

Tutorial WSim/WSNet et IPv6/Senslab

Journées IP Capteur

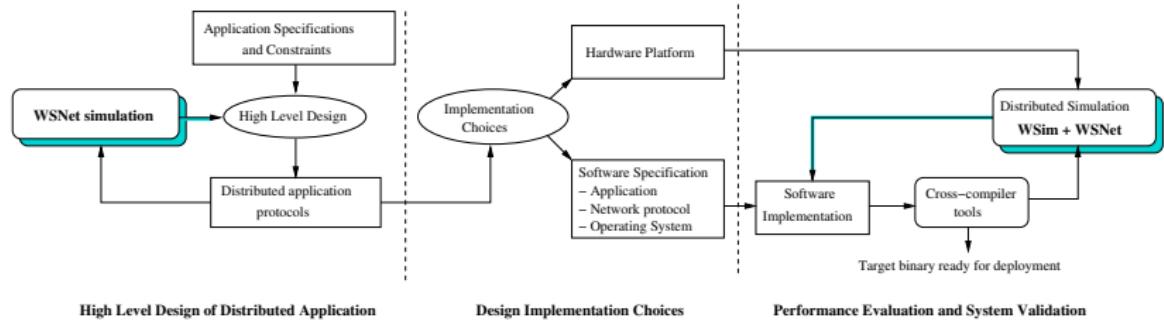
Antoine Fraboulet Julien Carpentier

`prenom.nom@inria.fr`
Labo CITI, INSA de Lyon



13 janvier 2011

Utilisation des outils dans la conception



Outils

WSim : simulation de la plate-forme matérielle

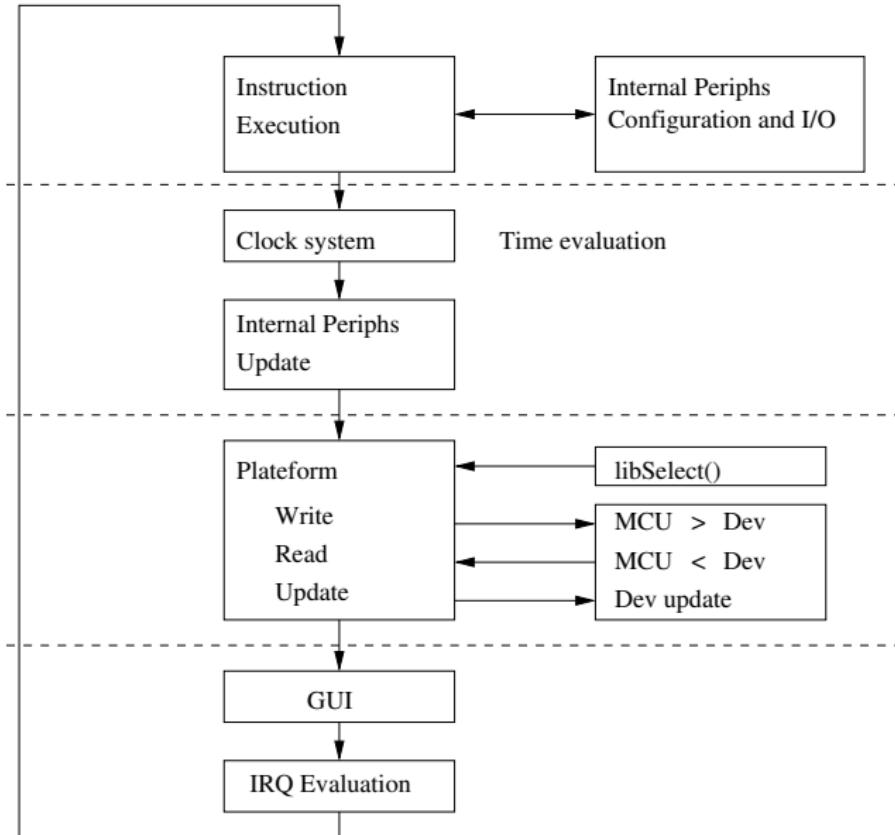
- Simulation précise en temps
- Plateforme complète
- Interaction avec WSNet pour la simulation du réseau
- Utilise le binaire final de l'application

WSNet : Simulateur de réseau radio

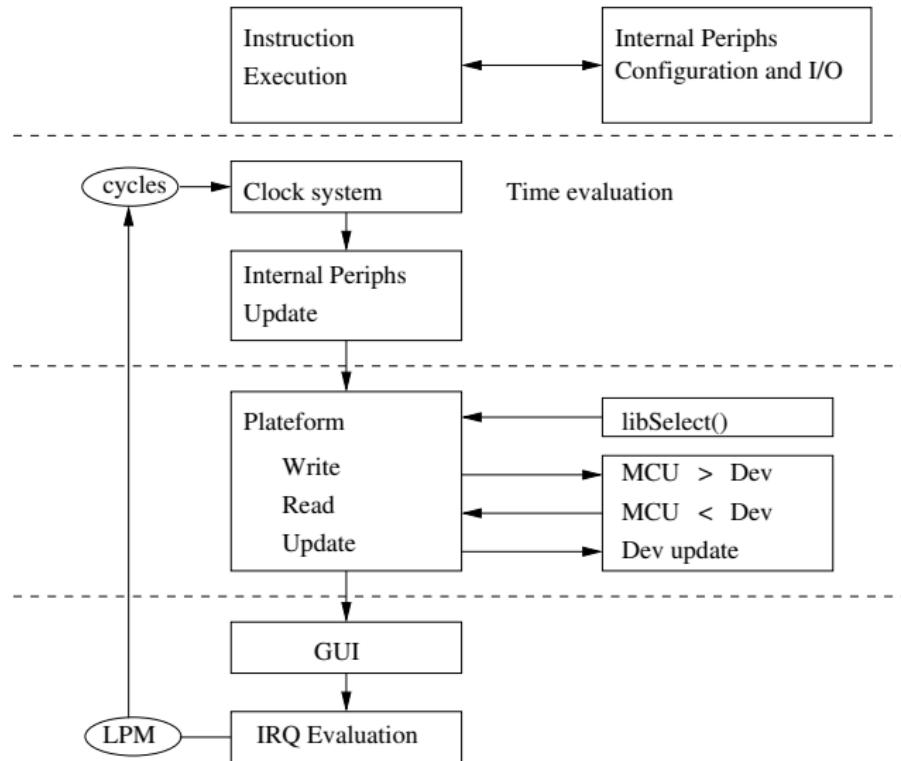
- Simulation événementielle
- Utilisation du simulateur pour les couches physique et radio
- Frontend UDP/IP (multicast)

WSNet + WSim : simulation complète d'un système distribué

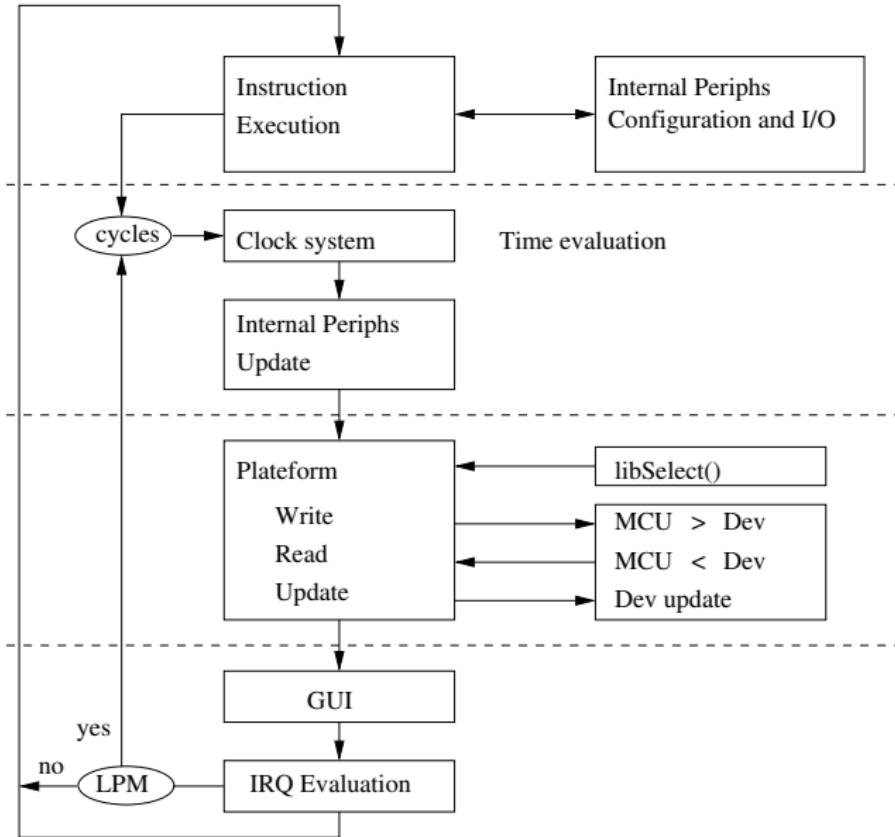
WSim : boucle de simulation



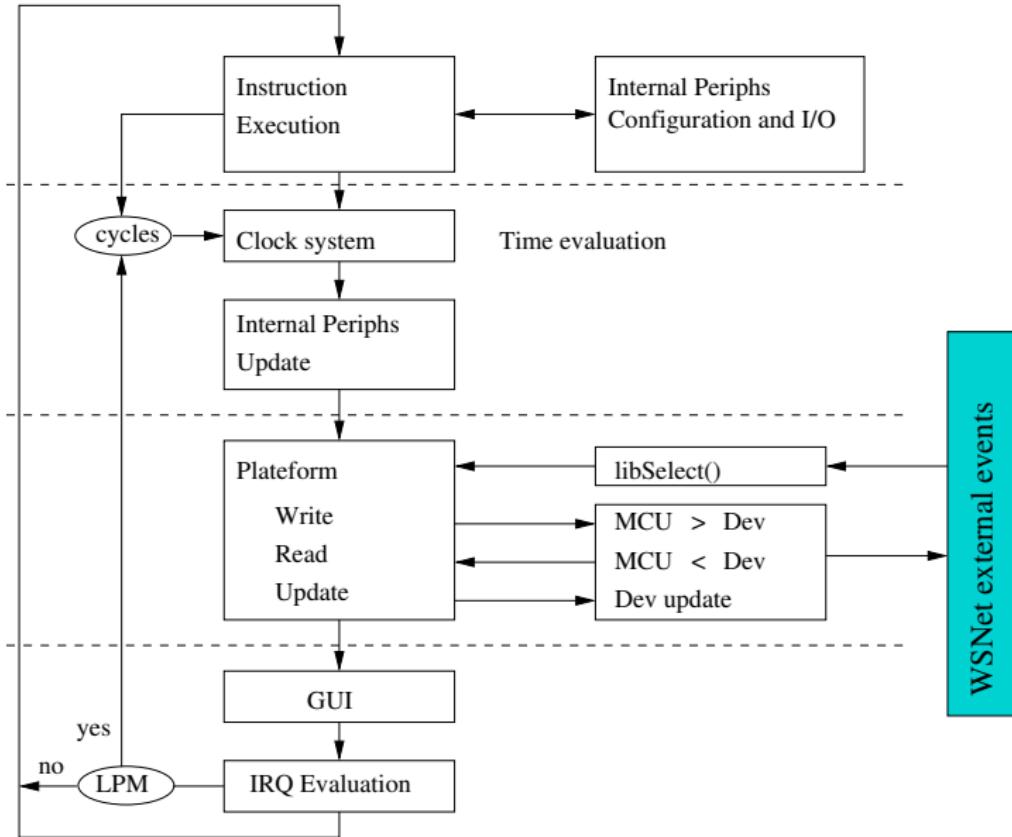
WSim : boucle de simulation



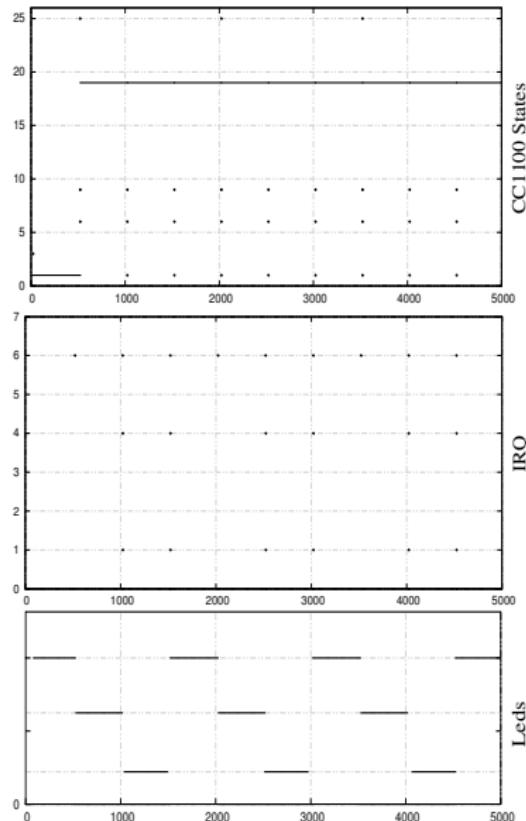
WSim : boucle de simulation



WSim : boucle de simulation



Simulation précise à l'instruction



Traces générées

- Interruptions / activité
- Mode de veille
- Communications

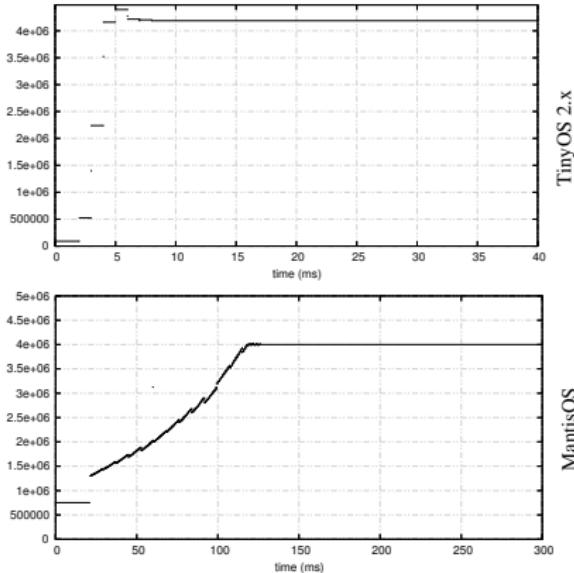
Activité des nœuds

- Evénements radio
- Analyse hors-ligne

Evaluation de performance

- Instructions
- Emprunte Mémoire
- *Consommation*

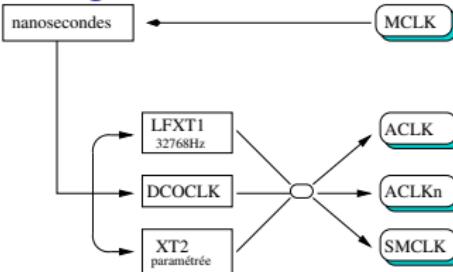
Gestion des fréquences des micro-contrôleurs



Simulation des blocs d'horloge

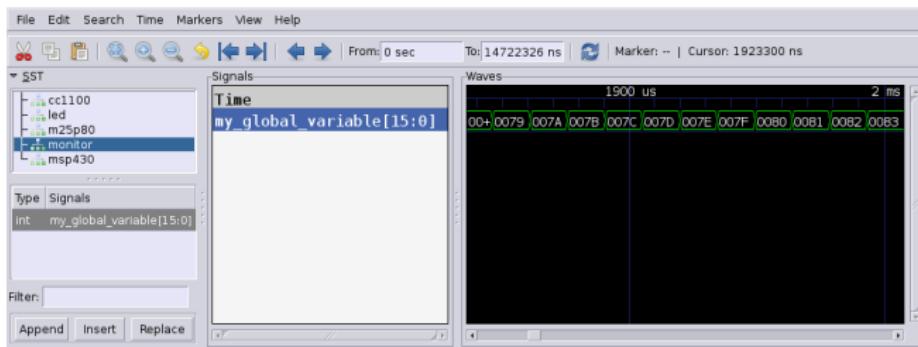
- Fréquences variables
- Dérive d'horloge
- Estimation de la puissance et énergie

Simulation du système d'horloge

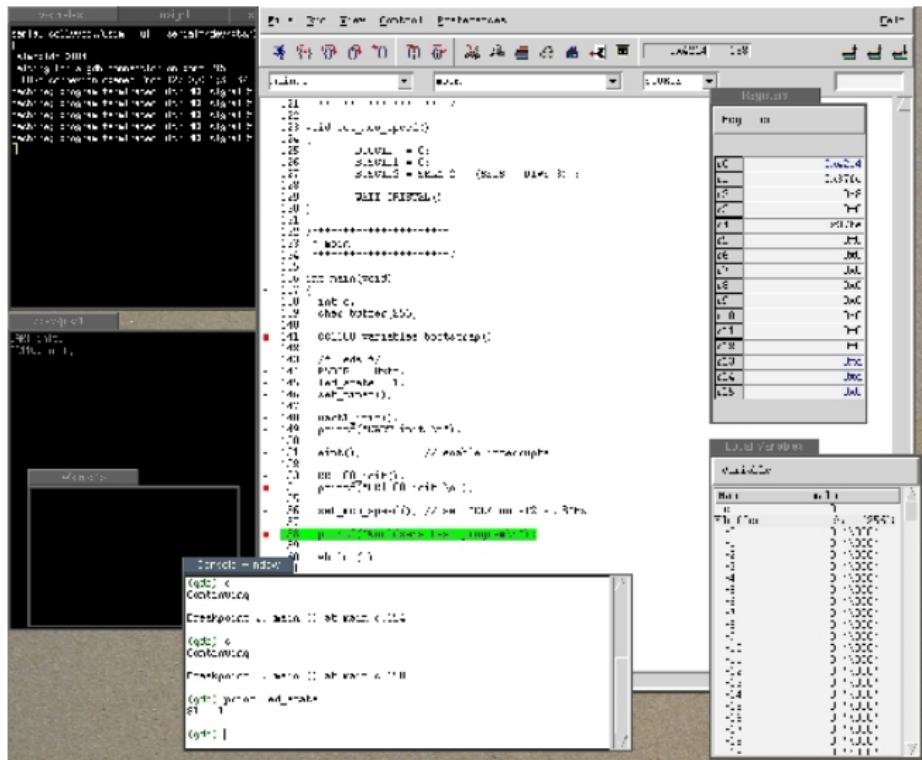


Observation non intrusive

- Possibilité de contrôle sur la mémoire (observation, modification)
- Détection des débordements de pile
- Gestion des erreurs courantes

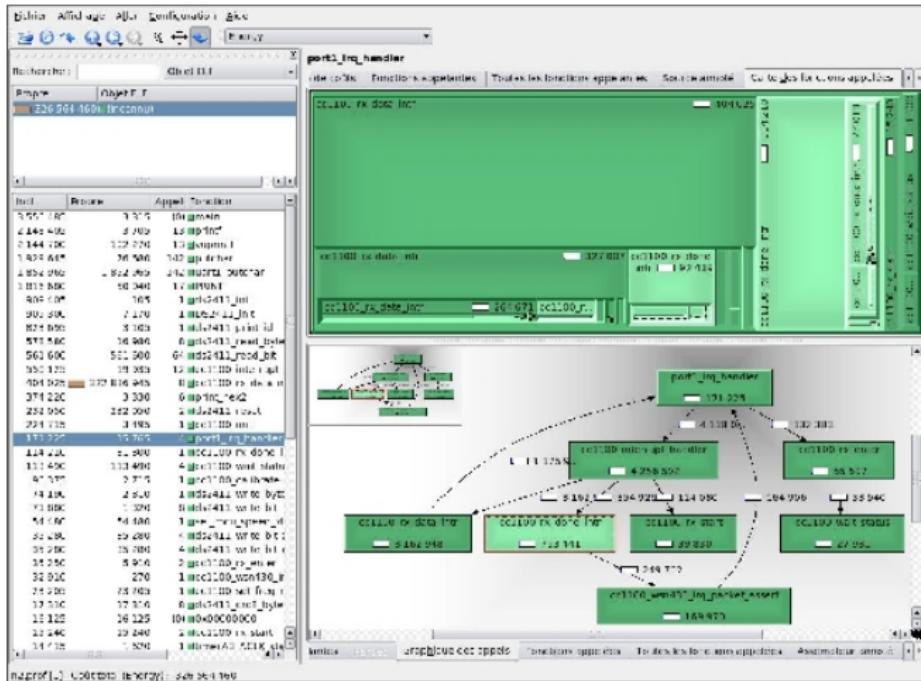


Debug



Annotation du code source

Utilisation d'eSimu, estimation de performance et de consommation.



WSNet : modèles

Modèles radio

- propagation (ex : range, Friis free space, Rayleigh, etc.)
- interférences / SiNR (ex : orthogonal, additive, partially additive, etc.)
- modulation (ex : bpsk, step function, etc.)

Modèles de haut niveau

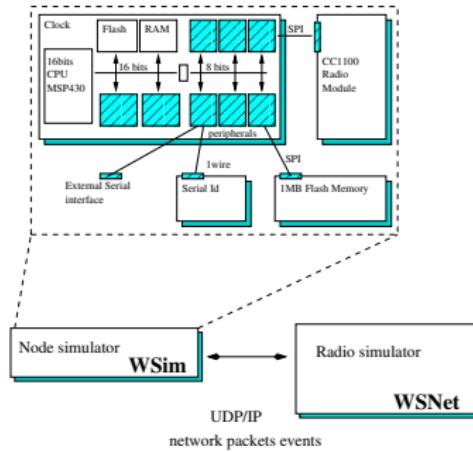
- radio (ex : full-duplex, half-duplex, etc.) ;
- antenna (ex : directionnal, etc.) ;
- environment (ex : battery, etc.)
- application (ex : sink, source, etc.) ;
- routing (ex : gradient, geographic, etc.) ;
- queue (ex : fifo, priorities, etc.) ;
- mac (ex : s-mac, 802.11DCF, Aloha, etc.) ;

WSNet : extensions pour WSim

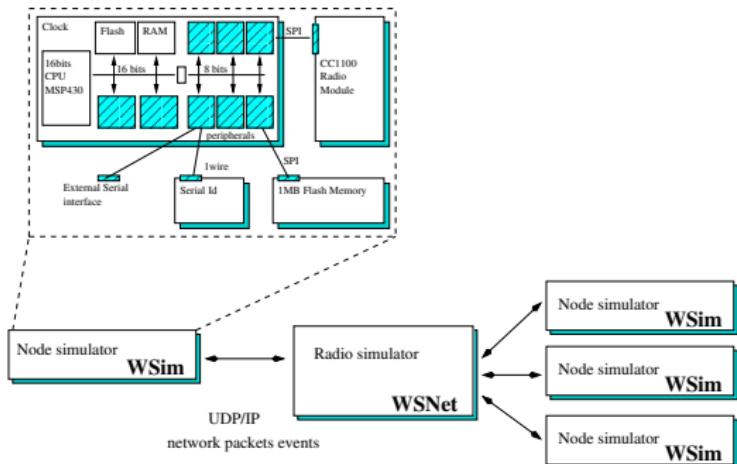
Simulation hybride

- Les nœuds sont émulés par des process externes
- Communications
 - Simulation distribuée
 - Communications multicast
- Synchronisation lâche
 - synchronisation sur les octets radios
 - synchronisation périodique des nœuds en cas d'absence de communication
 - reprise arrière (backtrack) des émulateurs lors d'une détection de perte de synchronisation

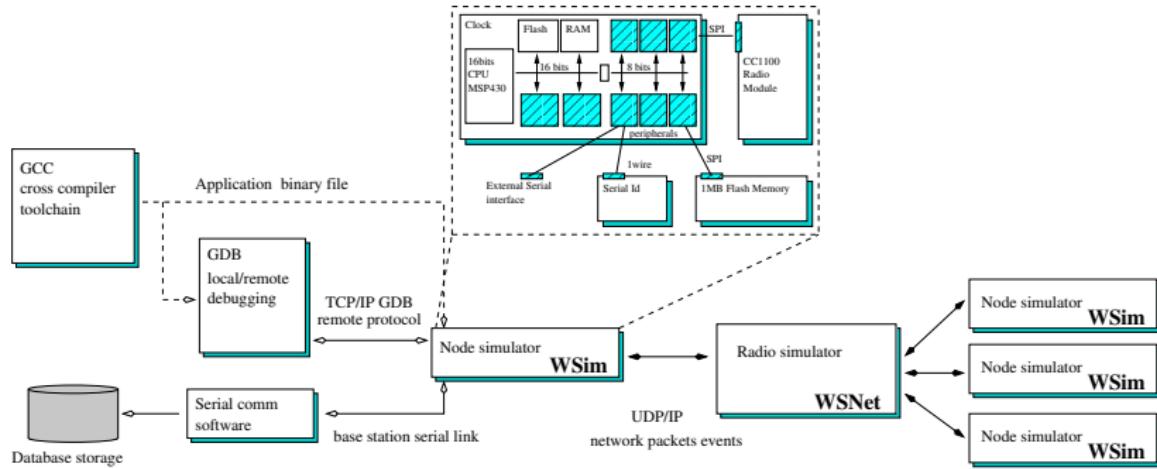
Simulation mixte



Simulation mixte



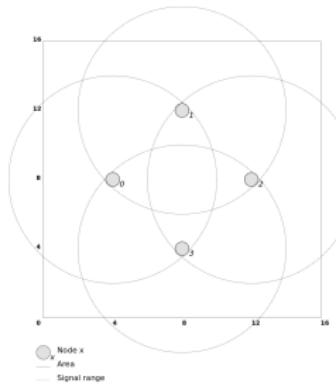
Simulation mixte



Simulation mixte

- La simulation distribuée permet de faire du pas à pas sur l'application de façon globale
- Les mesures de performances locales associées aux communications permettent de faire ressortir le comportement de l'application complète.
- La mesure de consommation en analyse hors-ligne permet de mesurer les divers compromis existant dans l'architecture entre calcul et communication.
- Des changements d'annotation de consommation permettent de détecter les composants du système qui sont les plus consommateurs en fonction de l'application.

Simulation distribuée



Four terminal windows are shown, each displaying a snippet of WSim/WSNet code and its corresponding output:

- netcat-0:**

```
WNETT: init, ds2411_init() ok.
ds2411_get_id(), ds2411_get_id(), trc a 291291010101010101 family 1
CC1100 init.
Wardines test program, id:0
cc1100_enter node 0
[...]
```

Output:

```
WNETT: nm TIME 53011802 (seq: 61)
WNETT: >>> SYN (seq: 62)
WNETT: <-- SYN (seq: 63)
WNETT: >>> SYN (seq: 64)
WNETT: <-- SYN (seq: 65)
WNETT: === TIME 530247984 (seq: 62)
WNETT: === received packet (seq: 63)
WNETT: >>> SYN (seq: 64 + RP (seq: 65, period: 3575), rcp: 530778054)
WNETT: --> RX (ipcl, sinr:300, freq:820MHz, modul:1)
WNETT: <-- SYN (seq: 65)
WNETT: >>> SYN (seq: 66)
WNETT: <-- SYN (seq: 67)
WNETT: === TIME 530378984 (seq: 63)
WNETT: >>> SYN (seq: 68 + RP (seq: 69, period: 100000000, rcp: 153577054)
WNETT: >>> SYN (seq: 70 + RP (seq: 71, period: 100000000, rcp: 153577054)
WNETT: <-- SYN (freq:820MHz, sinr:300, ipcl_size:1,data:52, freq:820MHz, modul:1)
WNETT: >>> CC1100_set_txpower (txpower: 5, period: 98743077, rcp: 1535771961)
WNETT: <-- SYN (seq: 65)
```
- netcat-1:**

```
WNETT: init, ds2411_init() ok.
ds2411_get_id(), ds2411_get_id(), trc a 291291010101010101 family 1
CC1100 init.
Wardines test program, id:1
[...]
```

Output:

```
WNETT: nm TIME 53011802 (seq: 61)
WNETT: >>> SYN (seq: 62)
WNETT: <-- SYN (seq: 63)
WNETT: >>> SYN (seq: 64)
WNETT: <-- SYN (seq: 65)
WNETT: === TIME 530247984 (seq: 62)
WNETT: === received packet (seq: 63)
WNETT: >>> SYN (seq: 64 + RP (seq: 65, period: 3575), rcp: 530778054)
WNETT: --> RX (ipcl, sinr:300, freq:820MHz, modul:1)
WNETT: <-- SYN (seq: 65)
WNETT: >>> SYN (seq: 66)
WNETT: <-- SYN (seq: 67)
WNETT: === TIME 530378984 (seq: 63)
WNETT: >>> SYN (seq: 68 + RP (seq: 69, period: 100000000, rcp: 153577054)
WNETT: >>> SYN (seq: 70 + RP (seq: 71, period: 100000000, rcp: 153577054)
WNETT: <-- SYN (freq:820MHz, sinr:300, ipcl_size:1,data:52, freq:820MHz, modul:1)
WNETT: >>> CC1100_set_txpower (txpower: 5, period: 98743077, rcp: 1535771961)
WNETT: <-- SYN (seq: 65)
```
- netcat-2:**

```
WNETT: init, ds2411_init() ok.
ds2411_get_id(), ds2411_get_id(), trc a 291291010101010101 family 1
CC1100 init.
Wardines test program, id:2
[...]
```

Output:

```
WNETT: nm TIME 53011802 (seq: 61)
WNETT: >>> SYN (seq: 62)
WNETT: <-- SYN (seq: 63)
WNETT: >>> SYN (seq: 64)
WNETT: <-- SYN (seq: 65)
WNETT: === TIME 530247984 (seq: 62)
WNETT: === received packet with correct id (64)\n, rx_buf[0] = 0x00, token = 1
WNETT: >>> SYN (seq: 66)
WNETT: <-- SYN (seq: 67)
WNETT: >>> SYN (seq: 68)
WNETT: <-- SYN (seq: 69)
WNETT: === TIME 530378984 (seq: 63)
WNETT: >>> SYN (seq: 70 + RP (seq: 71, period: 100000000, rcp: 153577054)
WNETT: <-- SYN (freq:820MHz, sinr:300, ipcl_size:1,data:52, freq:820MHz, modul:1)
WNETT: >>> CC1100_set_txpower (txpower: 5, period: 98743077, rcp: 1535771961)
WNETT: <-- SYN (seq: 65)
```
- main.c - Source Window:**

```
151         cc1100_size,
152         rx_buffer[cc1100_size + 1],
153         rx_buffer[cc1100_size + 2] & 0x80,
154         rx_buffer[cc1100_size + 2] & 0x7f);
155     */
156 }
157 else
158 {
159     printf("recv: error\n");
160 }
161 if (rx_buffer[0] == myid)
162 {
163     printf("received packet with correct id (%d)\n", rx_buf[0]);
164     token = 1;
165     rx_buffer[0] = 0;
166     rx_buffer[1] = 0;
167     LRD_RXDR_ON;
168 }
169 else
170 {
171     printf("recv: received packet with id %d (%d)\n", rx_buf[0], rx_buf[1]);
172     rx_buffer[0] = 0;
173     cc1100_rx_enter();
174     cc1100_rx_received = 0;
175     timer_wakeup = 0;
176 }
177 else if ((timer_wakeup == 1) && (token == 1)) /* timer wakeup
178 */
179 {
220 }
```

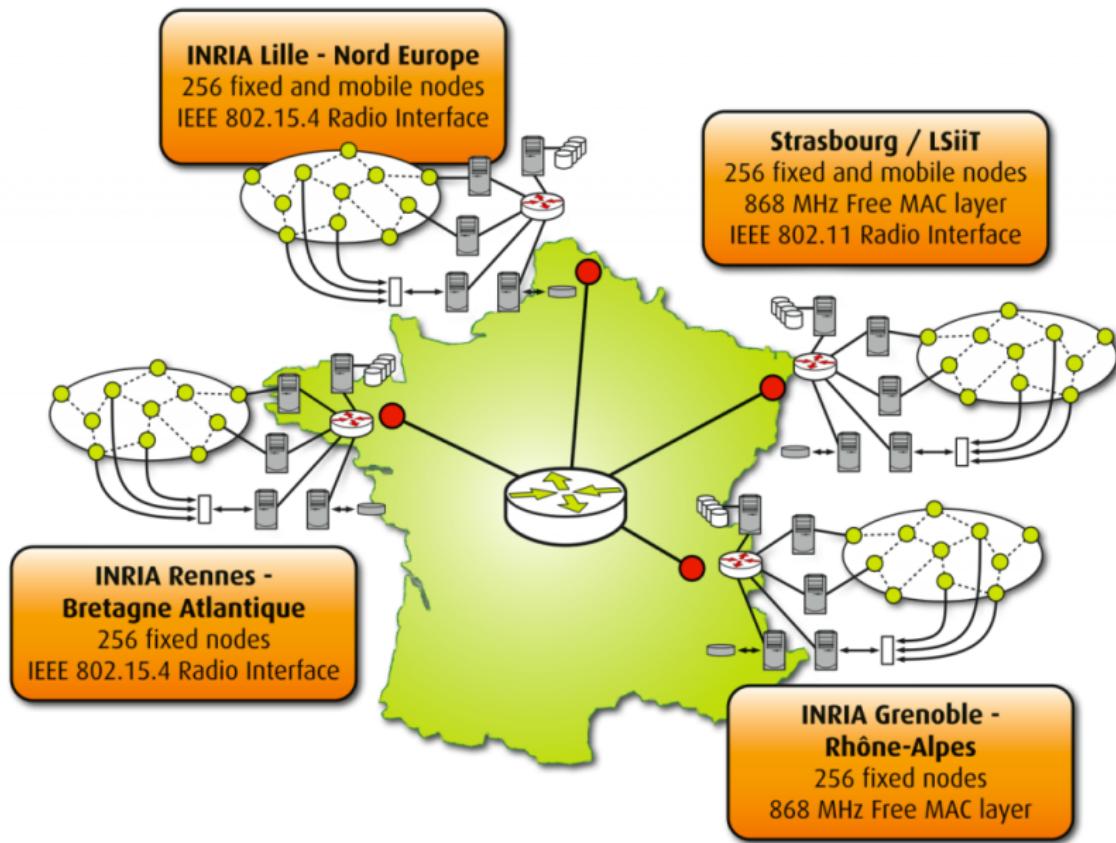
Output:

```
Program stopped at line 212
```



Plate-forme expérimentale

- But de SensLAB
 - Plate-forme d'expérimentation de réseau de capteurs Outils
 - Large échelle pour la communauté scientifique
- Spécificités Distribuée sur 4 sites distants Grande échelle avec 256 capteurs par site
- Automatisée / Ouverte
- Accessible à distance

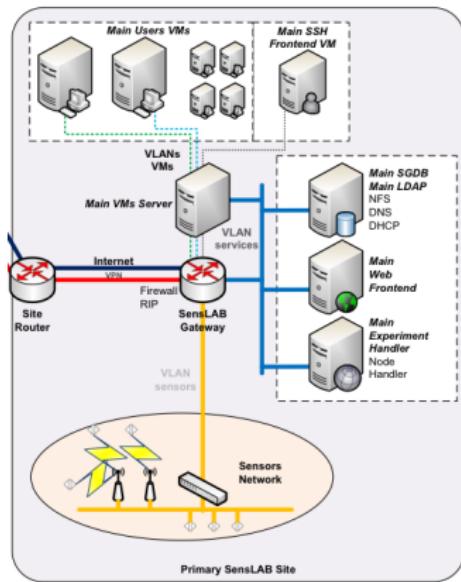


Senslab

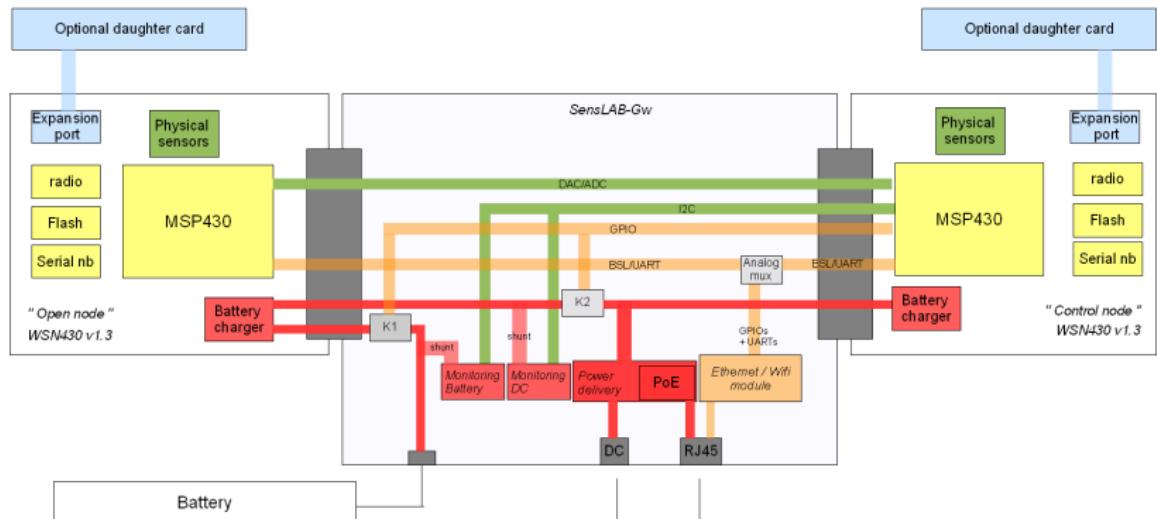
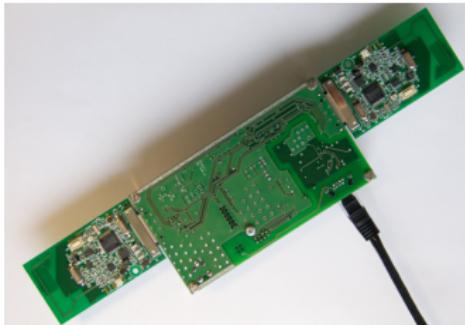


Services utilisateurs

- Portail Web
- Base de données de résultats des expérimentations
- Espace de stockage NFS
- Accès SSH
- Machine virtuelle (compte administrateur)
- Outils de développements adaptés à la plateforme



Architecture des noeuds



Utilisation des outils

Les outils de simulation servent à prototyper et à débugger les expérimentations de façon très fine. Les utilisations principales sont :

- **Debug distribué** en pas à pas synchronisé
- **Evaluation de performance** avec annotations
- Tests **d'inter-opérabilité** entre systèmes
- Ecriture de code de bas niveau

La simulation permet d'exhiber des effets de synchronisation difficiles à avoir en condition réelles.

- problèmes de synchronisation entre les noeuds
- détection de “race conditions” internes

Sites web

www.senslab.info, plateforme Senslab
Grenoble, Lille, Strasbourg sont disponibles

wsim.gforge.inria.fr, émulateur de plateforme de capteur

wsnet.gforge.inria.fr, simulateur de réseau radio

esimu.gforge.inria.fr, analyse de trace, annotation de code source avec information de performance et de consommation.