution sur une langue, une expression, un village, une coutume et une tradition

#### Diagramme de cas d’utilisation



#### Diagramme de séquence

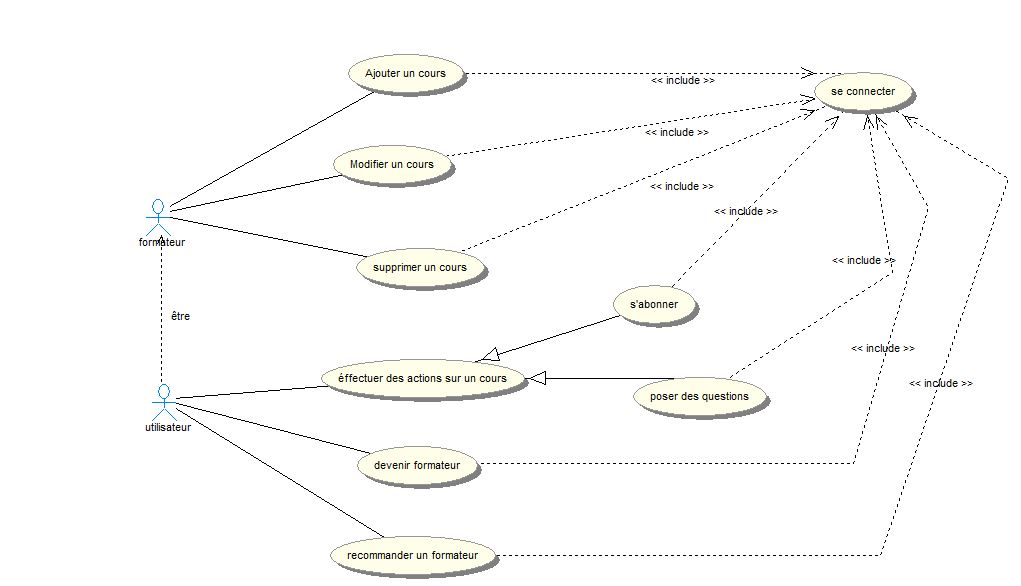
#### Diagramme de classe

#### Diagramme de déploiement

### Module formation

Un utilisateur peut voir la liste des cours et y adhéré. Ces cours sont publiés par des personnes qui ont le statut de formateur

#### Diagramme de cas d’utilisation



#### Diagramme de séquence

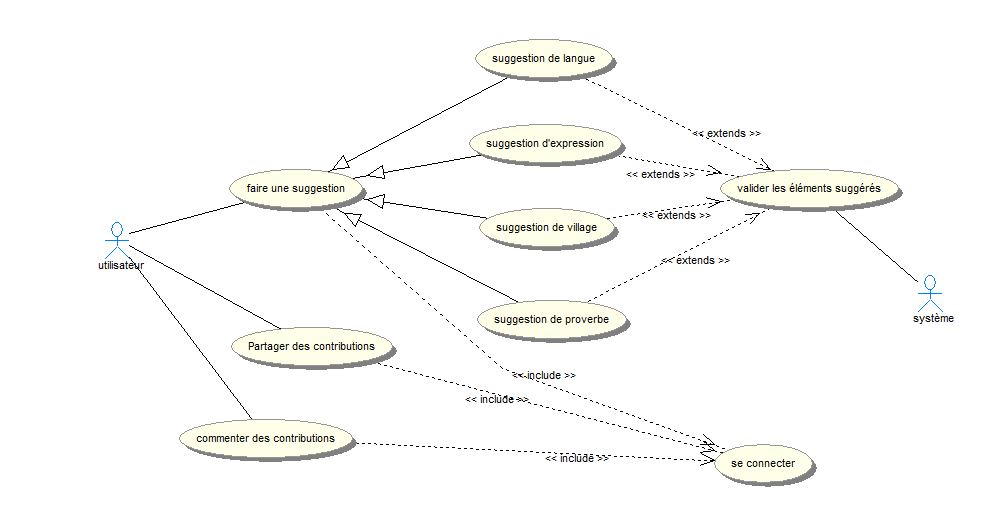
#### Diagramme de classe

#### Diagramme de déploiement

### Module contribution

Puisque nous n’avons pas toutes les expressions, les langues, les villages et les proverbes du monde, nous donnons la possibilité aux utilisateurs de nous faire des suggestions et ceux-ci sont affichés suite à une validation.

#### Diagramme de cas d’utilisation



#### Diagramme de séquence

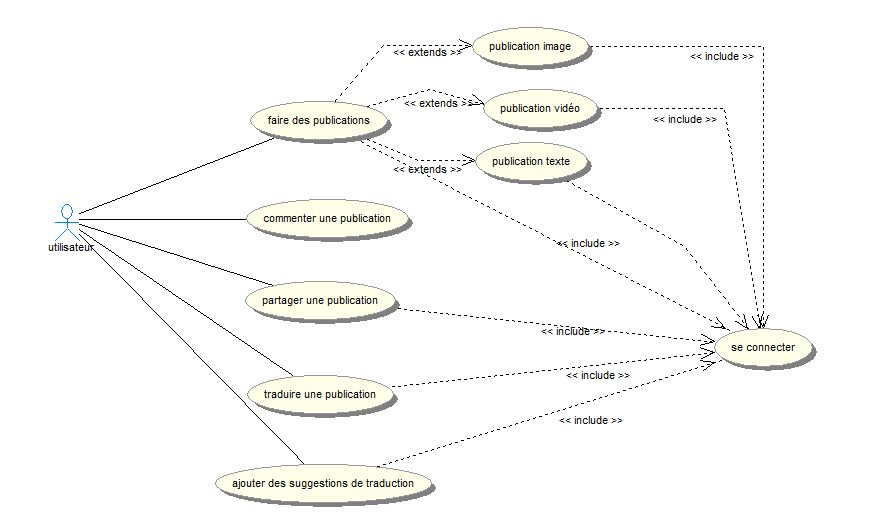
#### Diagramme de classe

#### Diagramme de déploiement

### Module fil d’actualité

Les utilisateurs voient les différentes publications en fonction de la page ou il se trouve.

#### Diagramme de cas d’utilisation



#### Diagramme de séquence

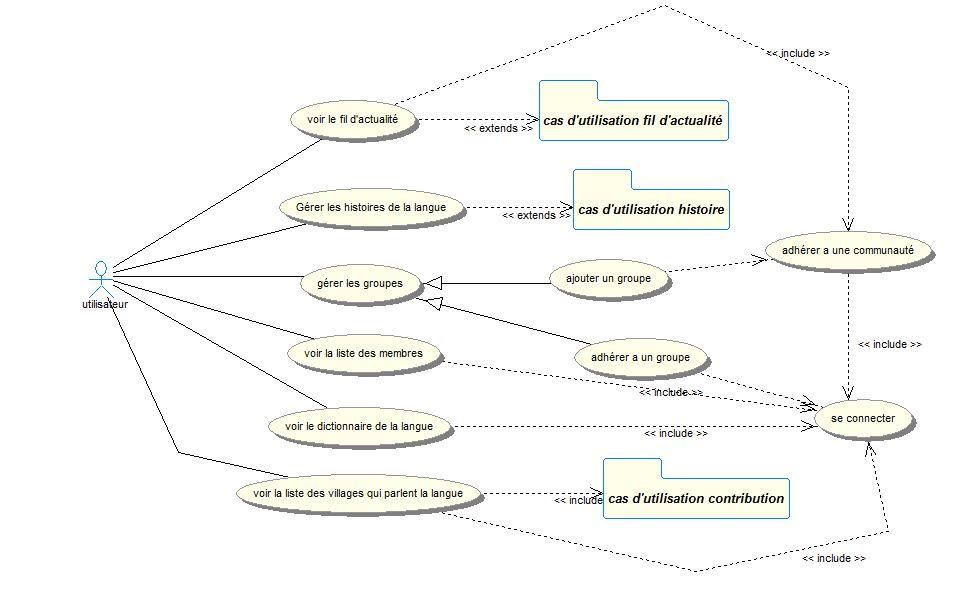
#### Diagramme de classe

#### Diagramme de déploiement

### Module communauté

Un regroupement d’utilisateur ayant pour but d’apprendre une langue. Les utilisateurs y accèdent après avoir adhéré. Le groupe est ensemble d’utilisateur qui a pour but d’apprendre une langue dont l’accès est fonction du type de groupe.

#### Diagramme de cas d’utilisation



#### Diagramme de séquence

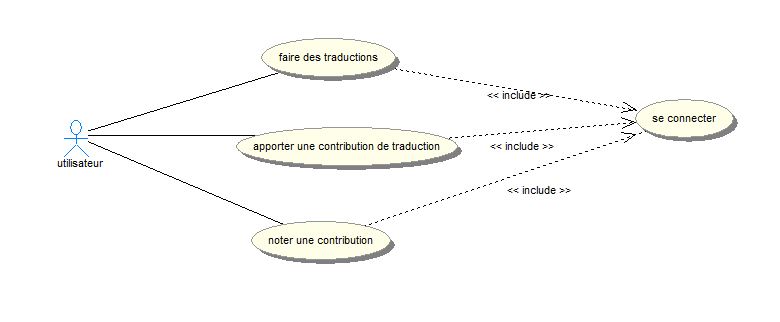
#### Diagramme de classe

#### Diagramme de déploiement

### Module traducteur

Fait des équivalences d’une expression en une langue bien précise vers une autre langue sélectionné.

#### Diagramme de cas d’utilisation



#### Diagramme de séquence

#### Diagramme de classe

#### Diagramme de déploiement

1. Diagramme de classe global
2. Modèle conceptuel des données

## Chapitre 4 : Implémentation de la solution

### Outil de développement et langage utilisée

### Architecture

### Algorithme développé

#### Algorithme de traduction écrite ou vocale

#### Algorithme de phonétique

### Interface graphique

# 

# CONCLUSION

# REFERENCES BIBLIOGRAPHIQUES

# Annexes