Eliminating State Entanglement with Checkpoint-based Virtualization of Mobile OS Services

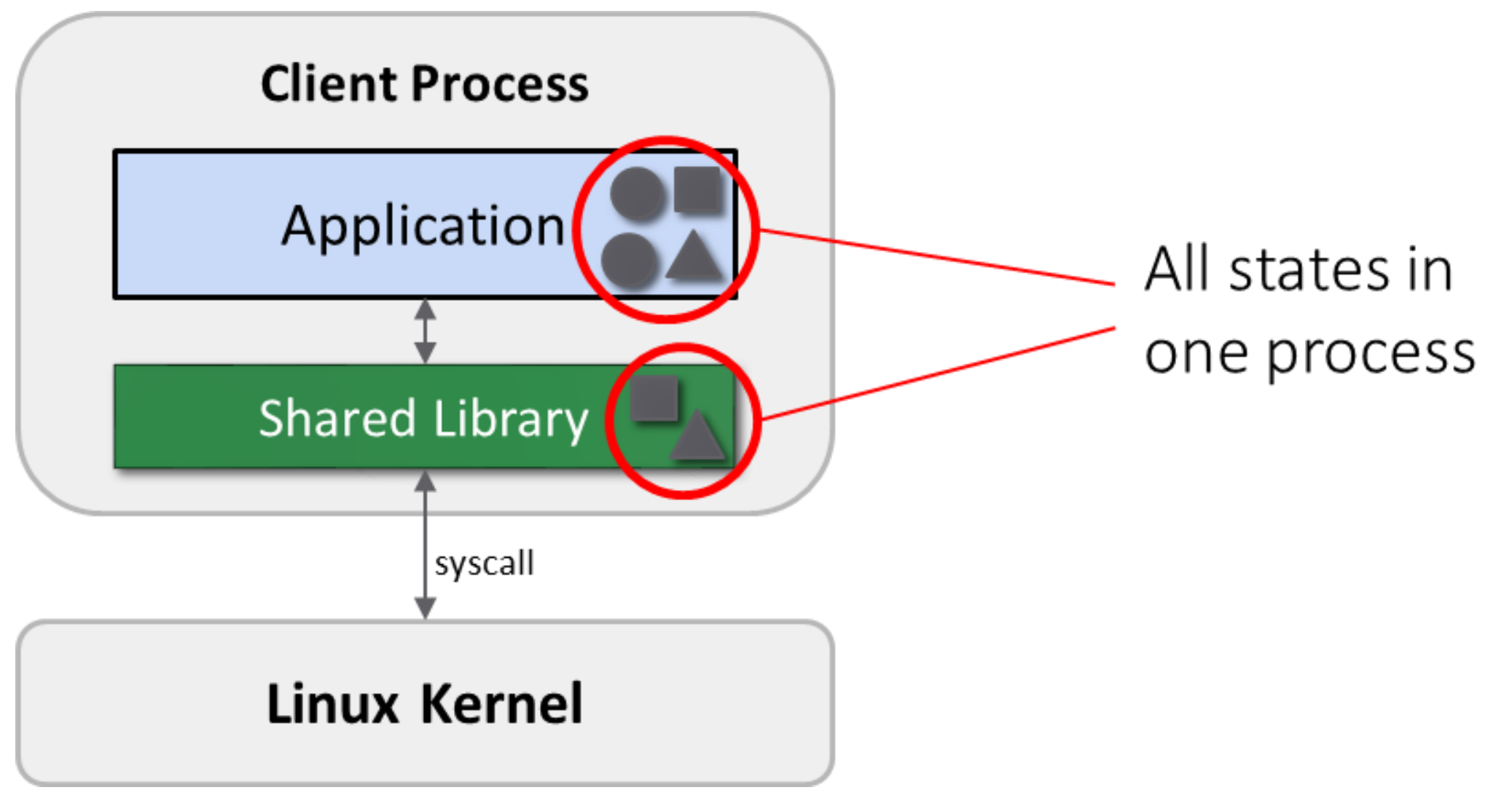
Kevin Boos & Lin Zhong

RICE Unconventional Wisdom
Rice University ECE
Houston, Texas

What is State Entanglement?

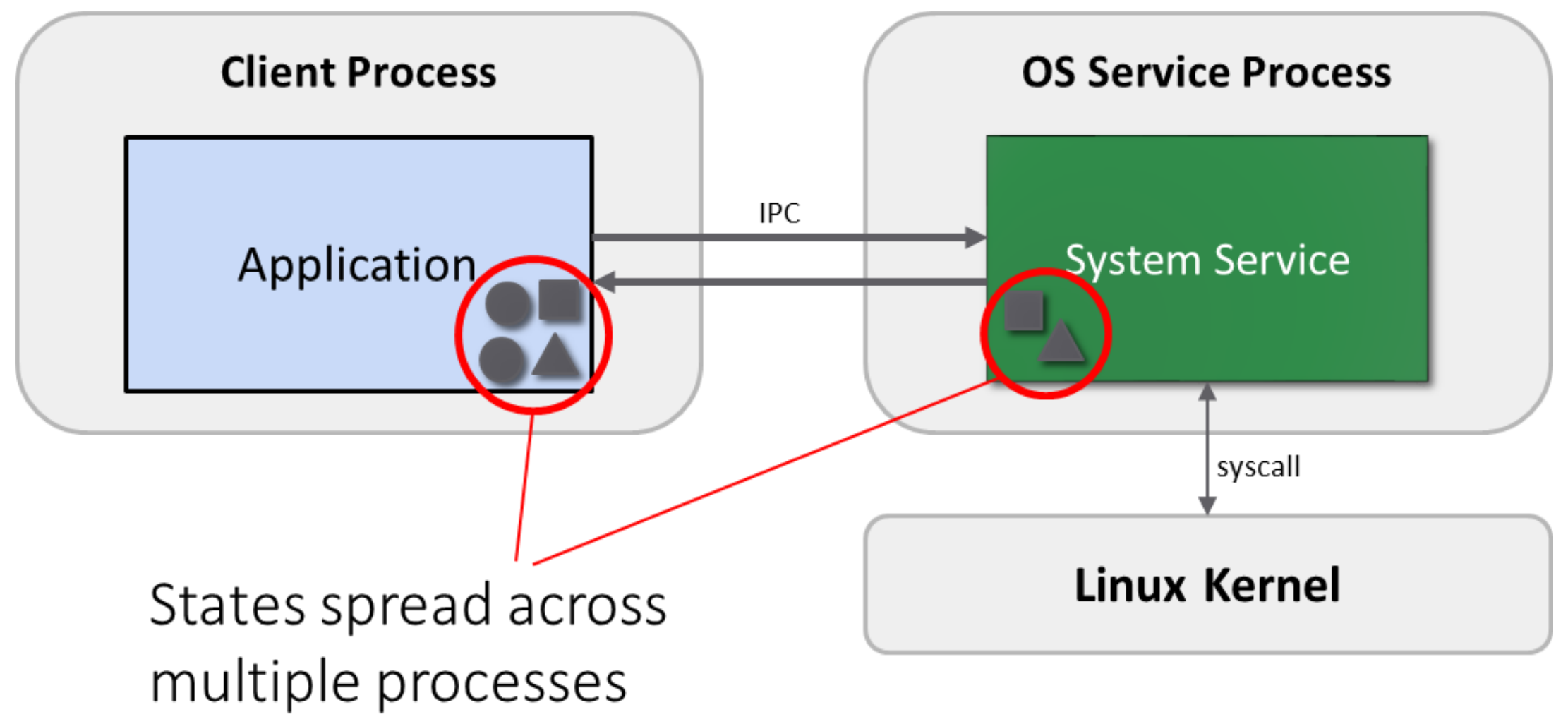
Application-relevant states are stored outside of the application's process memory

Shared Library Model



vs.

Mobile OS Service Model



Motivation – Why do we care?

State entanglement prevents the following:

- Fault isolation
- Fault tolerance
- Application migration
- Live update (of both apps and services)
- Whole-application speculation

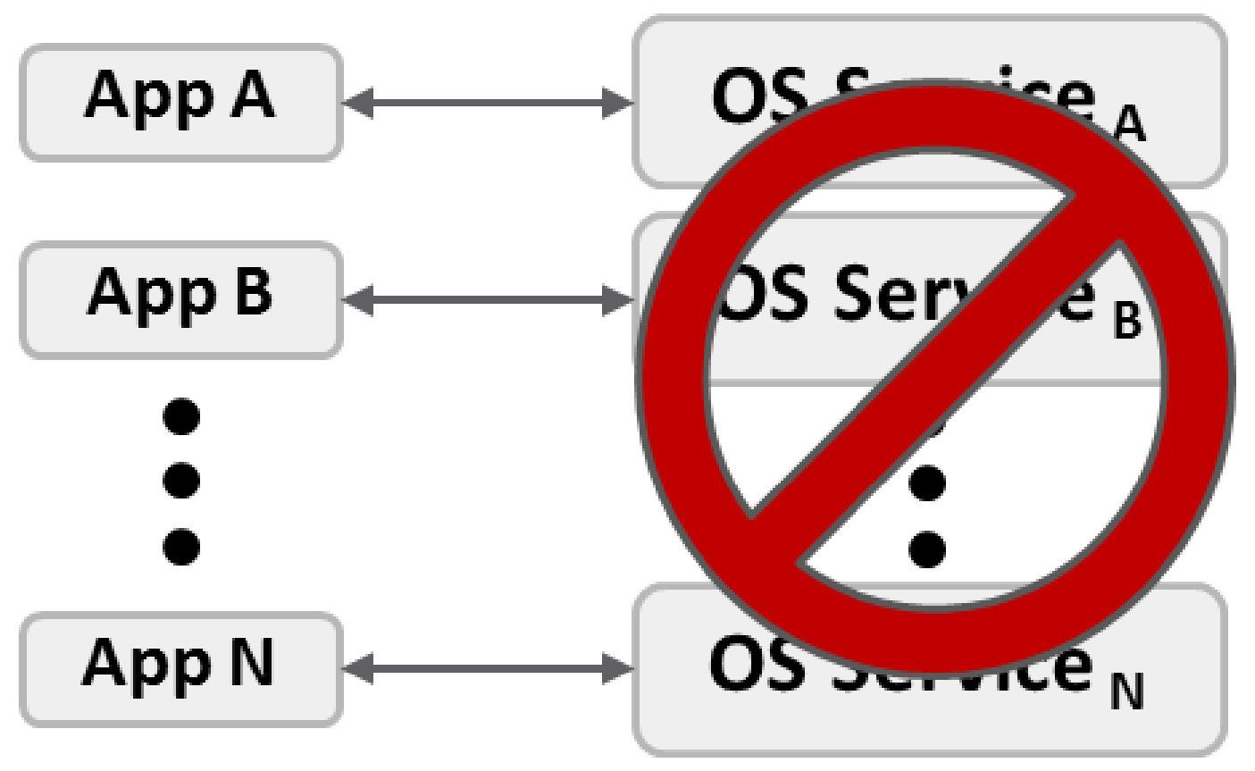
Solution: OS Service Virtualization

- Virtualize OS Service on a per-app basis
- Encapsulates *only one* app's states in each service instance
  - Disentangle states

www.owlnet.rice.edu/~kevinaboos

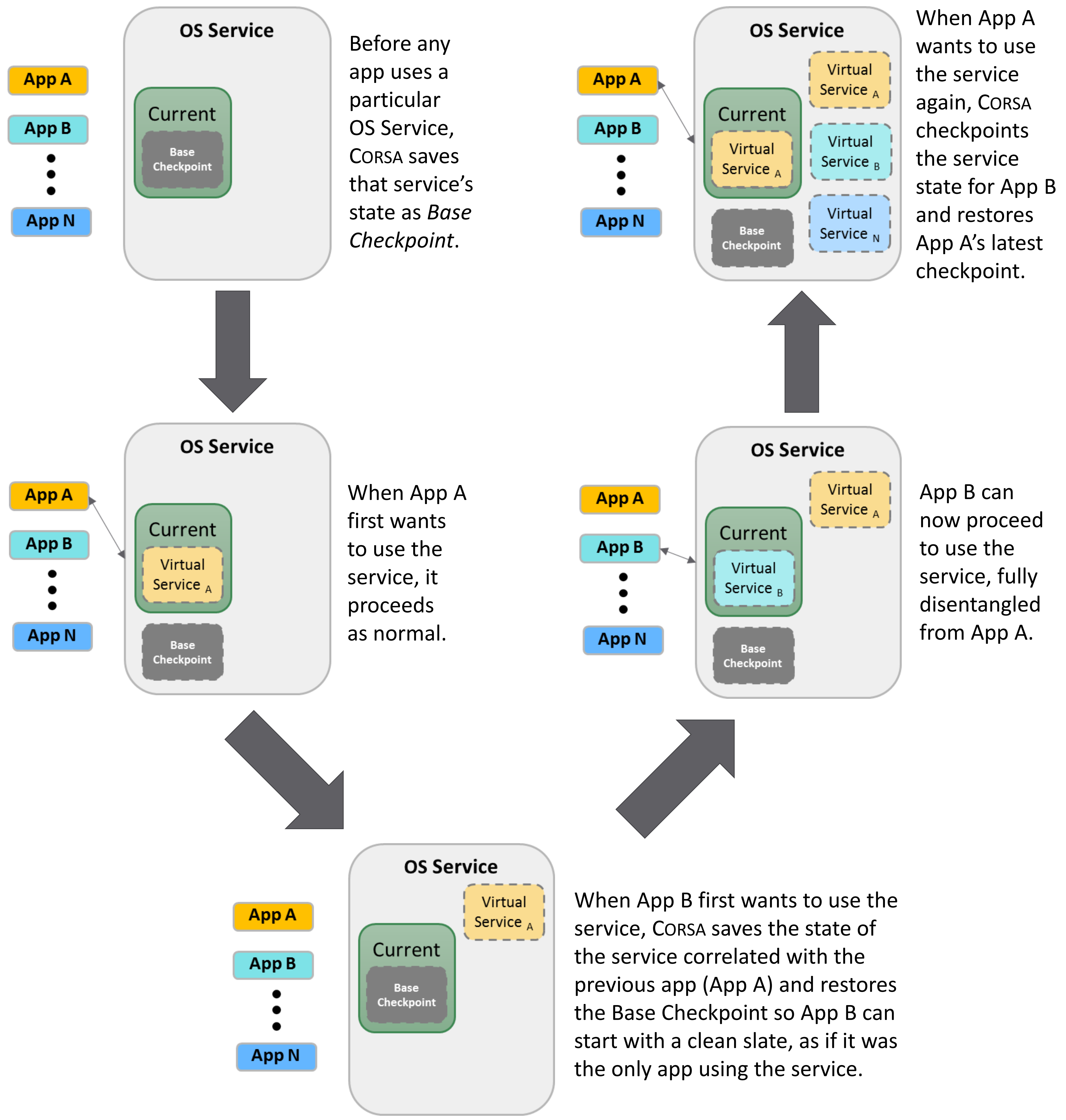
An OS Service cannot be instantiated multiple times!

Each service must be a singleton instance to ensure compatibility with the global service directory and other legacy OS components.



CORSA: Checkpoint-based Virtualization

- Virtualizes OS Services via checkpoint/restore
- Intercepts app-service transactions
- Maintains a per-app checkpoint history
- Only one service instance is active at a time
  - All other OS bodies see one service instance
  - Satisfies legacy expectations and constraints



Before any app uses a particular OS Service, CORSA saves that service's state as *Base Checkpoint*.

When App A first wants to use the service, it proceeds as normal.

When App B first wants to use the service, CORSA saves the state of the service correlated with the previous app (App A) and restores the Base Checkpoint so App B can start with a clean slate, as if it was the only app using the service.

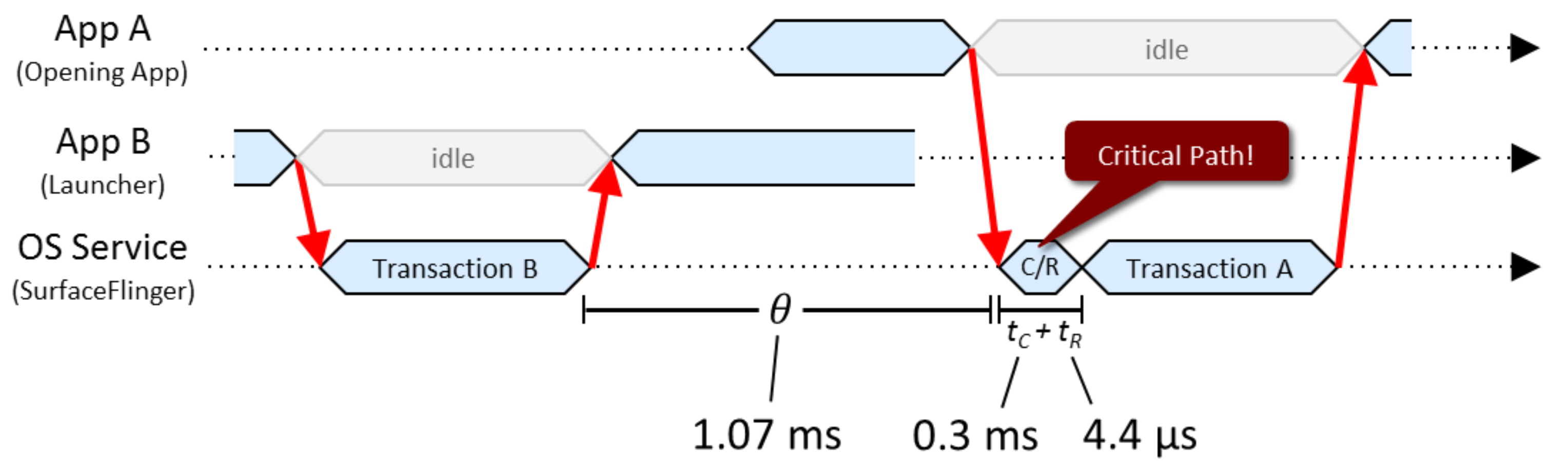
App B can now proceed to use the service, fully disentangled from App A.

When App A wants to use the service again, CORSA checkpoints the service state for App B and restores App A's latest checkpoint.

Ongoing Implementation

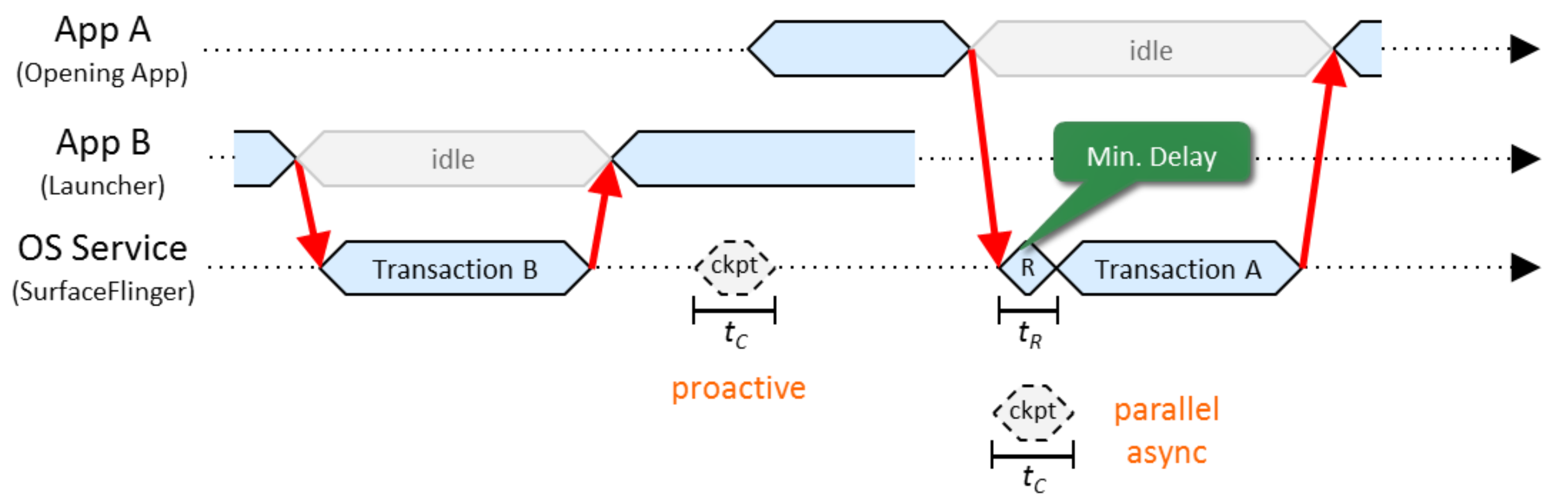
- Kernel-based C/R mechanism
  - **Checkpoint:** duplicates process structures, uses COW for speed
  - **Restore:** swaps process control block pointers to previous checkpoint
  - Triggered on Binder IPC transactions

Feasibility Measurement Study



Checkpoint and Restore can be parallelized

- Slow checkpoint, *fast* restore operation
- Only restore is on the critical path



- Checkpoint latency $t_C = 0.3$ ms
- Restore latency $t_R = 4.4$ µs
- Min. transaction interval: $\theta = 1.07$ ms
- Max transaction frequency: $f = 221$ Hz

CORSA Android Implementation is Feasible!

✓ Checkpoint latency (t_C) $<$ θ
✓ Restore latency (t_R) $<<$ θ
✓ No perceivable effect on user experience