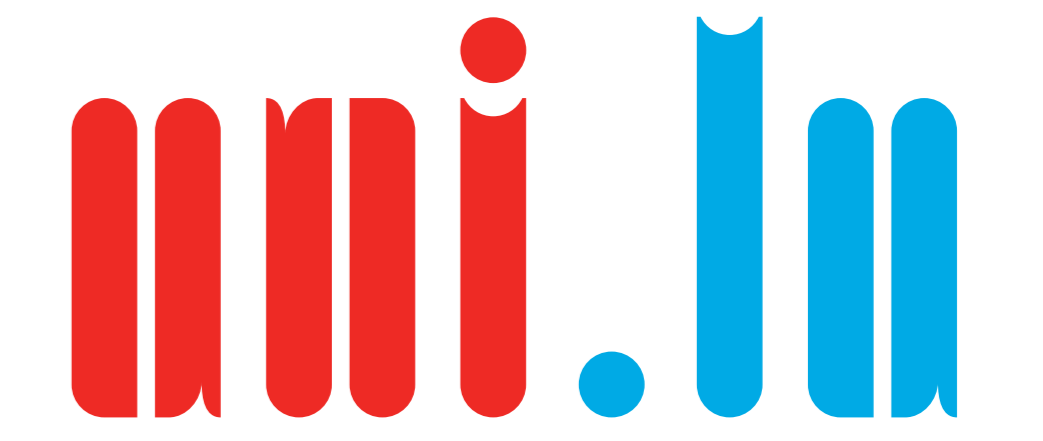


Laboratory for Advanced Software Systems

# Designing Reliable Real-Time Concurrent Object-Oriented Software Systems

Alfredo Capozucca, Nicolas Guelfi

LASSY - University of Luxembourg,  
6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg



UNIVERSITÉ DU  
LUXEMBOURG

## Introduction

- **Coordinated Atomic Actions conceptual framework (CaaFWrk)**: fault tolerance technique meant for increasing the reliability of concurrent object-oriented software systems

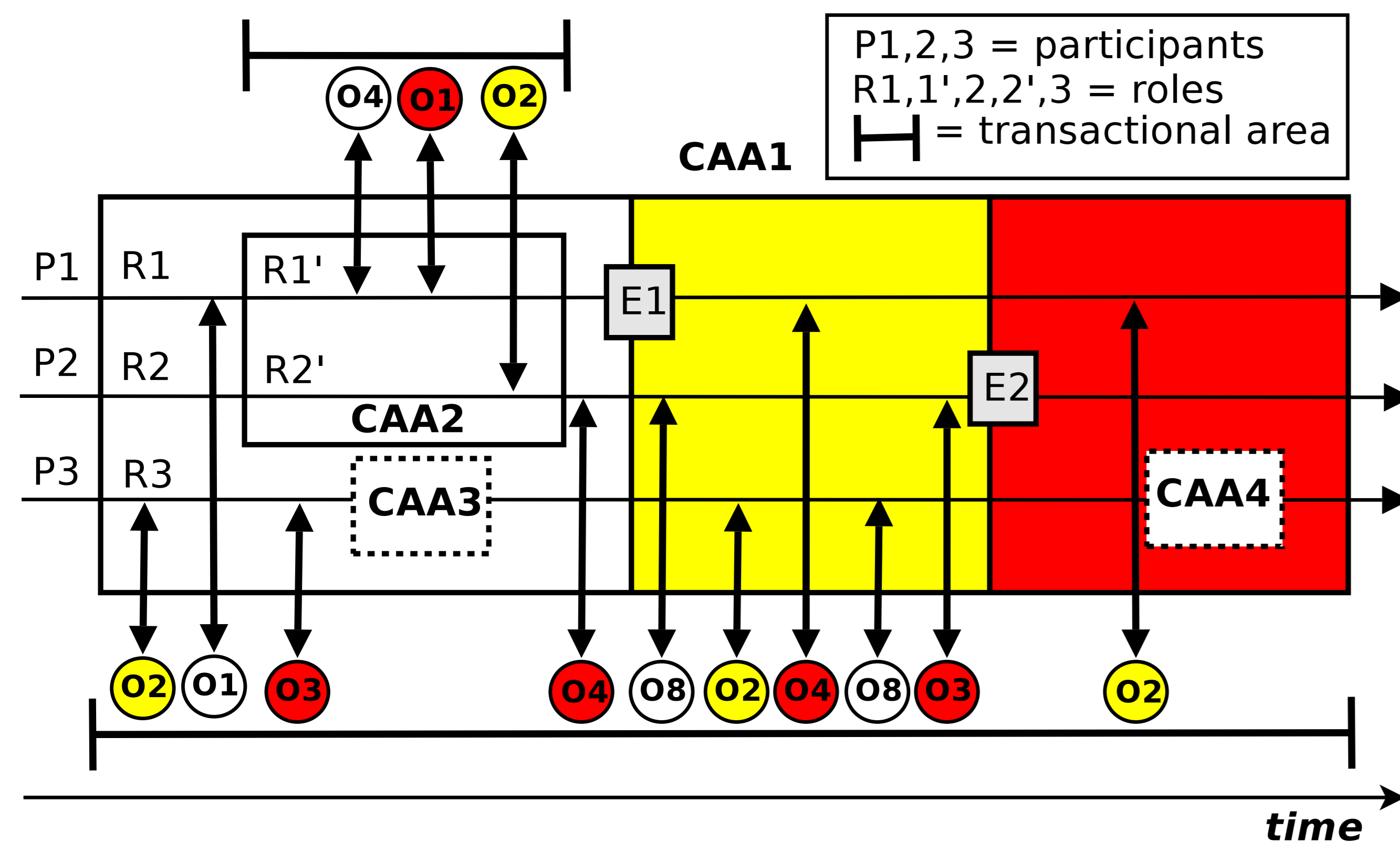


FIGURE 1: CaaFWrk core elements.

## Problem definition

Real-time software systems are concurrent (either inherent or imposed) and very often have reliability requirements. Thus, these types of software systems are first-class candidates to be designed using the CaaFWrk. **Timing requirements imposed by most real-time software systems cannot be modelled (or, at least, not easily) by the CaaFWrk as it is.**

## Work summary

1. Analysis of the first proposal for time extensions on the CaaFWrk (aka Timed-CaaFWrk)
2. Description of the open issues found on this proposal:
  - Timing constraints over roles
  - Recovery semantics
  - CAAs/Roles interleaving
  - Pre-emptive scheme to speed up the recovery process
3. Solutions to the open issues  $\Rightarrow$  **Timed-CaaFWrk++**

## Results

The **Timed-CaaFWrk++** conceptual framework:

- allows to set multiple timing constraints over a *Role* and specify those recovery actions to be taken in case a constraint is violated  $\Rightarrow$  *less(timeExpr){...} exceed{...}*
- extends the recovery semantics: handling an exception in the scope of a *Role* before starting the cooperative recovery
- includes the *Immediate Ceiling Priority Protocol* as scheduling policy to reduce the non-determinism found within a software system designed by several CAAs executing concurrently
- supports both the pre-emptive and blocking schemes, the decision about which one to use is made by the scheduler  $\Rightarrow$  *if  $t_E - t_e \leq t_A$  then complete else abort*, where
  - $t_e$  = Elapsed Time (measured by a timer at runtime)
  - $t_E$  = Maximum Elapsed Time (defined at design-time)
  - $t_A$  = Abortion Time (either:
    - \* calculated upon release of the CAA at runtime, or
    - \* defined at design-time)

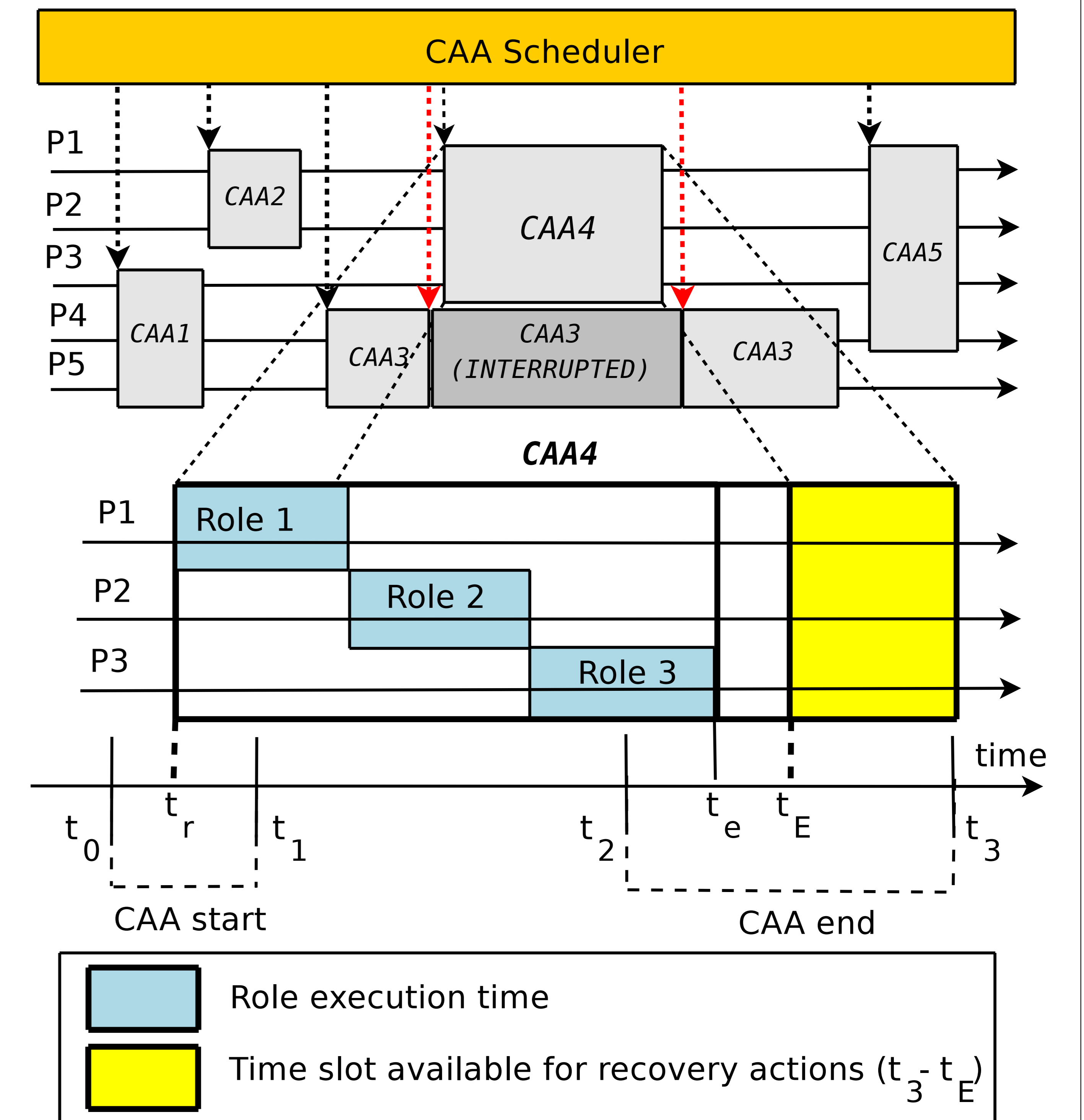


FIGURE 2: It shows a CAA (here  $CAA_4$ ) interrupting the execution of another CAA (here  $CAA_3$ ) with a lower priority. The figure also shows the internal execution of the CAA with higher priority and those timing constraints it may hold.

## Conclusion

- **Timed-CaaFWrk++** is a new conceptual framework to design reliable concurrent real-time software systems. Whether it covers all the needs and is desirable for constructing this kind of software system can only be determined from future practical experience.