
# Simulation de réseaux radio sous OPNET Modeler

ARC IRAMUS
19 Mai 2005

Fabrice Valois

fabrice.valois@insa-lyon.fr


# Agenda

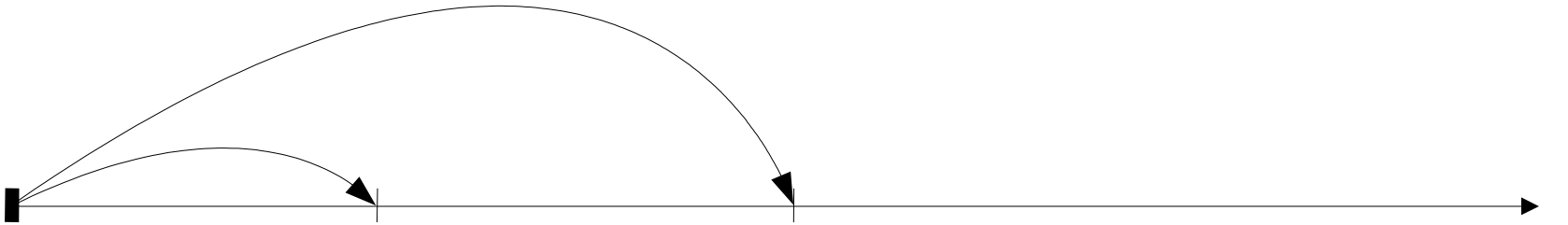
- Un peu de rappel sur la simulation
  - Échéancier à temps discret
  - Intervalles de confiance et fiabilité des résultats
- Un modèle OPNET Modeler
- Modélisation radio sous Modeler
  - Pipeline stage
  - Modèle par défaut

# I - Simulateur à événement discret


***T=0***
*Début de la simulation*
*Initialisation*

- 1 échéancier d'événements *à temps discret*
- 1 simulation d'horloge
- Des variables d'états du système
- Des routines liées aux événements
- Des routines I/O
- 1 outil de *reporting*
- Un gestionnaire de traces
- Une gestion de la mémoire dynamique
- Un programme principal
- Des fonctions d'initialisation

# Simulateur à événement discret




***T=0***
*Début de la simulation*
*Initialisation*

⇓

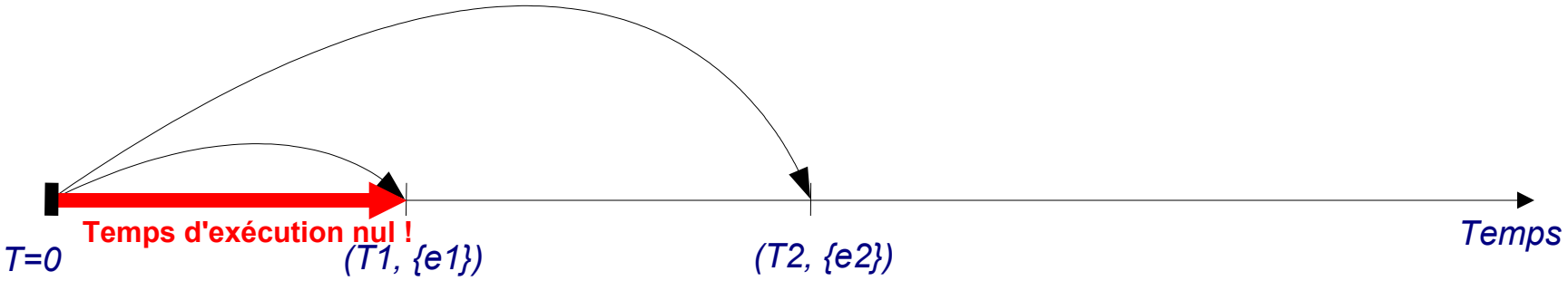
*SIMULATION*

*(T1, {e1})*

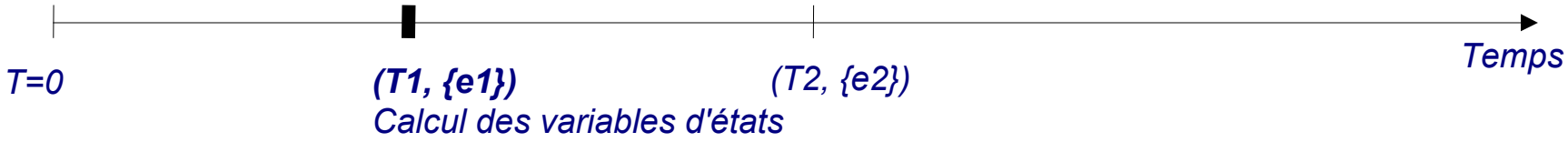
*(T2, {e2})*

*Événements futurs*

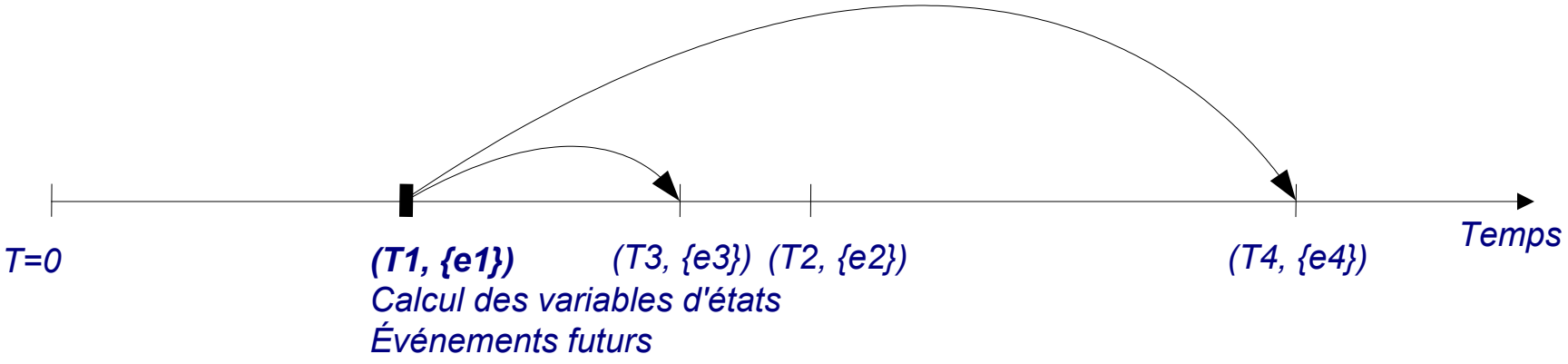
# Simulateur à événement discret



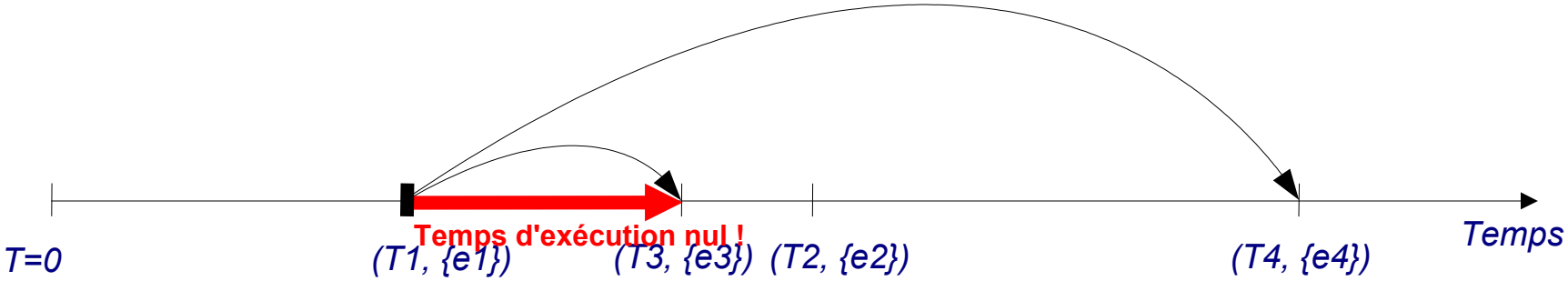
# Simulateur à événement discret



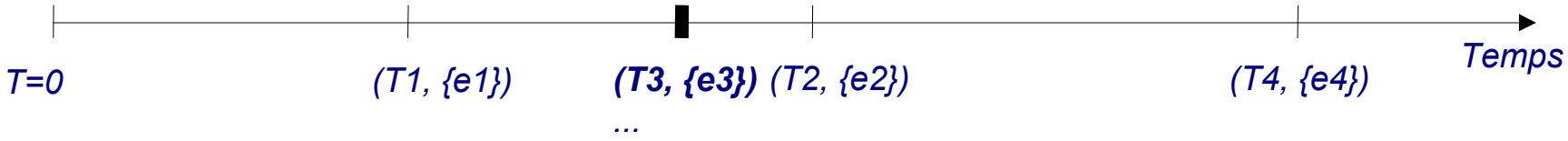
# Simulateur à événement discret



# Simulateur à événement discret



# Simulateur à événement discret

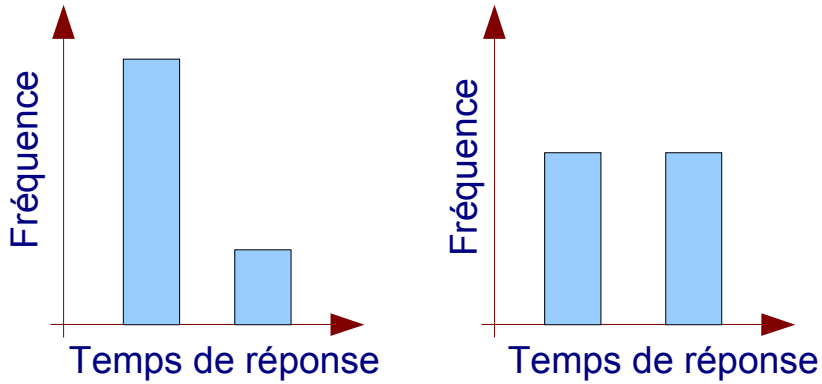


# II - Intervalles de confiance (1/3)

- Pour avoir des résultats fiables il faut :
  - Simuler suffisemment longtemps (mais pas trop...)
  - Vérifier qu'un nombre significatif d'événements ont été réalisés
  - Vérifier le temps moyen entre les événements
  - Tester plusieurs trajectoires du processus
- Notion d'intervalles de confiance et/ou d'indices de dispersion

# Intervalles de confiance (2/3)

- Sur une courbe, on résume un ensemble de valeurs obtenues par un point... Est-ce vraiment suffisant ?




Traduction de la variabilité des indicateurs de performances par des *indices de dispersion* :

  - Valeur mini-maxi observée
  - Variance ou déviation standard
  - Déviation absolue moyenne

# Intervalles de confiance (3/3)

- Soit un échantillon de n observations $\{x_1, x_2, ..., x_n\}$

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \qquad \text{où : } \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

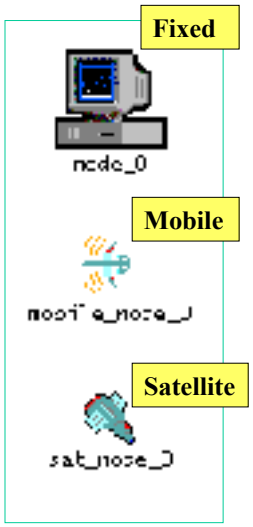
$s^2$ : variance simple et $\sqrt{s}$ : déviation standard (~écart-type)

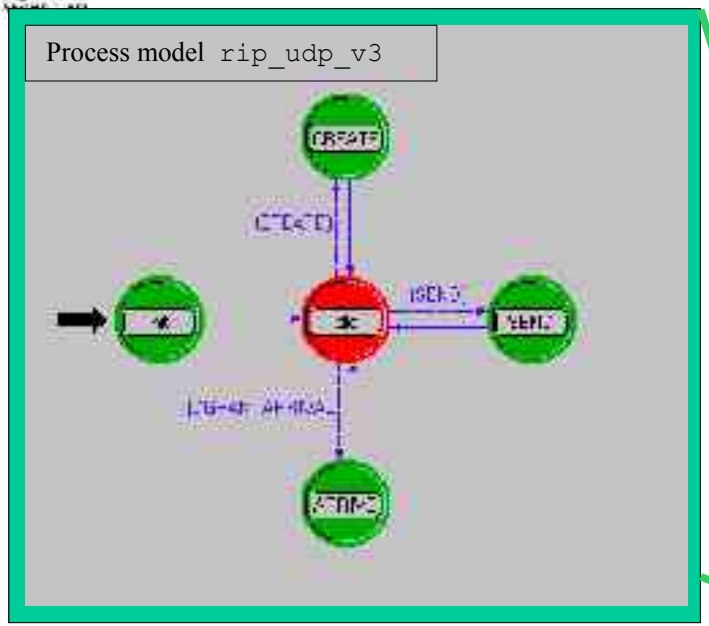
pour des raisons d'unité, on prend plutôt la déviation standard

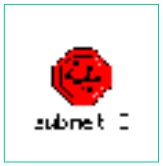
- Déviation absolue moyenne :

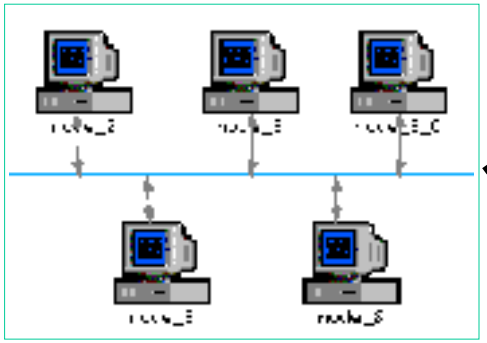
$$\text{Mean absolute deviation} = \frac{1}{n}\sum_{i=1}^{n}|x_i - \bar{x}|$$

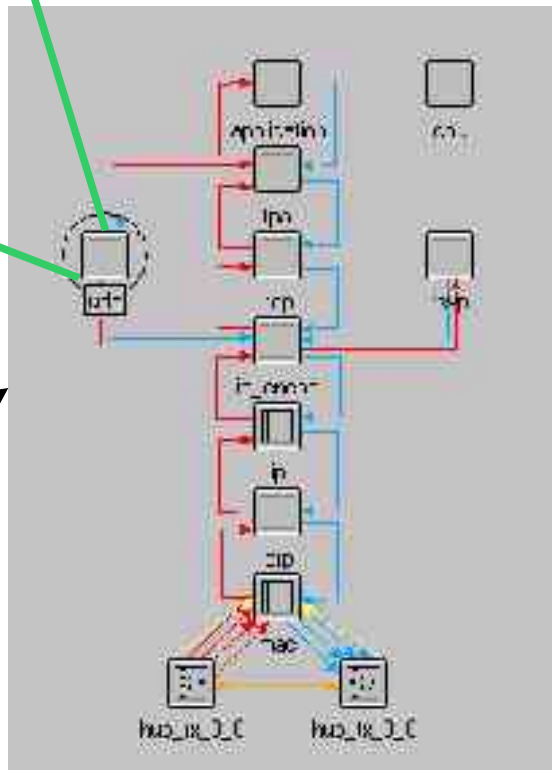
# III - Un modèle OPNET



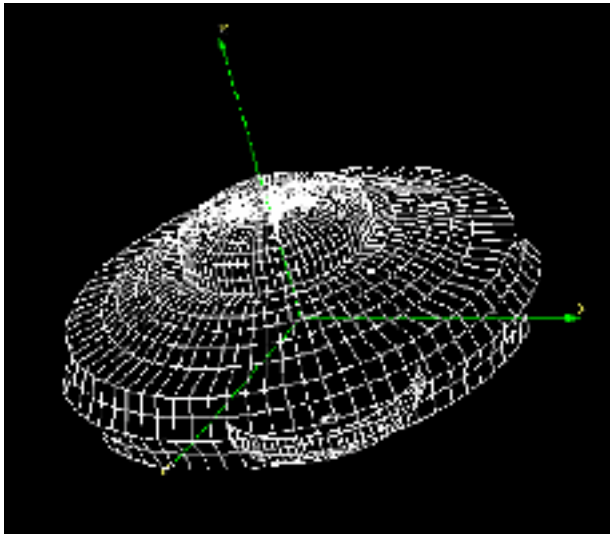







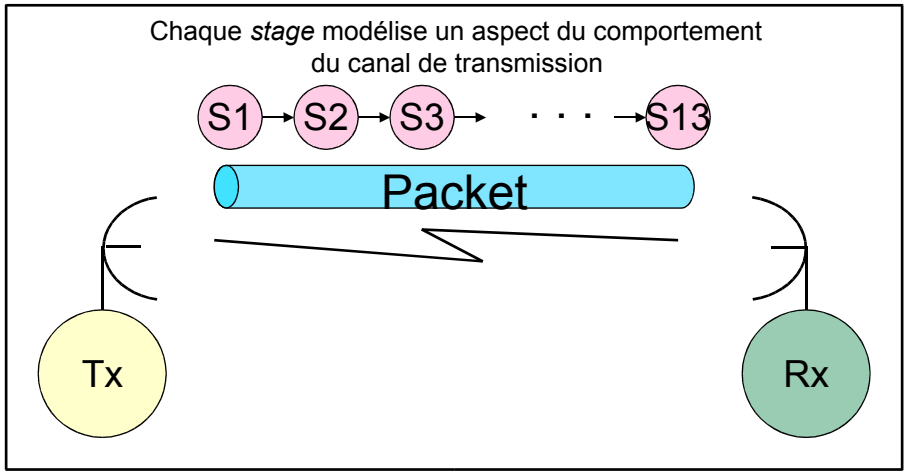
# IV - Modélisation radio

- Modélisation radio basée sur la notion de *pipeline stage :*
  - Prise en compte de l'environnement radio
  - Modélisation du diagramme de rayonnement des antennes
  - Intégration du gain des antennes Tx/Rx
  - etc.



# Pipeline de transmission

- Modélisation de la transmission d'un paquet à travers un canal de transmission
- Prise en compte des caractéristiques de la couche physique
- Divisé en étapes : les *pipeline stage*
- Détermine si un paquet peut être reçu



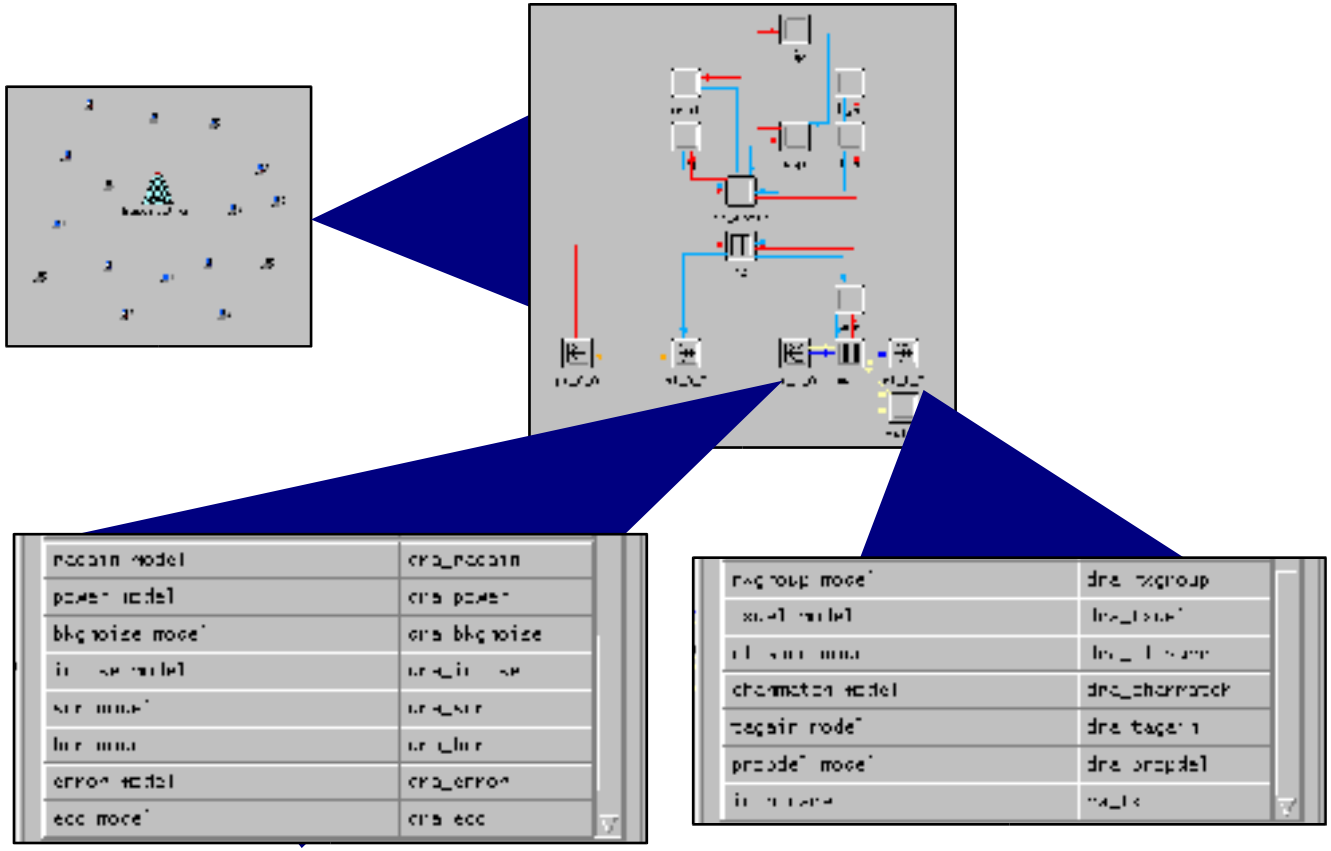
# Pipeline stage

- Procedures C / C++
  - Calcul de : voisinage radio, force du signal, bit erronés, etc.
  - Permet d'indiquer si un paquet n'est pas recevable
- Prototype commun pour les procédures :
  - Argument = paquet
  - Informations stockées dans un *Transmission Data Attributes* (TDA)

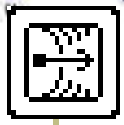
```
void
pipeline_stage (pkptr)
    Packet *     pkptr;
{
    double  dbl_val;
    int     int_val;
    double  result_val;

    /** Implements access of communication channel. **/
    clb (pipeline_stage (pkptr));

    /* Obtain input information.                    */
    dbl_val = op_td_get_dbl (pkptr, OPC_TDA_INT_VAL);
    int_val = op_td_get_int (pkptr, OPC_TDA_INT_VAL);

    /* Compute results.                             */
    result_val = dbl_val * int_val;

    /* Record an output information.                */
    op_td_set_dbl (pkptr, OPC_TDA_RESULT_VAL, result_val);

    FOUT;
}
```

# Pipeline stage attributs

- Modélisation radio en 13 (+1) points
- Propriétés intrinséques d'un émetteur



# Pipeline radio



émetteur

(0) Groupe de récepteurs

(1) Délai de transmission

(2) Établissement du lien

(3) Compatibilité des canaux

(4) Gain de l 'antenne émettrice

(5) Délai de propagation

(6) Gain de l 'antenne réceptrice

(7) Puissance reçue

(8) Bruit de fond

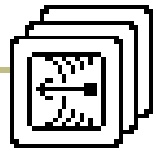
(9) Bruit dû aux interférences

(10) Rapport signal sur bruit

(11) Taux d 'erreur bits

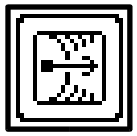
(12) Modèle d'erreurs

(13) Correction d 'erreurs



récepteurs

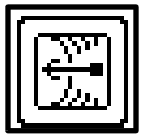
# Pipeline radio : émetteur - récepteur



Radio Transmitter

- *Receiver Group*
- Transmission Delay
- Link Closure
- Channel Match
- Tx Antenna Gain
- Propagation Delay

1+5 *stages* associés à l'émetteur



Radio Receiver

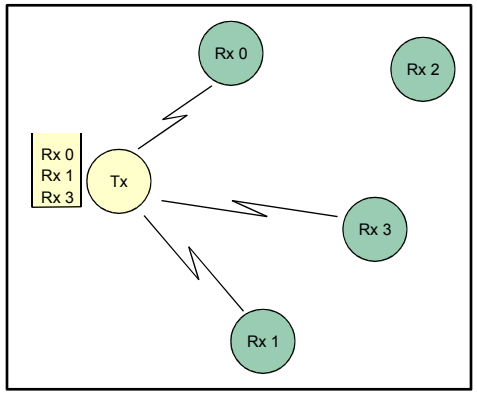
- Rx Antenna Gain
- Received Power
- Background Noise
- Interference Noise
- Signal-to-Noise Ratio
- Bit Error Rate
- Error Allocation
- Error Correction

8 *stages* associés à chaque récepteur

# Stage 0: Receiver Group

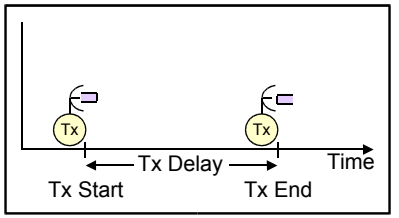
- Objet
  - Élimine les récepteurs non éligibles
    - Optimisation du temps de simulation
  - Utilisation dans les cas suivants :
    - Bande de fréquences disjointes
    - Noeuds non connexes
    - ...
- Résultats
  - Nécessite de définir des *destination channel set* pour chaque transmetteur

Exemple:



# Stage 1: Transmission Delay

- Appel
  - Exécuté dès le début de la transmission
  - 1 seul appel quelque soit le nombre de destinataires
- Objet
  - Évalue le temps de transmission d'un paquet
- Résultats
  - Rajoute un événement de fin de transmission
    - Signale le début de transmission du paquet suivant




- Par défaut :

$$\text{Delay} = \frac{\text{Packet Length}}{\text{Data Rate}}$$

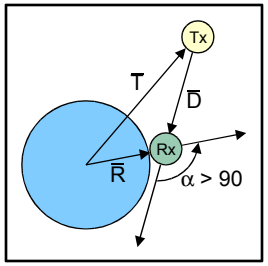

# Stage 2: Closure

- Appel
  - Pour chaque destinataire et sans condition après stage 1
- Objet
  - Détermine si le signal physique peut atteindre la destination
  - Permet de gérer dynamiquement le voisinage radio
- Résultats
  - Transmission ou *obstruction* (Paquet détruit par le kernel et fin du pipeline stage)

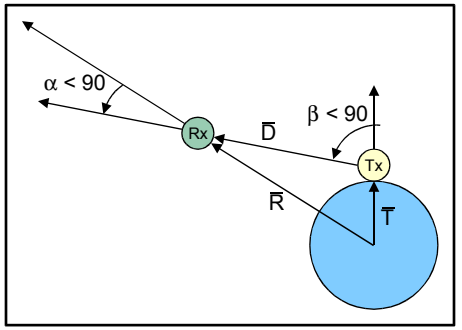
# Stage 2: Closure (defaut)

- Évaluation (défaut)
  - Basé sur un modèle de lancer de rayon pour évaluer la vue directe
  - Hypothèse : Terre sphérique
  - 3 cas *d'obstruction*
    - Cas 1: $\alpha > 90^o$
    - Cas 2: $\alpha < 90^o$ , $\beta < 90^o$
    - Cas 3: $\alpha < 90^o$, $\beta > 90^o$, d ≥ rayon Terre

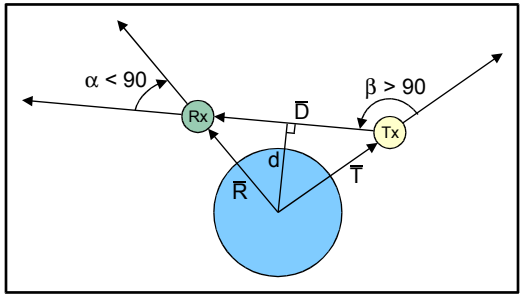
Cas 1:




Cas 2:

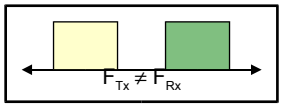


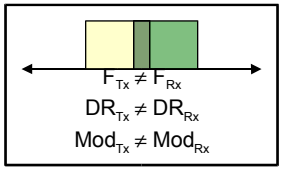

Cas 3:

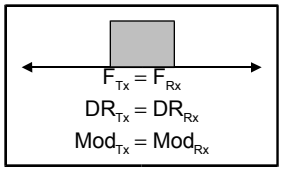


# Stage 3: Channel Match

- Objet
  - Détermine la compatibilité et la qualité de transmission entre le canal radio de l'émetteur et celui du récepteur
  - Basé sur la fréquence, bande passante, débit, codage, etc.
- Résultats
  - Caractérise la transmission : *Valide*, *bruitée* ou *ignorée*
  - Paquets ignorés détruits par le kernel (fin du pipeline)
- Évaluation (défaut)
  - Prise en compte des fréquences utilisées, du débit, de la modulation et du codage
    - Cas 1: Fréquence différente - *ignorée*
    - Cas 2: Caractéristique partiellement commune - *bruitée*
    - Cas 3: Adéquation - *valide*

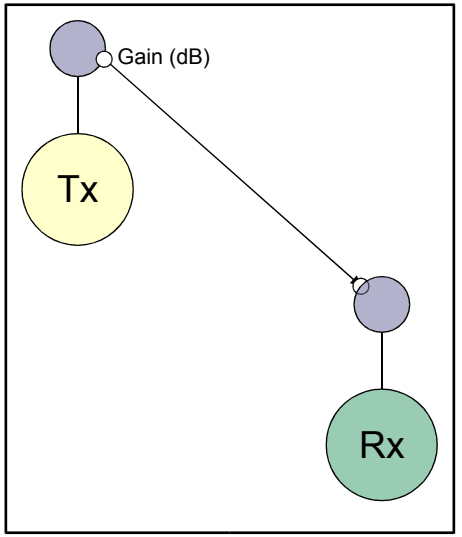


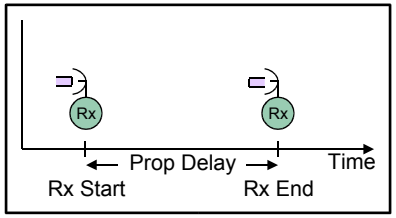
# Stage 4: Transmitter Antenna Gain

- Objet
  - Calcul du gain de l'antenne émettrice (db) en tenant compte de la direction du récepteur
- Résultats
  - Ré-utilisé dans le *stage* 7 pour évaluer la puissance du signal reçu
- Évaluation
  - Distance entre Tx et Rx
  - Diagramme de rayonnement de l'antenne, orientation, positionnement des noeuds



# Stage 5: Propagation Delay

- Objet
  - Calcul du temps de propagation émetteur → récepteur

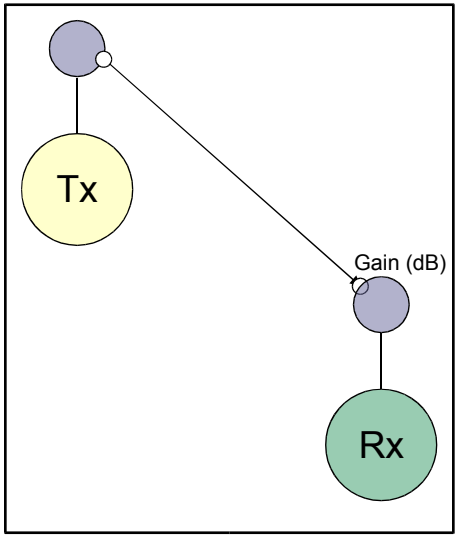



- Évaluation
  - Fonction de la distance et de la vitesse de propagation du milieu
  - Calcul du début et de la fin de la transmission
  - Tiens compte de la mobilité

$$\text{Delay} = \frac{\text{Distance}}{\text{Velocity}}$$

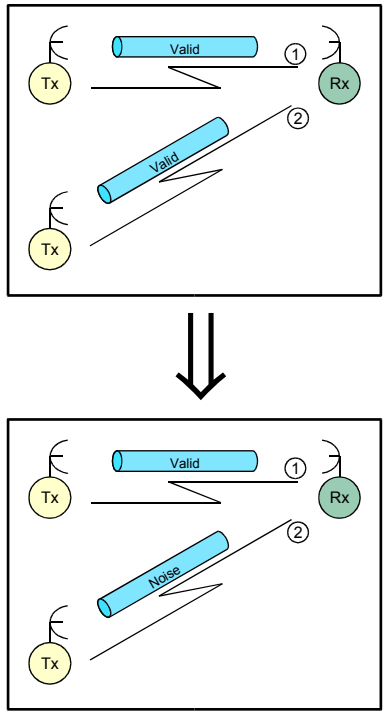
# Stage 6: Receiver Antenna Gain

- Appel
  - Pour chaque récepteur
  - 1er *stage* après le début de la réception
- Objet
  - Calcul du gain de l'antenne réceptrice (db) en tenant compte de l'orientation de l'antenne émettrice
- Résultats
  - Utilisé dans le *stage* 7 pour évaluer la puissance du signal reçu
- Évaluation
  - Distance entre Tx et Rx
  - Diagramme de rayonnement de l'antenne, orientation, positionnement des noeuds



# Notion de *Signal Lock* ?

- But
  - Attribut de chaque canal radio
  - Possibilité pour chaque récepteur de bloquer le canal lorsqu'un paquet arrive
  - L'état *lock* est maintenu pendant toute la durée de la transmission
- Procédure
  - 1er paquet valide arrivant au destinataire
    - Signal lock ok
  - Pour les autres canaux :
    - Transmission bruitée (channel match status)
  - 1er paquet valide complétement arrivé
    - Signal lock relâché



# Stage 7: Received Power

- Objet
  - Calcul du niveau de puissance du signal reçu
  - Basé sur la puissance d'émission, la fréquence, la distance, les gains des antennes
  - Calculé pour tous les paquets *valide* ou *bruité*
- Évaluation (défaut)
  - Détermine si le *signal lock* est actif
  - Calcule la puissance reçue
    - Paquets *valide* et *bruité*
    - Calcul l'affaiblissement (free space)
    - Utilise les gains des antennes rx et tx

$$L_p = \left( \frac{\lambda}{4\pi D} \right)^2$$

$$\lambda = \frac{C}{f_c}$$

$$P_i = \frac{P_{tx}\left(f_{\max} - f_{\min}\right)}{B}$$

$$P_{rx} = P_i G_{tx} L_p G_{rx}$$

$L_p$ = Pathloss
$\lambda$ = Wavelength
$D$ = Distance
$C$ = Speed of Light
$f_c$ = Center Frequency
$P_i$ = Inband Frequency
$P_{tx}$ = Transmitted Power (watts)
$f_{\max}$ = Maximum Frequency (Hz)
$f_{\min}$ = Minimum Frequency (Hz)
$B$ = Bandwidth
$P_{rx}$ = Received Power (watts)
$G_{tx}$ = Transmitter Antenna Gain (watts)
$G_{rx}$ = Receiver Antenna Gain (watts)

# Stage 8: Background Noise

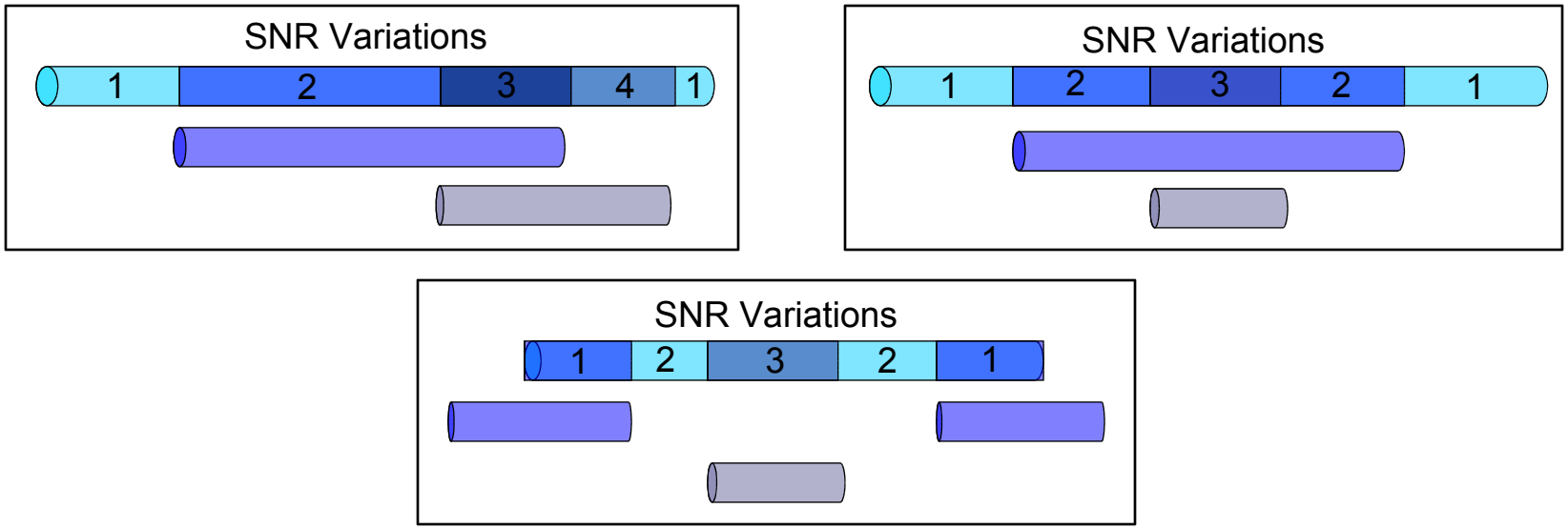
- Objet
  - Intégre les différentes sources de bruit (background)
  - Prend en compte :
    - Bruit thermique (et bruit galactique)
    - Émissions électroniques du voisinage
    - Possibilité de rajouter autres sources de bruit
- Résultats
  - Utilisé dans le *stage* 10 pour évaluer le rapport signal s/ bruit
- Évaluation (défaut)
  - Bruit ambiant (constant)
  - Bruit de fond (constant)
  - Bruit thermique (constant)

$$T_{rx} = (NF - 1.0) * 290.0$$
$$T_{bk} = 290.0$$
$$k = 1.379\mathrm{E}^{-23}$$
$$N_b = (T_{rx} + T_{bk}) B_{rx} k$$
$$N_a = B_{rx}(1.0\mathrm{E}^{-26})$$
$$N = N_b + N_a$$

$NF$ = Noise Figure
$T_{rx}$ = Receiver Temperature
$T_{bk}$ = Background Temperature
$k$ = Boltzmann's Constant
$B_{rx}$ = Receiver Bandwidth
$N_b$ = Background Noise
$N_a$ = Ambient Noise
$N$ = *Noise*

# Notion de *Packet Segments*

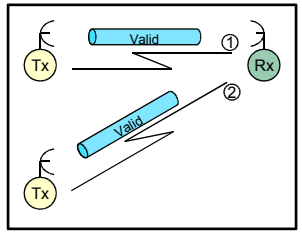
- Sous-ensemble d'un paquet ayant un rapport signal s/ bruit constant :
  - Évaluation du SNR dans le pipeline *ad hoc*



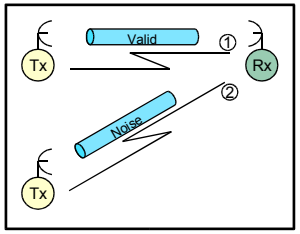
# Stage 9: Interference Noise

- Appel
  - En cas de transmissions simultanées
- Objet
  - Tient compte des transmissions concurrentes
  - Calcul les effets du bruit sur les paquets valides
- Résultats
  - Accumule le bruit des paquets interférants
  - Utilisé dans le *stage* 10 pour l'évaluation du rapport signal s/ bruit

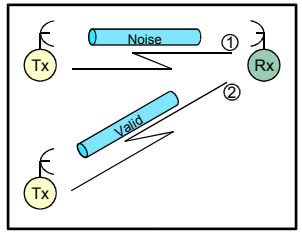
Cas 1:




Cas 2:




Cas 3:



# Stage 10: Signal-to-Noise Ratio

- Appel
  - Uniquement pour les paquets *valide*
- Objet
  - Calcul du SNR
  - Basé sur puissance de réception et bruit
- Évaluation (défaut)
  - Puissance de réception (*stage* 7)
  - Bruit de fond (*stage* 8)
  - Interférences (*stage* 9)
  - Calcul du rapport signal-to-noise (dB)

$$SNR = 10\log_{10}[P_r/(P_b + P_i)]$$

$P_r$ = Received Power (watts)

$P_b$ = Background Noise (watts)

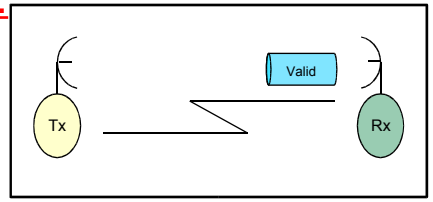
$P_i$ = Interference Noise (watts)

# Stage 11: Bit Error Rate

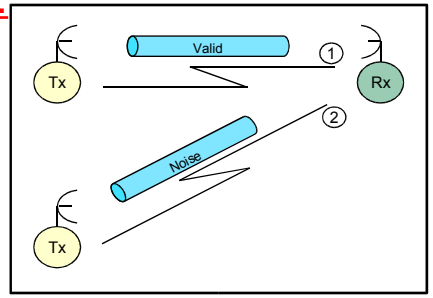
- Appel
  - Uniquement pour les paquets *valide*
- Objet
  - Calcule la probabilité d'erreurs bit
  - Calculé pour chaque segment de paquet (à SNR constant)
- Résultats
  - Utilisé dans le *stage* 12 for déterminer les erreurs
- Évaluation (défaut)
  - Signal-to-noise ratio (*stage* 10)
  - Gain récepteur
  - Calcul du SNR effectif
  - Détermine le taux d'erreur bit

$$SNR_{effective} = SNR_{actual} + G_p$$
$$G_p = \text{Processing Gain}$$

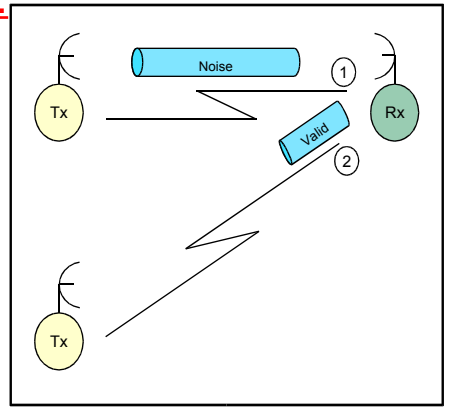
Cas 1:




Cas 2:




Cas 3:



# Stage 12: Error Allocation

- Objet
  - Évalue les bits erronés pour un segment de paquet
- Résultats
  - Utilisé dans le *stage* 13 pour la correction d'erreurs

- Évaluation (défaut)
  - Utilise la probabilité d'erreur (BER)
  - Calcul de la probabilité d'avoir k erreurs
  - Tirage suivant une loi uniforme

$$P_k = p^k (1-p)^{N-k} \binom{N}{k}$$

$$\sum_{k=0}^{N} P_k \geq r$$

$P_k$ = Probability of $k$ Errors
$p$ = Probability of Error
$N$ = Packet Length (bits)
$r = op\_dist\_uniform(1)$
$k$ = Number of Errors

# Stage 13: Error Correction

- Appel
  - Uniquement pour les paquets valide
- Objet
  - Déterminer si un paquet est acceptable ou non

- Évaluation (défaut)
  - Utilise une variable **`ECC_TRESH`** indiquant le taux d'erreur acceptable
  - Calcul du nombre d'erreurs (*stage* 12)
  - Calcul du taux d'erreurs pour le paquet en cours

$$\%_{\text{Error}} = \frac{N_{errors}}{P_{length}}$$

# Synthèse

- Environnement radio :
  - 3D, isotrope, homogène, géométrie euclidienne
  - Canal radio symétrique
  - Pas de corrélation temporelle pour le canal radio
  - Caractérisation *assez* fine des interférences
  - Prise en compte des diagrammes de rayonnement
  - Évaluation des gains des antennes