Formation d'ingénieur en partenariat

UV SIT 210

Module SIT 213

Atelier logiciel : *simulation d'un système de transmission*

— Énoncé —

Année scolaire 2019-2020

Équipe pédagogique :

Éric Cousin
Bruno Fracasso
Julien Mallet
François-Xavier Socheleau

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

1. Présentation du module

Ce module est un atelier logiciel organisé en 8 séances, dont une dédiée à la restitution du travail. Il permet d'illustrer et d'approfondir de manière transversale et expérimentale les activités des deux autres modules de l'UV SIT210, à savoir :

- SIT 211 : modélisation et validation des logiciels
- SIT 212 : simulation de signaux et briques de transmission.

- **Objectifs**

Il s'agit de réaliser, par équipe de 3 ou 4 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.

2. Principe de l'atelier

2.1 Un travail d'équipe

Dans le cadre du module SIT211, des outils/méthodes d'ingénierie logicielle sont abordées. Sur la base de ces compétences et en tenant compte de ses acquis antérieurs, chaque équipe établira son propre cadre de travail. Elle s'y conformera tant que faire se peut, ou le fera évoluer si besoin et analysera sa pertinence en fin d'atelier.

2. 2 Un travail itératif

Le principe de l'atelier repose sur une élaboration par étapes du système, en introduisant au fur et à mesure des spécifications plus précises du système demandé. Ces étapes sont définies à l'avance par l'équipe pédagogique. Cinq itérations sont prévues. En début de chaque étape devra être explicitée la **liste des paramètres pertinents** de la simulation et les résultats attendus par le groupe, que l'on confrontera systématiquement aux **résultats observés en fin d'étape**.

Les instructions valables pour chaque itération sont les suivantes :

a) une livraison intervient à la fin de chaque itération ;

b) la livraison consiste à déposer sur Moodle 2 fichiers : le rapport et le logiciel. Chacun de ces fichiers devra respecter les exigences indiquées ci-après.

c) le rapport est un **compte-rendu** actualisé de projet donnant au moins les faits marquants et les observations faites sur la base du simulateur. Comme indiqué précédemment, ce compte-rendu devra **confronter** les résultats observés aux prévisions initiales ;

d) les délais doivent bien sûr être respectés. Par défaut, la date limite de dépôt est fixée à 23h55 la veille du début de l'itération (séance) suivante.

La dernière séance de l'atelier est dédiée à la présentation du travail effectué. À titre indicatif, le déroulement en 2018 était le suivant :

a) Les équipes **présentent** à tour de rôle, en dix minutes :
- un bilan organisationnel,
- une présentation technique du logiciel au client, avec démonstration possible.

b) Cinq minutes de **questions/réponses** suivent chaque présentation.

Format des livrables attendus

Le **logiciel** sera livré dans une archive de nom : `Nom1-Nom2-Nom3-Nom4.tar.gz`, où les `Nomi` sont les noms de famille des membres de l'équipe commençant par une majuscule, classés par ordre alphabétique. L'archive contient, directement à sa racine :

- un dossier `src` avec votre code Java ;
- un dossier `bin` où seront générés – à la demande – les .class ;
- un dossier `docs` où sera générée – à la demande – la javadoc ;
- un script de compilation du logiciel nommé `compile` ;
- un script de génération de la javadoc nommé `genDoc` ;
- un script de nettoyage du dossier nommé `cleanAll`, qui supprime tous les fichiers générés par les autres scripts ;
- un script de lancement d'une simulation nommé `simulateur` et conforme aux exigences indiqués spécifiquement dans le document commande unique ;
- un script d'autotests nommé `runTests` pour exécuter les éventuels auto-tests du logiciel ; ceux-ci devront clairement indiquer si tout est OK ou s'il y a une anomalie, et s'il n'y a pas d'auto-test (pour l'instant), ce script devra afficher un message le notifiant (mais le script lui-même devra être présent) ;
- un fichier `readme` explicatif global du contenu du dossier et de l'utilisation du logiciel.

Le **rapport** sera livré au format `pdf`, et à chaque itération :

- il contiendra une nouvelle section concernant la nouvelle étape,
- les éventuelles modifications apportées à la partie préexistante seront clairement mises en évidence (surlignage ou utilisation d'une couleur spécifique).

Ce rapport sera nommé `Nom1-Nom2-Nom3-Nom4-X.pdf` où X est le numéro d'étape (1, 2…) et `Nomi` sont les noms de famille des membres de l'équipe commençant par une majuscule et classés par ordre alphabétique.

Critères d'évaluation

Le travail est jugé autant sur la qualité des produits livrés que sur la rigueur des modalités suivies.

3. Travail demandé

3.1. Chaîne de transmission : généralités et vocabulaire

Le schéma général d'un système de transmission est rappelé sur la figure 1.

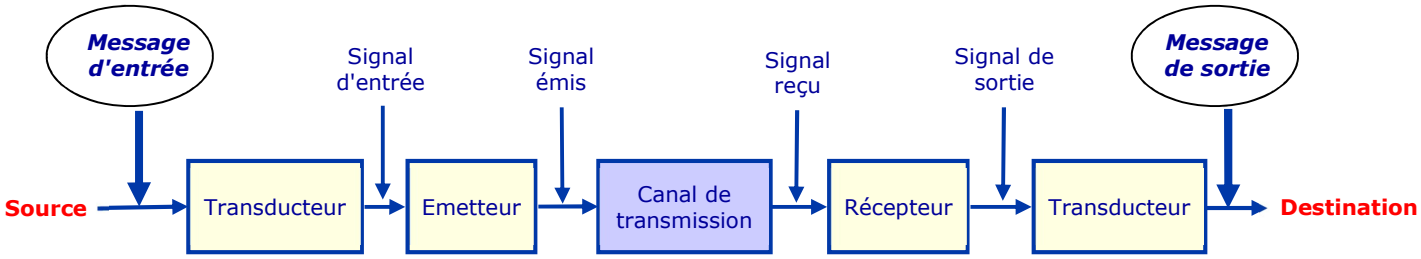

Figure 1. *schéma élémentaire d'une chaîne de transmission*

L'objectif consiste à transmettre un **message** d'un point d'entrée à un point de sortie, via un **canal de transmission** (ou de communication). Le message d'entrée est émis par une **source** d'entrée. Les messages considérés dans cet atelier seront des suites de **symboles binaires** (0 ou 1) correspondant à des informations échantillonnées et quantifiées sur deux niveaux logiques. Le message de sortie sera – autant que faire se peut - semblable au message d'entrée. Ce dernier étant incapable de traverser le canal de propagation tel quel, on l'adaptera aux caractéristiques physiques du canal en le convertissant au moyen d'un **transducteur** en un « vecteur » adapté à la transmission, appelé **signal**. Ce dernier sera injecté dans le canal au moyen d'un **émetteur**. À l'autre extrémité du canal, il sera récupéré et traité par le **récepteur** et le transducteur de réception.

Les principaux canaux de transmission rencontrés dans la nature sont : le canal Hertzien (espace libre), le canal guidé électrique (câble), le canal guidé Optique (fibre), le canal acoustique aérien et le canal acoustique sous-marin. Chaque canal de propagation devant être utilisé à une fréquence bien particulière, le message est transposé autour de cette fréquence par l'opération de **modulation**. En outre, le canal sera une source de bruit pour les signaux qu'il transporte. Les principales sources de bruit rencontrées en pratique sont : la dispersion de trajets, la dispersion chromatique, le bruit de détection (grenaille), le bruit thermique et le bruit d'amplification.

Par la suite, chaque composant du système de transmission entre la source et la destination sera dénommé **transmetteur**.

3.2. Modélisation « objet » de la chaîne de transmission

Dans le cadre de l'atelier, il conviendra de se conformer à la modélisation pré-établie indiquée sur la figure 2. Une chaîne de transmission sera ainsi composée :

- d'une source (une instance d'une classe héritant de la classe abstraite `Source`)

- d'un ou plusieurs transmetteurs (des instances de classes héritant de la classe abstraite `Transmetteur`)
- d'une destination (une instance d'une classe héritant de la classe abstraite `Destination`)
- d'une à plusieurs sondes (des instances de classes héritant de la classe abstraite `Sonde`). Plusieurs sondes sont mises à votre disposition.

Votre travail consiste donc à élaborer les parties permettant de simuler la chaîne de transmission complète.

Le programme principal lancera la simulation demandée et calculera le taux d'erreur binaire (TEB) du système en comparant la séquence émise par la source et celle reçue par la destination.

À l'issue de chaque étape de l'atelier, ce TEB devra être commenté dans le compte-rendu de projet.

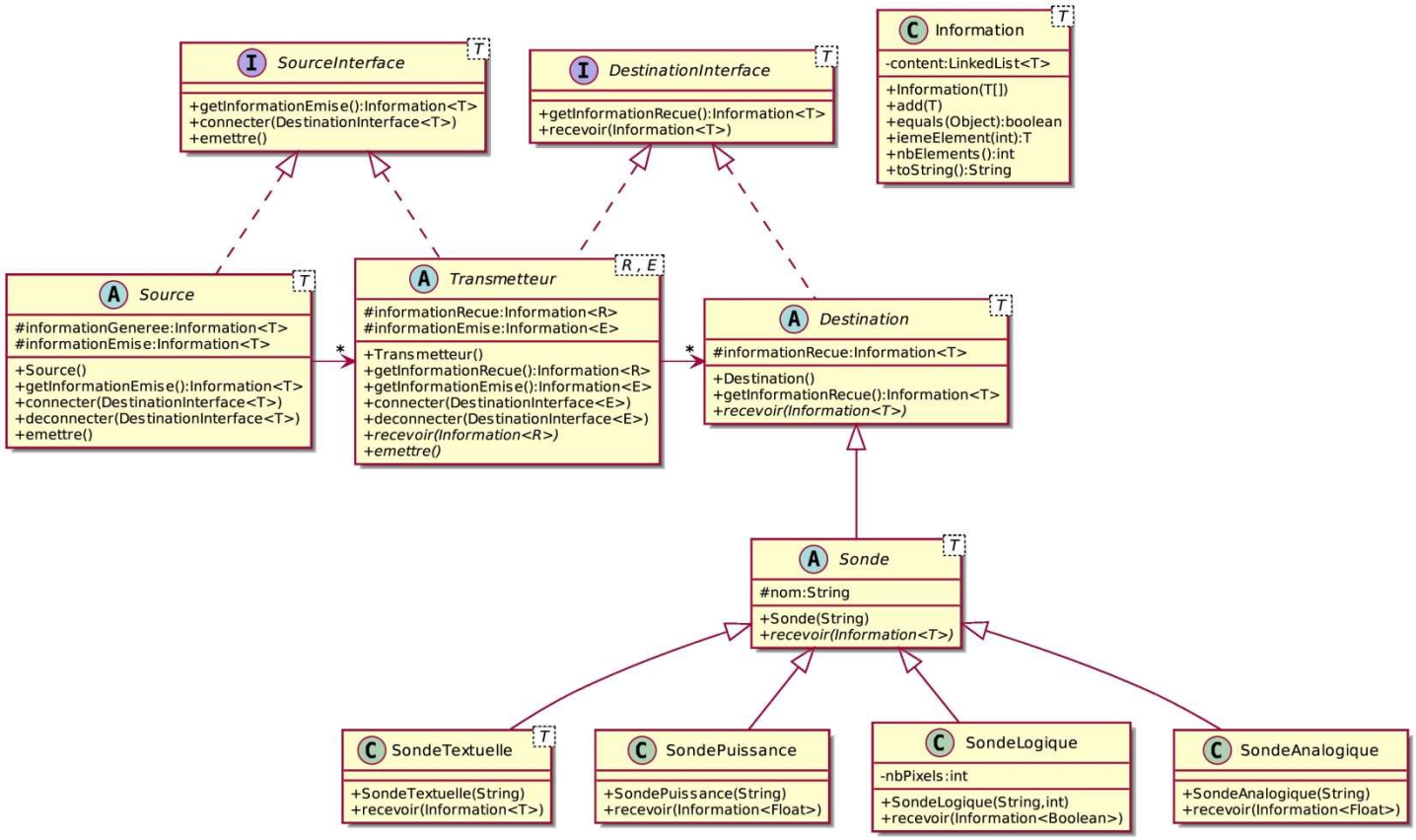

Figure 2. *Arborescence des classes de la chaîne de transmission proposée*

3.3 Structuration de l'atelier

L'atelier logiciel s'articulera autour des huit étapes suivantes :

► Étape 0 : *déploiement de logiciel.* L'objectif de la séance est de préparer le processus de livraison/déploiement des différentes étapes du projet. En effet, à chaque étape – et donc pour chaque livraison - le respect des exigences de mise en forme des livrables en conditionnera la prise en compte. Ce TP doit vous permettre de prendre un bon départ.

► Étape 1 : *transmission élémentaire "back-to-back"*. On introduira le premier modèle de transmission schématisé sur la figure 3. 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.

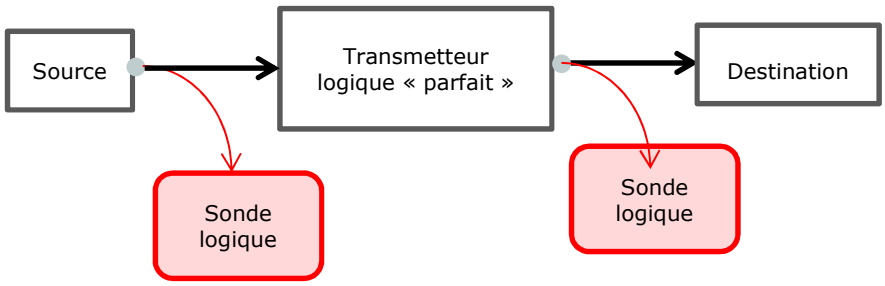

Figure 3. *Modélisation de la chaîne de transmission à l'étape 1.*

► Étape 2 : *transmission non bruitée d'un signal analogique*. On prendra en compte la nature analogique du canal de transmission en faisant évoluer la chaîne de transmission par l'adjonction de **deux étages** (logique → analogique et analogique → logique), comme indiqué sur le schéma de la figure 4.

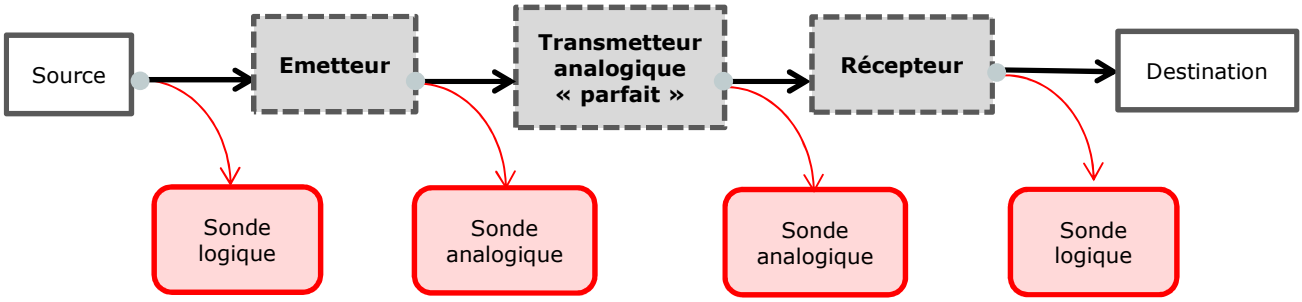

Figure 4. *Modélisation de la chaîne de transmission à l'étape 2.*

► Étape 3 : *transmission non-idéale avec canal bruité de type « gaussien »* (cf figure 5). La propagation dans le canal est modélisée de manière théorique par un bruit blanc additif gaussien qu'il conviendra de régler en fonction des paramètres du transmetteur vus à l'étape précédente. Vous devrez particulièrement surveiller le TEB en réception.

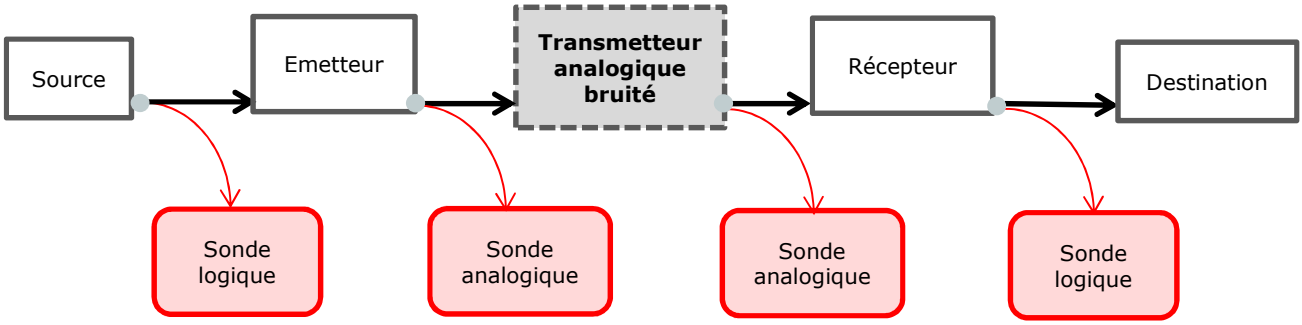

Figure 5. *Modélisation de la chaîne de transmission à l'étape 3.*

► Étape 4a : *transmission non-idéale avec divers bruits « réels »*. Le transmetteur est maintenant bruité selon un ou plusieurs schémas de perturbations aléatoires rencontrées dans la nature. Des modèles physiques de bruit seront considérés à ce stade tels que : trajets-multiples, dispersion chromatique, bruit électronique (thermique, quantique). Vous pourrez les sélectionner à partir de vos cours d'ELP et des activités vues dans le module SIT 212.

► Étape 4b : transmission non-idéale avec divers bruits « réels » plus le bruit « gaussien » réalisé à l'étape 3.

► Étape 5a : *codage de canal*. Pour prendre en compte les propriétés typiques du message émis par une source, il est parfois utile d'adapter la séquence booléenne transmise. Ceci peut se faire en rajoutant ou supprimant certaines valeurs à la séquence à transmettre, et, bien évidemment, en réalisant une opération réciproque à l'arrivée (figure 6).

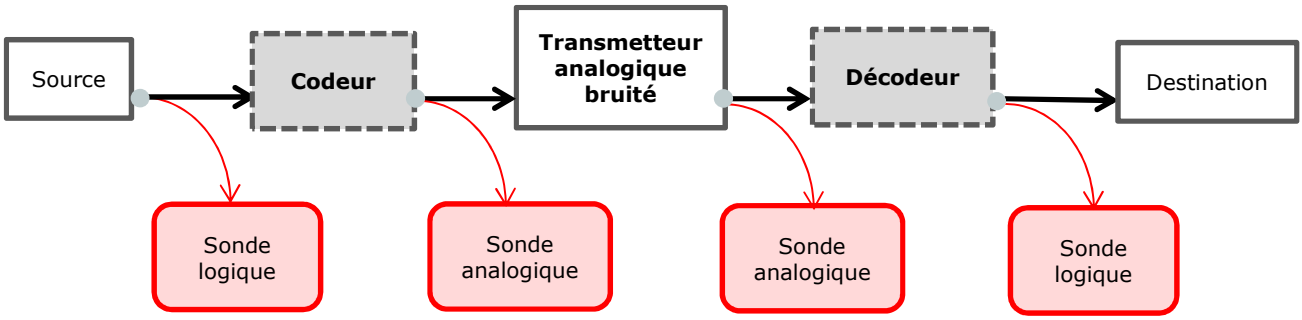

Figure 6. *Modélisation de la chaîne de transmission à l'étape 5a.*

Cette étape doit permettre de diminuer le TEB de votre chaîne de transmission bruitée à l'étape précédente. Cela peut également être l'occasion d'utiliser le formalisme des automates pour une implémentation concrète.

► Étape 5b : *codage de canal avec canal bruité de type « gaussien »*. Cette étape doit permettre de diminuer le Taux d'Erreur Binaire de votre chaîne de transmission bruitée de l'étape 3.

► Étape 6 : il s'agira d'utiliser votre simulateur sur un cas d'étude soumis par le client. Cet exercice sera l'occasion de prendre du recul sur votre réalisation et d'analyser l'impact de ses ingrédients techniques sur le résultat final.

► Étape 7 : séance de livraison du logiciel au client (présentation orale et démonstration).