Out-of-memory must fail fast

Mark S. Miller, JF Paradis — Agoric
tc39 October 2019

JS: **Cannot implement language correctly**

Java: **Cannot write a correct program**

```
function splice(left, newRight) {
  const oldRight = left.right;
  newRight.left = left;
  newRight.right = oldRight;
  left.right = newRight;
  console.log(whatever);  // Zalgo beckons
  oldRight.left = newRight;
}
```

```
function splice(left, newRight) {
  const oldRight = left.right;
  newRight.left = left;
  newRight.right = oldRight;
  left.right = newRight;
```

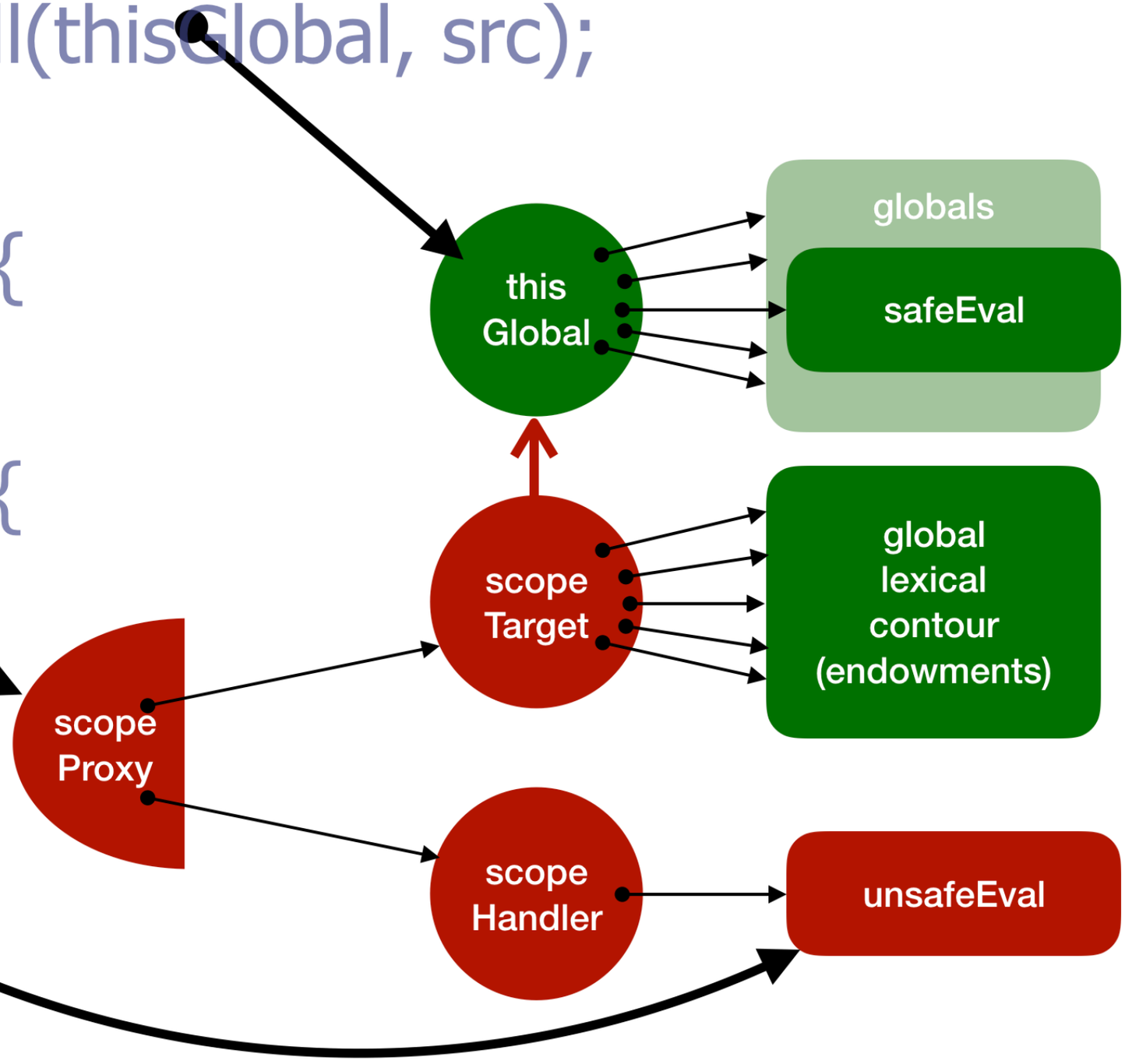
```
function splice(left, newRight) {
  const oldRight = left.right;
  newRight.left = left;
  newRight.right = oldRight;
  left.right = newRight;
  try {
    console.log(whatever);
    oldRight.left = newRight;
  } catch or finally {
    WHAT?  // Zalgo laughs
  }
}
```

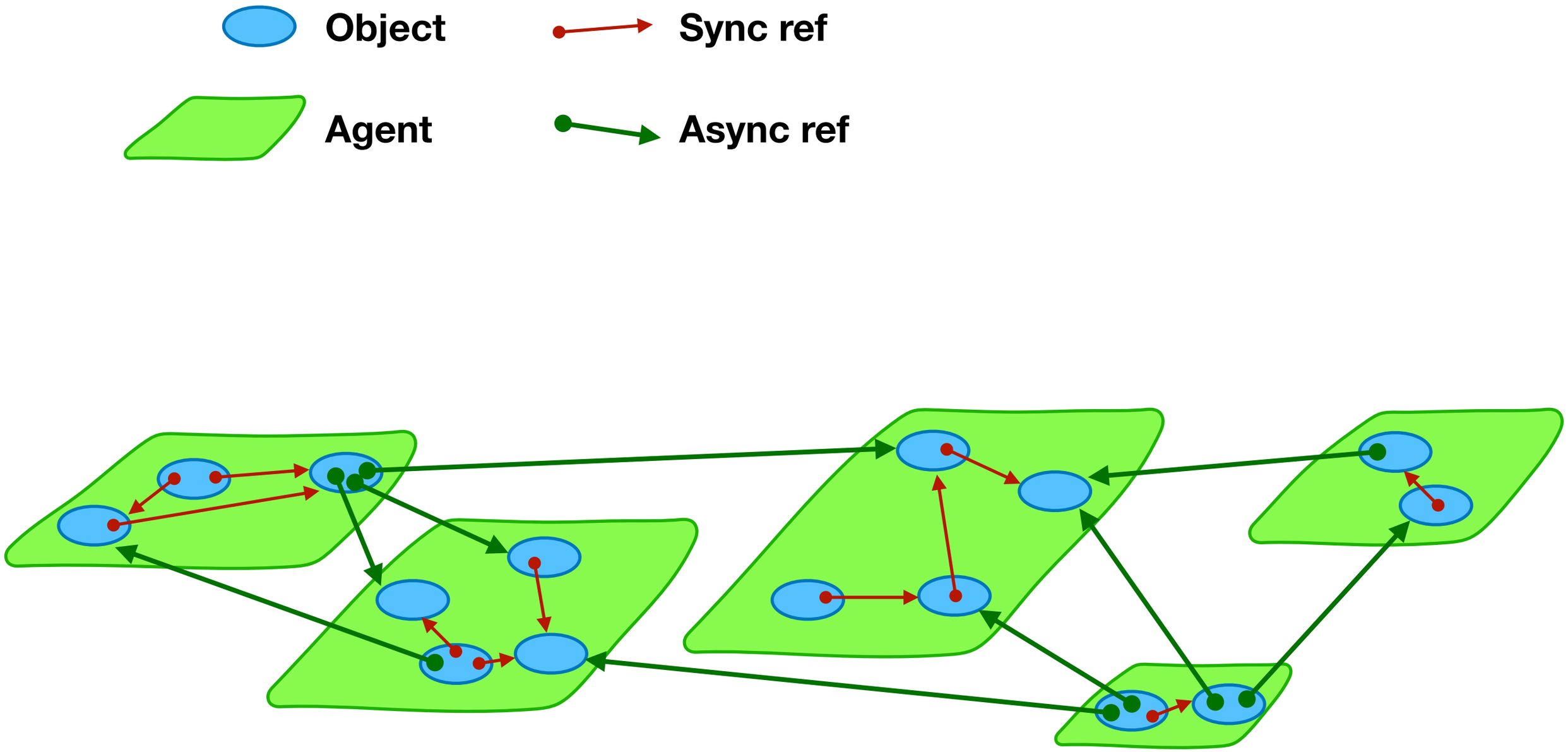
1.1 Exploit by forcing an out-of-memory exception in `safeEval`

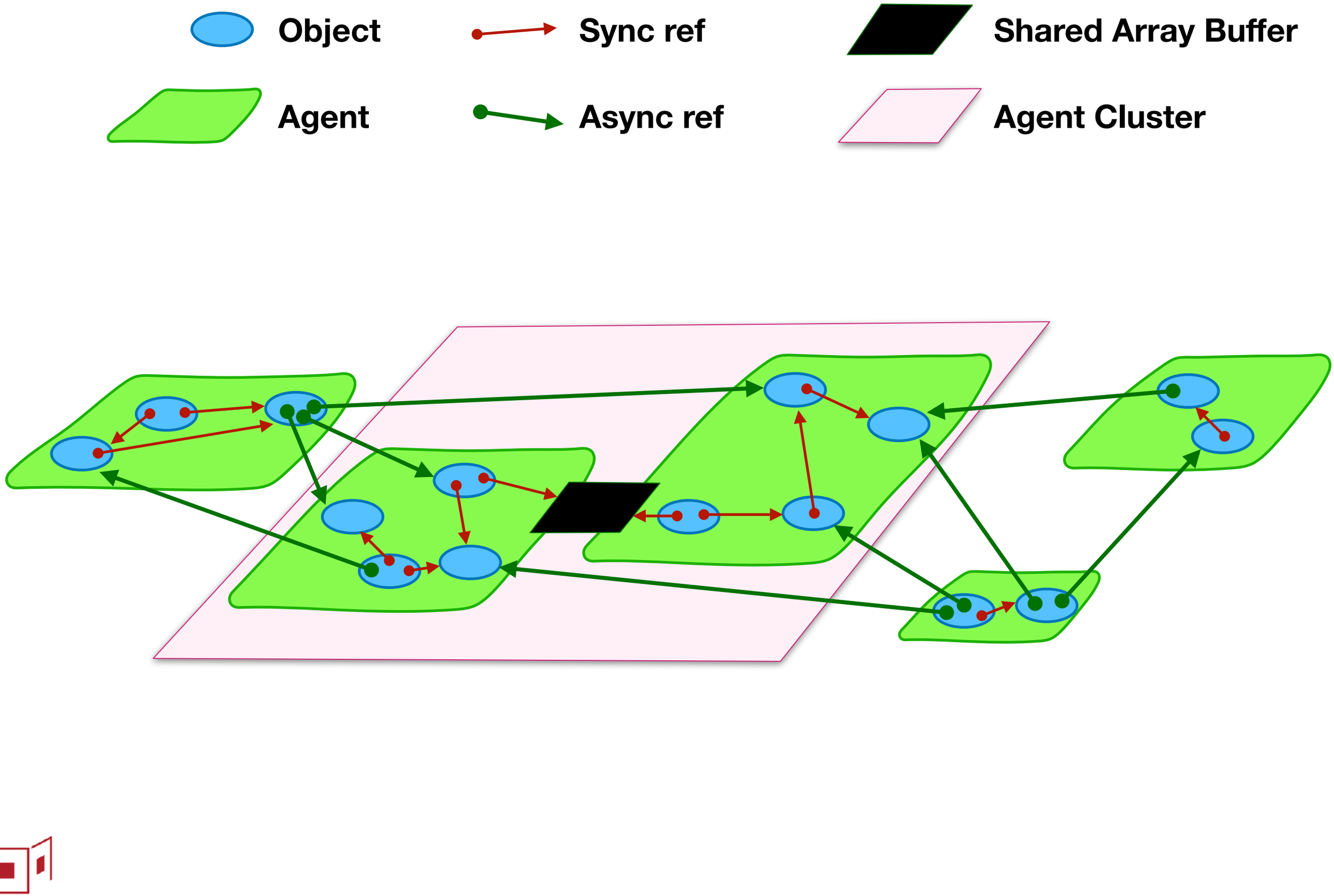
```
function loop(){
  (0,eval)('1');
  loop();
}

try{
  loop();
} catch(e) {}
eval + '' // "function eval() { [native code] }"
```

```
return f(scopeProxy).call(thisGlobal, src);
...
function f(scopeProxy) {
  with (scopeProxy) {
    return function(src) {
      "use strict";
      return eval(src);
    };
  }
};
```




Object
Sync ref
Agent
Async ref


Object
Sync ref
Shared Array Buffer
Agent
Async ref
Agent Cluster

Refresh page

Dead tabs

} **Site Isolation?**

Refresh page

Dead tabs

Reboot device

} **Site Isolation?**

Abort transaction

Refresh page

Dead tabs

Reboot device

} **Site Isolation?**

Abort transaction

Erlang Supervisor / KeyKOS Keeper

Pre-mortem *vs* post-mortem “finalization” generalized

Pre-mortem *vs* post-mortem “finalization” generalized

```
function memKeeper(...) {...}
const agentCluster = new AgentCluster({oom: memKeeper});
```

Pre-mortem *vs* post-mortem “finalization” generalized

```
function memKeeper(...) {...}
const agentCluster = new AgentCluster({oom: memKeeper});
```

Fault-tolerant systems from fail-stop components

Questions?