

|  |
| --- |
| SIT213 Etape 1  IMT Atlantique |
| Chaire M@D - Maintien à domicile | IMT Atlantique |
| 5 septembre  Créé par :  LE DUC Elouan LE GRUIEC Clément DEMOULIN BENJAMIN |

Table des matières

[Objectifs généraux 3](#_Toc50224016)

[1ère étape du projet 4](#_Toc50224017)

[ Principe 4](#_Toc50224018)

[ Analyse des actions à mener 5](#_Toc50224019)

[ Programmation et connexion des blocs 6](#_Toc50224020)

[ Mise en place des liaisons entre les blocs 6](#_Toc50224021)

[ Calcul du TEB 7](#_Toc50224022)

[ Réalisation des tests 7](#_Toc50224023)

[Capitalisation du travail réalisé 8](#_Toc50224024)

[Conclusion 9](#_Toc50224025)

[Table des illustrations 10](#_Toc50224026)

# Objectifs généraux

Il s’agit de réaliser, par équipe de 4 ou 5 élèves, une maquette logicielle (en Java) simulant un système de transmission numérique élémentaire. On intégrera donc dans la chaîne un bloc de modulation numérique.

Le système sera assemblé suivant une bibliothèque de modules comportant des ports d’entrée, des ports de sortie et des paramètres physiques. Ces derniers pourront être déterminés à partir des activités du module SIT 212. Le système global sera mis au point progressivement sur 5 séances au cours desquelles les modules seront raffinés, complétés, validés et connectés selon un schéma de transmission de type « point-à-point ».

Outre la qualité technique de la réalisation, on insistera sur les points suivants :

1. La qualité de documentation de la maquette logicielle (notamment la Javadoc).
2. Les efforts de validation des résultats de simulation produits par la maquette.
3. La maîtrise du processus de travail : gestion des versions successives de la maquette logicielle et du dossier technique afférent, synergie de l'équipe, démarche qualité. Concernant ce tout dernier critère, le respect des exigences de mise en forme du livrable sera primordial.

# 1ère étape du projet

## Principe

Transmission élémentaire "back-to-back". On introduira le premier modèle de transmission schématisé sur la figure 1. Il vérifie les propriétés suivantes :

* La source émet une séquence booléenne soit fixée, soit aléatoire.
* Le transmetteur logique parfait se contente, à la réception d’un signal, de l’émettre tel quel vers les destinations qui lui sont connectées.
* La destination se contente de recevoir le signal du composant sur lequel elle est connectée.
* Des sondes logiques permettent de visualiser les signaux émis par la source et le transmetteur parfait.
* L’application principale calcule le taux d’erreur binaire (TEB) du système.



Figure 1 Modélisation de la chaîne de transmission à l’étape 1.

Par défaut le simulateur doit utiliser une chaîne de transmission logique, avec un message aléatoire de longueur 100, sans utilisation de sondes et sans utilisation de transducteur.

L’option *-mess m* précise le message ou la longueur du message à émettre :

* Si *m* est une suite de 0 et de 1 de longueur au moins égale à 7, m est le message à émettre.
* Si m comporte au plus 6 chiffres décimaux et correspond à la représentation en base 10 d'un entier, cet entier est la longueur du message que le simulateur doit générer et transmettre. Par défaut le simulateur doit générer et transmettre un message de longueur 100.

L’option *-s* indique l’utilisation des sondes. Par défaut le simulateur n’utilise pas de sondes

## Analyse des actions à mener



Figure Diagramme de Classe du projet simulateur

Ainsi nous devons mettre en œuvre 4 blocs de fonction pour mener à bien l’’étape1.

## Programmation et connexion des blocs

### Mise en place des liaisons entre les blocs

public Simulateur(String[] args) throws ArgumentsException {

        //Analyse des arguments

        analyseArguments(args);

        //Instanciations des differents blocs de traitement

        if (!messageAleatoire) {

            source=new SourceFixe(messageString);

        } else {

            source=new SourceAleatoire();

        }

        transmetteurLogique = new TransmetteurParfait();

        destination = new DestinationFinale();

        //Instanciations des differentes sondes

        SondeLogique viewSrc = new SondeLogique("ViewSrc", 720);

        SondeLogique viewTransmit = new SondeLogique("ViewTransmit", 720);

        SondeLogique viewDest = new SondeLogique("ViewDest", 720);

        //connexion des blocs ensembles

        source.connecter(transmetteurLogique);

        source.connecter(viewSrc);

        transmetteurLogique.connecter(destination);

        transmetteurLogique.connecter(viewTransmit);

}

### Calcul du TEB

public float calculTauxErreurBinaire() {

        int nbErr=0;

        float TEB=0.0f;

        for (int i = 0; i < destination.getInformationRecue().nbElements(); i++) {

            if(destination.getInformationRecue().iemeElement(i)!=source.getInformationEmise().iemeElement(i)) nbErr++;

        }

        TEB=(nbErr\*1.0f)/(source.getInformationEmise().nbElements());

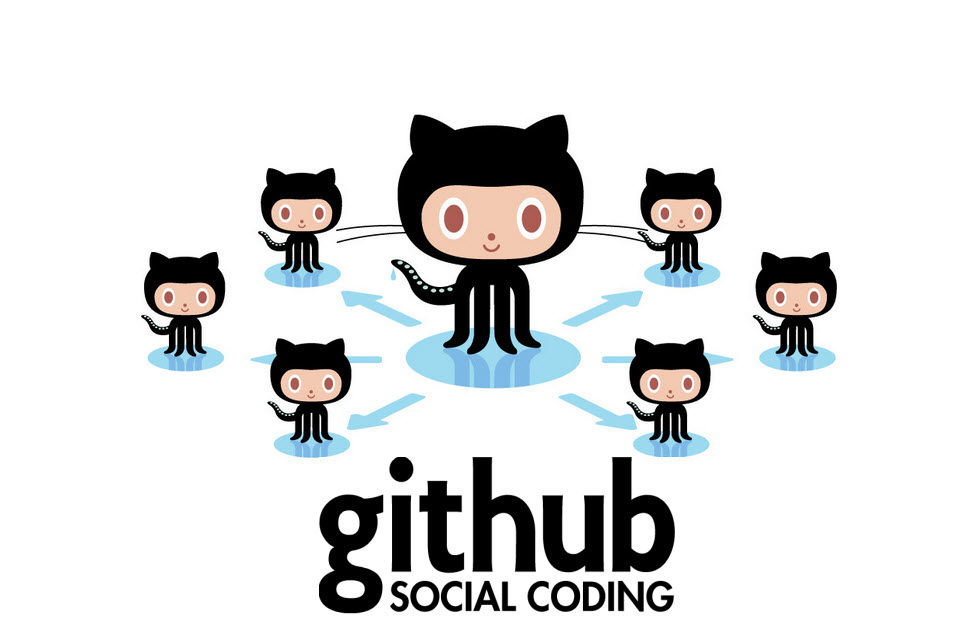
        return TEB;

}

## Réalisation des tests

# Capitalisation du travail réalisé

Github



# Conclusion

# Table des illustrations

[Figure 1 Modélisation de la chaîne de transmission à l’étape 1. 4](#_Toc50222925)

[Figure 2 Diagramme de Classe du projet simulateur 5](#_Toc50222926)