AADL resource requirements analysis with Cheddar

F. Singhoff, J. Legrand, L. Nana

University of Brest, France

LYSIC/EA 3883

Introduction and motivations

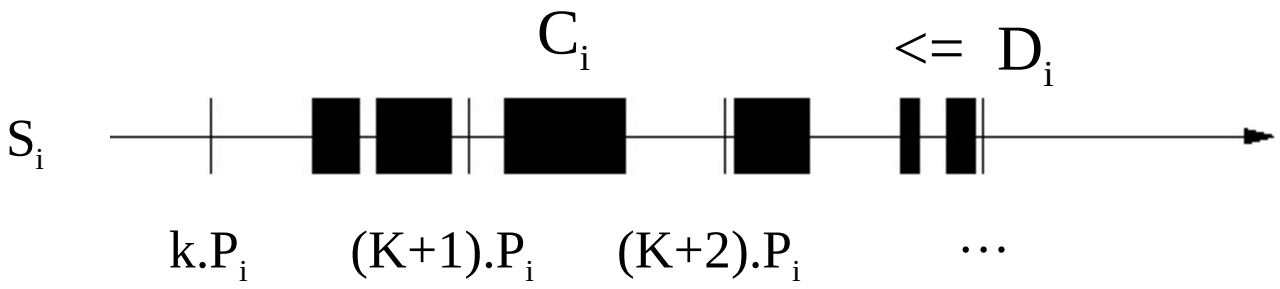
- **Real time scheduling Analysis** :
  - Provides a way to predict if temporal constraints will be met.
  - First results 30 years ago (Liu & Layland). Still sometimes unapplied. Unknown method ? Sometimes unpractical ?

  - ***Aims at providing tools to teach and apply real time scheduling analysis*:**
    - Should contain foundation that students/engineers have to know.
  - ***Aims at applying real time scheduling on practical cases* :**
    - How to investigate applications which are « outside » the theory ?
    - How to extend real time scheduling analysis to take distribution and buffers into account ?

Talk overview

- Introduction and project motivations
- Usual performance analysis methods
- Cheddar : a resource requirements analyzer
- Examples of AADL analysis :
  - AADL threads scheduling analysis
  - Event data port memory analysis
- Conclusion and ongoing works

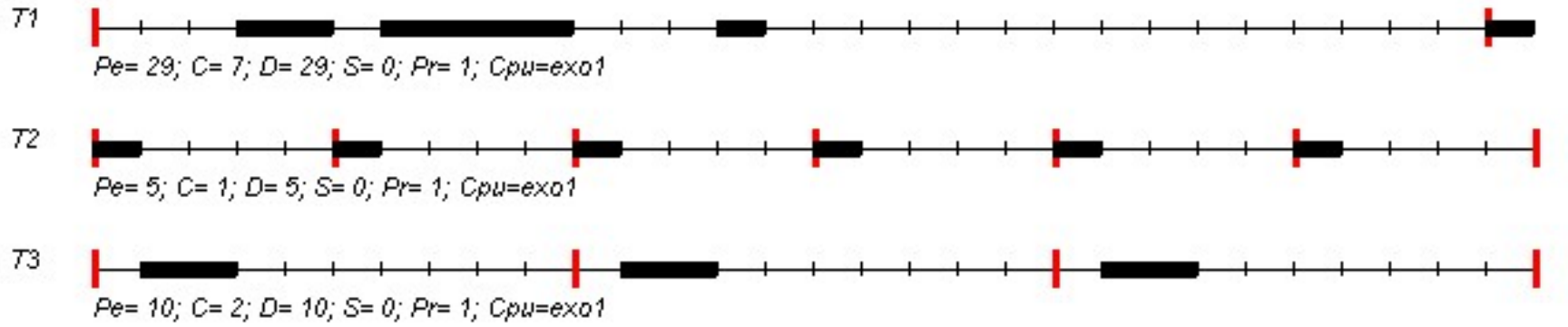
Usual performance analysis methods : real time scheduling (1/3)



- **The periodic task model :** (Liu & Layland, 1974)
  - Bound on execution time (capacity) : C_i
  - Delay between two wake-up times (period) : P_i
  - Temporal constraint to meet (deadline) : D_i
- **Classical real time scheduling algorithms :** Rate Monotonic, Earliest Deadline First, …
- Simulation vs analytical analysis (feasibility tests).

Usual performance analysis methods : real time scheduling (2/3)

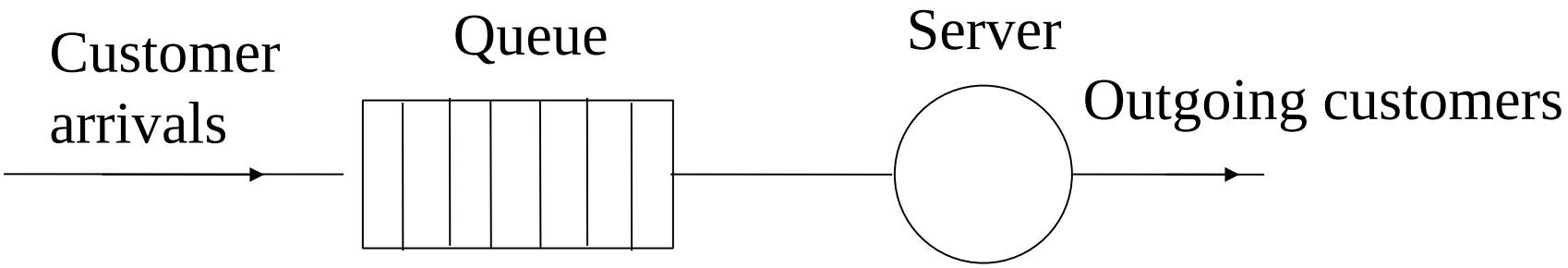
- **Simulation** : Rate Monotonic (RM, Liu & Layland 1974), run task with the smallest period



- **Analytical/Feasibility tests example :** the processor utilization factor test

$$\sum_{i=1}^{n} \frac{Ci}{Pi} \leq n(2^{1/n} - 1) \approx 69\%$$

Usual performance analysis methods : queueing systems (3/3)

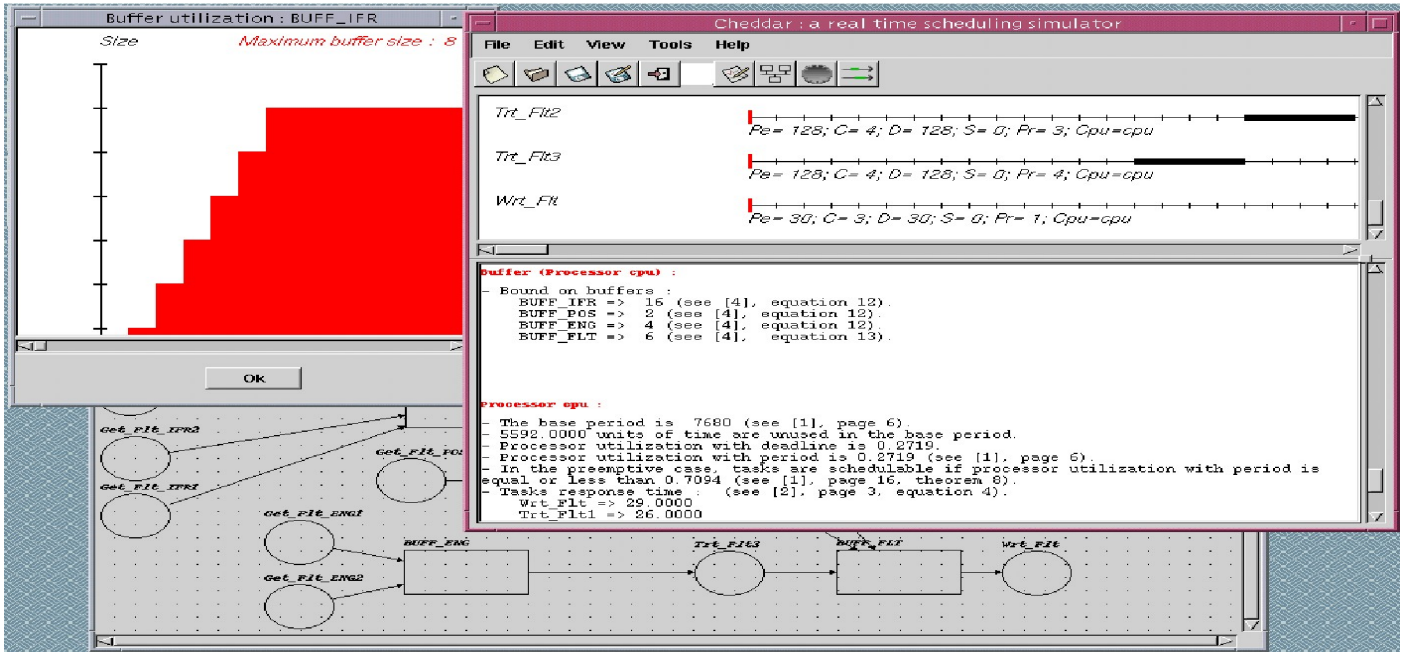


- **Queueing system Kendall's notation :** X/Y/n.
  - X : customer arrival rate (M,G,D).
  - Y : service time rate (M,G,D).
  - n : number of servers.
  - Examples : M/M/1, M/D/1, M/G/1, …
- **Goal :** From a given customer arrival/service time rate, compute analytical criterion such as customer waiting time and number of waiting customers.

Talk overview

- Introduction and project motivations
- Usual performance analysis methods
- Cheddar : a resource requirements analyzer
- Examples of AADL analysis :
  - AADL threads scheduling analysis
  - Event data port memory analysis
- Conclusion and ongoing works

Cheddar : a resource requirements analyzer (1/4)



- **Cheddar** : provides **analytical** and **simulation** performance analysis methods/tools. Focuses on tasks, processors, shared resources, buffers and task dependencies.
- First release on oct. 2002.

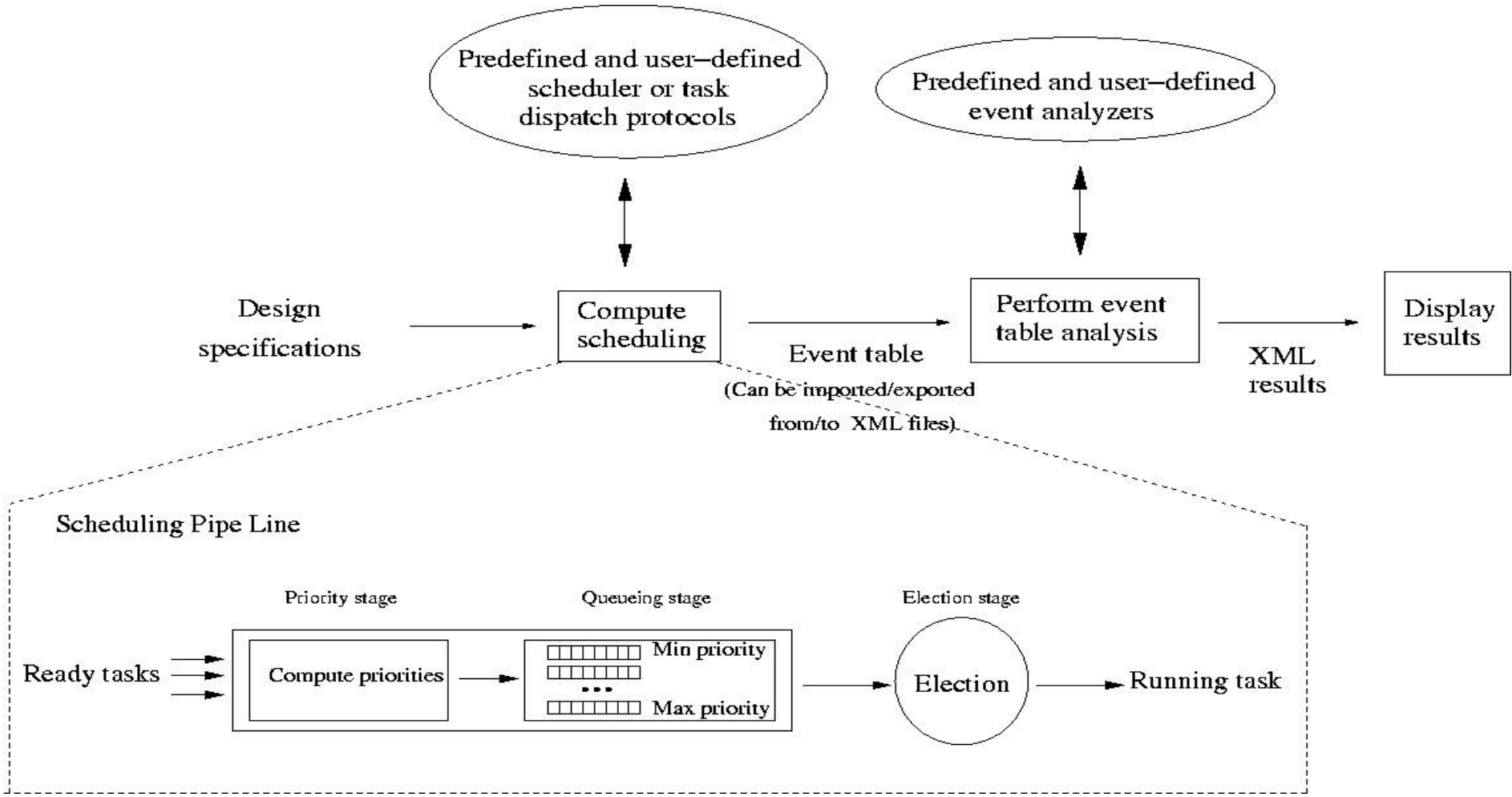
Cheddar : a resource requirements analyzer (2/4)

- **Provides classical schedulers/task dispatching policies :** periodic/aperiodic tasks, RM/DM/POSIX 1003.1b, EDF/LLF, ...
- Provides many analytical analysis/feasibility tests on different resources :
  - **Tasks/processors :** processor utilization factor,worst case response time, task priorities/deadlines assignment algorithms, tasks partitionning.
  - **Shared resources :** worst case blocking time (PIP/PCP).
  - **Buffers :** worst case/average case message waiting time and number of messages (P/P/1, M/P/1, M/M/1, …).

Cheddar : a resource requirements analyzer (3/4)

- Provides an extensible simulation engine :
  - When no analytical/feasibility test exist.
  - Compute scheduling time lines and run time line analyzers **(not a proof !)** :
    - **Processors/tasks :** worst/best/average response time, number of context switches/preemptions, missed deadlines, ...
    - **Buffers :** maximum/average message waiting time, maximum/average number of messages ...
    - **Shared resources :** worst/best/average shared resource blocking task, priority inversion, deadlock ...
  - Can be extended with user-defined schedulers, task dispatching policies and time line analyzers Ada like piece of code.

Cheddar : a resource requirements analyzer (4/4)



Cheddar and AADL

- Cheddar was not originally designed to work with AADL. How the tool can be applied to such design language ?
- In the sequel, we consider the following points :
  - AADL thread scheduling analysis.
  - Buffer requirements of AADL event data ports.

Talk overview

- Introduction and project motivations
- Usual performance analysis methods
- Cheddar : a resource requirements analyzer
- Examples of AADL analysis :
  - AADL threads scheduling analysis
  - Event data port memory analysis
- Conclusion and ongoing works

AADL threads scheduling analysis (1/9)

- AADL includes most of the features used in the context of real time scheduling analysis.
- Nevertheless, the following questions have to be investigated :
  - Can we model any built-in Cheddar's scheduler/task dispatching protocols ?
  - Are standard properties enough to perform analytical/feasibility tests on any resources ?
  - How to express user-defined scheduler/task dispatching protocols ?

=> we need some new AADL properties

AADL threads scheduling analysis (2/9)

- **Example 1 :** a set of periodic/aperiodic threads scheduled with POSIX1003.1b and Rate Monotonic schedulers.

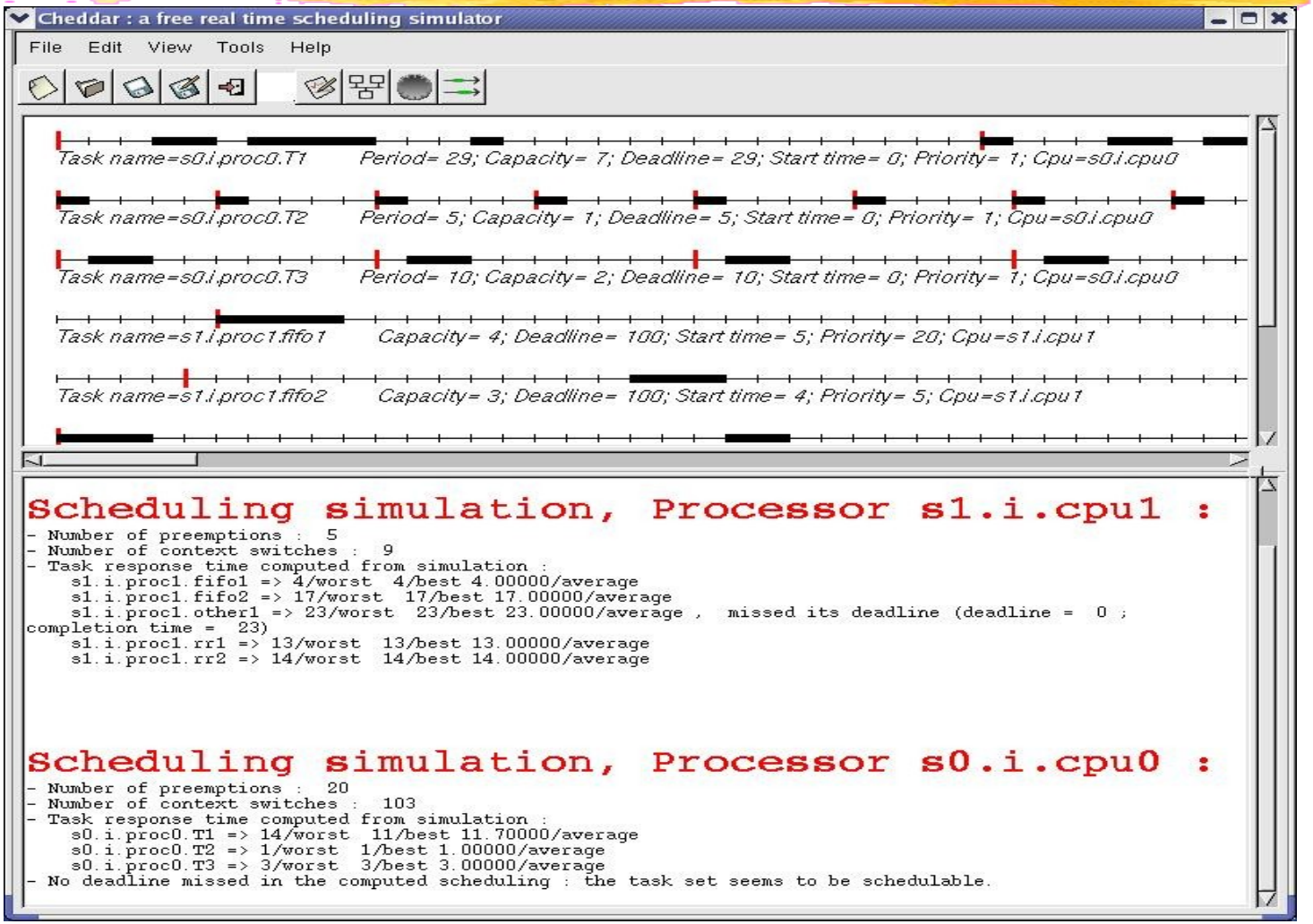
```
thread implementation T3.i
    properties
        Dispatch_Protocol => Periodic;
        Compute_Execution_time =>  1 ms ..  2 ms;
        Deadline  =>  10;
        Period =>  10;
end T3.i;
thread implementation fifo2.i
    properties
        Dispatch_Protocol => Background;
        Compute_Execution_time => 1 ms ..  3 ms;
        Cheddar_Properties::POSIX_Scheduling_Policy =>
                SCHED_FIFO;
        Cheddar_Properties::Fixed_Priority =>  5;
        Cheddar_Properties::Dispatch_Absolute_Time =>  4;
        Deadline => 100;
end fifo2.i;
```

```
process implementation proc0.i
    subcomponents
        T1 : thread T1.i;
            ....
processor implementation rma_cpu.i
    properties
        Scheduling_Protocol => RATE_MONOTONIC;
        Cheddar_Properties::Preemptive_Scheduler => true;
        Cheddar_Properties::Scheduler_Quantum =>  0;

end rma_cpu.i;
processor implementation posix_cpu.i
    properties
        Scheduling_Protocol => HIGHIEST_PRIORITY_FIRST;
        Cheddar_Properties::Preemptive_Scheduler => true;
        Cheddar_Properties::Scheduler_Quantum =>  2;

end posix_cpu.i;
```

AADL threads scheduling analysis (3/9)

Compute simulation

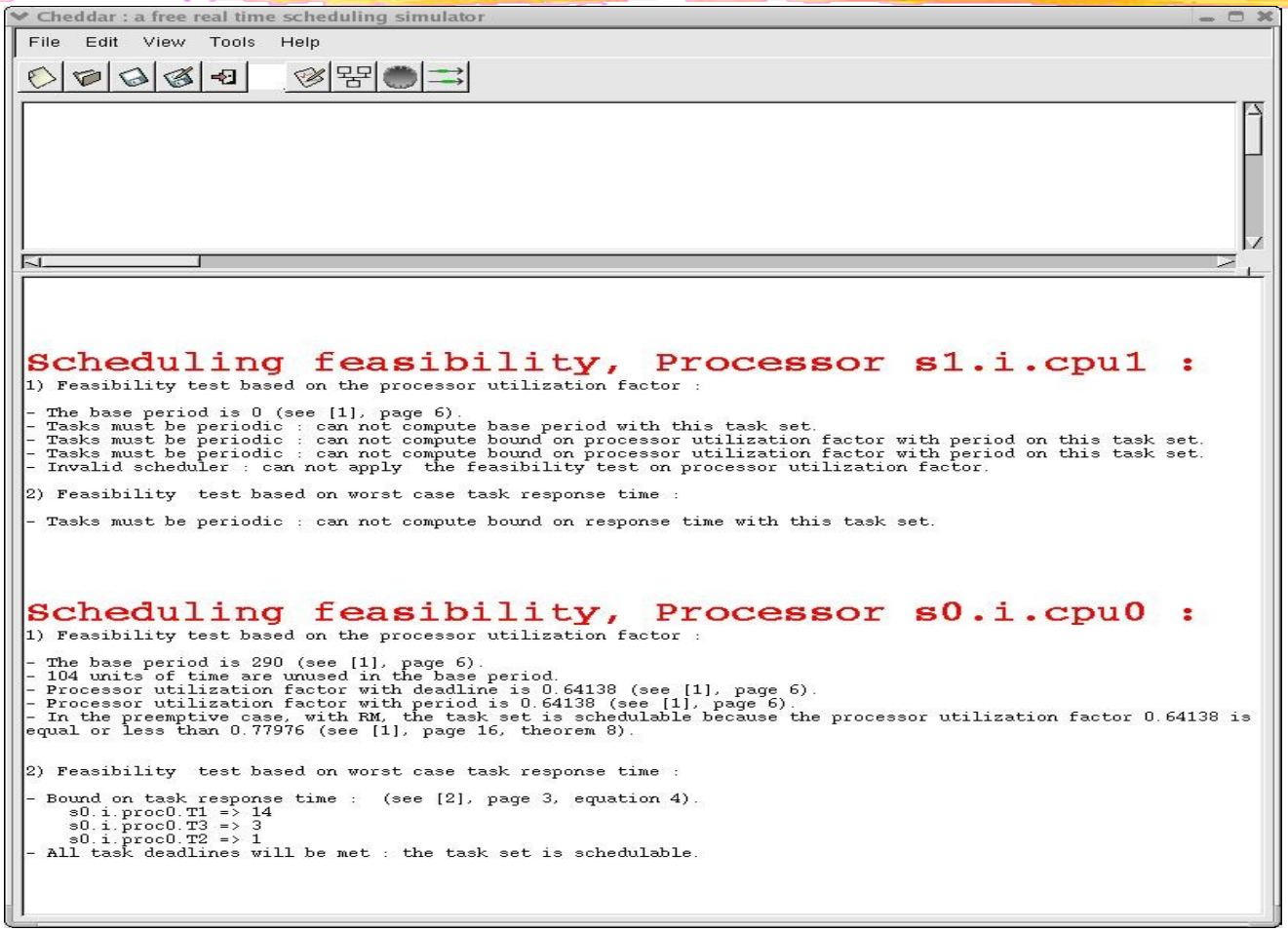
Analysis (eg. deadlines, response times)



AADL threads scheduling analysis (4/9)

No scheduling required

Analytical analysis (periodic only)



AADL threads scheduling analysis (5/9)

- **Example 2 :** a set of periodic threads sharing a PCP data.

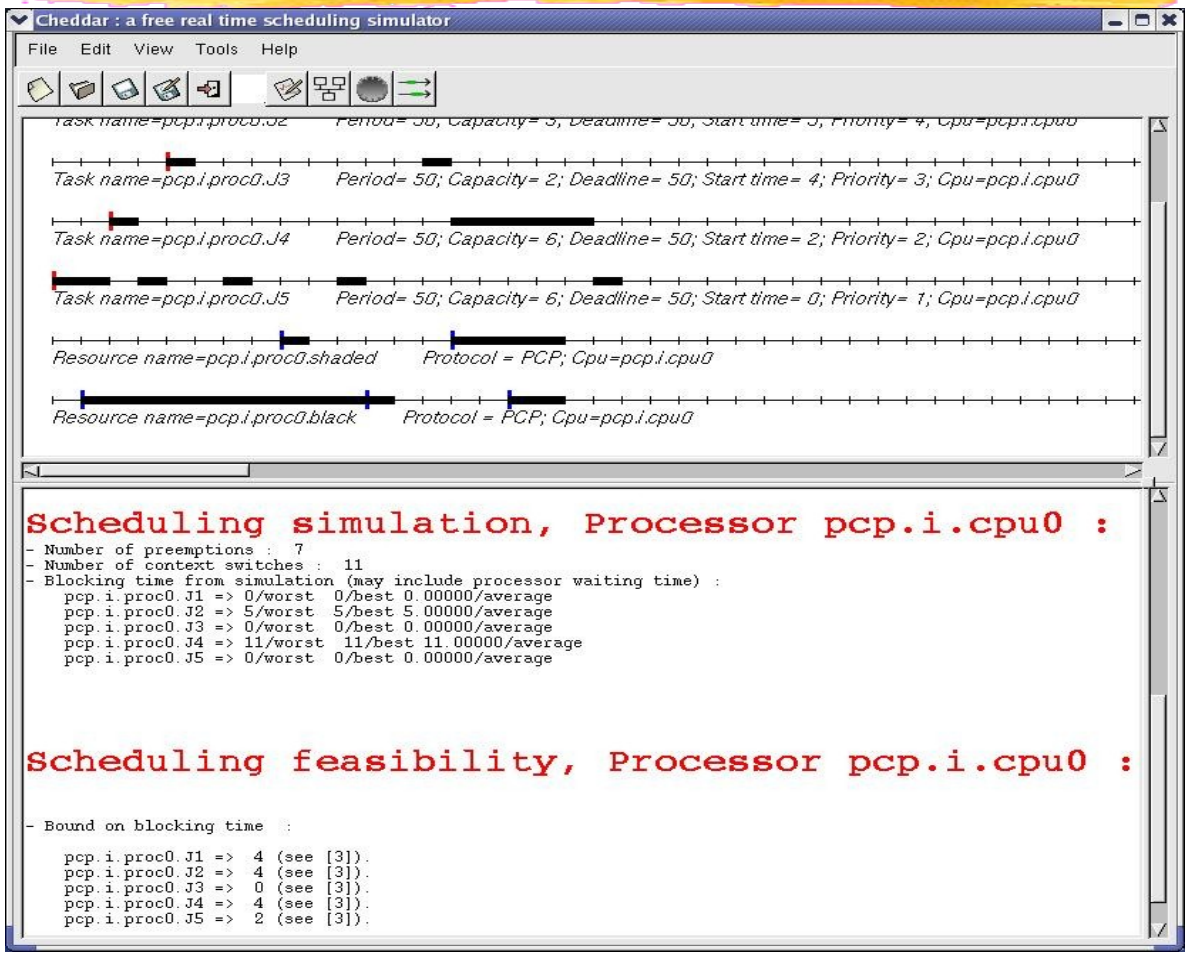
```
data implementation black.i
    properties
        Cheddar_Properties::Data_Concurrency_State =>  1;
        Concurrency_Control_Protocol =>
                PRIORITY_CEILING_PROTOCOL;
end black.i;

thread J2
    features
        black_features : requires data access black.i;
end 24;

thread implementation J4.i
  properties
    Dispatch_Protocol => Periodic;
    Cheddar_Properties::Fixed_Priority =>  2;
    Cheddar_Properties::Bound_On_Date_
            Blocking_Time =>  5 ms;
    ....
```

```
process implementation proc0.i
    subcomponents
        J1 : thread J1.i;
        J2 : thread J2.i;
        ...
        shaded : data shaded.i;
        black : data black.i;
    connections
        data access shaded -> J1.shaded_features;
        data access black -> J2.black_features;
        ...
    properties
        Cheddar_Properties::Critical_Section => (
            "shaded",  "J1","2","3",
            "shaded",   "J4","1","5",
            "black",    "J2","1","2",
            ...          );
end proc0.i;
```

AADL threads scheduling analysis (6/9)

Data access

Bound on data waiting time (analytical & simulation)



AADL threads scheduling analysis (7/9)

- **Example 3 :** user-defined schedulers, task dispatching protocols and analyzers.

```
system s0
end s0;

system implementation s0.i
      subcomponents
            cpu : processor mixed.i;
            p1  : process proc.i;
      properties
            Actual_Processor_Binding =>
                  reference cpu applies to p1;
            Source_Text =>
                  "number_of_sporadic_activations.sc";
end s0.i;
```

```
thread implementation T1.i
      properties
        Compute_Execution_time =>  1 ms ..  3 ms;
        Cheddar_Properties::Fixed_Priority =>  1;
        Dispatch_Protocol => Parametric;
        Source_Text => "sporadic_activation";
        Deadline  =>  100;
        Period =>  5;
end T1.i;

processor implementation mixed.i
      properties
        Scheduling_Protocol => parametric;
        Cheddar_Properties::Preemptive_Scheduler => true;
        Source_Text =>
      "mixed_time_sharing_and_real_time.sc";
end mixed.i;
```

AADL threads scheduling analysis (8/9)

start_section:

```
a_max : integer;
i : integer;
...
exponential(gen1, 10);
current_activation:=integer'last;
dynamic_priority : array (tasks_range) of integer;

number_of_activation : array (tasks_range) of integer;
number_of_activation:=0;
```

priority_section:

```
for i in  tasks_range loop
    if tasks.activation_number(i)<current_activation
      then current_activation:=tasks.activation_number(i);
    end if;
end loop;

dynamic_priority:=0;
for i in  tasks_range loop
    if tasks.activation_number(i)=current_activation
      then dynamic_priority(i):=tasks.priority(i);
    end if;
end loop;
```

election_section:

```
return max_to_index(dynamic_priority);
```

task_activation_section:

```
set sporadic_activation max(tasks.period, gen1);
set random_activation gen1;
```

gather_event_analyzer_section:

```
if events.type = "task_activation"
        then
        id := get_task_index(events.task_name);
        number_of_activation(id):=number_of_activation
                (id)+1;
end if;
```

display_event_analyzer_section:

```
put(tasks.name,0,2);
put(number_of_activation,0,2);
```

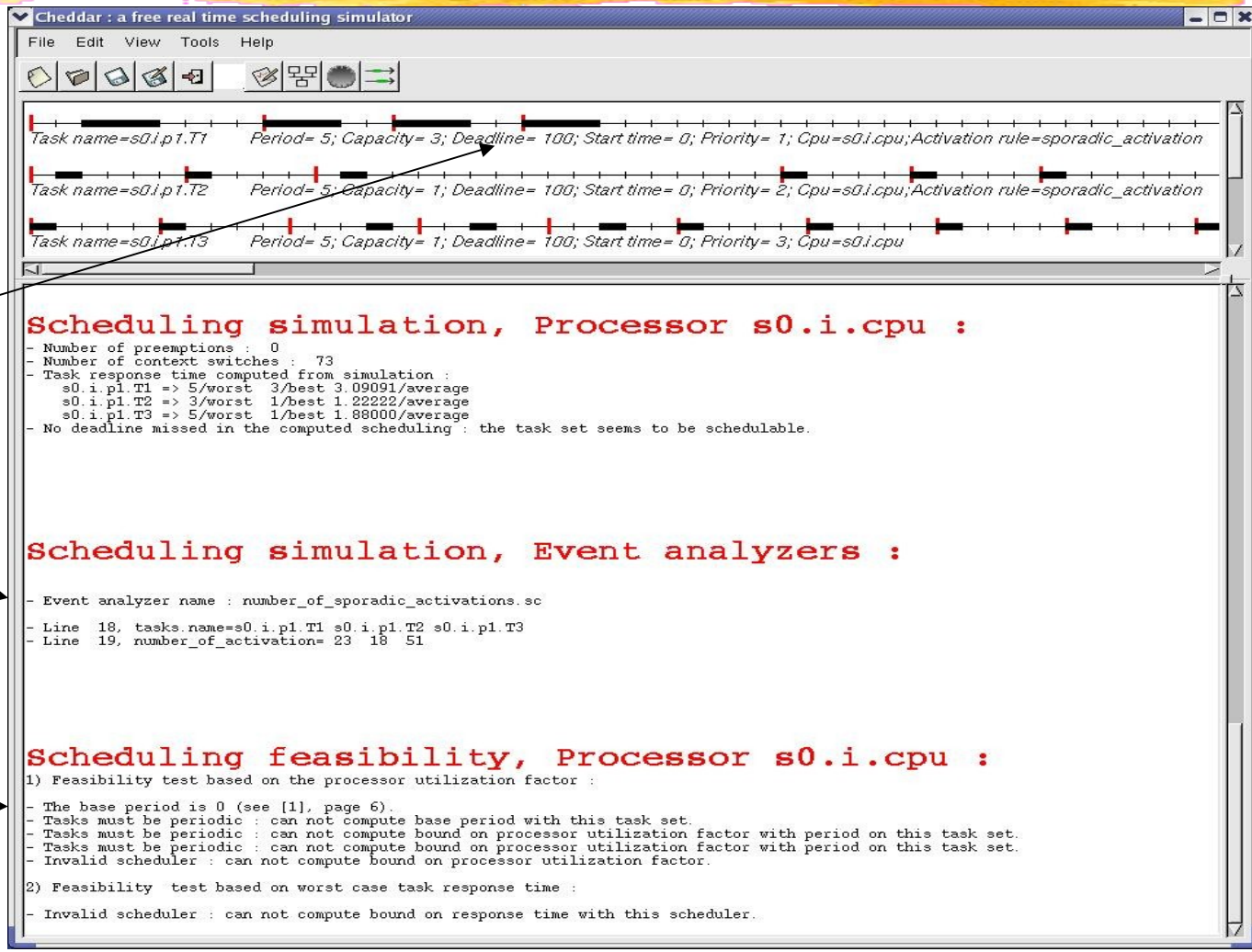
AADL threads scheduling analysis (9/9)

Mixed user-defined real time/time sharing scheduler

User-defined sporadic task

User-defined analyzer

Don't expect any analytical result !



Talk overview

- Introduction and project motivations
- Usual performance analysis methods
- Cheddar : a resource requirements analyzer
- Examples of AADL analysis :
  - AADL threads scheduling analysis
  - Event data port memory analysis
- Conclusion and ongoing works

Event data ports memory requirement analysis (1/5)

- Event data port are used for message transmission between threads. Events are queued ... Queueing system may be able to predict memory requirement .... But we have to answer two questions :
  - How to take into account thread dispatching (eg. periodic) ?
  - How to take into account thread scheduling (eg. RM) ?
- **Cheddar provides :**
  - Buffer simulation services.
  - Feasibility tests (Legrand & Singhoff & Nana & Marcé 2003).

Event data ports memory requirement analysis (2/5)

- **Consumers or producers may be periodic RM scheduled tasks** :
  - Define a new consumption/arrival rate : the P rate.
  - Define new queueing systems based on the P rate.

- **Worst case analytical analysis based on P/P/1 :**
  - Periodic arrivals assumption : minimum time between 2 message arrivals is known. Worst case number of messages/message waiting time.
  - P/P/1 Resolution : based on ATM/AAL1.
- **Average case analytical analysis based on M/P/1 :**
  - Random arrivals assumption : mean time between 2 message arrivals
  - M/P/1 approximation : M/G/1 with P average service time.

Event data ports memory requirement analysis (3/5)

- **Example of buffer feasibility test (P/P/1 queueing system analysis) :**
  - The maximum number of messages in a buffer shared by N periodic producers and 1 periodic consumer (with deadline <= period) is :
    - 2.N (harmonic thread set)
    - 2.N+1 (other cases)

Event data ports memory requirement analysis (4/5)

- **Example 4 :** event data port communications

```
processor implementation cpu_rm.i
      properties
            Scheduling_Protocol => Rate_Monotonic;
            ...
end cpu_rm.i;
process implementation p0.i
 subcomponents
   Producer1 : thread Producer.i;
   Producer2 : thread Producer.i;
   Consumer1 : thread Consumer.i;
connections
   event data  port Producer1.Data_Source ->
      Consumer1.Data_Sink;
   event data  port Producer2.Data_Source ->
      Consumer1.Data_Sink;
end p0.i;
```

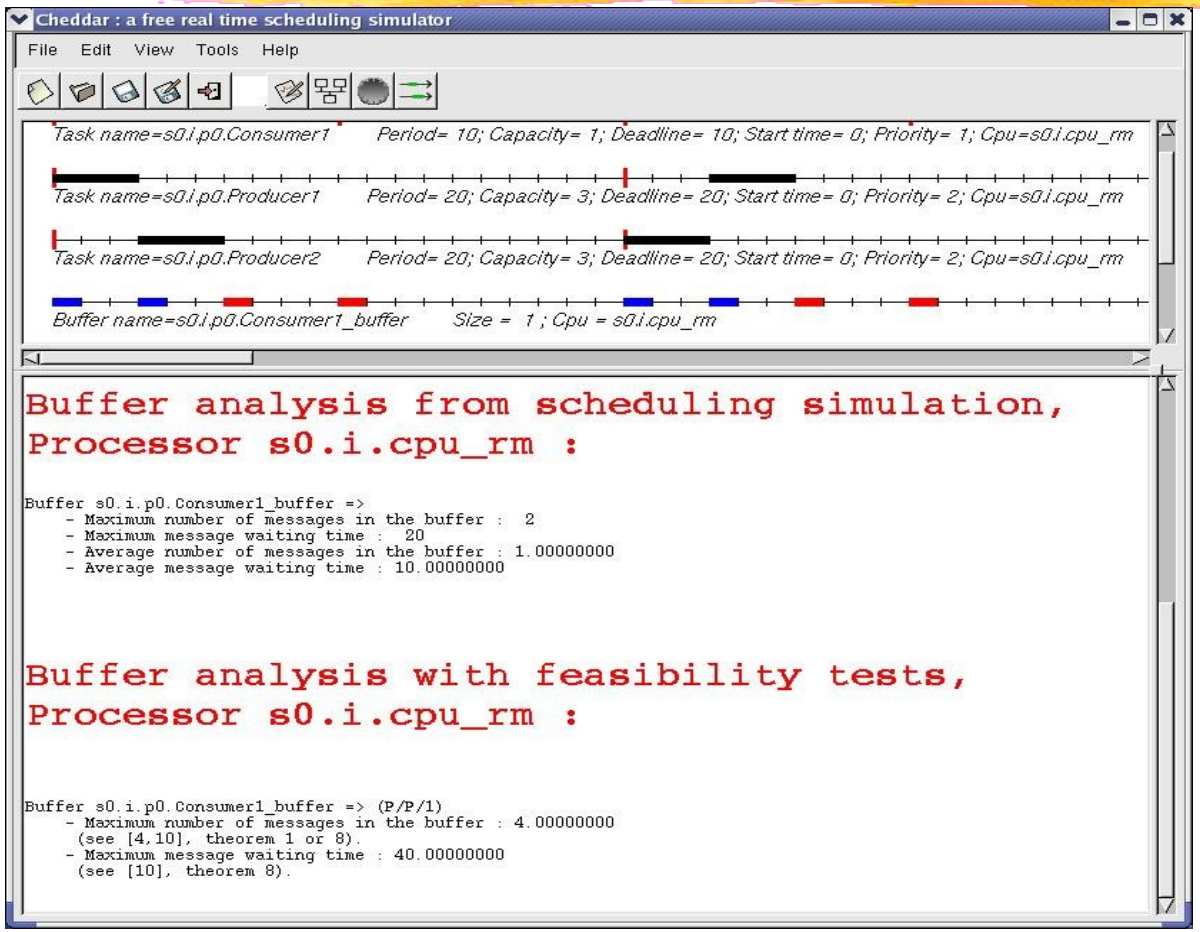
```
thread Producer
 Features
   Data_Source : out event data port;
end Producer;

thread implementation Producer.i
      properties
            Dispatch_Protocol=>periodic;
            ...
end Producer.i;

thread Consumer
 features
   Data_Sink : in event data port;
end Consumer;
thread implementation Consumer.i
      properties
            Dispatch_Protocol=>periodic;
            ...
end Consumer.i;
```

Event data ports memory requirement analysis (5/5)

Buffer simulation

Analysis from simulation

Worst case queueing system analysis (based on P/P/1)



Talk overview

- Introduction and project motivations
- Usual performance analysis methods
- Cheddar : a resource requirements analyzer
- Examples of AADL analysis :
  - AADL threads scheduling analysis
  - Event data port memory analysis
- Conclusion and ongoing works

Conclusion and ongoing works

- **Cheddar's current status :**
  - Provides feasibility tests and simulation features on different AADL resources (see the SIGADA'05 paper for details).
  - This AADL analyzer will be distributed by the end of october ... but it has to be tested !!!
  - Implementation based on Ocarina (AADL parser). Stood plug-in.
- **Ongoing works :**
  - Related to task precedency relationships (AADL connections)
  - Scheduling according to task precedency and end to end task response time (analytical Holistic computation).