Adding Time-Step Management to Distributed Ptolemy-HLAcerti framework

Yanxuan LI, Janette Cardoso, Pierre Siron

11th Biennial Ptolemy Miniconference

Berkeley, October 16, 2015

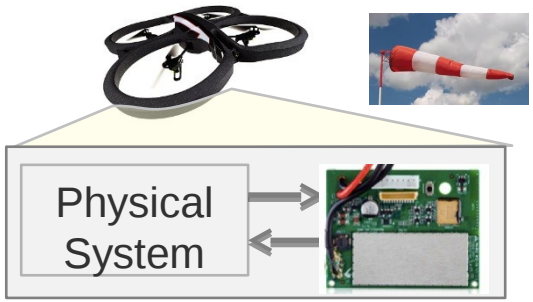
Plan

- Introduction
- High Level Architecture (HLA-CERTI)
- PTII-HLA Framework
- Example Producer/consumer (NER, TAR)
- Conclusion & Perspectives

Distributed Simulation

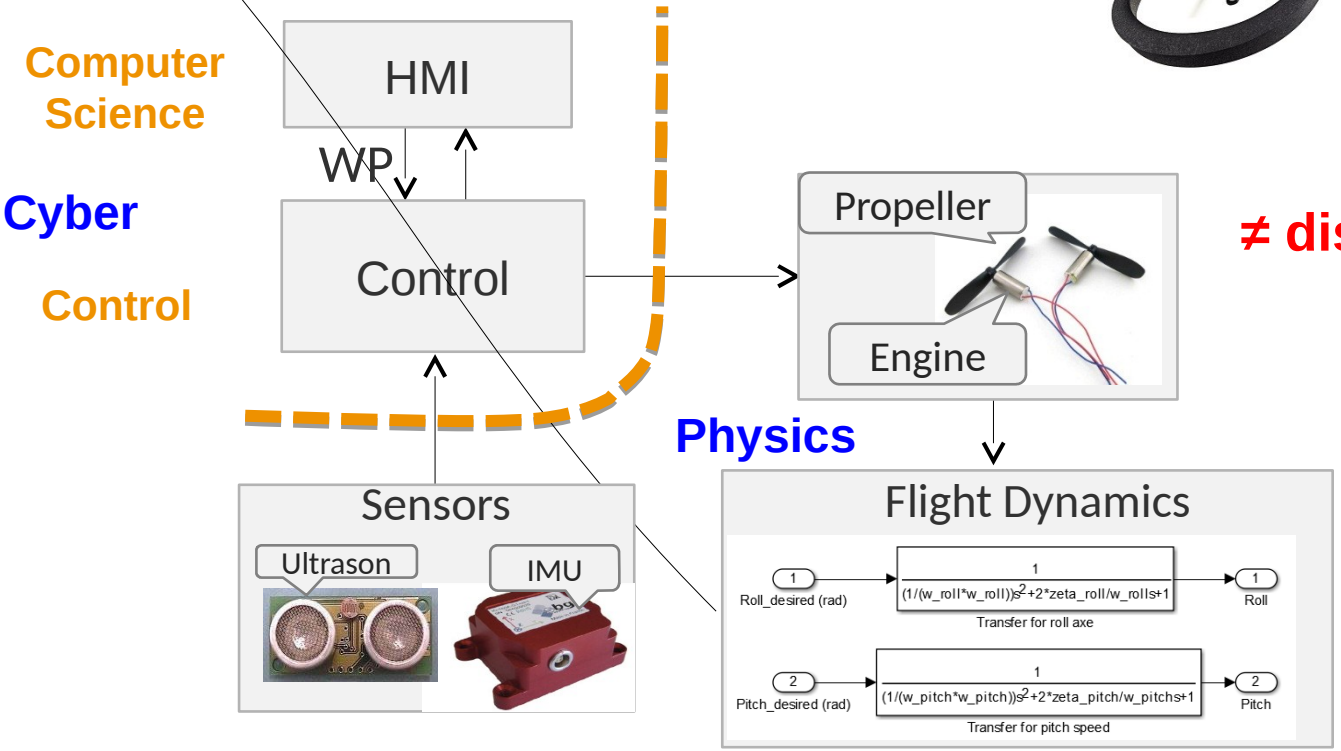
WHY ?

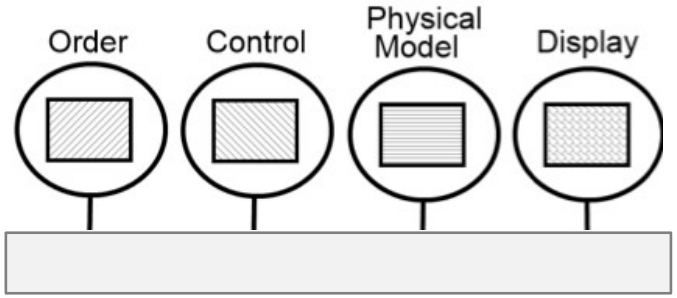
- System itself is distributed



- System is too complex and/or too much models
  - Reduce the simulation time
  - Enable larger simulations
  - Integrate several (different) simulators into a single simulation environment.

Cyber-Physical System



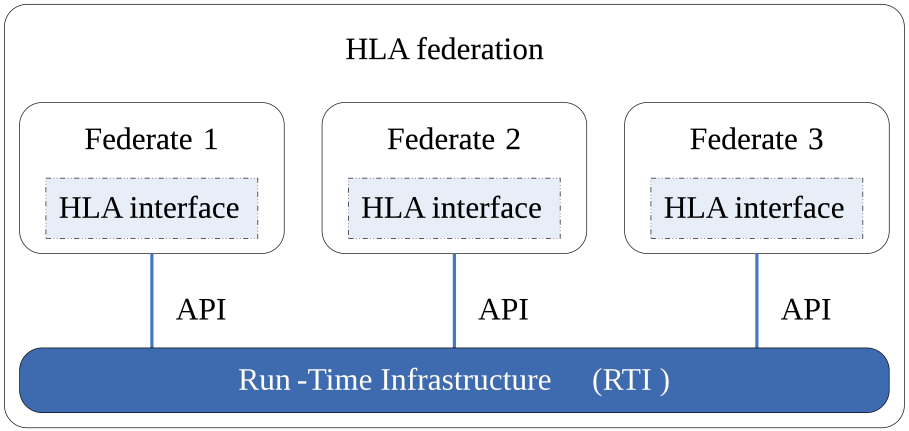


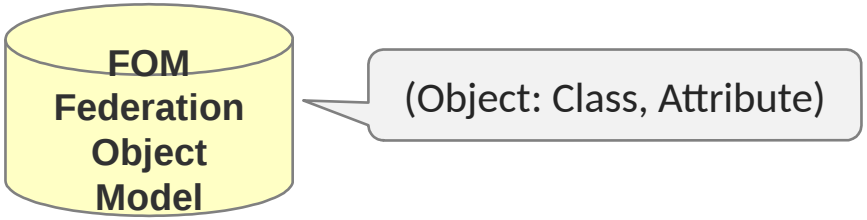
- **Heterogeneous models**
- **Distributed simulation** HLA CERTI

HLA High Level Architecture

- High Level Architecture for distributed discrete event simulations
- IEEE standard (1516)
- Interoperability and reuse

Federate = Simulator
Federation = Distributed simulation

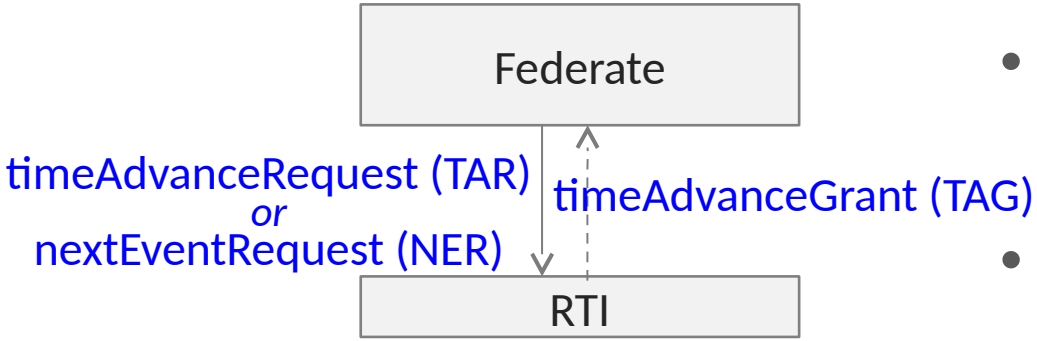





```
(FED
   (Federation QuadrirotorSimple)
   (FEDversion v1.3)
   (objects
       (class ObjectRoot
         (attribute privilegeToDelete  reliable timestamp)
         (class RTIprivate)
         ;; ****************************************
         ;; QUADROTOR
         ;; ****************************************
         (class QUADROTOR
             (attribute NAME RELIABLE TIMESTAMP)
           | ;; ************************************************
             ;; DRONE STATE AND VARIABLES
             ;; ************************************************
             (class DRONE_POSITION
                 (attribute X reliable timestamp)
                 (attribute Y reliable timestamp)
                 (attribute Z reliable timestamp)
                  )
             (class DRONE_ATTITUDE
                 (attribute ROLL reliable timestamp)
                 (attribute PITCH reliable timestamp)
                 (attribute YAW reliable timestamp)
                  )
             (class DRONE_CALCULATED
                 (attribute ROLL_CALCULATED reliable timestamp)
                 (attribute PITCH_CALCULATED reliable timestamp)
                 (attribute YAW_CALCULATED reliable timestamp)
                  )
             (class DRONE_POWER
                 (attribute THRUST reliable timestamp)
                  )
             ;; ************************************************
             ;; AUTOPILOT
             ;; ************************************************
             (class AUTOPILOT
                 (attribute INIT reliable timestamp)
                 (attribute X_ORDER reliable timestamp)
                 (attribute Y_ORDER reliable timestamp)
```

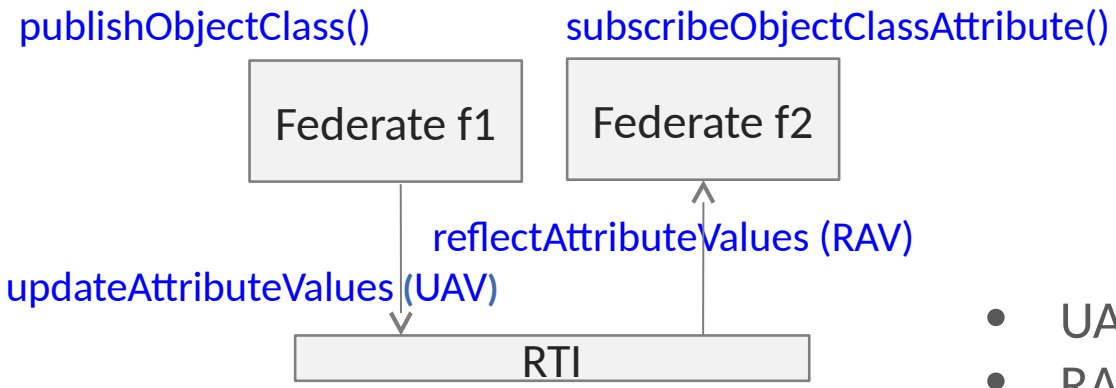
Areas	Services	Description (non-formal)
Federation	createFederationExecution()	create a federation
	joinFederationExecution()	join a federation
	resignFederationExecution()	quit a federation
	destroyFederationExecution()	destroy a federation
	registerFed...Sync...Point()	register a synchronization point
	sync...PointReg...Succeeded()⋆	register synchro point succeeded
	announceSynchro...Point() ⋆	wait a synchronization point
	synchronizationPointAchieved()	release from a synchro. point
	federationSynchronized() ⋆	announce synchronization
	tick()	allow to get callbacks from RTI
Declaration	publishObjectClass()	declare publication of a class
	subscribeObj..ClassAttributes()	subscribe to a class
	unsubscribeObjectClass()	unsubcribe to a class
	unpublishObjectClass()	unpublish a class
Object	registerObjectInstance()	register an object instance
	discoverObjectInstance() ⋆	for object instances discovering
	updateAttributeValues(), UAV	send & update value
	reflectAttributeValues() RAV ⋆	receive updated value
Time	enableTimeRegulation()	declare federate is regulator
	timeRegulationEnabled() ⋆	federate as regulator succeeded
	enableTimeConstrained()	declare federate constrained
	timeConstrainedEnabled() ⋆	federate as contrained succeeded
	timeAdvanceRequest(), TAR	ask to advance federate's time
	timeAdvanceGrant() TAG ⋆	notify time advancement granted
	nextEventRequest(), NER	ask to advance federate's time

HLA Time Management



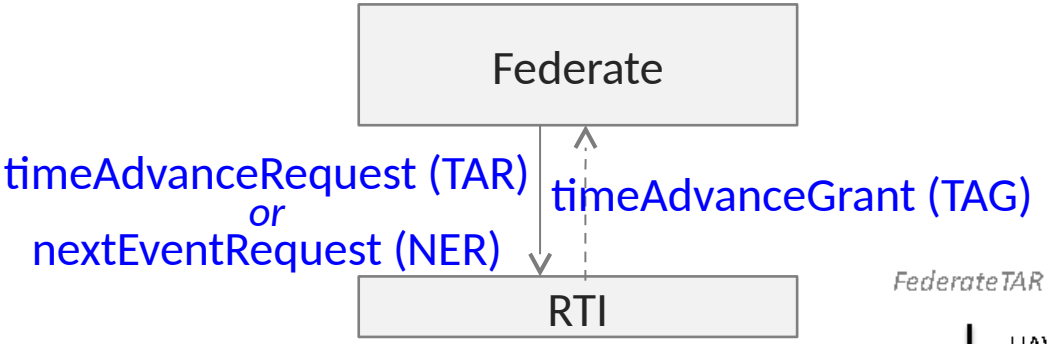
- Time Advancing mechanisms: the federation will be conservative, de terministic and repeatable.
- The federates can be:
  - Time-Stepped (TAR)
  - Event-Driven (NER)

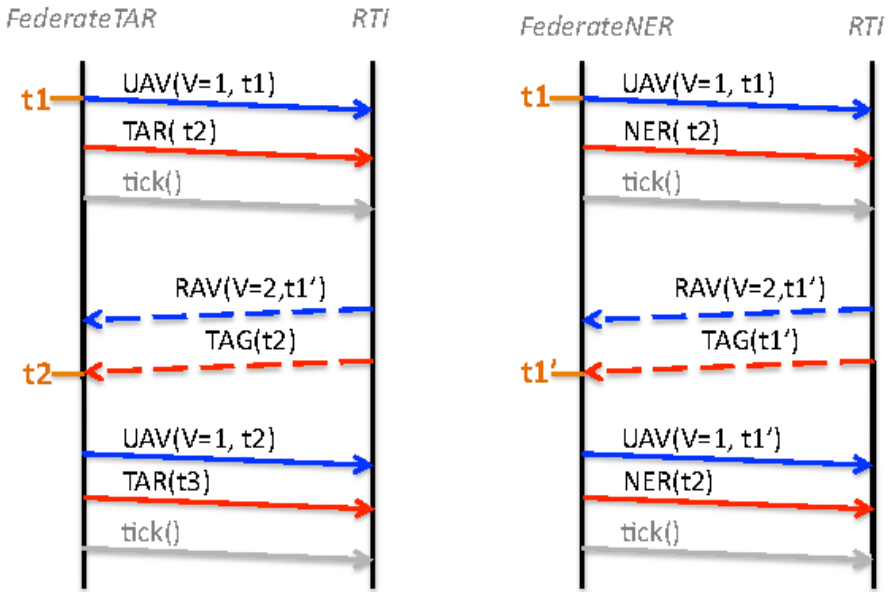
HLA Object Management



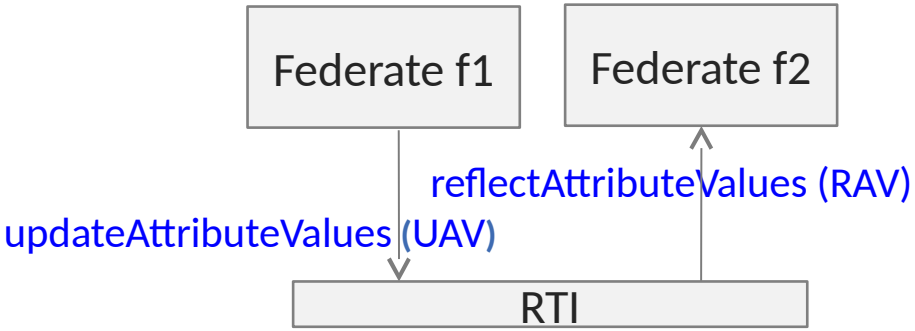
- UAV(object,attribute,timestamp)
- RAV(object,attribute,timestamp)

HLA Time Management





HLA Object Management



Bridges between PTII and HLA standard

PTII	HLA standard
Model Instance	Federate
Model Time: **t**	Logical Time: **Lt**
Data (Token)	Data (Attribute of an object class instance)
Event e(t, n, val)	**Event** e(obj, val, Lt)
Director advances time	RTI advances time
Input & Output ports	HlaPublisher + HlaSubscriber (UAV+RAV)

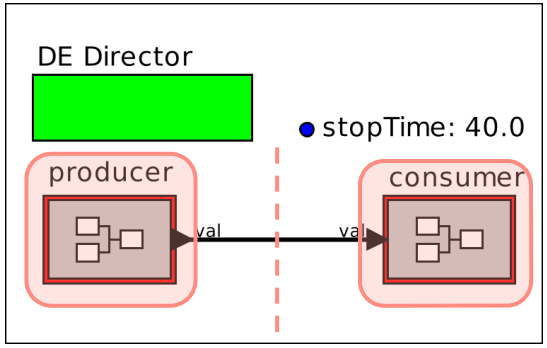
Time Management

- Time advancing in Ptolemy > uses the advancing time services of HLA
- Interface between DE director and RTI: decorator **HlaManager**

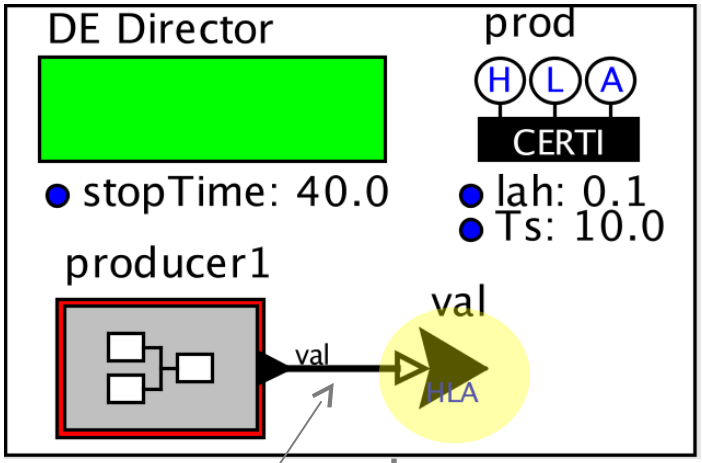
Object Management -> actors for translating:

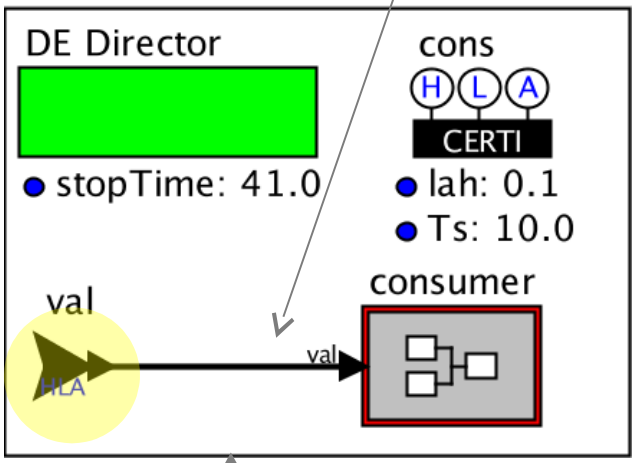
- A ptII-event from an out-port to a UAV service: new actor **HlaPublisher**
- RAV service from RTI to a ptII-event in an in-port: new actor **HlaSubscriber**

PtII-HLA framework



$e(t''=f_{rav}(g_{uav}(Lt'))\ n, val)$





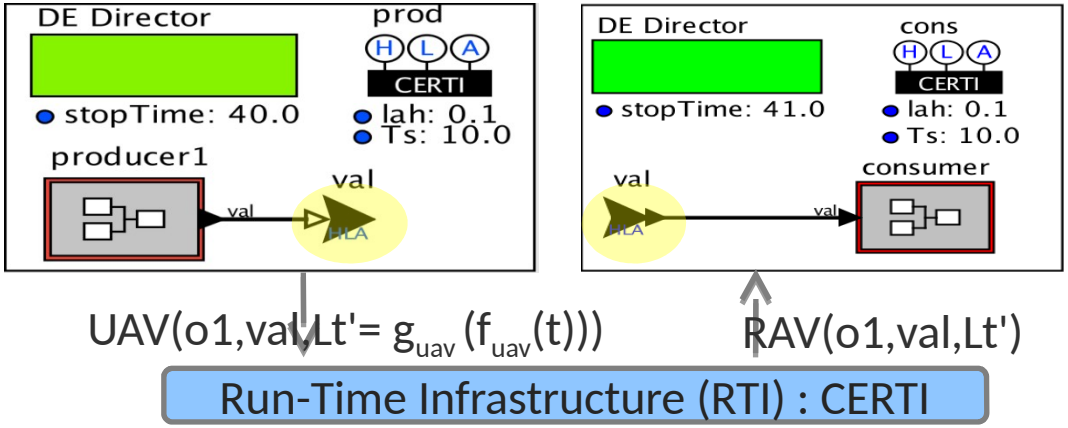
$e(t, n, val)$

$UAV(o1,val,Lt'= g_{uav}(f_{uav}(t))$

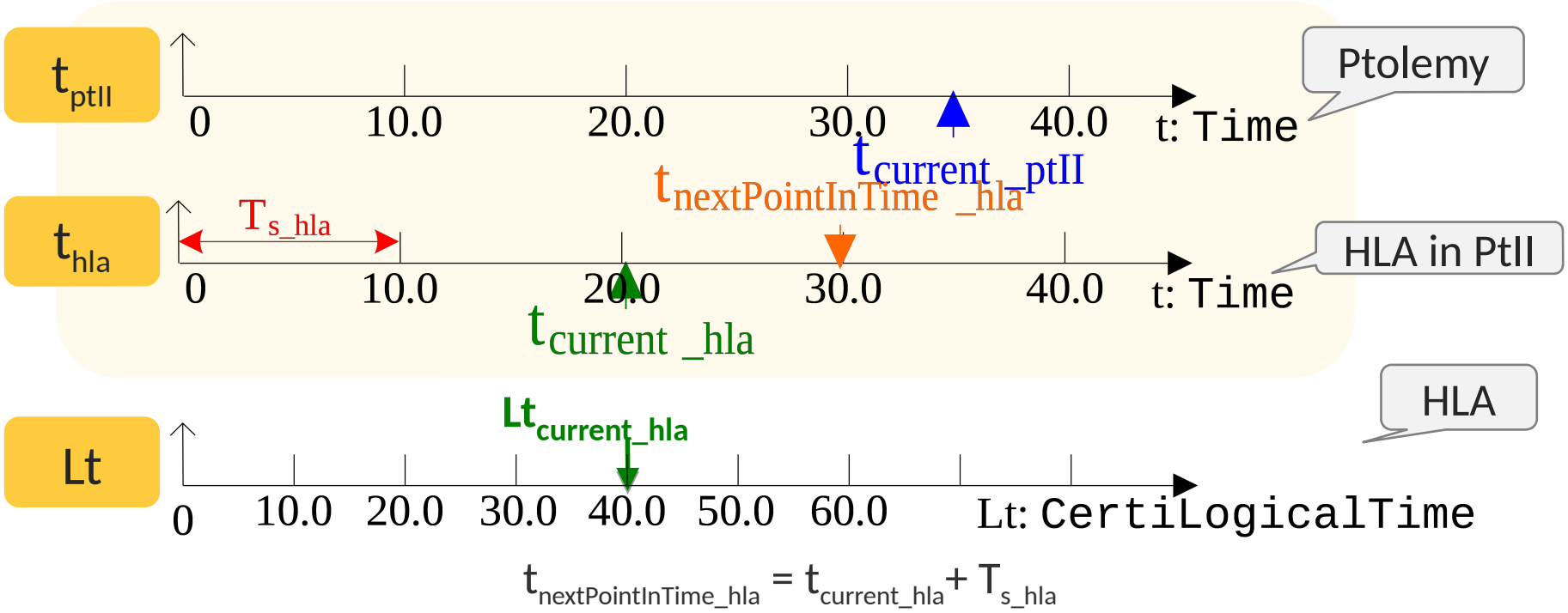
$RAV(o1,val,Lt')$

Run-Time Infrastructure (RTI) : CERTI

PtII-HLA framework



Time lines



$$t_{nextPointInTime\_hla} = t_{current\_hla} + T_{s\_hla}$$

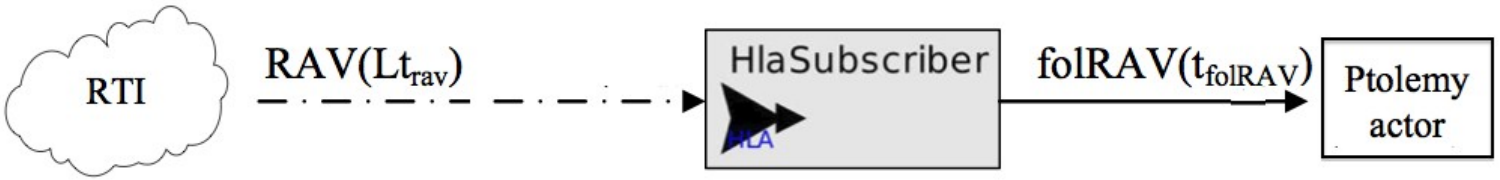
PtII-HLA framework

Sending

preUAV(t_{preUAV}: `Time`) $\xrightarrow{f_{UAV}}$ pUAV(t_{uav}: `Time`) $\xrightarrow{g_{UAV}}$ UAV(Lt_{uav}: `CertiLogicalTime`)

ptolemy-event — *uav-event* — TSO messages

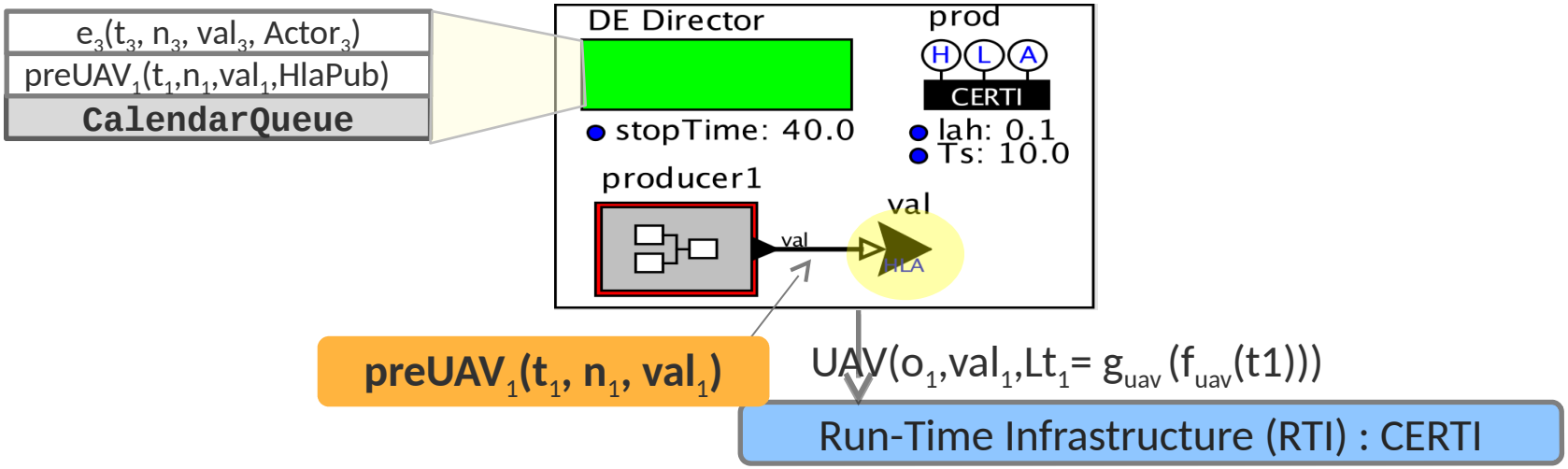
Receiving



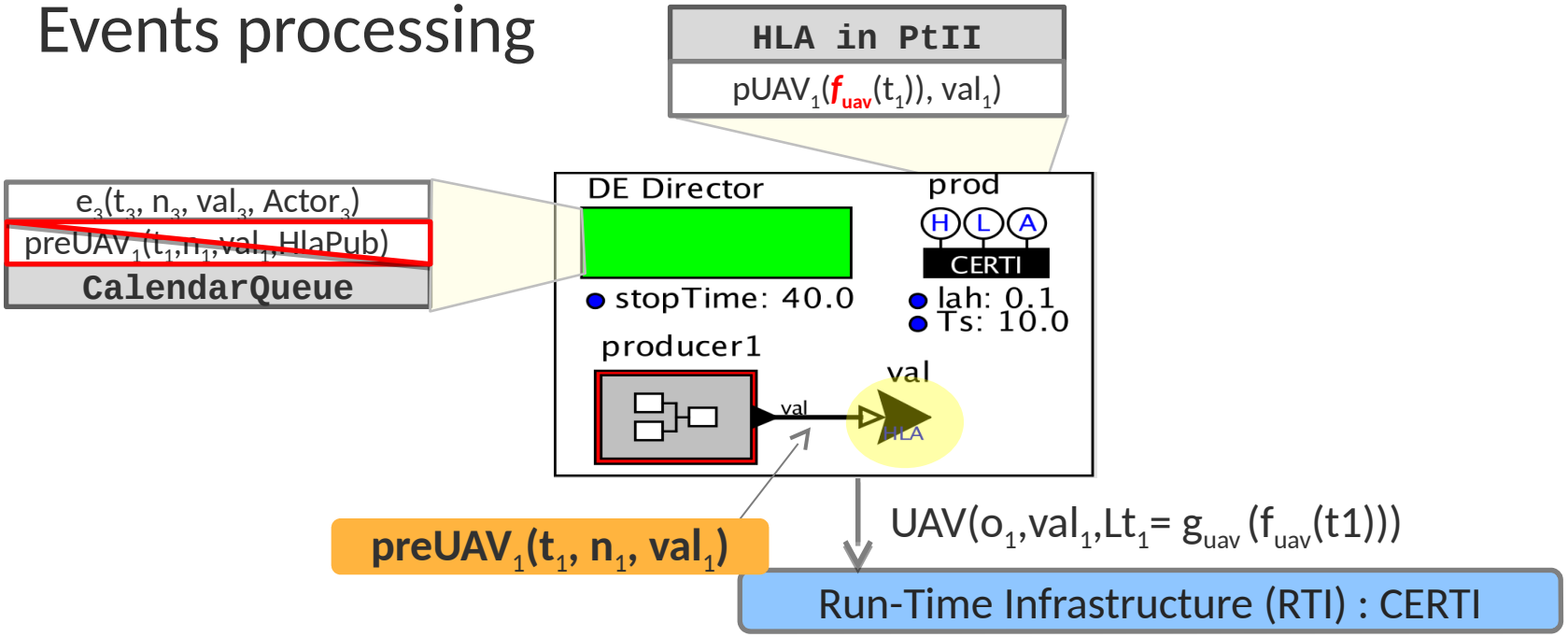
RAV(Lt_{rav}: `CertiLogicalTime`) $\xrightarrow{g_{RAV}}$ pRAV(t_{rav}: `Time`) $\xrightarrow{f_{RAV}}$ folRAV(t_{folRAV}: `Time`)

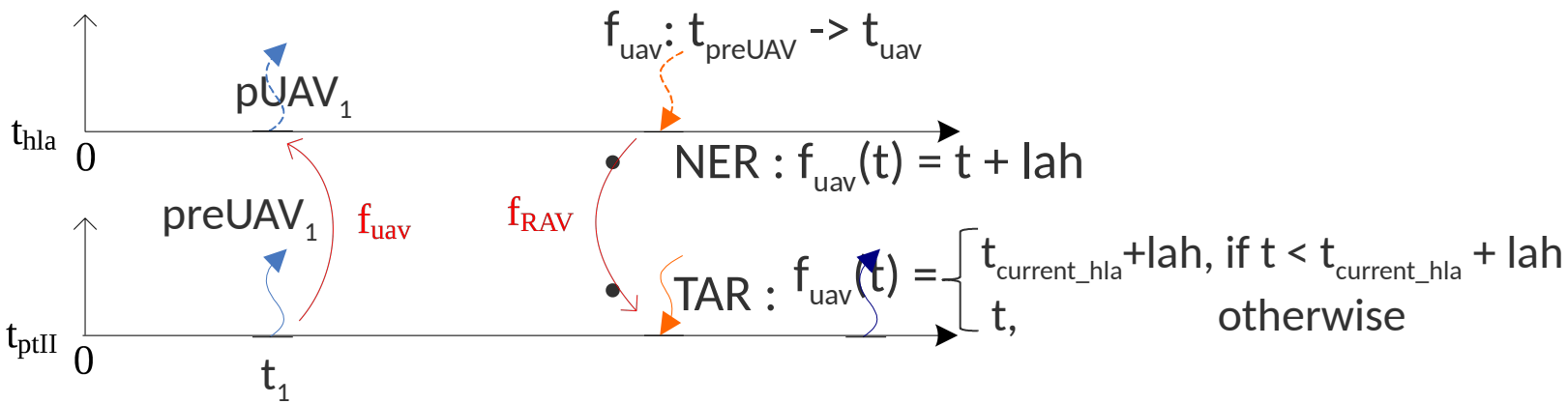
TSO messages — *rav-event* — *ptolemy-event*

Events processing

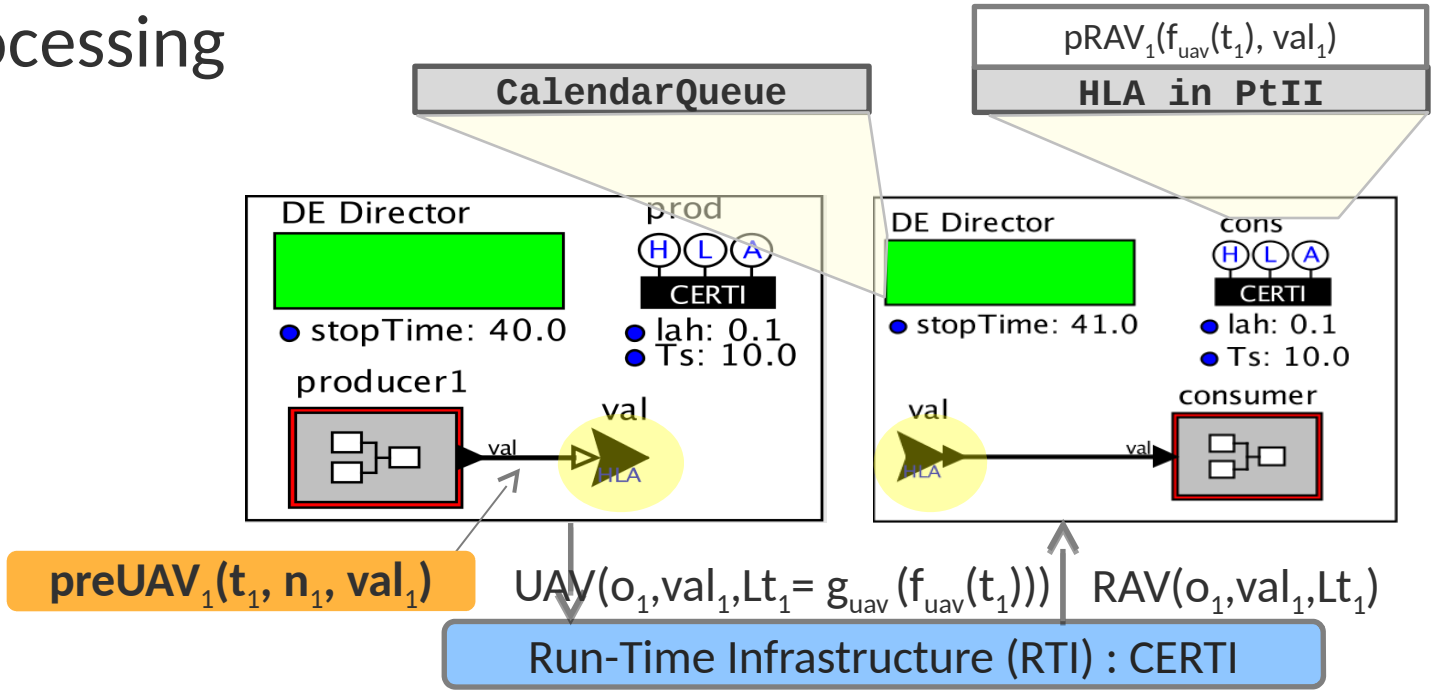


Events processing

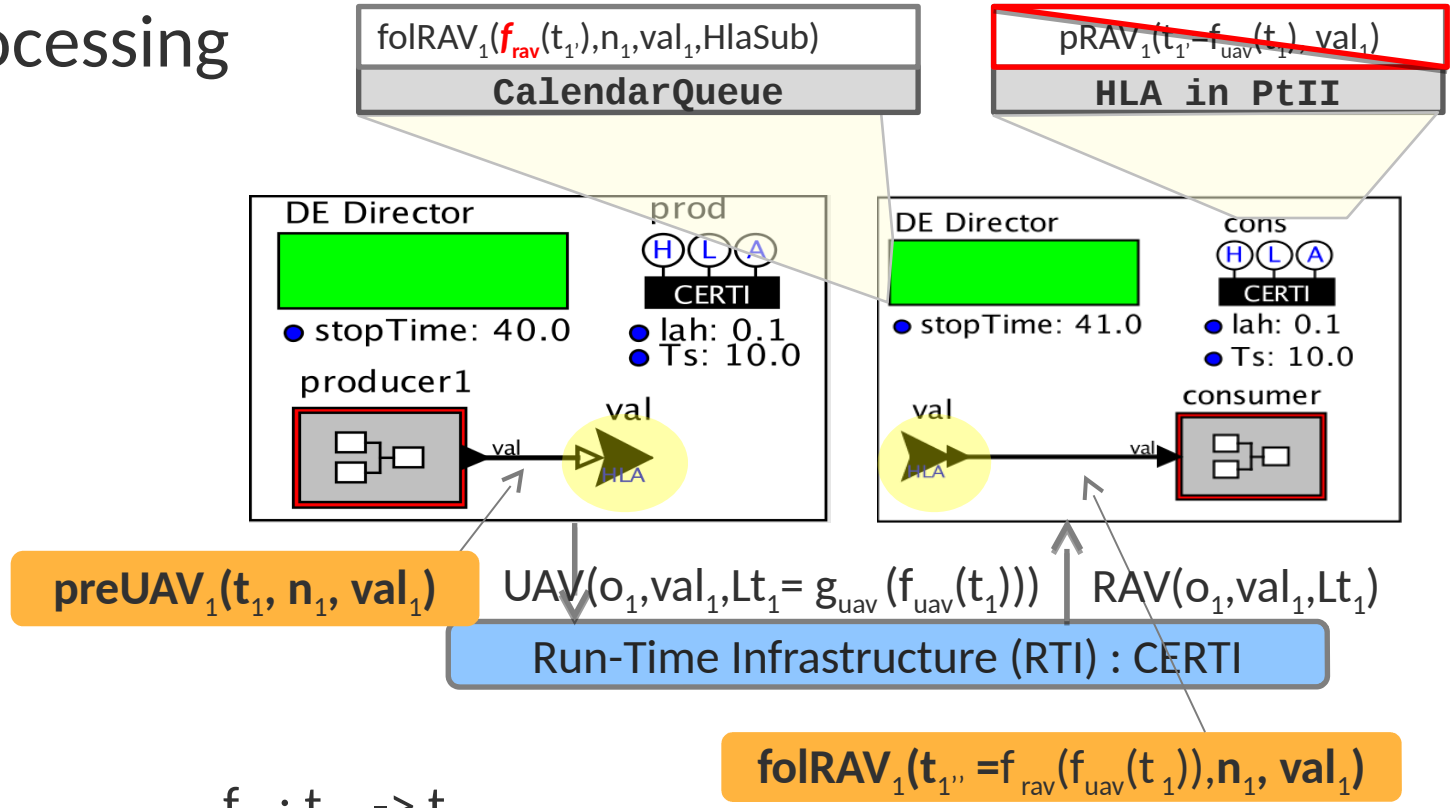




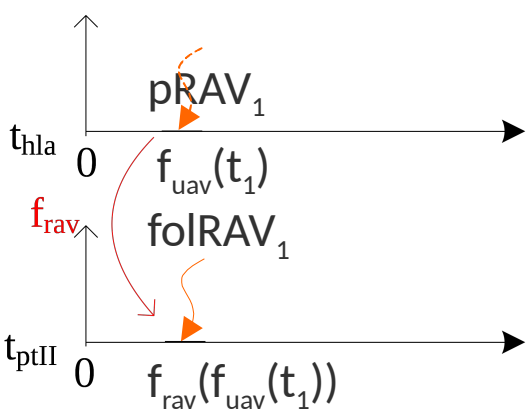
Events processing



Events processing



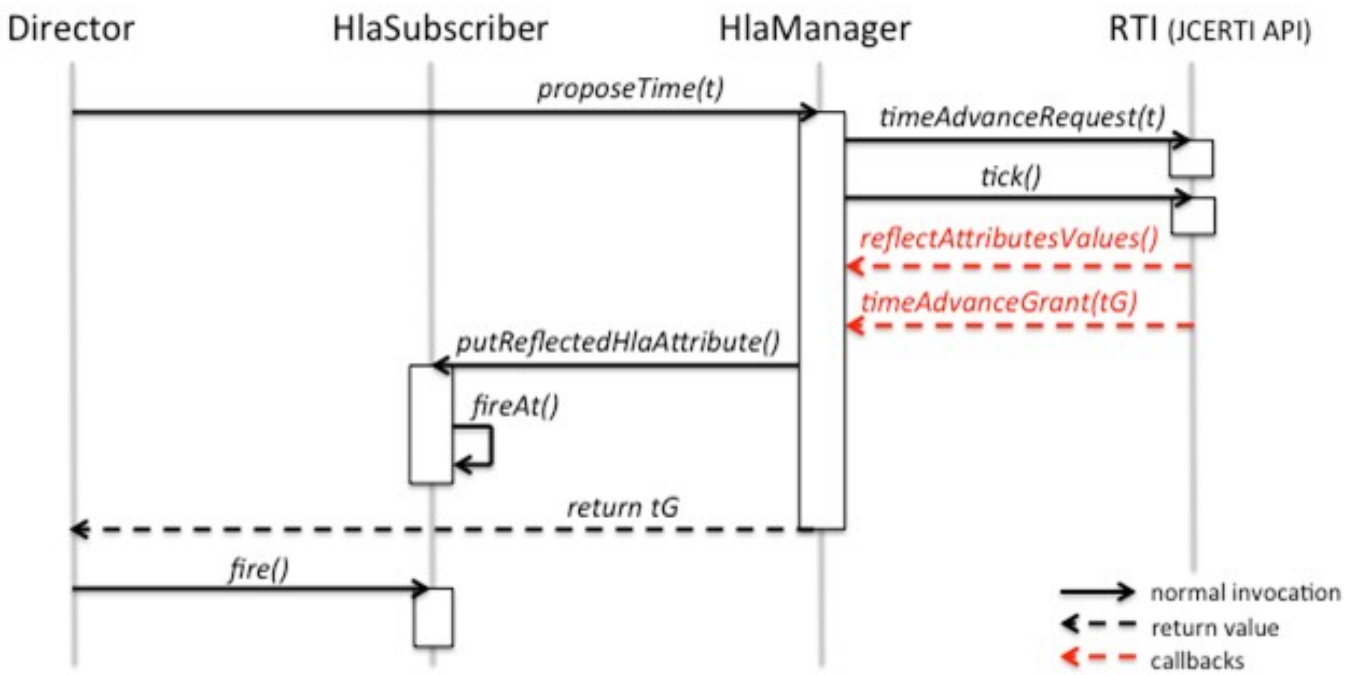
$f_{rav}: t_{rav} \rightarrow t_{folRAV}$



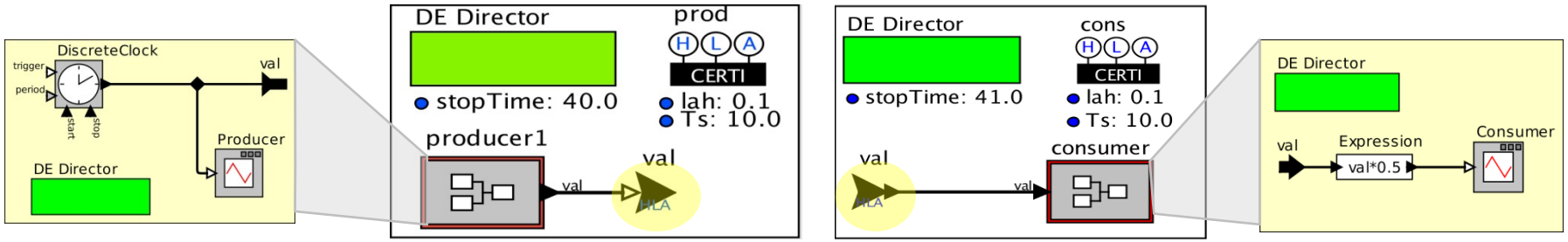
- NER : $f_{rav}(t) = t$
- TAR : $f_{rav}(t) = \begin{cases} t_{nextPinTime\_hla}, & \text{if } t \in]t_{current\_hla}\ t_{nextPInTime\_hla}] \\ t & \text{if } t > t_{nextPointInTime\_hla} \end{cases}$

Data Management with timestamps

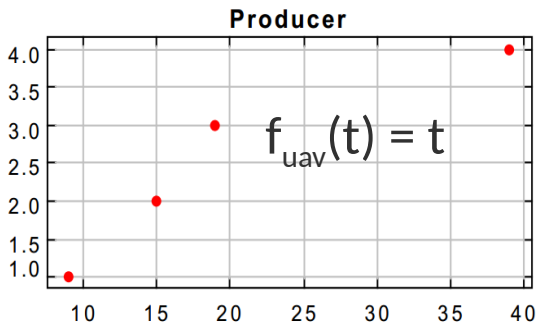
- Time "synchronization"
  - A local PtII event is safely computed if no (external) HLA events can arr ive with a smaller timestamp
  - A PtII federate declares its time advancement proposal to the federati on through **`proposeTime`**()



Distributed Producer/Consumer

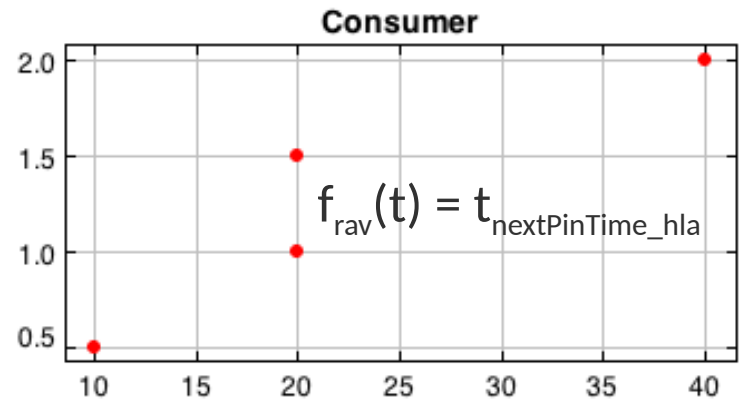


$e_1(9, 1, 1.0)$, $e_2(15, 1, 2.0)$ $e_3(19, 1, 3.0)$, $e_4(39, 1, 4.0)$



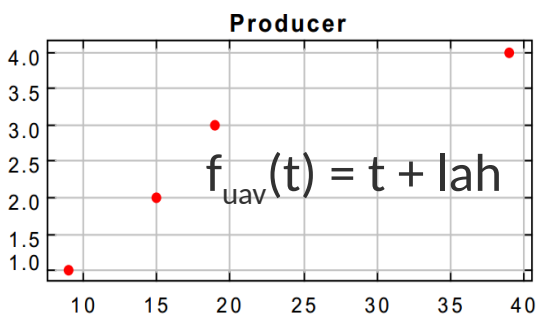
$f_{uav}(t) = t$

TAR

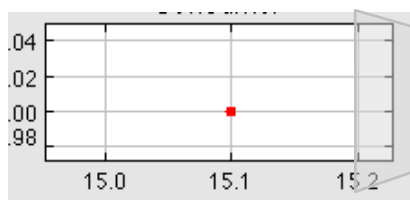


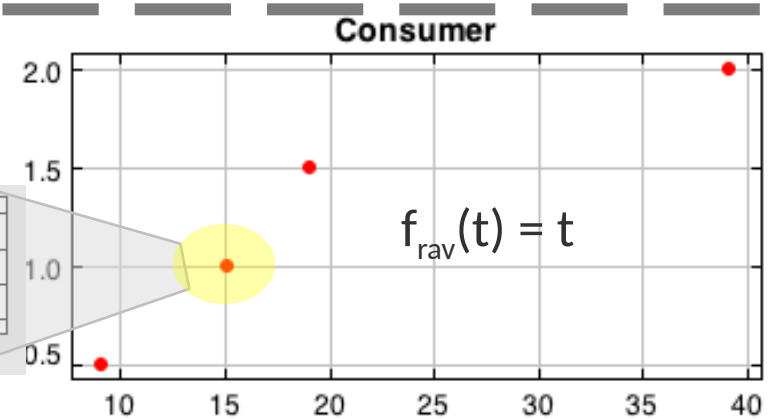
$f_{rav}(t) = t_{nextPinTime\_hla}$

NER



$f_{uav}(t) = t + lah$





$f_{rav}(t) = t$

Conclusion

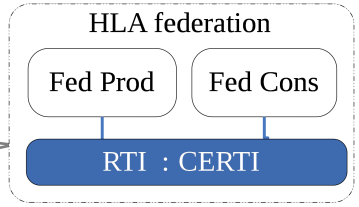
- An easy way to produce a HLA federate from a Ptolemy model
- A way to distribute the execution of a Ptolemy model + hardware-in-the-loop
- Time management extend for time-stepped federates (TAR mechanism)
- On-going work, started with G. Lasnier, J. Cardoso, P. Siron, C. Pagetti and P.Derler. Distributed simulation of heterogeneous and real-time systems. In DS-RT13.

Perspectives

- Compare TAR and NER performance and define related applications
- Implement TARA and NERA mechanisms: TAR and NER with lookahead = 0.
- Make a more complex application: multi-periodic flight controller ROSACE (Research Open-Source Avionics and Control Engineering).

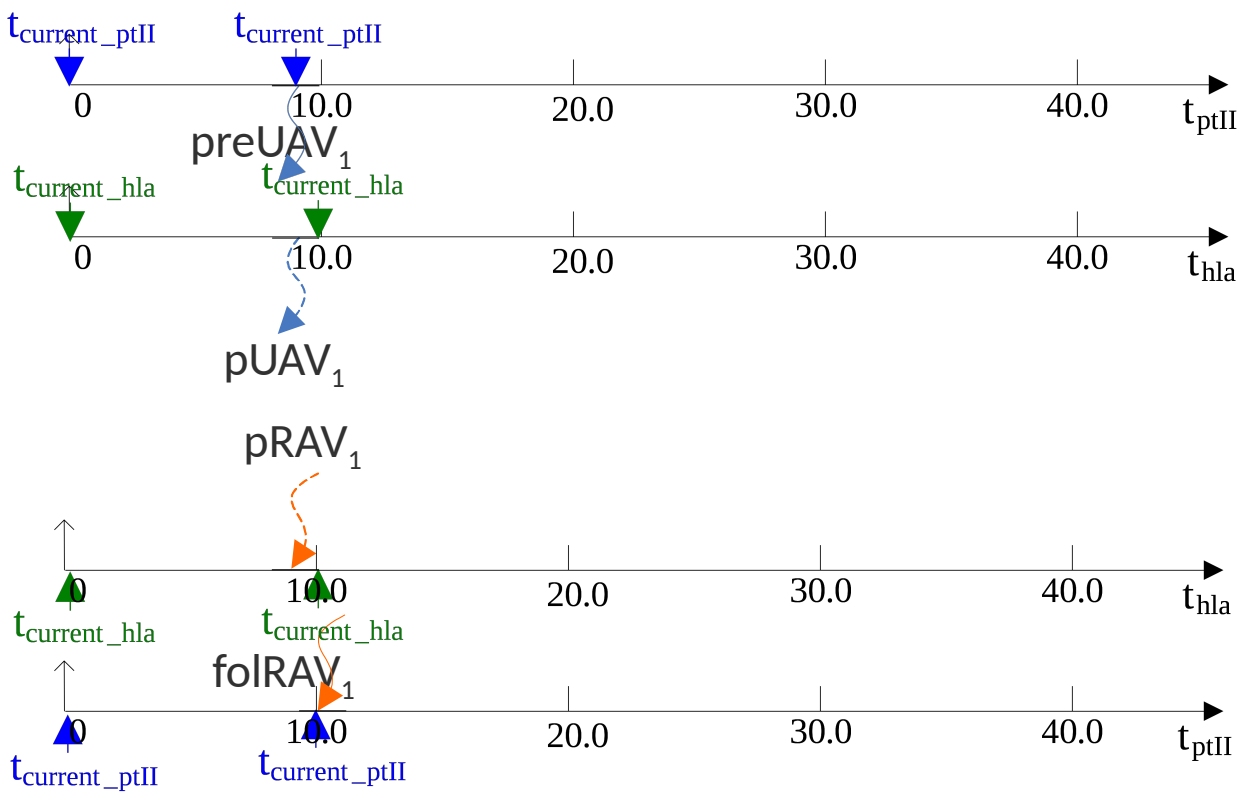
Thank you for your attention.

Distributed Simulation



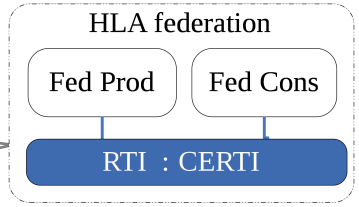
Ts = 10, lah= 0.1, TAR

Events List	$e_1(9, 1, 1.0)$, $e_2(15, 1, 2.0)$, $e_3(19, 1, 3.0)$, $e_4(39, 1, 4.0)$



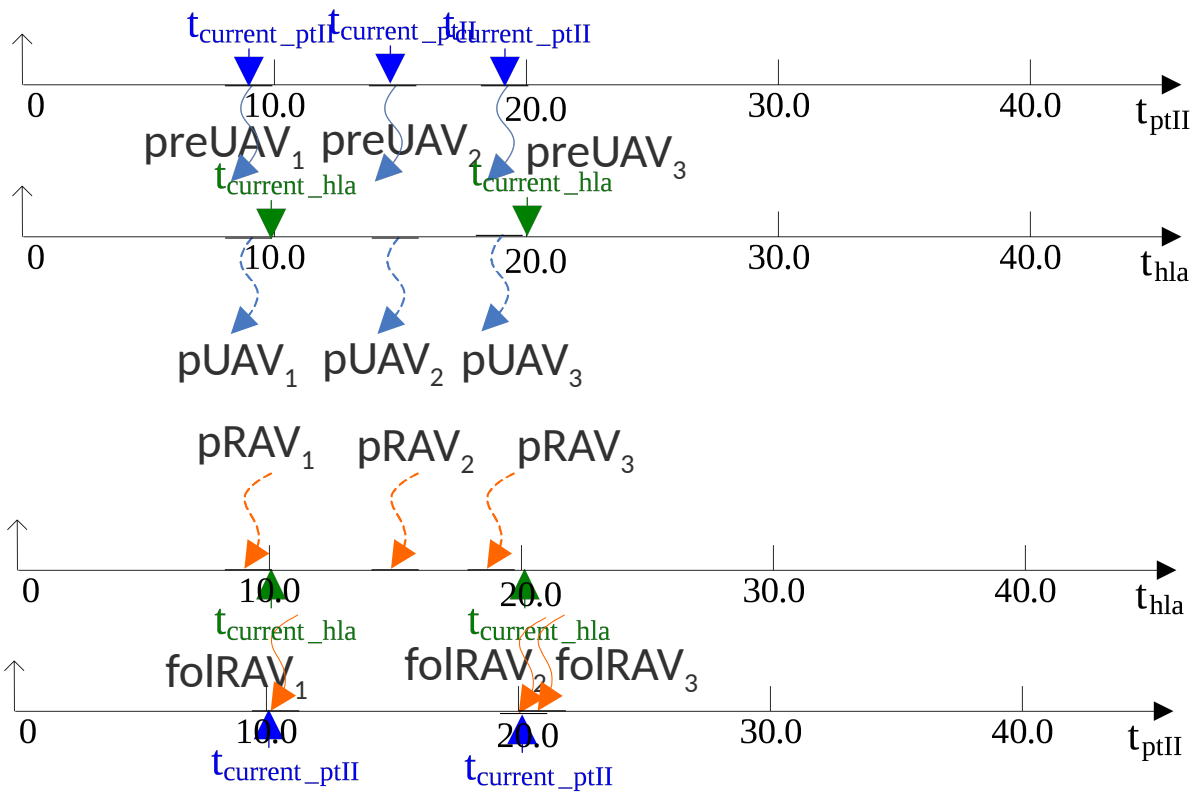
Events List	$e_1(10, 1, 0.5)$

Distributed Simulation



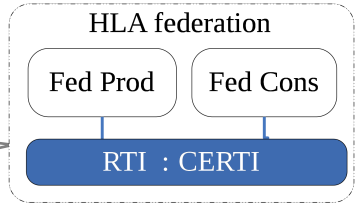
Ts = 10, lah= 0.1, TAR

Events List	$e_1(9, 1, 1.0)$, $e_2(15, 1, 2.0)$, $e_3(19, 1, 3.0)$, $e_4(39, 1, 4.0)$



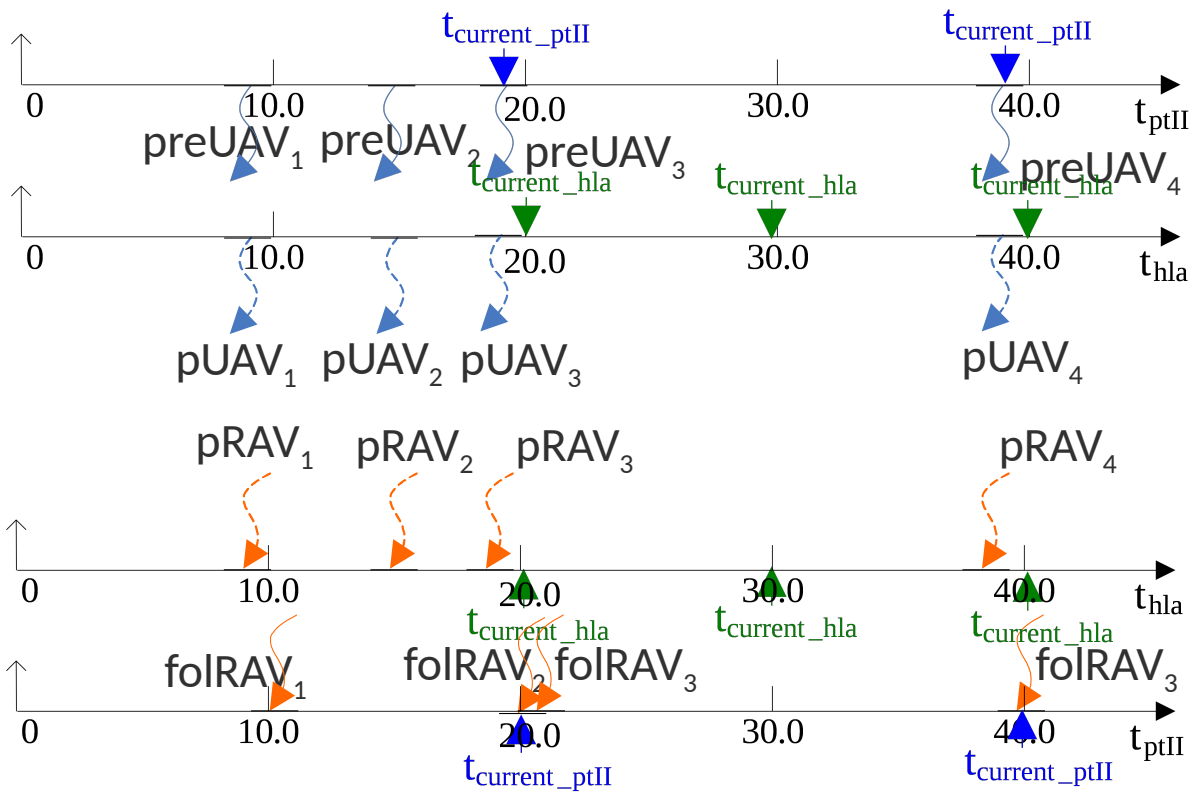
Events List	$e_1(10, 1, 0.5)$, $e_2(20, 1, 1.0)$, $e_3(20, 2, 1.5)$

Distributed Simulation



Ts = 10, lah= 0.1, TAR

Events List	$e_1(9, 1, 1.0)$, $e_2(15, 1, 2.0)$, $e_3(19, 1, 3.0)$, $e_4(39, 1, 4.0)$



Events List	**$e_1(10, 1, 0.5)$, $e_2(20, 1, 1.0)$, $e_3(20, 2, 1.5)$**, $e_4(40, 1, 2.0)$

Algorithms of time management

Algorithms		HLA	HLA/CERTI	PTII/HLA
Optimistic	Jefferson 85	X	Non	Non
Conservative 1st generation	Chandy, Misra & Bryan 79 (lookahead > 0)	X	X	X
Conservative 2nd generation	State global computation, Mattern	X	Non	Other RTI?
	Null Prime Message protocol	X	X	X

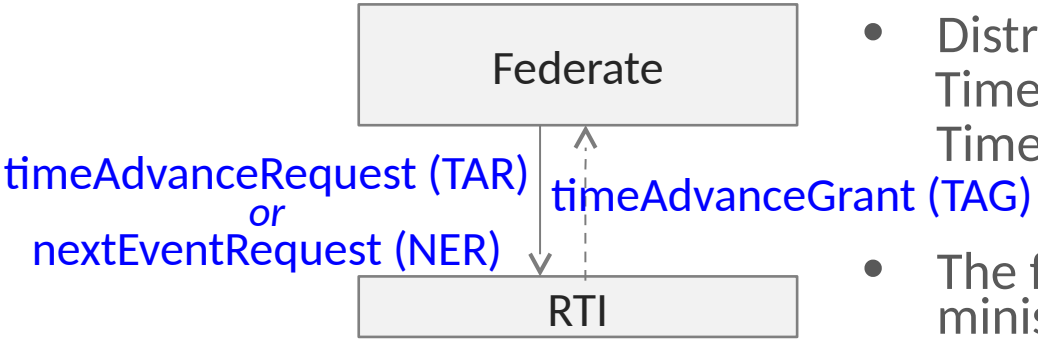
Superdense time?

- In accordance with the HLA standard, time representation could be PtII superdense time
  - A requirement ?
  - Requires very high C++ skills
- Sometimes, events with the same HLA timestamp are transformed in events with different PtII timestamps
- Actor's ranking
  - Lost in the distribution
  - Problems for an automatic distribution of PtII (work in progress)

Reference

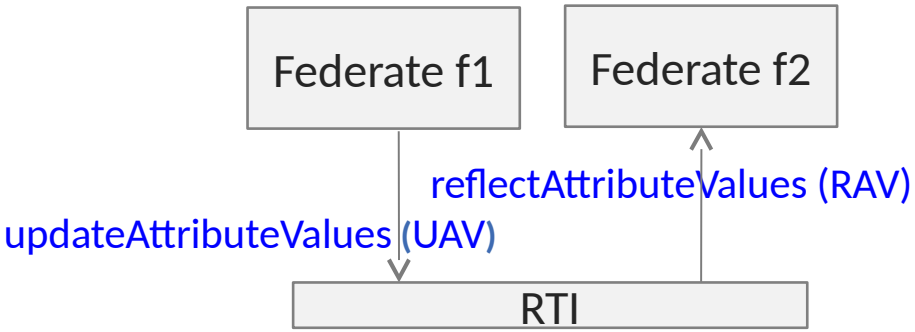
1. R. M. Fujimoto. Time Management in the high level architecture
2. G. Lasnier, J. Cardoso, P. Siron, C. Pagetti and P.Derler. Distributed simulation of heterogeneous and real-time systems. In DS-RT13
3. U. of Berkeley. The ptolemy project. http://ptolemy.eecs.berkeley.edu/ptolemyII/index.htm
4. ONERA. Certi. http://www.nongnu.org/certi/certi_doc/Install/html/intro.html
5. D. Come. Improving hla-ptolemy co-simulation framework.
6. Y. LI. A Distributed Simulation Environment for Cyber-physical systems.

HLA Time Management



- Distributed Events Ordering <->
  Time Management <->
  Time Advancing mechanisms
- The federation will be conservative, deter ministic and repeatable.
- The federates can be:
  - Time-Stepped (TAR)
  - Event-Driven (NER)

HLA Object Management



- UAV(object,attribute,timestamp)
- RAV(object,attribute,timestamp)

Data & Time Management

- Time "synchronization"
  - A local PtII event is safely computed if no (external) HLA events can arr ive with a smaller timestamp
  - A PtII federate declares its time advancement proposal to the federati on through **`proposeTime`**()