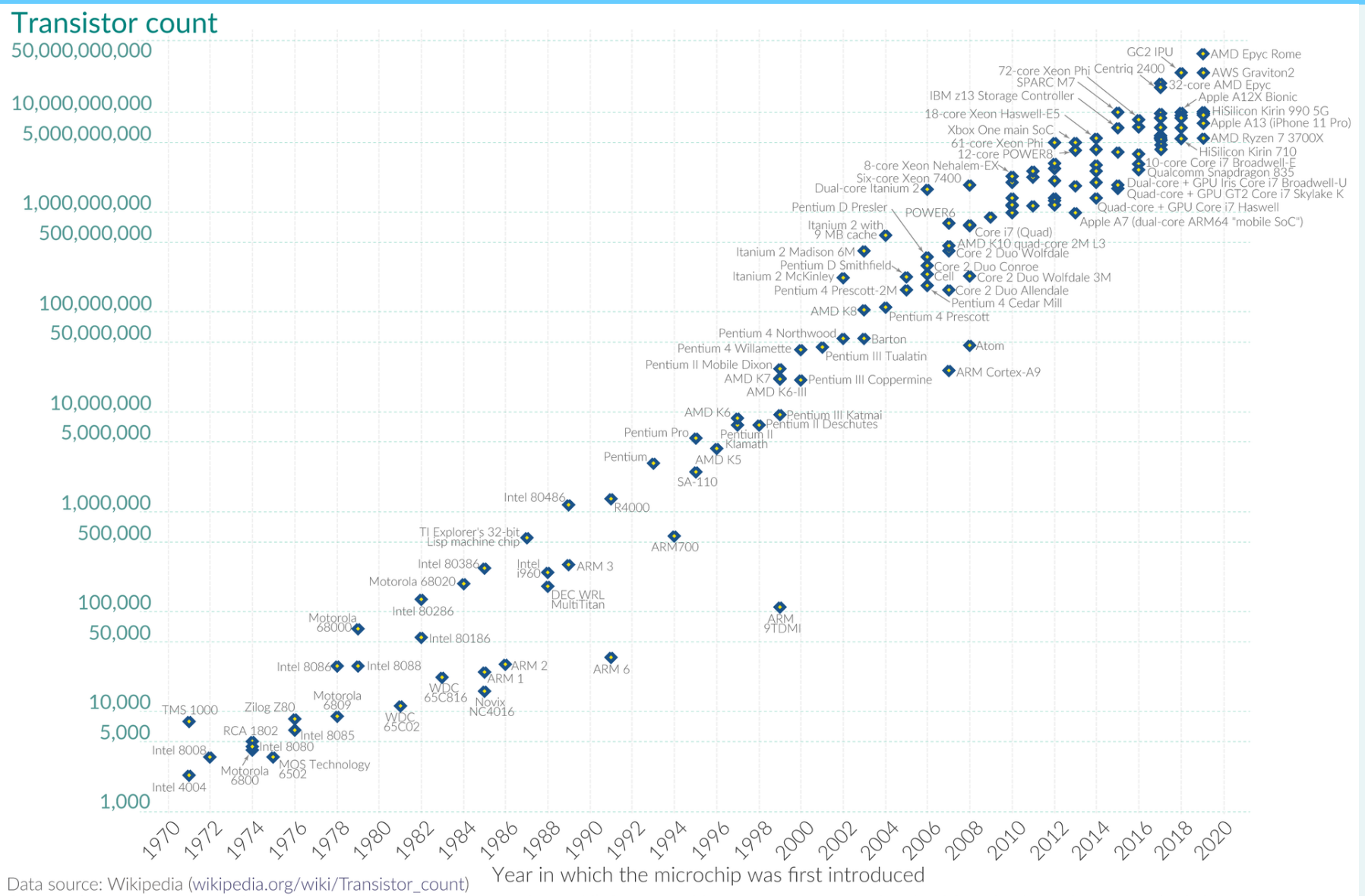
Threading: COM, COBIA, Status, interoperability and challenges

Bill Barrett – EPA
Michael Hlavinka- Bryan Research & Engineering
Jasper van Baten - AmsterCHEM
Michel Pons – CO-LaN
Mark Stijnman – Shell Global Solutions

21 September 2022

Moore's Law



Data source: Wikipedia (wikipedia.org/wiki/Transistor_count)
OurWorldinData.org – Research and data to make progress against the world's largest problems.

CO LaN

But:

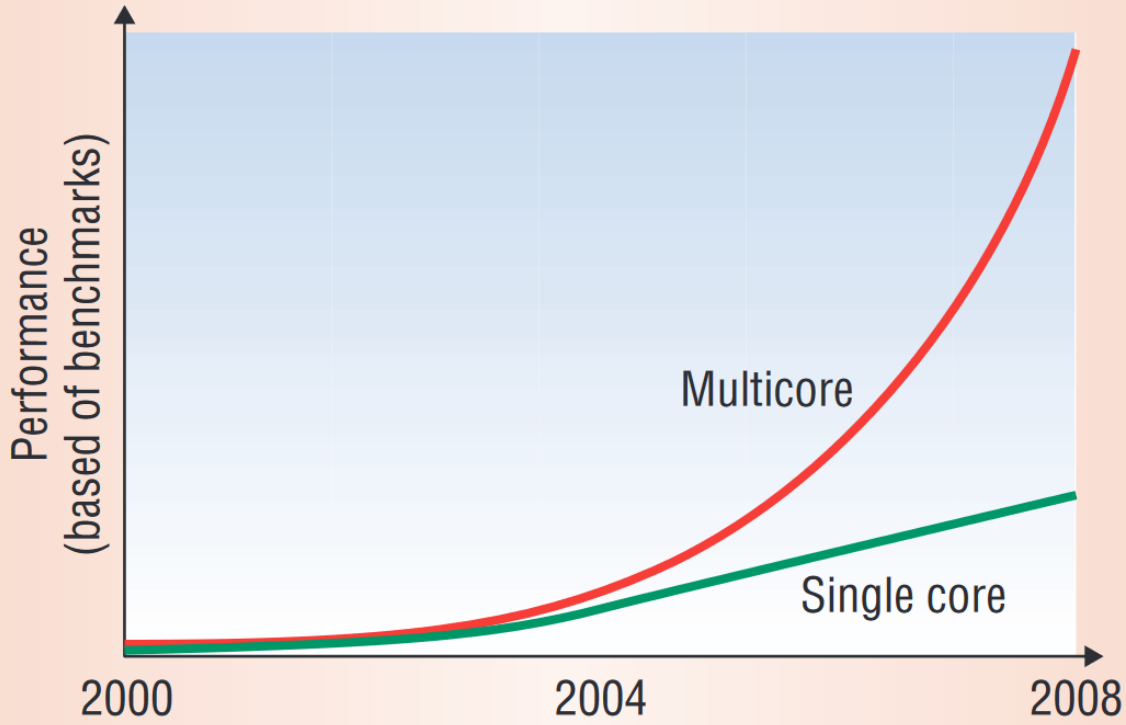


Figure 1. Multicore chips perform better—based on Intel tests using the SPECint2000 and SPECfp2000 benchmarks—than single-core processors. And, Intel says, multicore chips' relative advantage will increase during the next few years.

Source: Industry Trends: Chip Makers Turn to Multicore Processors, 10.1109/MC.2005.160

But:

- Instruction level parallelism progress stalled ~ 1990
- Parallel computing uses less energy than faster computing
- Transistors cannot shrink anymore, latency cannot reduce anymore
- Cores can run threads partially in parallel (Hyperthreading)
- Computers are equipped with multiple cores (Previous slide)

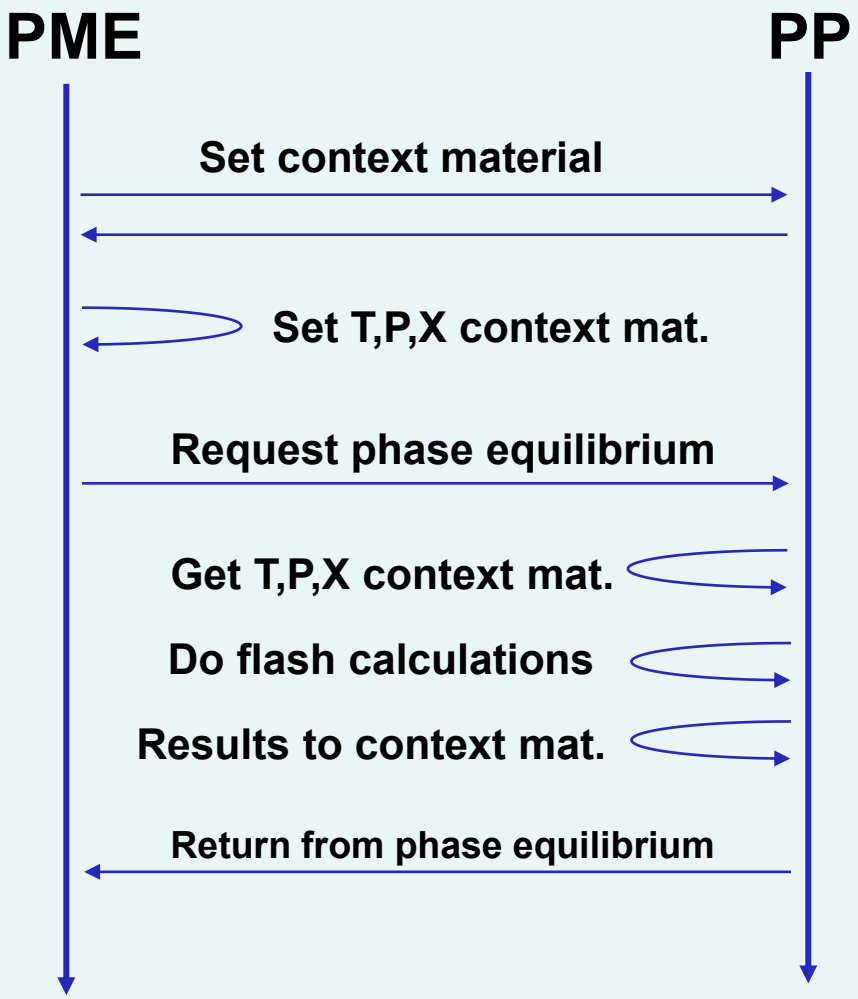
One process can utilize multiple cores, from multiple concurrent threads.

To utilize the speed of modern day computers, we need to consider multi threading!

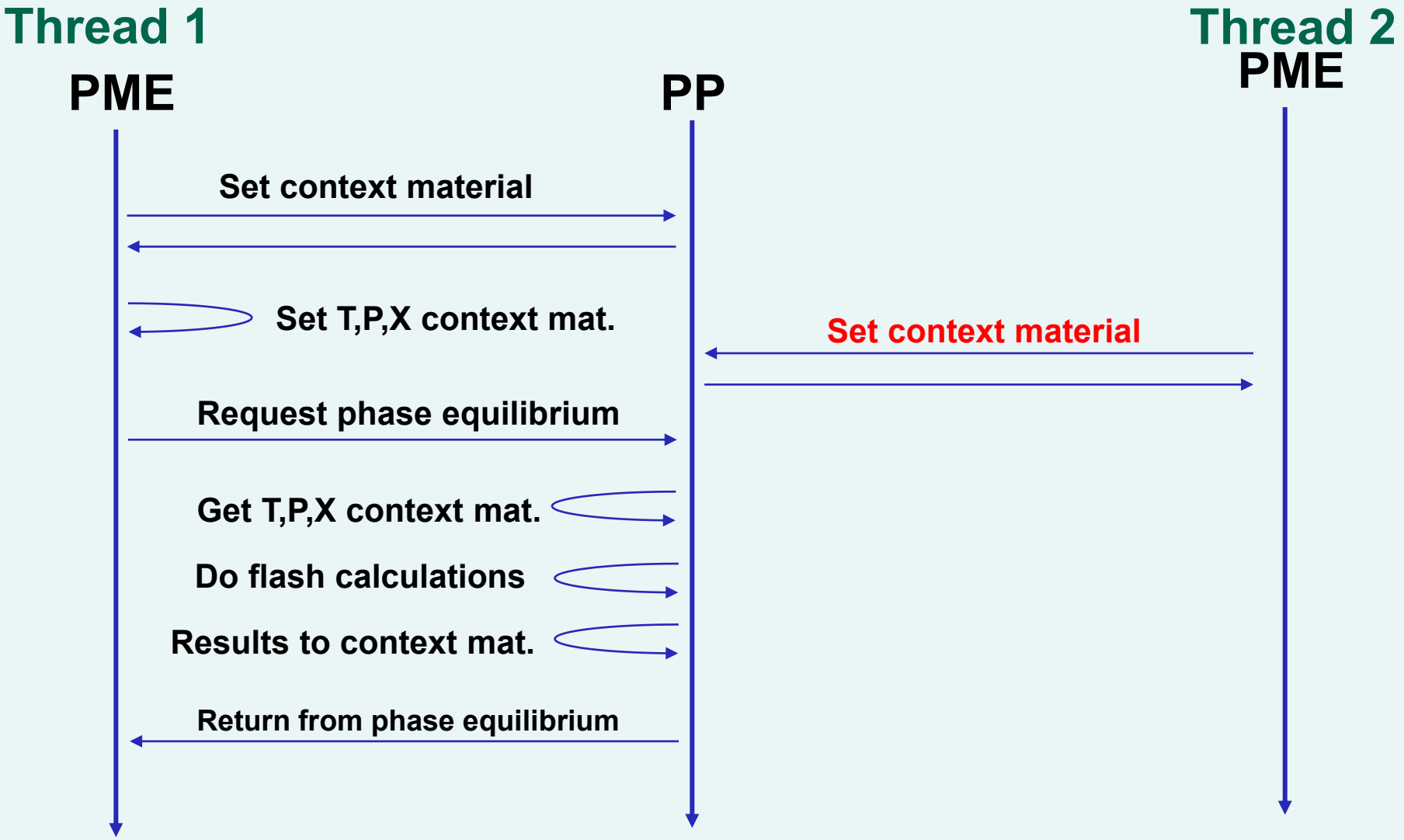
Outline

- CAPE-OPEN interoperability is not stateless!
- COM threading models
- Current status of multi-threading in CAPE-OPEN
  - Issues hampering performance in major PMEs
- COBIA threading models
- Interoperability between COM and COBIA
- Recommendations
- Outlook

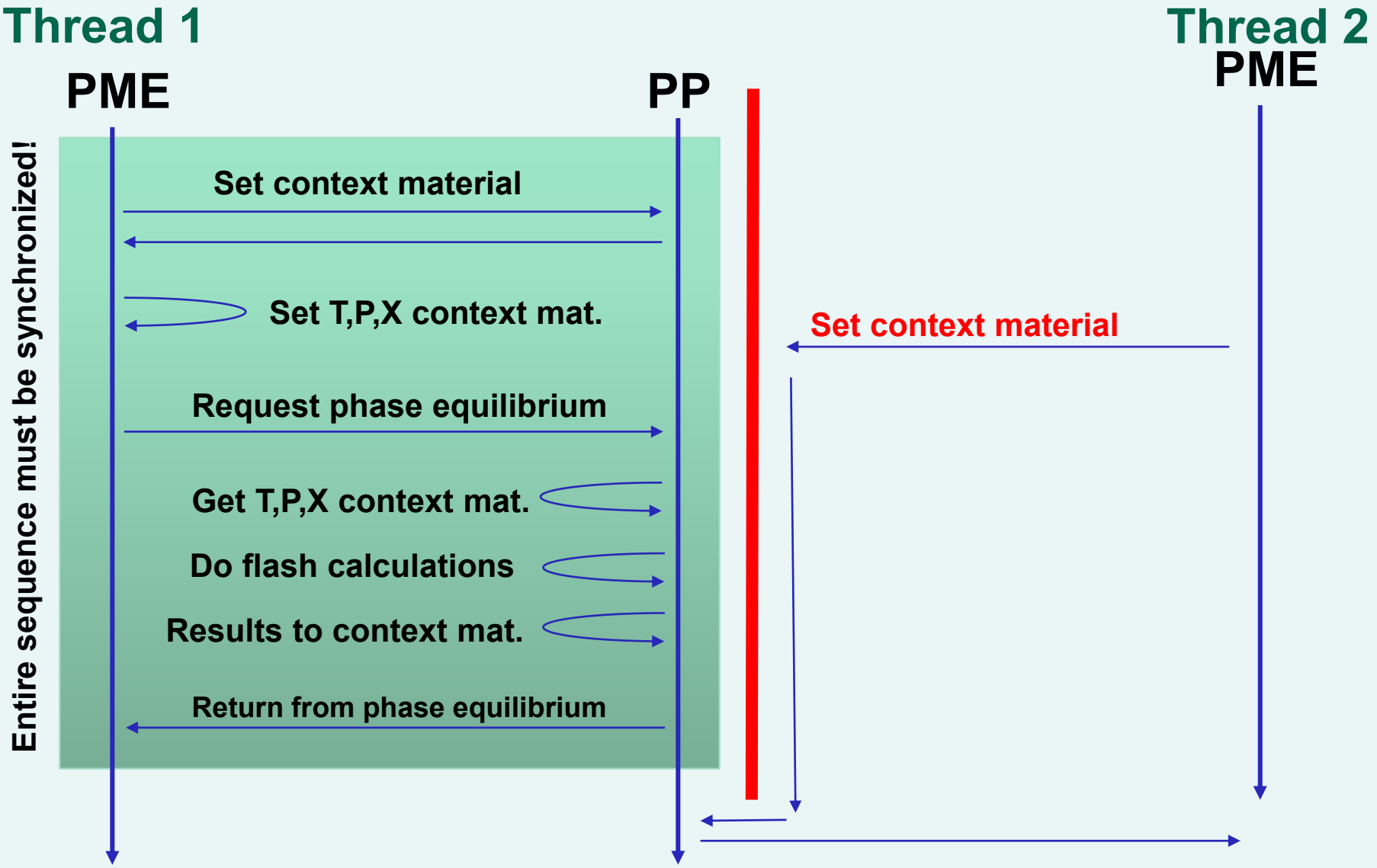
CAPE-OPEN interoperability is not stateless!



CAPE-OPEN interoperability is not stateless!



CAPE-OPEN interoperability is not stateless!



CAPE-OPEN interoperability is not stateless!

- Property package: state carried by context Material Object
- Unit operations: state carried by port connections
- All PMCs: simulation context
- All PMCs: state carried by configuration
  - ICapeUtilities::Edit
  - Public Parameters

Synchronization around multi-threaded calls seems inevitable.

COM threading models

- COM is initialized per thread
  - Thread is Single Threaded Apartment: STA
  - Or part of Multi Threaded Apartment: MTA
- In-process COM servers (PMCs) advertise threading ability:
  - *Apartment*: each instance called only from one thread
    - But multiple threads can exist with multiple instances
  - *Free*: each instance is thread safe, ok to call concurrently
  - *Both*: Apartment of Free – depending on the creating thread
  - *Single/Neutral*: special case, only one single thread

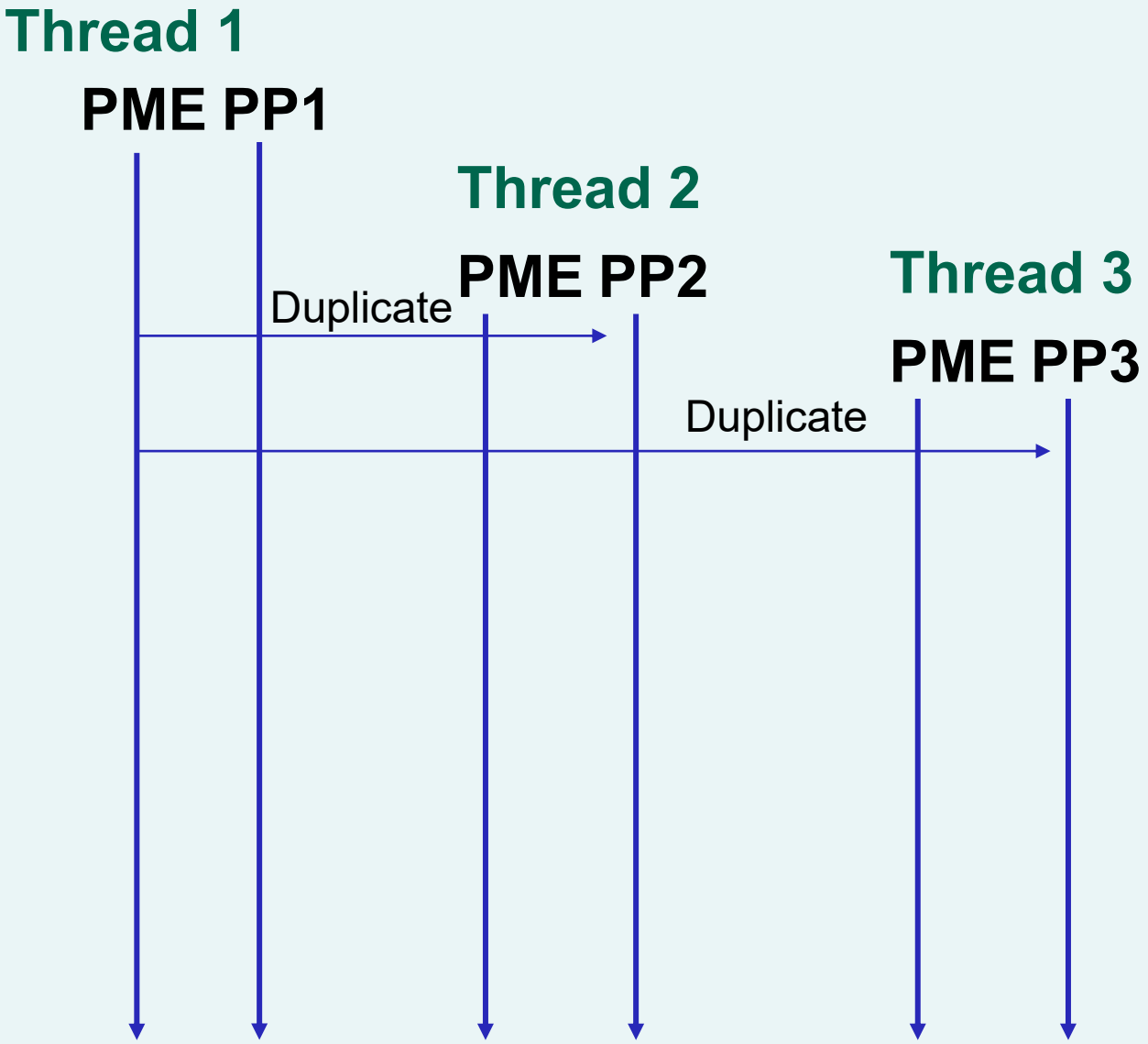
COM threading models

- Multiple STA apartments may exist
- There is only one MTA apartment
- Marshaling between threads is costly: involves synchronization between threads and placing a thread on hold.
- PMC created in ‘incompatible’ compartment is placed in a COM-created thread, and marshaled (costly)
- PMCs can be accessed from outside their apartment:
  - This will incur marshalling (costly)

Current status of multi-threading in CAPE-OPEN

- CO-LaN has never identified in documentation that PMCs have a state, and implied synchronization of call sequences
- Perhaps because of this, nearly all COM PMCs are implemented as Apartment threaded
- Some implications follow

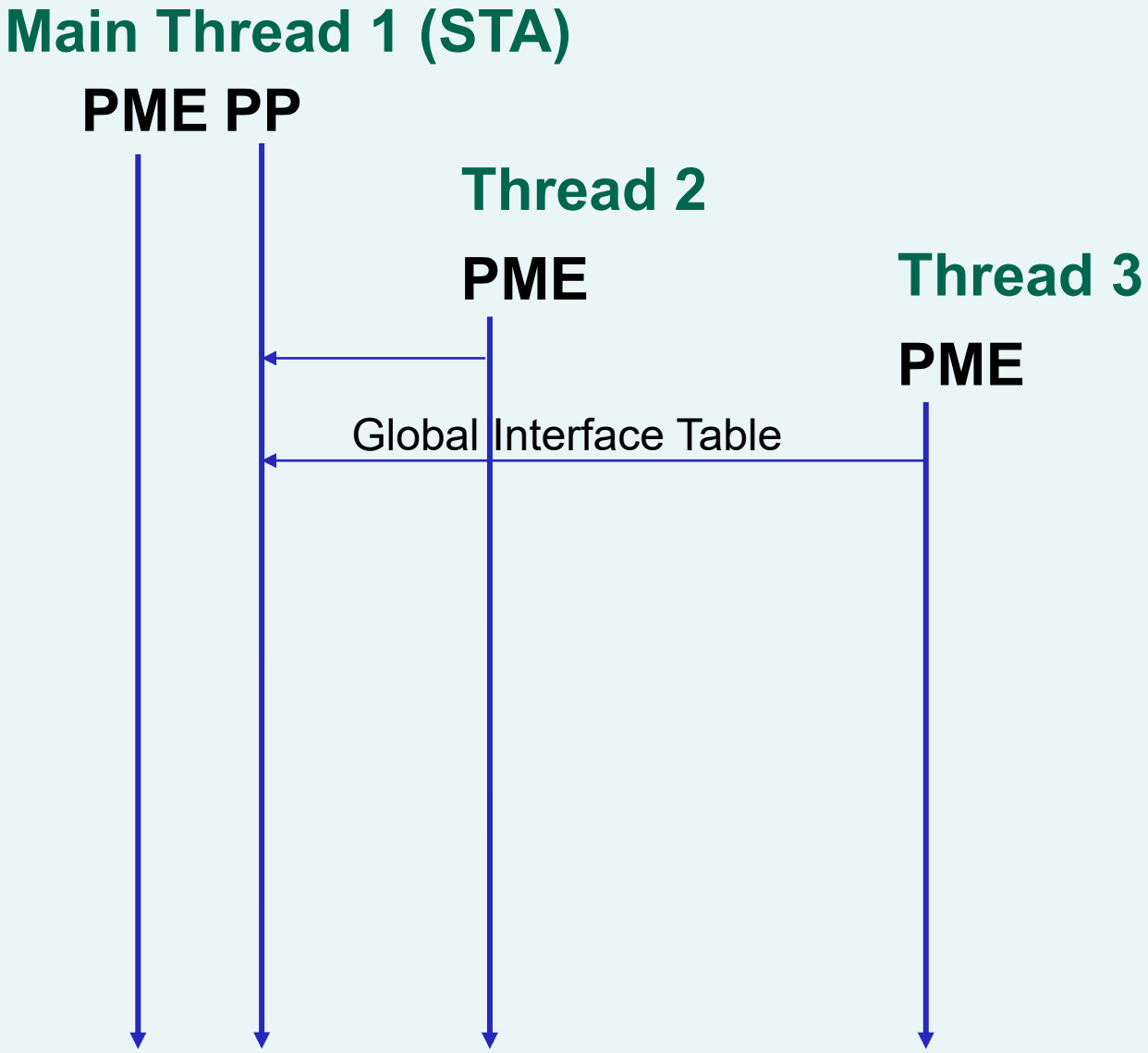
Per-thread PMC



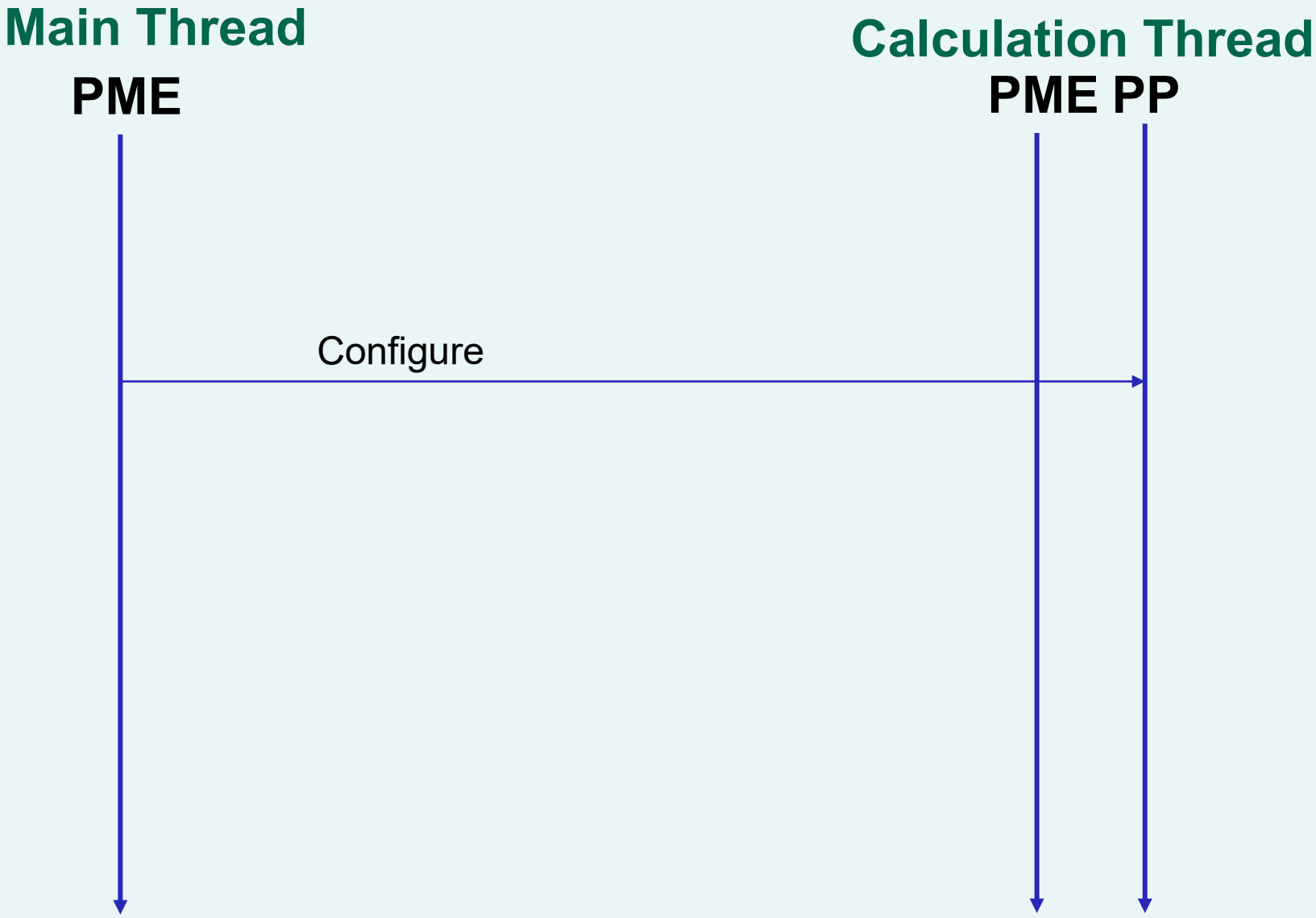
Per-thread PMC

- The duplicate PMC can be created via persistence: persisted PP1 in Thread 1, depersist PP2 in Thread 2 and PP3 in Thread 3
- If Threads 1, 2 and 3 are *not* all STA, marshaling will take place for Apartment Threaded PMCs
- If Threads 1, 2 and 3 are all STA, If the PMC is “Free” (not Apartment or Both), single or neutral, marshalling will take place
- For efficient operation, PME and *all* PMCs must use compatible threading model.

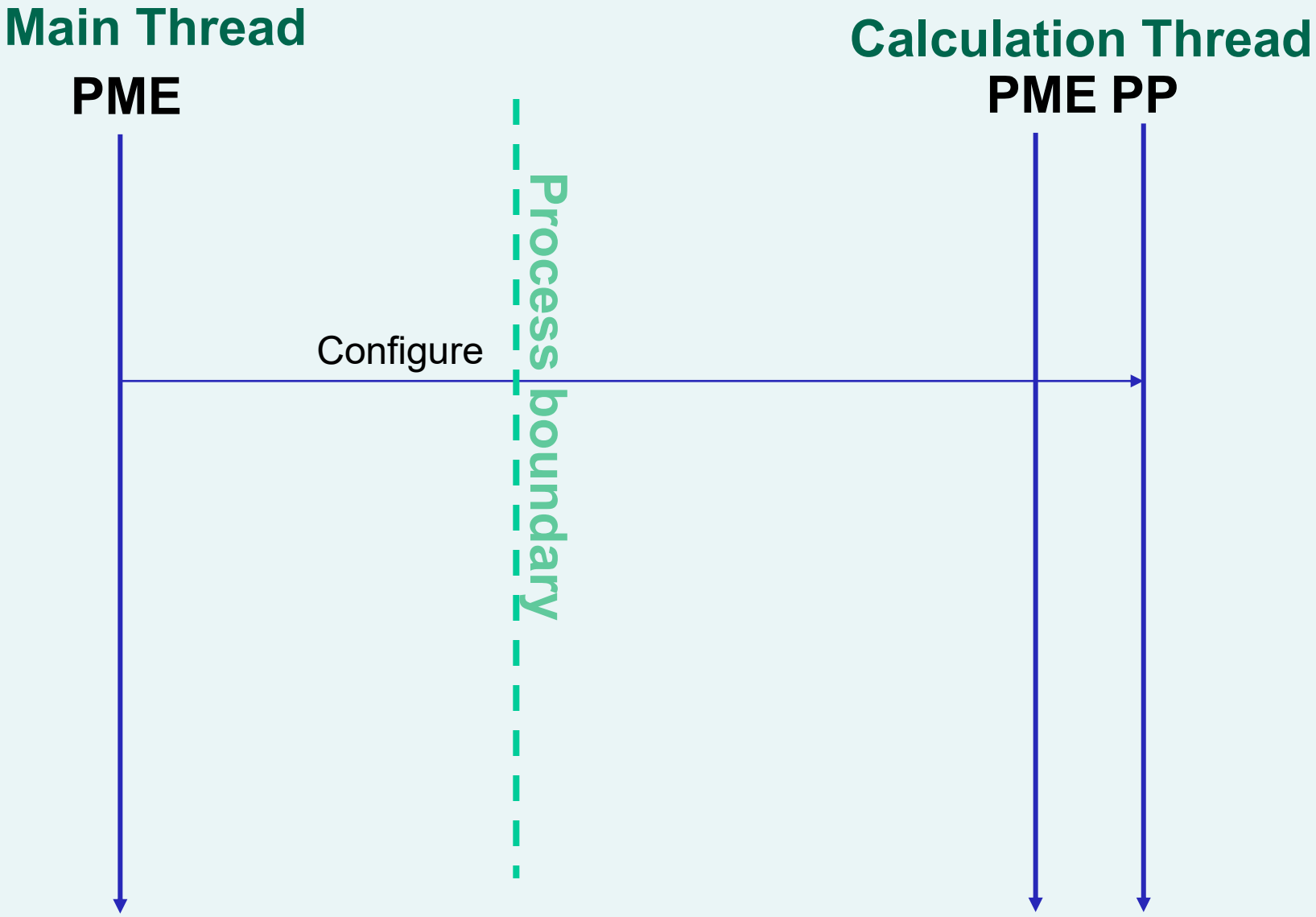
Global interface table



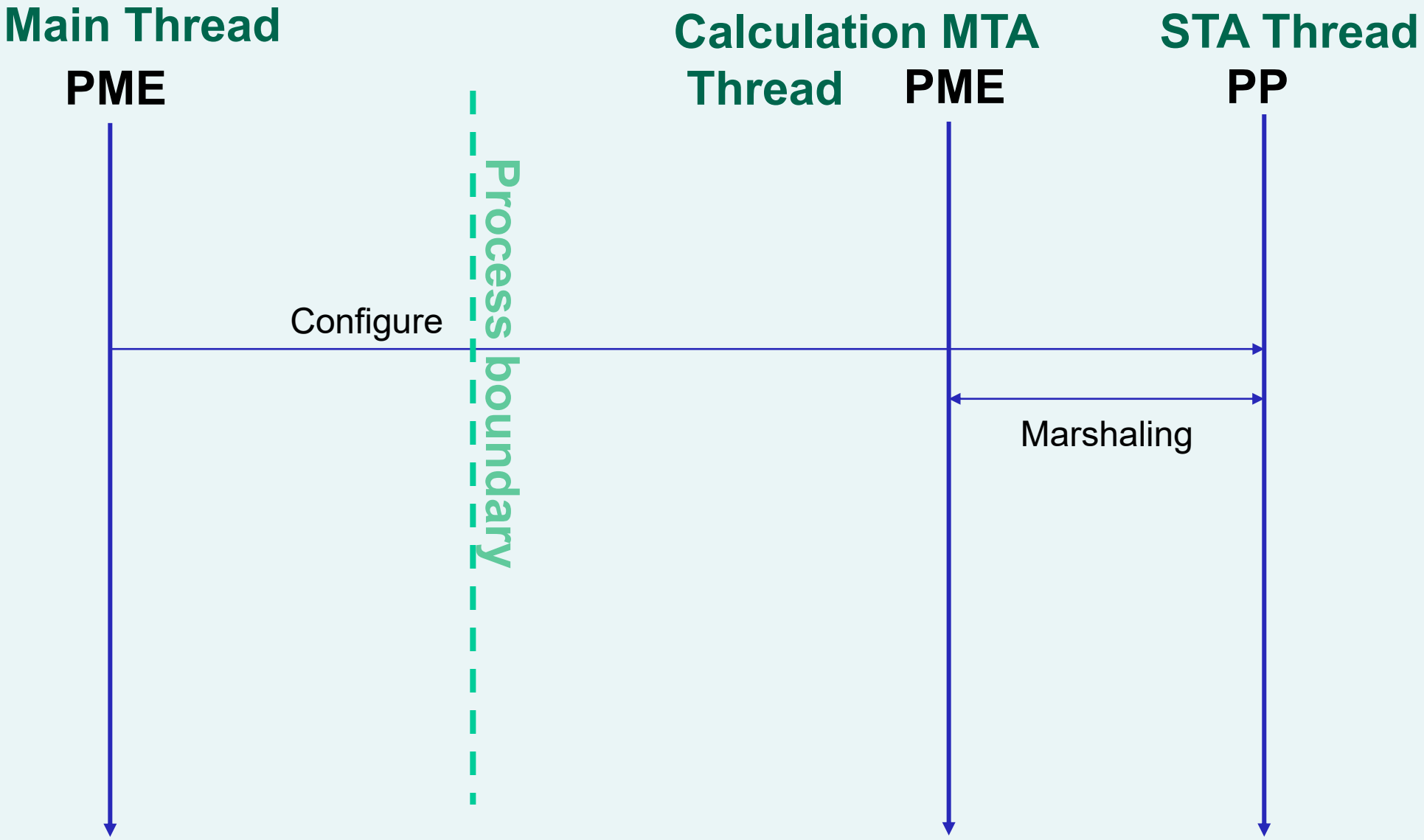
Single calculation thread



Single calculation thread



Single calculation thread



COBIA threading models

- **COBIA threading model per PMC – PME can mix and match different threading models per thread**

COBIA threading models

- **COBIA threading model per PMC – PME can mix and match different threading models per thread**
- **DEFAULT:**
  - **PME can call PMC from any thread**
  - **PME may not make concurrent calls on PMC**
    - **… or any of its secondary objects!**

COBIA threading models

- COBIA threading model per PMC – PME can mix and match different threading models per thread
- DEFAULT:
  - PME can call PMC from any thread
  - PME may not make concurrent calls on PMC
    - … or any of its secondary objects!
- RESTRICTED:
  - PME can only call PMC from one thread

COBIA threading models

- COBIA threading model per PMC – PME can mix and match different threading models per thread
- DEFAULT:
  - PME can call PMC from any thread
  - PME may not make concurrent calls on PMC
    - … or any of its secondary objects!
- RESTRICTED:
  - PME can only call PMC from one thread
- PME states intent on PMC creation: DEFAULT or RESTRICTED
  - RESTRICTED PMC in DEFAULT context is marshaled

Interoperability between COM and COBIA

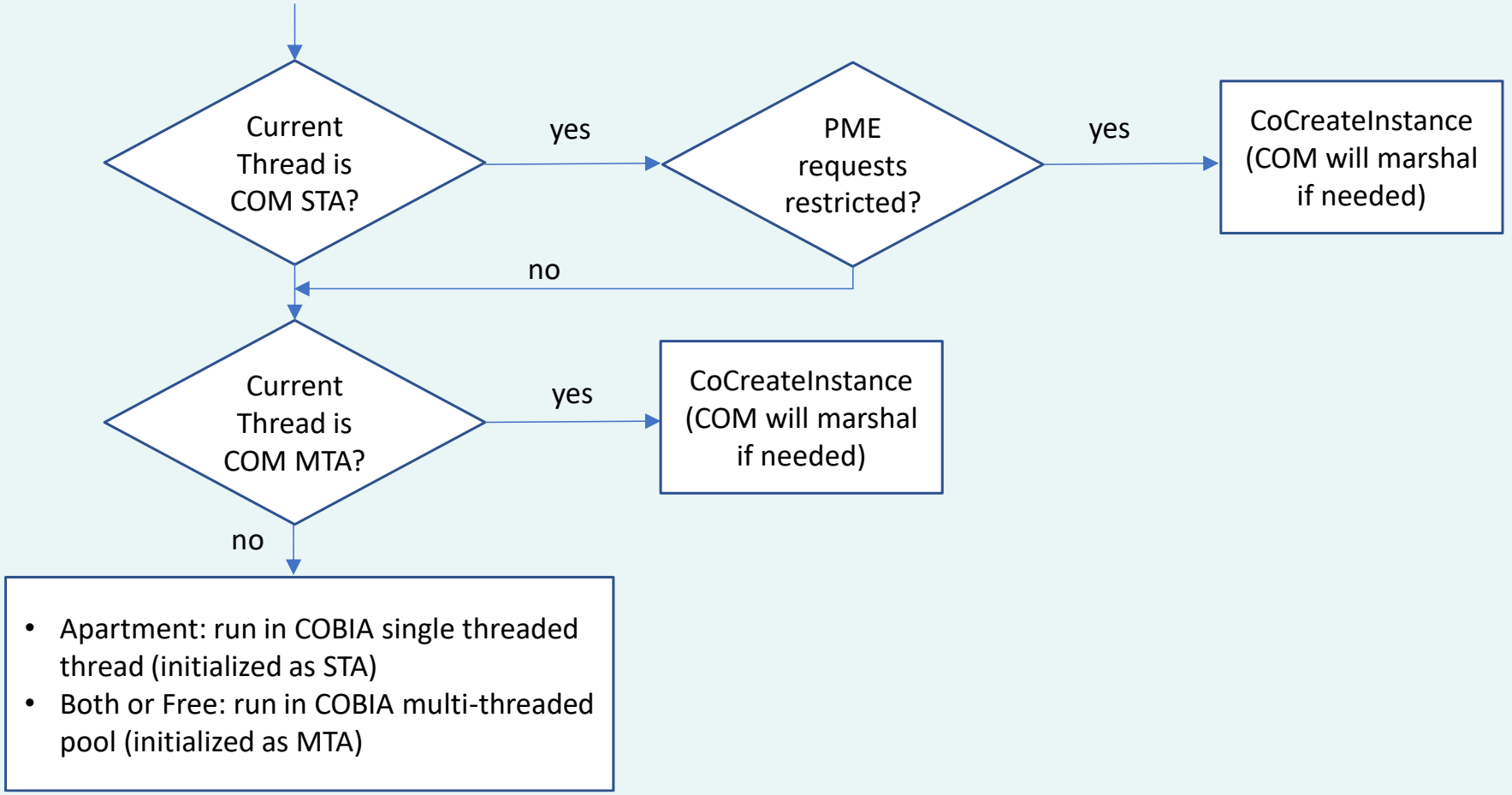
- COBIA PMCs using RESTRICTED threading model are advertised as COM “Apartment” threaded
- COBIA PMCs using DEFAULT threading model are advertised as COM “Both” threaded
  - As Both allows for concurrent access, COMBIA must take care of synchronization

Interoperability between COM and COBIA

- COM PMCs in COBIA require more attention:
  - Never call CoInitialize on a PME thread
    - May poorly interact with later PME COM initialization
    - CoInitialize must be matched by CoUninitialize
  - CoInitialize can be called on COBIA internal threads
- Synchronization of “Free” (MTA) PMC not needed:
  - COBIA PME may not make concurrent calls
  - COM Free threaded objects should be ready for concurrent calls

Threading

COMBIA creates
COM based PMC



Recommendations

- PME developers:
  - Check your threading model assumptions
  - New developments should assume multi-threading

- PMC developers
  - Use COBIA for new developments
  - Advertise your COM PMCs as “Both”
    - Be prepared for concurrent access

Outlook

- Threading Note to be released by M&T
- Expect feedback on COBIA threading model
  - RFC?
  - This presentation
- Implement COMBIA COM/COBIA interoperability
  - COBIA marshaling machinery already in place (PhaseIII)
- Gradual change in current ecosystem