Trace d'exécution de systèmes temps-réel

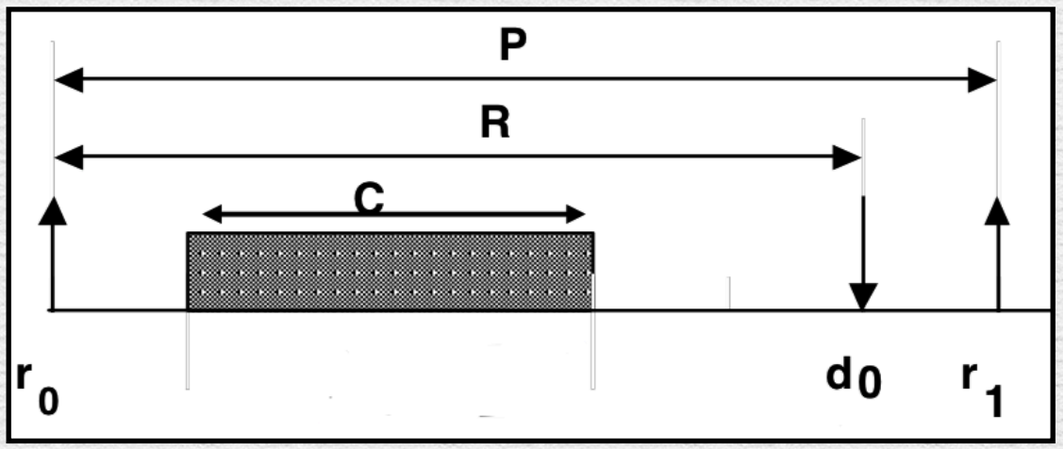
IMA5 2018/2019 P10

Amine El Messaoudi

Encadrant : Julien Forget

Context

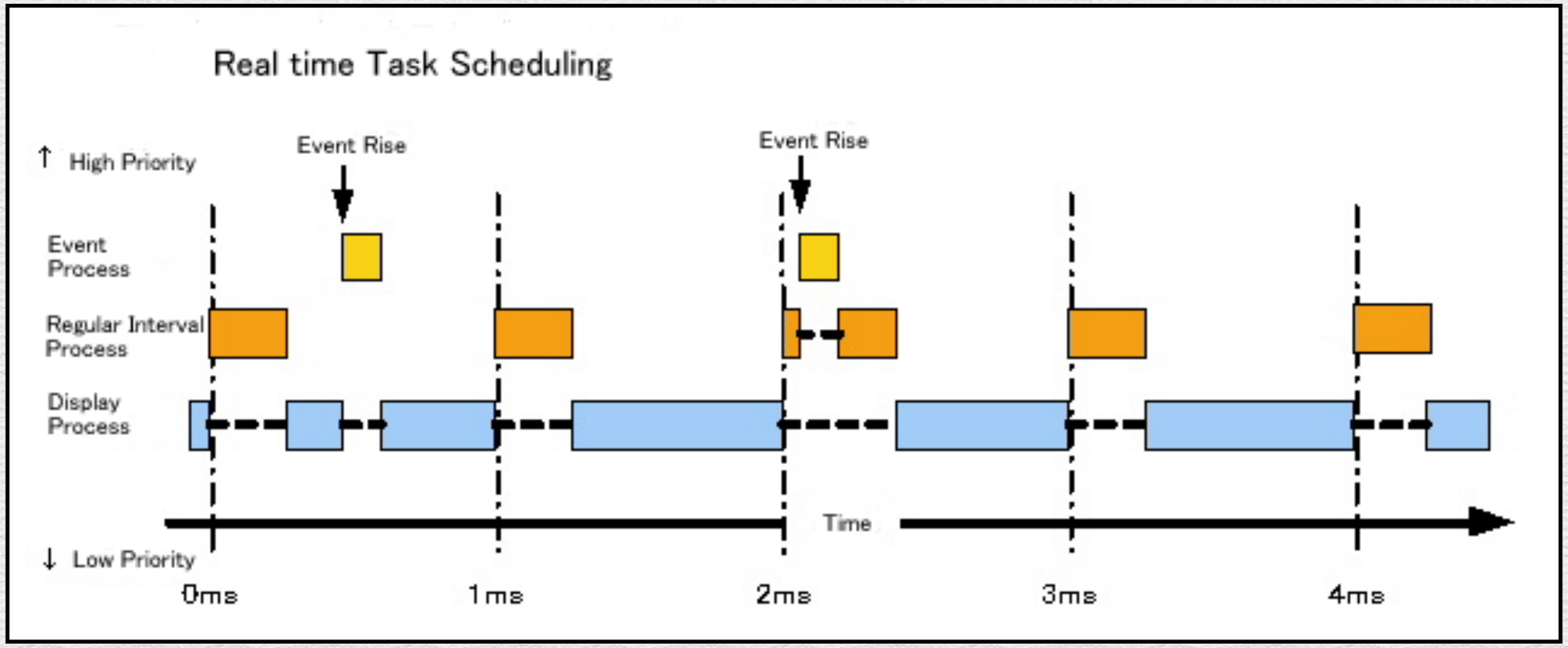
- Système temps-réel :
  - contrôler un procédé physique à
    - vitesse adaptée à l'évolution du procédé
- Tâches :
  - temps de calcul (C)
  - échéance (R)
  - période (P)



Objectif

Mettre au point un outil de trace d'un système temps-réel basé sur l'API ptask.

Objectif : aide au débbugage d'un système temps-réel.



Sommaire

Cahier de charges :

1. Documentation sur ptask et LTTng.
2. Implémentation d'un outil d'extraction des traces.
3. Implémentation d'un outil d'analyse et de visualisation des traces.

ptask

- Bibliothèques C pour le développement de tâches temps-réel pour linux.
- Surcouche de la bibliothèque pthread.
- Tâche périodique et tâche apériodique.

Tâche périodique

```
ptask my_periodic_task() {
    while(1) {
        do_the_job();
        ptask_wait_for_period();
    }
}

int main() {
    ptask_init(SCHED_FIFO, GLOBAL, PRIO_INHERITANCE);
    ptask_create(my_periodic_task, PER, PRIO, NOW);

    while(1) ;
    return 0;
}
```

LTTng

- Toolkit opensource pour le traçage du noyau et de l'espace utilisateur linux.
- Dépend de :
  - une session : temps du traçage ;
  - un domaine : espace utilisateur, kernel … ;
  - un channel : repartition entre buffers ;
  - un événement : points d'instrumentations.

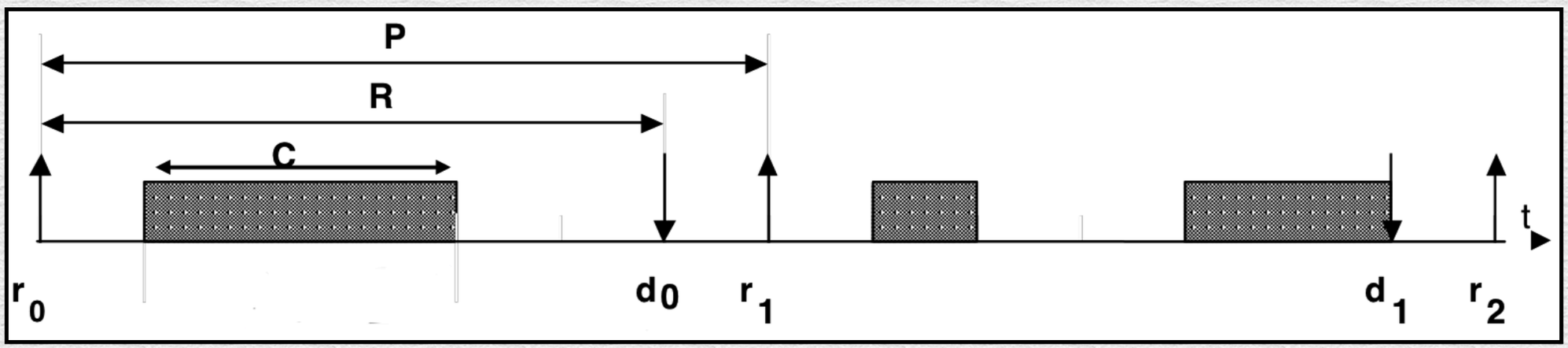
LTTng & ptask

Événements :

- Tracepoint ptask_wait_for_period()

```
ptask my_periodic_task() {
    while(1) {
        do_the_job();
        ptask_wait_for_period();
    }
}
```

- Appel système clone
- Tracepoint sched_switch



Récupération de traces

Format CTF (binaire).

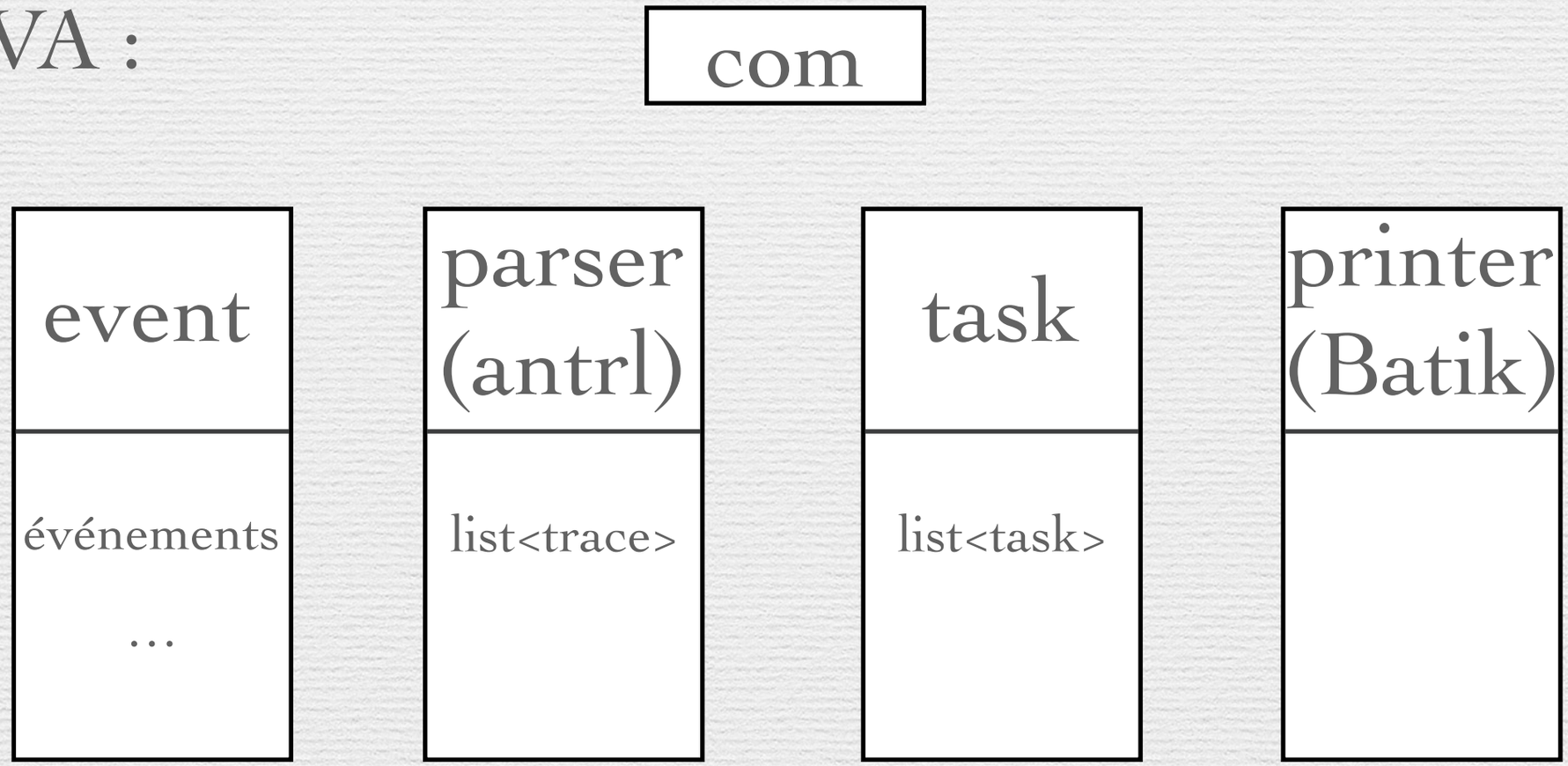
Babeltrace : Implementation de référence CTF

```
[12:20:16.002620588] (+0.000004208) debian-vm syscall_exit_clone: { cpu_id = 0 }, { pid =
16226, tid = 16308 }, { ret = 0 }
[12:20:16.002593777] (+0.000034078) debian-vm syscall_entry_clone: { cpu_id = 0 }, { pid =
16226, tid = 16226 }, { clone_flags = 0x3D0F00, newsp = 0x7FC06FF5CF70, parent_tid =
0x7FC06FF5D9D0, child_tid = 0x7FC06FF5D9D0 }
[12:20:16.842354195] (+0.000009715) debian-vm ptask_provider:ptask_tracepoint: { cpu_id =
0 }, { ptask_flag = "DEFERRED", ptask_state = "b_wait_period", ptask_pid = 16226,
ptask_tid = 16308, ptask_index = 0, ptask_time = 150276421, ptask_priority = 80,
ptask_period = 20000, ptask_deadline = 20000 }
[12:20:16.842358357] (+0.000004162) debian-vm sched_switch: { cpu_id = 0 }, { pid = 16226,
tid = 16308 }, { prev_comm = "ball", prev_tid = 16308, prev_prio = -81, prev_state = 1,
next_comm = "ball", next_tid = 16309, next_prio = 20 }
```

ptaskTracer → svg

ptaskTracer

- ANTL4 (analyseur syntaxique) : parser.g4, lexer.g4
- JAVA :



Conclusion

Bilan :

- Instrumentation de ptask.
- ptaskTracer :
  - Traçage avec LTTng.
  - Exploitation de la trace.

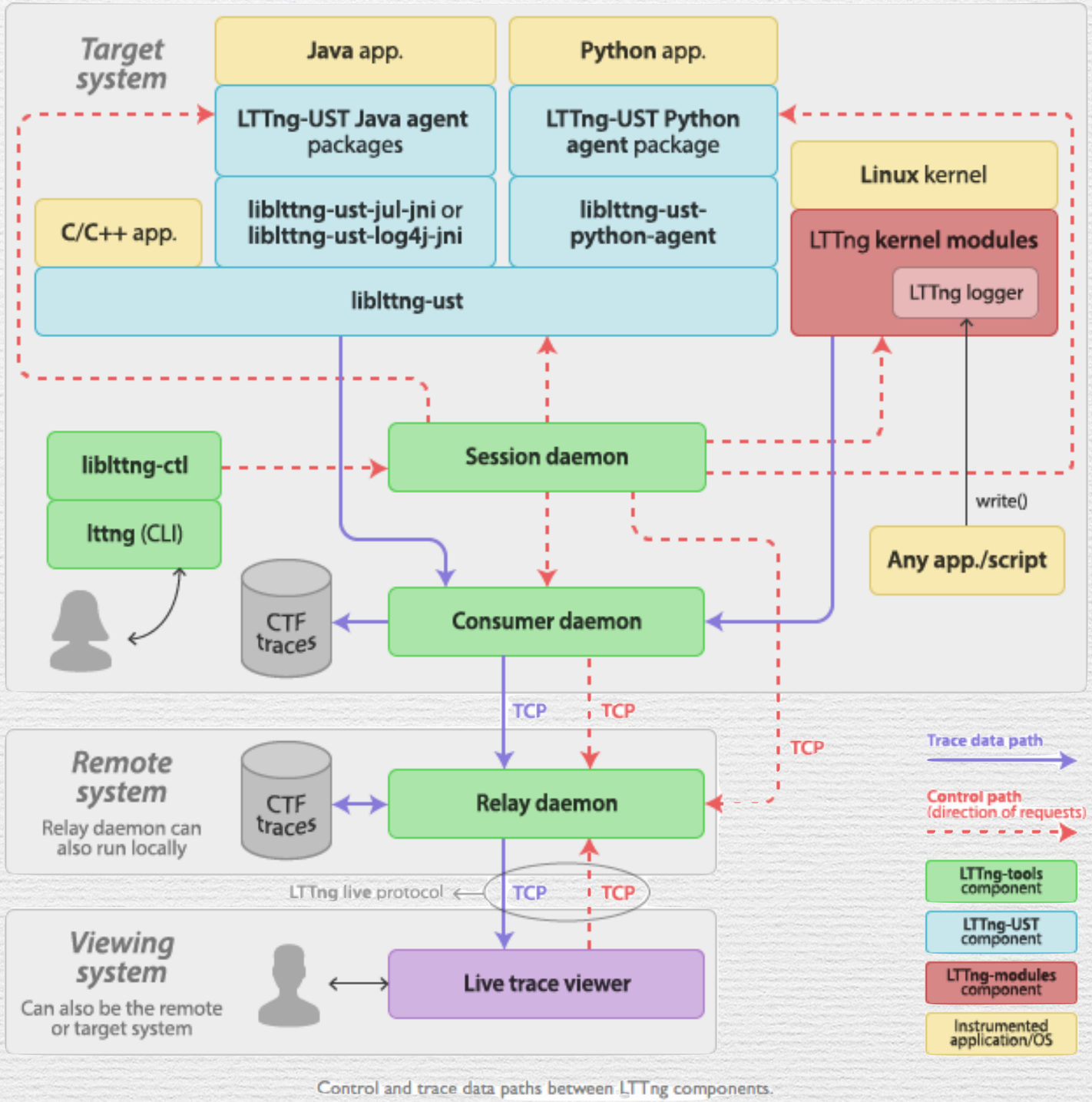
Perspective :

- Amélioration du stockage des informations lors de l'analyse.

Packages :

- LTTng-tool.
- LTTng-modules.
- LTTng-UST.

Contrôle



Control and trace data paths between LTTng components.