2024 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

MUNICH, GERMANY
OCTOBER 15-16, 2024

Simulation Phases

Mark Burton
Qualcomm France SARL.

Mark Glasser

Karsten Einwich
Coseda Technologies.

Ramzi Karoui
ZettaScale.

accellera SYSTEMS INITIATIVE

Phases to Stun !

This all started as a simple question between Mark and Mark – we should know, right?

What are the phases of a simulation?

(small print: you can call them states, or anything you like, I'll call them phases for now)

Easy as A B C

Children's wooden blocks spelling ABCD

- The real question is really, can we find some similarities between different simulators.
- How are we going to handle connecting different simulators if we don’t know their different phases…

The question is about Federation


1970’s teacher drawing a flowchart on a chalk board

Current systems of “Phases”

- Static Phases rule – Everywhere
  - Most are “organically” grown
    - (we’ll just add these extra phases to handle that….)
- UVM has “user defined” phases
  - Still need to be defined a-priori – before simulations start.
  - That’s interesting – but seems they are not heavily used
- We all ‘live’ within the phases provided by the simulation environment, and it’s always ‘awkward’
  - Hands up who has run-out-of-phases
    - “this needs to happen after one thing before another, but we don’t have a phase between them….”
- If you think about it – is there ever a simple solution?

So federating simulation phases – a mess?


Uncle Sam paying the bill in an Italian restaurant

Birth, Death and Tax

- If we go for an list of common states, we end up with start, run, stop….
- Can we do any better?

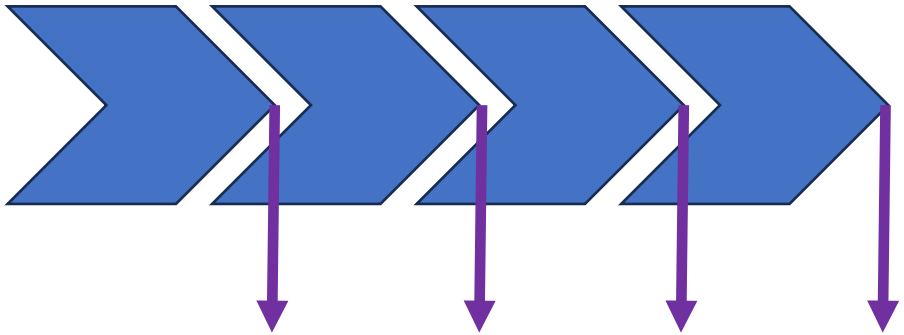
Start at the beginning:

- The role of the phases is to enforce ordering of operations and to ensure that dependencies are satisfied.

Dependencies…. Making sure one thing happens before another….

MMmmm – makes you think, isn't that what simulators normally do?

MoC : Tags

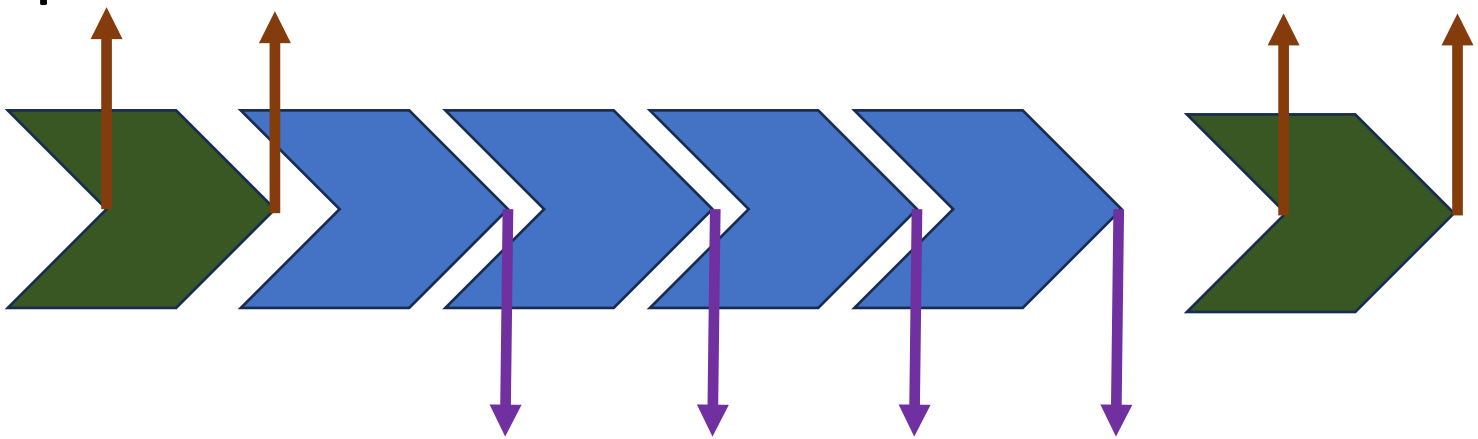


The ‘famous’ Models of Computation (Lee and Sangiovanni-Vincentelli) proposes a way to categorise different simulations based on how they treat “Tags” – the (simulation) events in the system.

This marked a move between ‘statically’ building simulators, and using dynamic events to comminicate events between different model components.

Could we do the same for “phases”?

Tags for phases



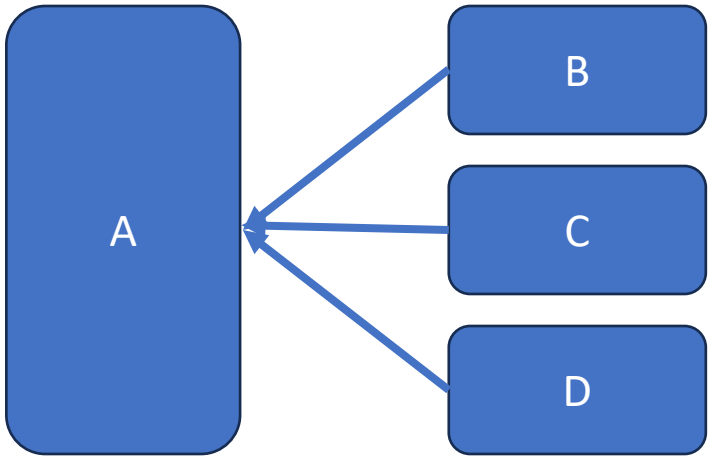
So – the jaw dropping insightfull proposition of this paper…..

Simulation phases are a little like Tags too !

NB, they are not the same, “Tags” are about the models themselves, and what is being modelled. Phases are at a ”meta” level – they are about how to co-ordinate the simulation.

So what?

- Lets say, A needs to be constructed after B C and D



- So, B C and D need to tell A when they are constructed.
- A needs to be ‘connected’ B C and D.

The proposition is to remove global phases, and use dynamic – local phases.

Dynamic phases : good for Federation?

- The ordering of the phases is not fixed.
- The ‘state machine’ is an emerging property, not a pre-defined monolith.
  - Therefore it’s complexity is not an issue. Federated components are added by connecting phases.
- Neither the individual component writer, nor the federator is ‘stuck’ within a pre-defined set of phases, nor do they need to handle the complexity of the full system.

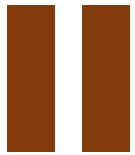
- The federator does not need to understand the equivalence of simulators (and how their phases may have been used/misused/abused).
- Rather models will require and emit specific phase events, which need to be ‘connected’.
- The conjecture here is that these specific phase events will be more robustly defined (as they relate to specific models). And hence making federation easier.

Handling dynamic phases: We need some servcies?

- Communication Service to carry ‘phases’ and share their semantic underlies everything

- Models need to emit ‘phases’ when important things happen (construction, binding, …)

- Models need to be able to find each other – and listen for those ‘control’ events.



- **Finally – most importantly – Models need to be able to start and pause or accellerate the simulation**

  “Don’t continue, until I have finished doing this task”

An example?

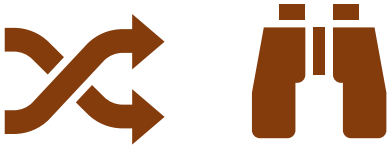
- Take a SystemC that uses “end-of-elaboration” :
  - First, ideally, it should be finding and connecting to the actual models it needs to have done their “elaboration” after, rather than this “global”.
  - Second, while this model is doing it’s “end-of-elaboration” certainly it does not want systemc time to advance, so simulation must be “paused”.

  - When it’s finished, it can remove it’s request to pause simulation,
  - Once all models agree to run, then the simulation starts…

Federation …
Is the name of the game!

- How simulators SystemC deals with its phases isn't our problem….
- To federate simulators together, we're asserting that you need :

Summary:
We should move from Static to Dynamic Phases



- Let the system designers decide what phases they need and Wire up ‘phases’ like any other events and communication paths.

- Use ‘phases’ to control how the simulation advances

The interfaces are open to discussion, different simulators may have different interfaces, and different notions of how they advance. The question posed here is whether theoretically such a simple framework would be sufficient to federate simulators in terms of their control, not in terms of how data flows between them. Please ask the bill payers permission before calling. For other terms and conditions, read the paper…

Results…

NONE

Question… Suggestions… Ideas…

MANY