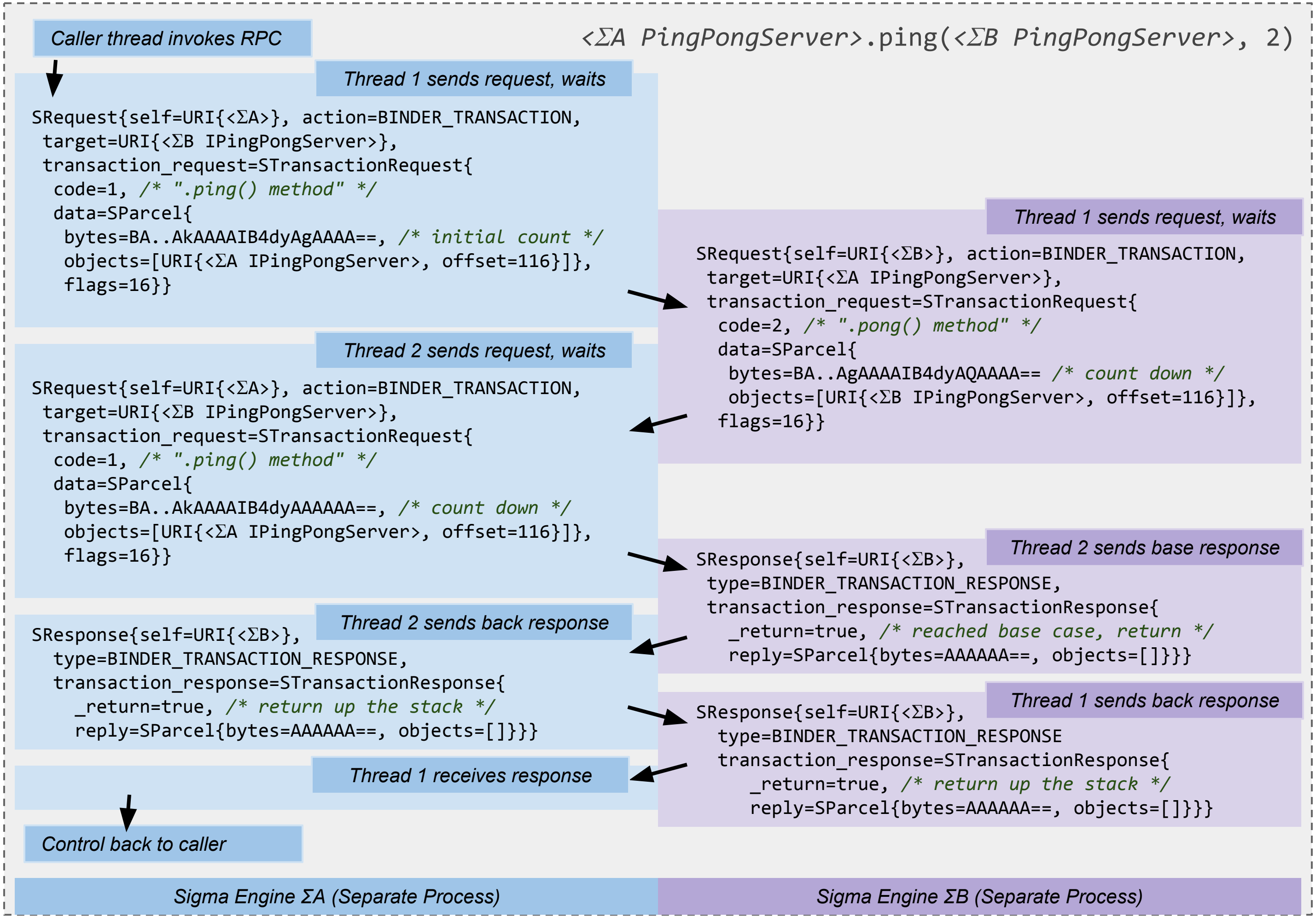
Caller thread invokes RPC
Thread 1 sends request, waits
Thread 1 sends request, waits
Thread 2 sends request, waits
Thread 2 sends base response
Thread 2 sends back response
Thread 1 sends back response
Thread 1 receives response
Control back to caller
Sigma Engine ΣA (Separate Process)
Sigma Engine ΣB (Separate Process)

```
<ΣA PingPongServer>.ping(<ΣB PingPongServer>, 2)
```

```
SRequest{self=URI{<ΣA>}, action=BINDER_TRANSACTION,
 target=URI{<ΣB IPingPongServer>},
 transaction_request=STransactionRequest{
  code=1, /* ".ping() method" */
  data=SParcel{
   bytes=BA..AkAAAAIB4dyAgAAAA==, /* initial count */
   objects=[URI{<ΣA IPingPongServer>, offset=116}]},
   flags=16}}
```

```
SRequest{self=URI{<ΣB>}, action=BINDER_TRANSACTION,
 target=URI{<ΣA IPingPongServer>},
 transaction_request=STransactionRequest{
  code=2, /* ".pong() method" */
  data=SParcel{
   bytes=BA..AgAAAAIB4dyAQAAAA== /* count down */
   objects=[URI{<ΣB IPingPongServer>, offset=116}]},
  flags=16}}
```

```
SRequest{self=URI{<ΣA>}, action=BINDER_TRANSACTION,
 target=URI{<ΣB IPingPongServer>},
 transaction_request=STransactionRequest{
  code=1, /* ".ping() method" */
  data=SParcel{
   bytes=BA..AkAAAAIB4dyAAAAAA==, /* count down */
   objects=[URI{<ΣA IPingPongServer>, offset=116}]},
   flags=16}}
```

```
SResponse{self=URI{<ΣB>},
 type=BINDER_TRANSACTION_RESPONSE,
 transaction_response=STransactionResponse{
   _return=true, /* reached base case, return */
   reply=SParcel{bytes=AAAAAA==, objects=[]}}}
```

```
SResponse{self=URI{<ΣB>},
  type=BINDER_TRANSACTION_RESPONSE,
  transaction_response=STransactionResponse{
    _return=true, /* return up the stack */
    reply=SParcel{bytes=AAAAAA==, objects=[]}}}
```

```
SResponse{self=URI{<ΣB>},
  type=BINDER_TRANSACTION_RESPONSE
  transaction_response=STransactionResponse{
     _return=true, /* return up the stack */
     reply=SParcel{bytes=AAAAAA==, objects=[]}}}
```