Eventual Send

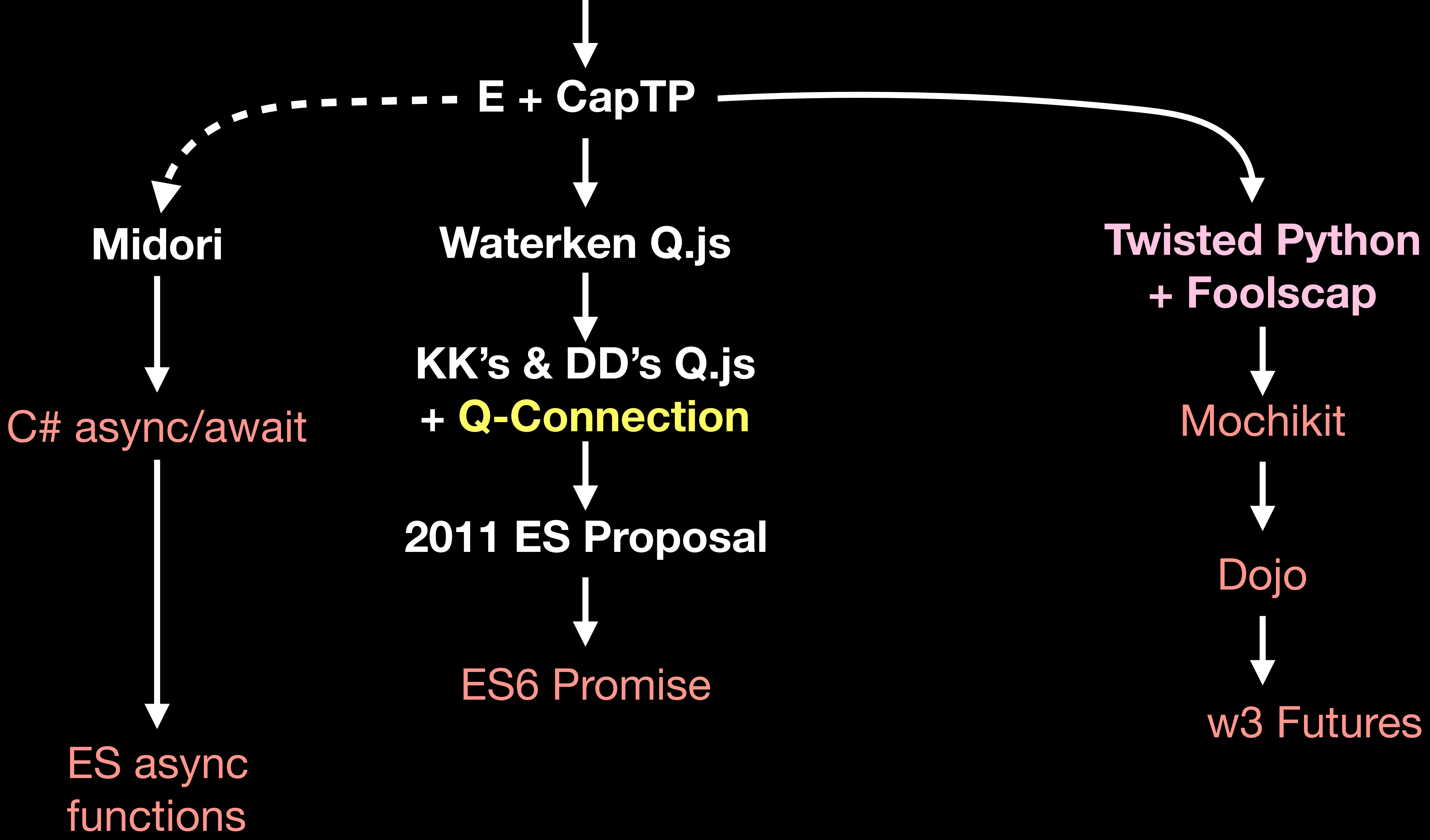
`HandledPromise` API for promise pipelining

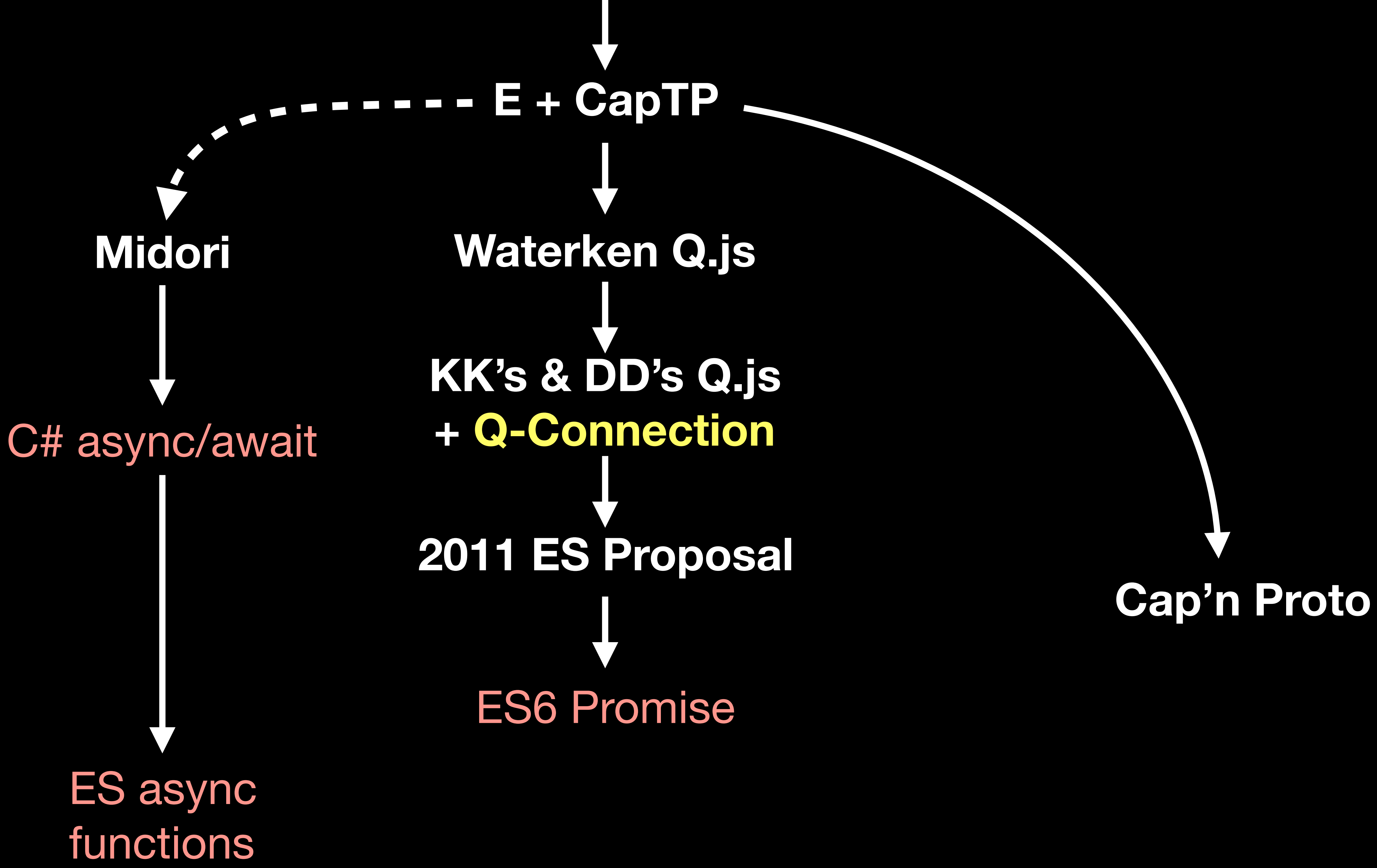
Mark S. Miller, Michael Fig — Agoric
Chip Morningstar — Evernote
tc39 October 2019

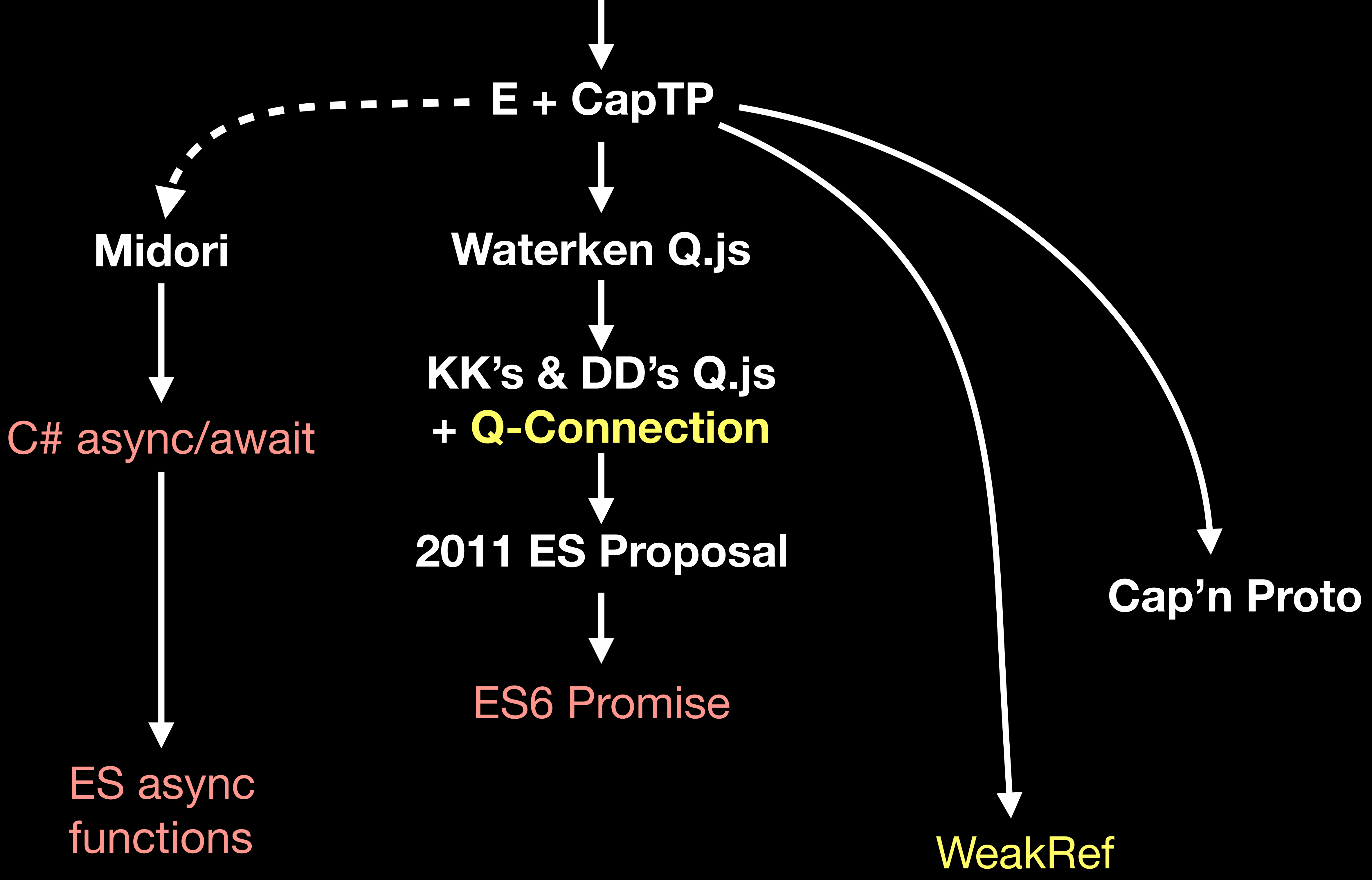
ES6 Promise

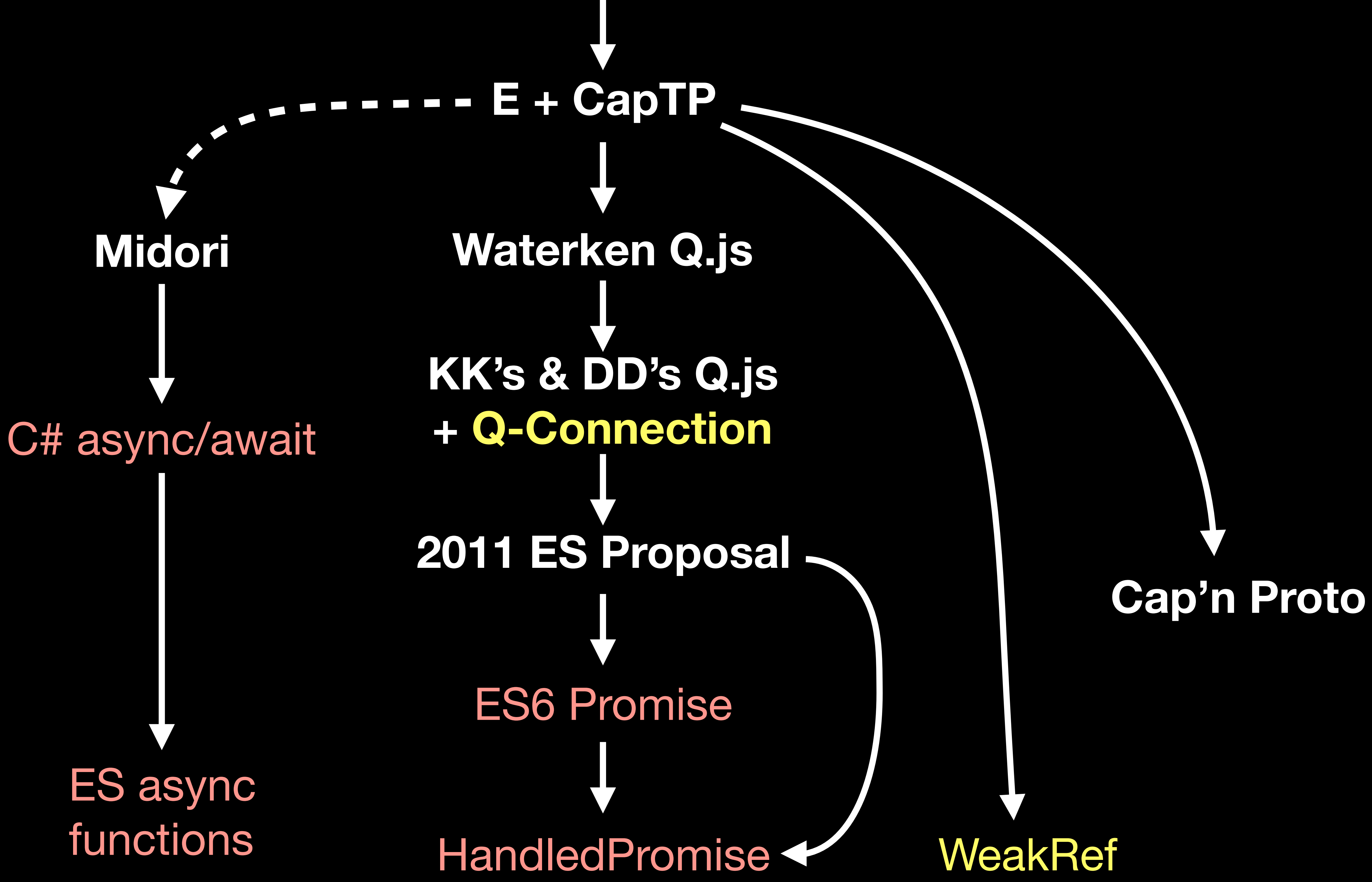
$$\begin{tikzcd}
{} \arrow[d] \\
\text{E + CapTP} \arrow[d] \\
\text{Waterken Q.js} \arrow[d] \\
\text{KK's \& DD's Q.js + Q-Connection} \arrow[d] \\
\text{2011 ES Proposal} \arrow[d] \\
\text{ES6 Promise}
\end{tikzcd}$$

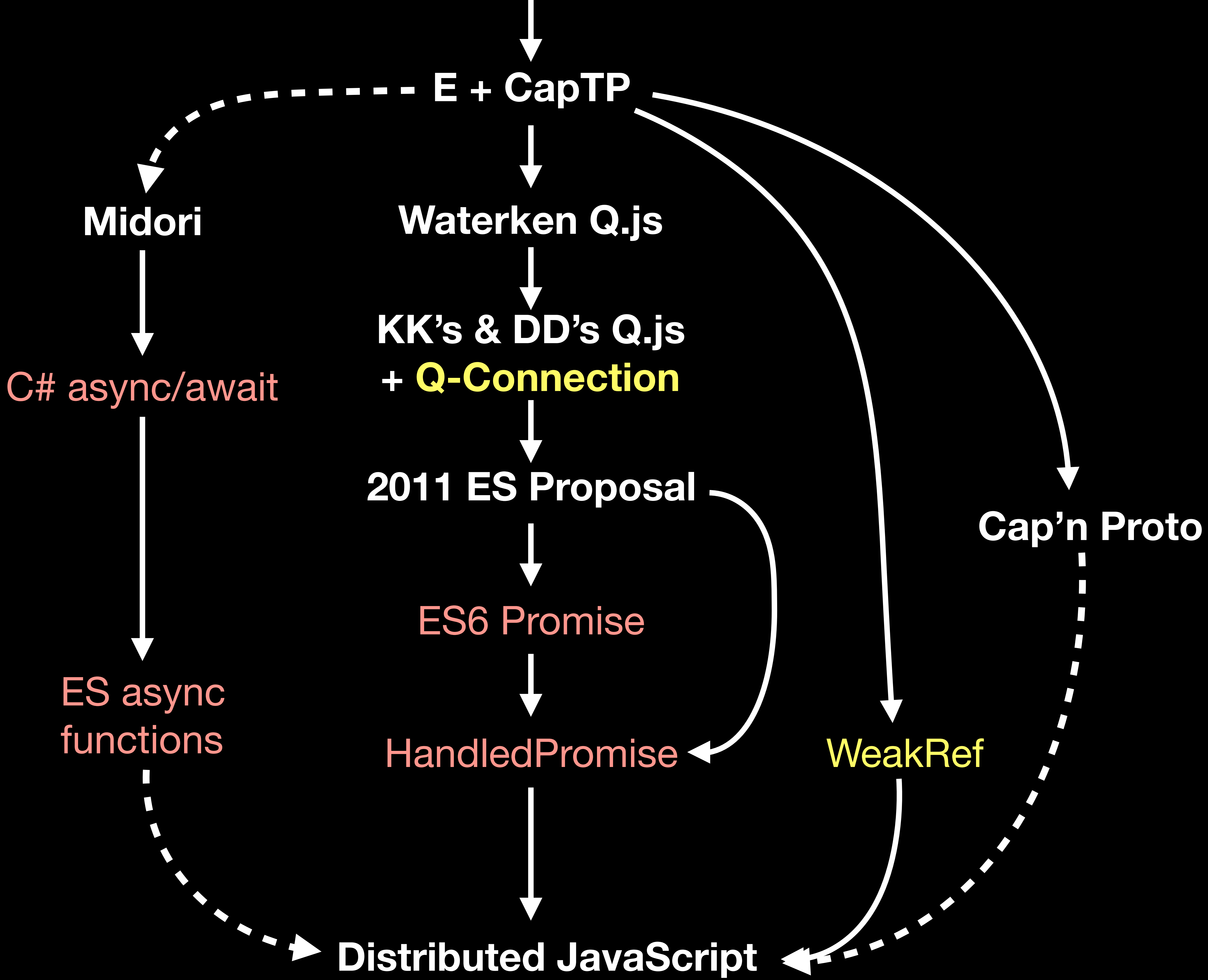
$$\begin{tikzcd}
 & {} \arrow[d] \\
 & \text{E + CapTP} \arrow[dl, dashed, bend right] \arrow[d] \\
\text{Midori} \arrow[d] & \text{Waterken Q.js} \arrow[d] \\
\text{C\# async/await} \arrow[d] & \text{KK's \& DD's Q.js + Q-Connection} \arrow[d] \\
\text{ES async functions} & \text{2011 ES Proposal} \arrow[d] \\
 & \text{ES6 Promise}
\end{tikzcd}$$


E + CapTP
Midori
Waterken Q.js
Twisted Python + Foolscap
C# async/await
KK's & DD's Q.js + Q-Connection
Mochikit
2011 ES Proposal
Dojo
ES6 Promise
w3 Futures
ES async functions

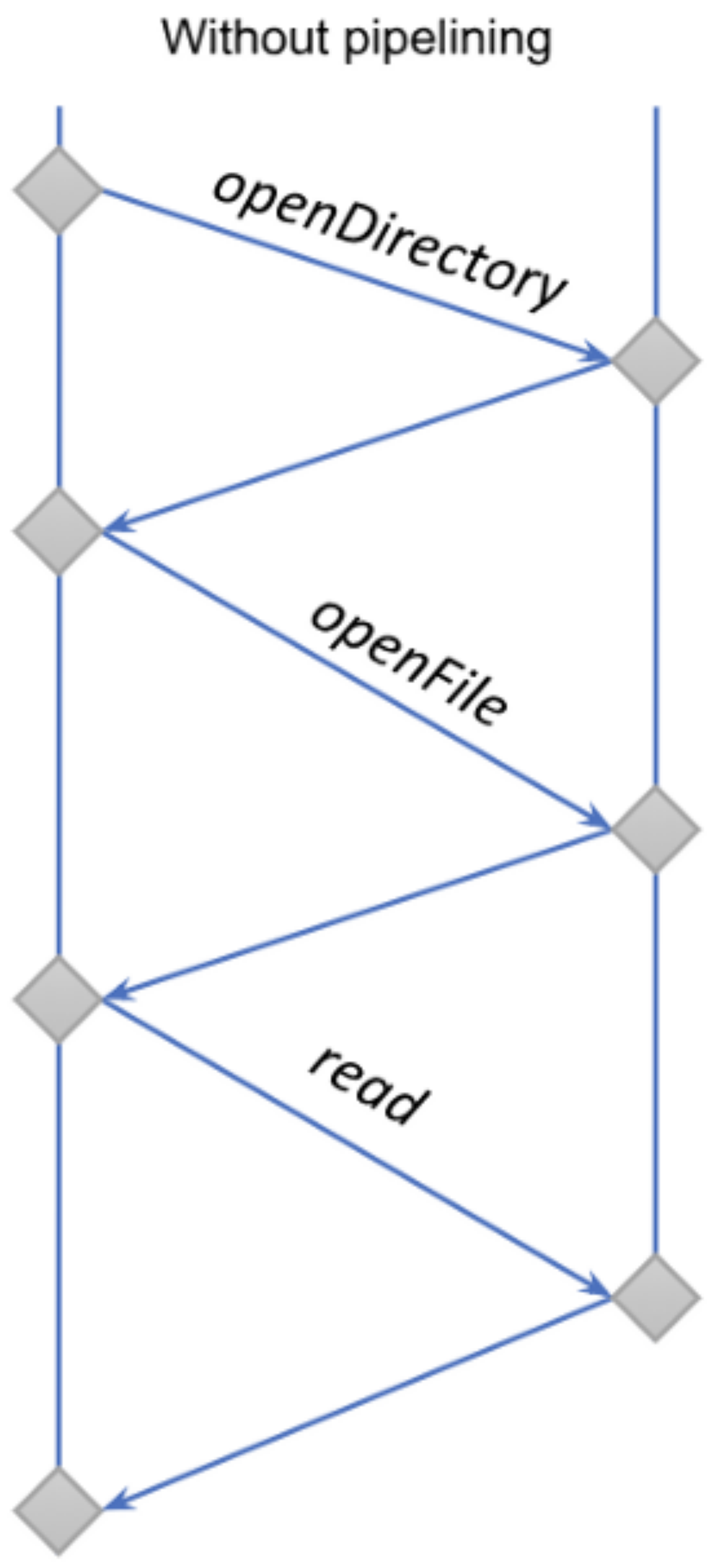

E + CapTP
Midori
C# async/await
ES async functions
Waterken Q.js
KK’s & DD’s Q.js
+ Q-Connection
2011 ES Proposal
ES6 Promise
Cap’n Proto

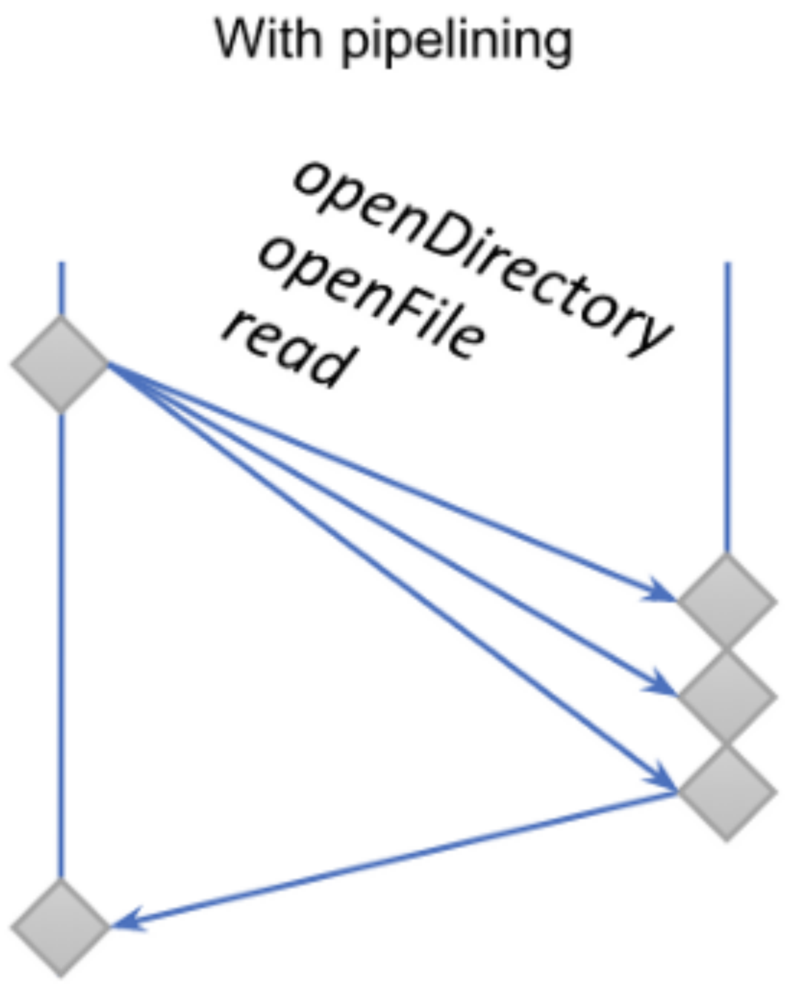

E + CapTP
Midori
Waterken Q.js
KK's & DD's Q.js
+ Q-Connection
2011 ES Proposal
ES6 Promise
C# async/await
ES async
functions
WeakRef
Cap'n Proto

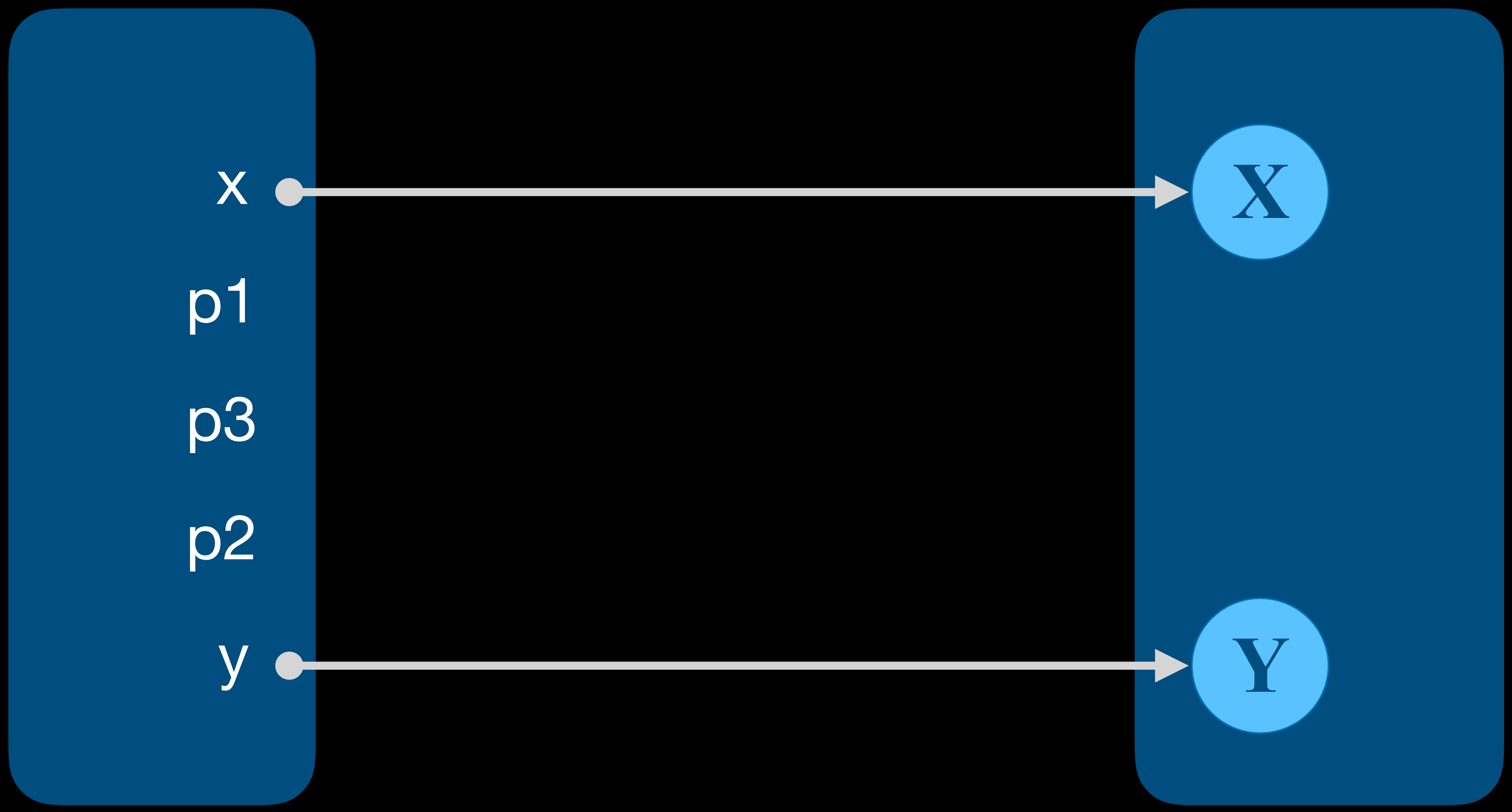

E + CapTP
Midori
Waterken Q.js
C# async/await
KK's & DD's Q.js
+ Q-Connection
2011 ES Proposal
Cap'n Proto
ES6 Promise
ES async
functions
HandledPromise
WeakRef


E + CapTP
Midori
Waterken Q.js
KK’s & DD’s Q.js
+ Q-Connection
C# async/await
2011 ES Proposal
Cap’n Proto
ES6 Promise
ES async
functions
HandledPromise
WeakRef
Distributed JavaScript

disk ~. openDirectory('foo') ~. openFile('bar.txt') ~. read()

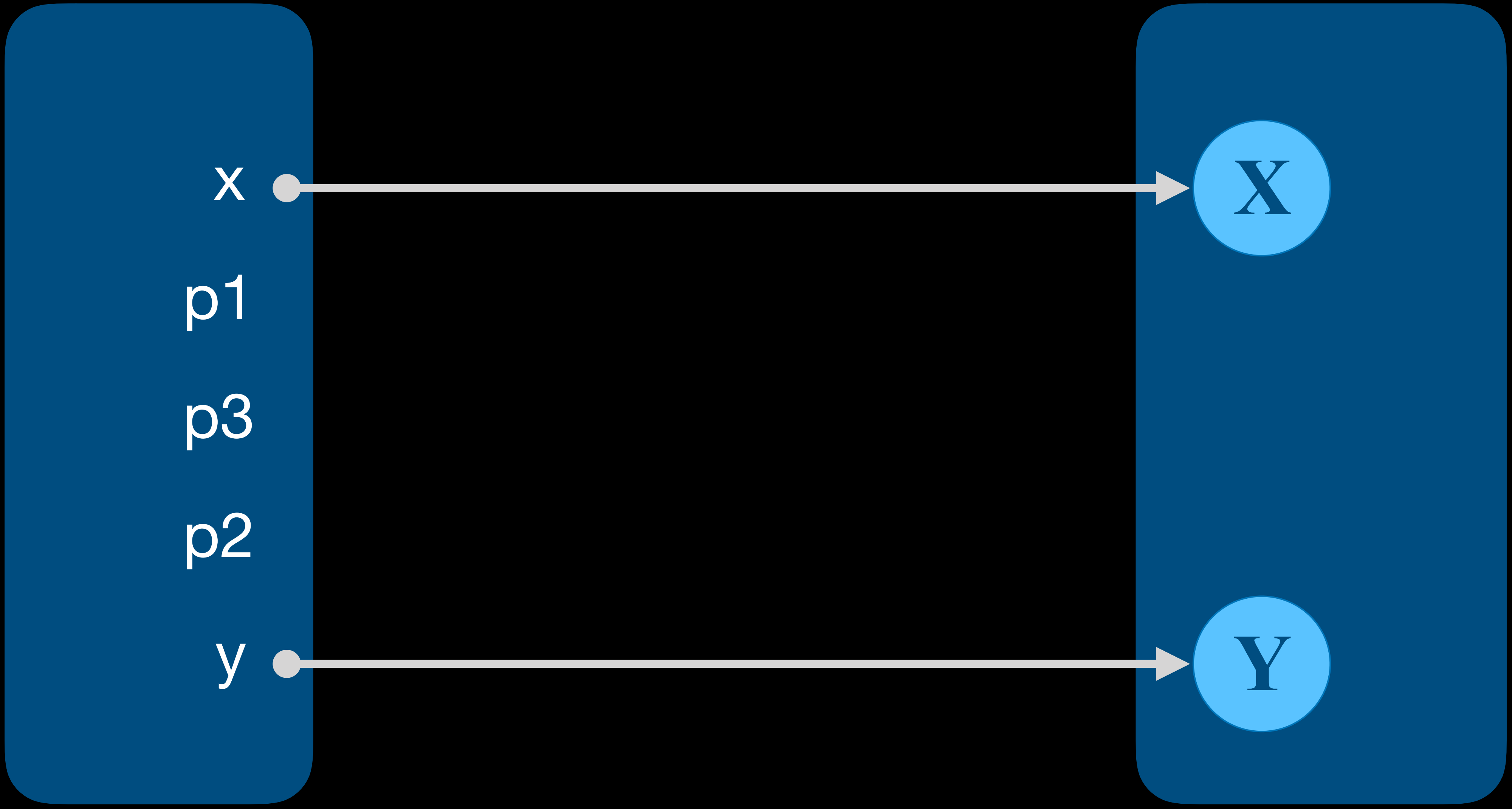





x
p1
p3
p2
y
X
Y

```
const t3 = (x.a()).c(y.b());

const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

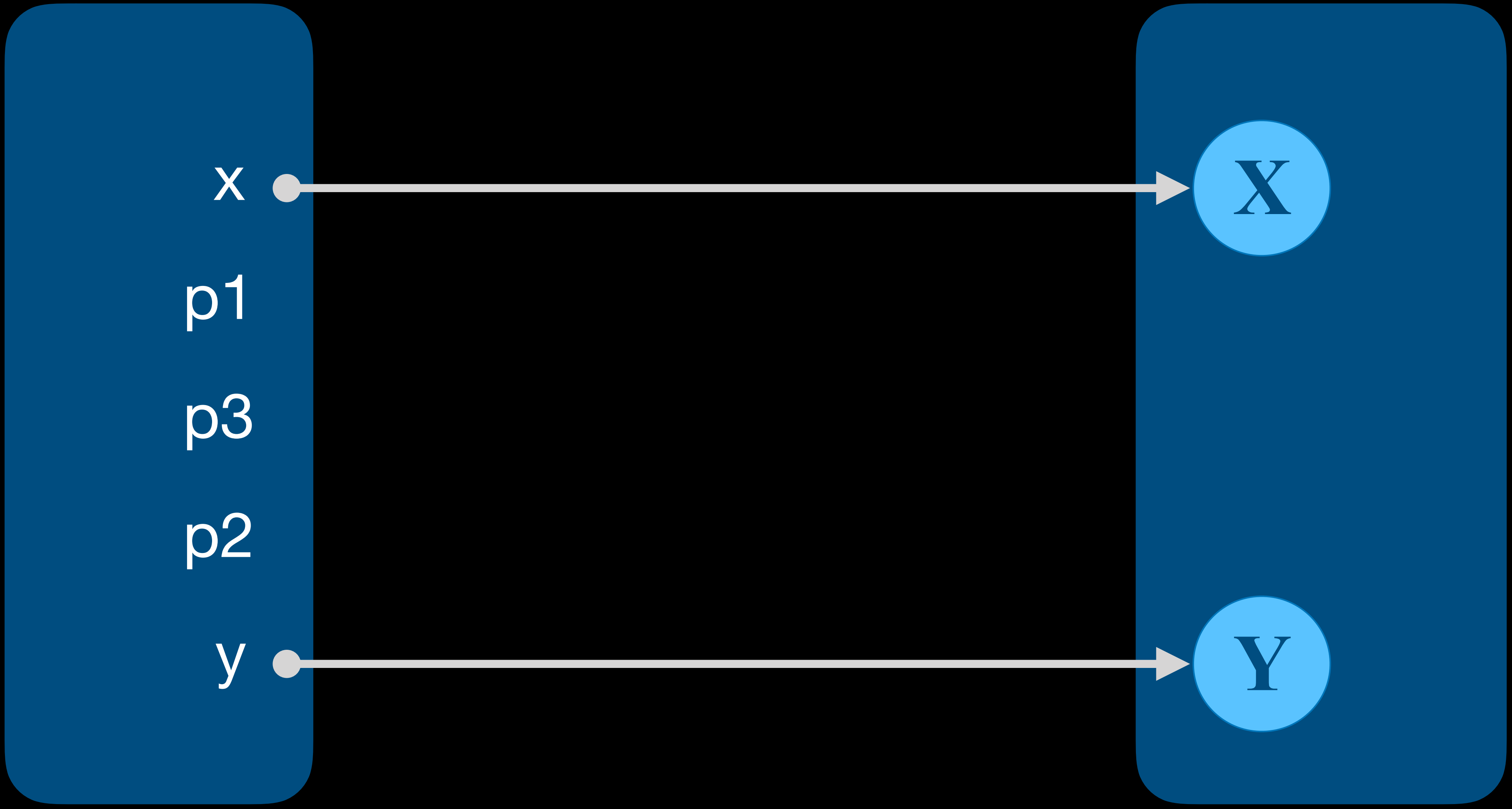


```
const t3 = (x.a()).c(y.b());
```

```
const p3 = (await (await x).a()).c((await y).b());
```

```
const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

```
const p1 = (await x).a();
const p2 = (await y).b();
const p3 = (await p1).c((await p2));
```

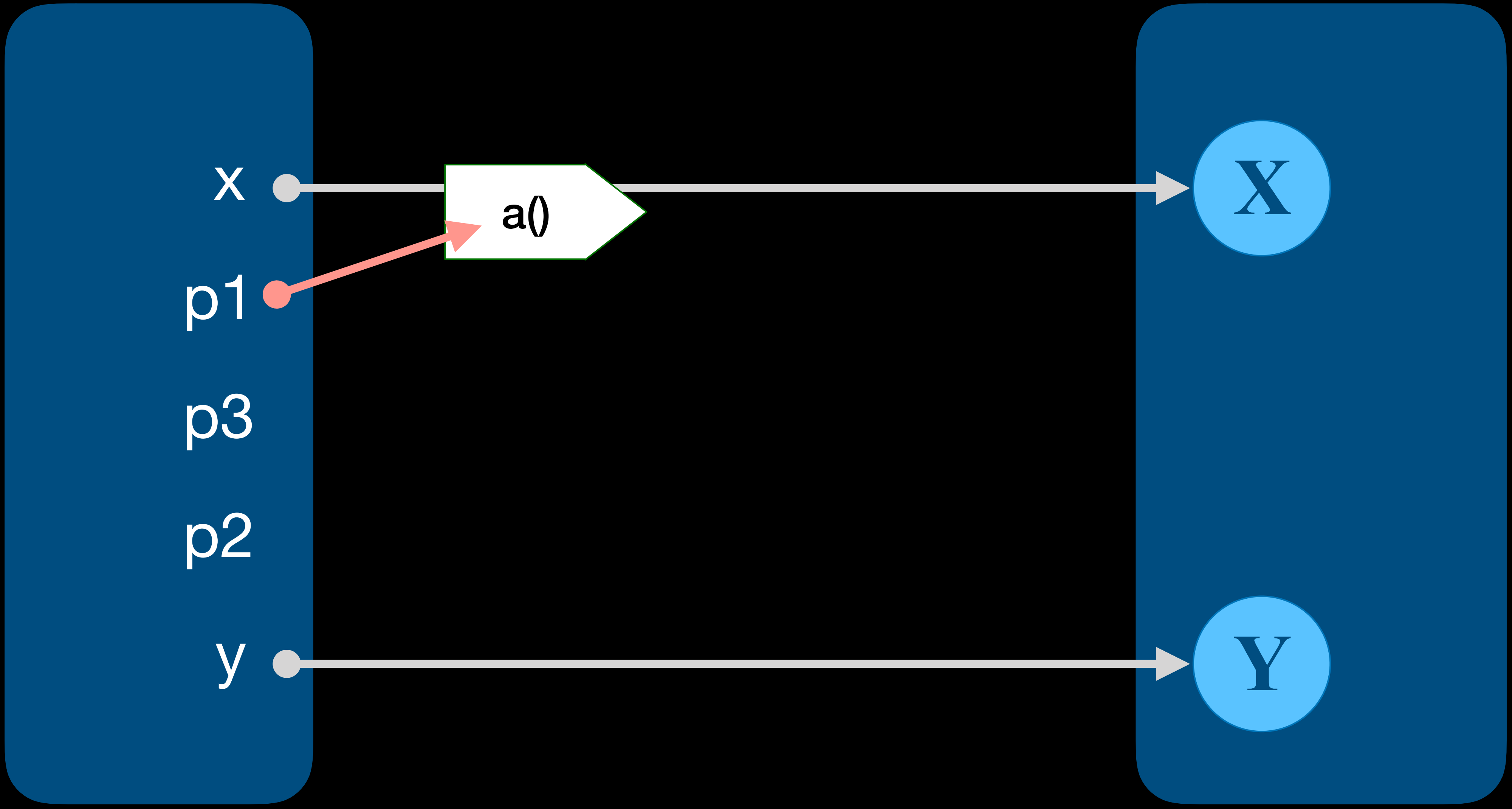


```
const t3 = (x.a()).c(y.b());

const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

```
const p3 = (await (await x).a()).c((await y).b());

const p1 = (await x).a();
const p2 = (await y).b();
const p3 = (await p1).c((await p2));
```

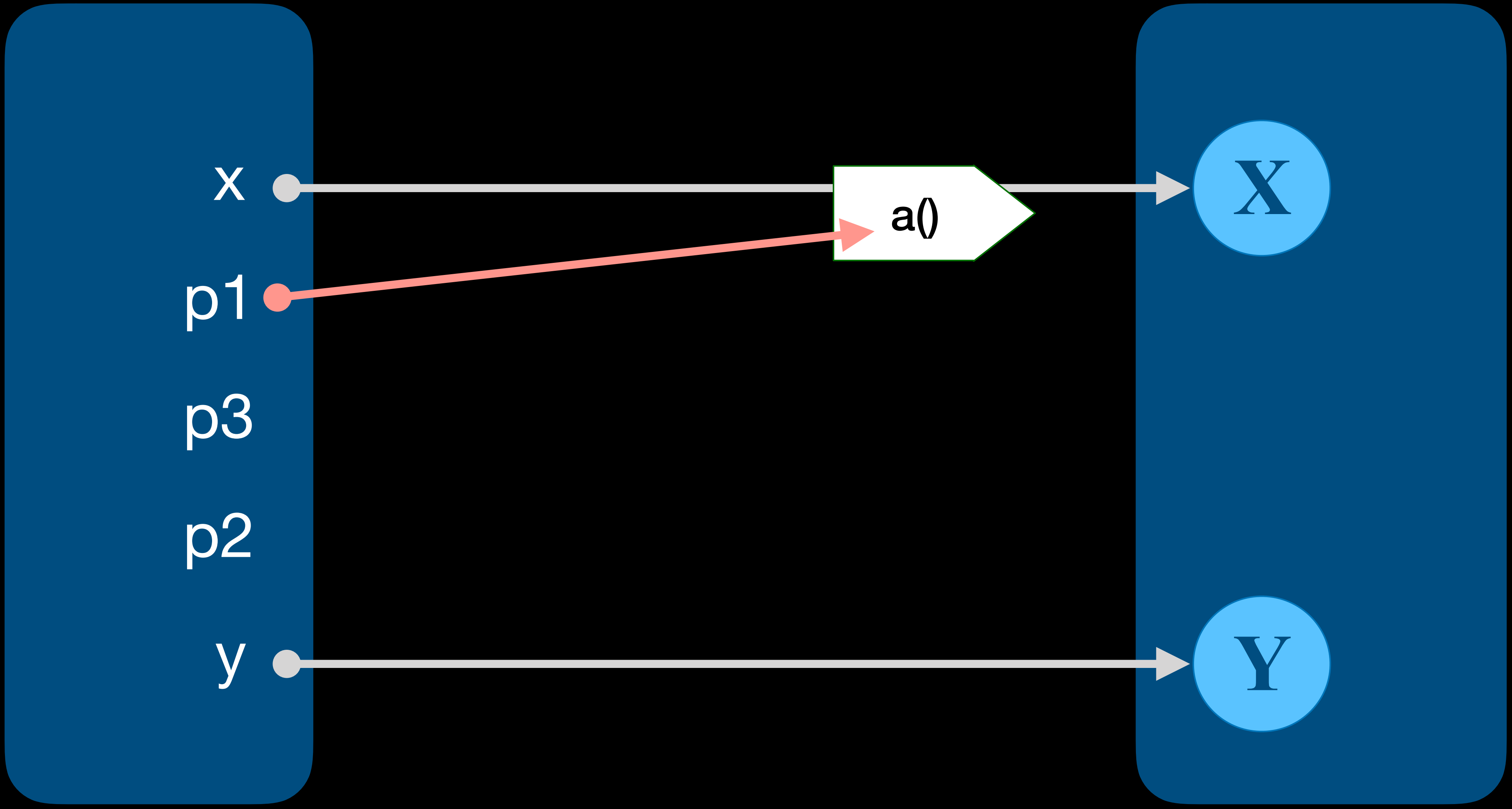


```
const t3 = (x.a()).c(y.b());
```

```
const p3 = (await (await x).a()).c((await y).b());
```

```
const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

```
const p1 = (await x).a();
const p2 = (await y).b();
const p3 = (await p1).c((await p2));
```

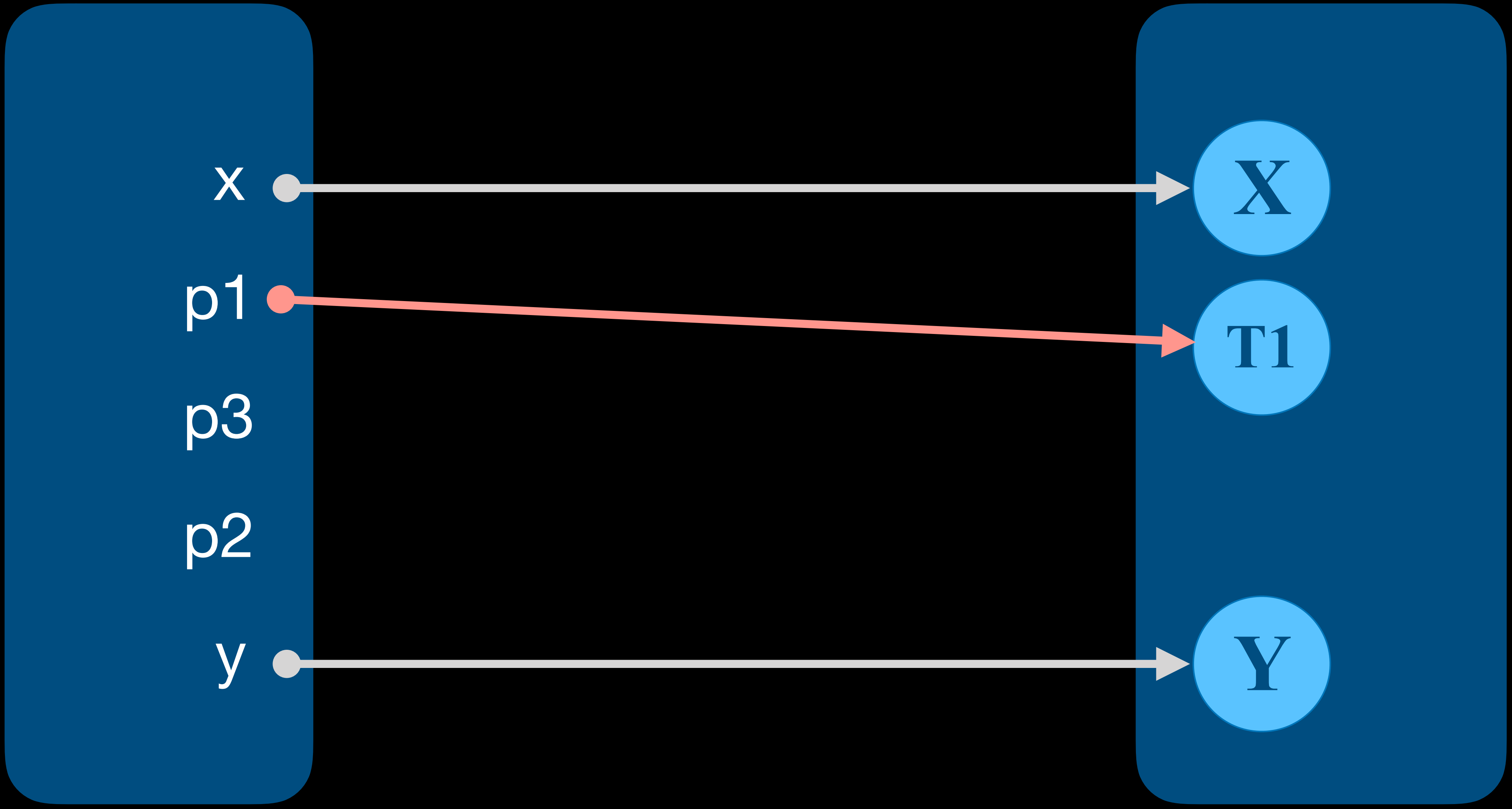


```
const t3 = (x.a()).c(y.b());
```

```
const p3 = (await (await x).a()).c((await y).b());
```

```
const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

```
const p1 = (await x).a();
const p2 = (await y).b();
const p3 = (await p1).c((await p2));
```

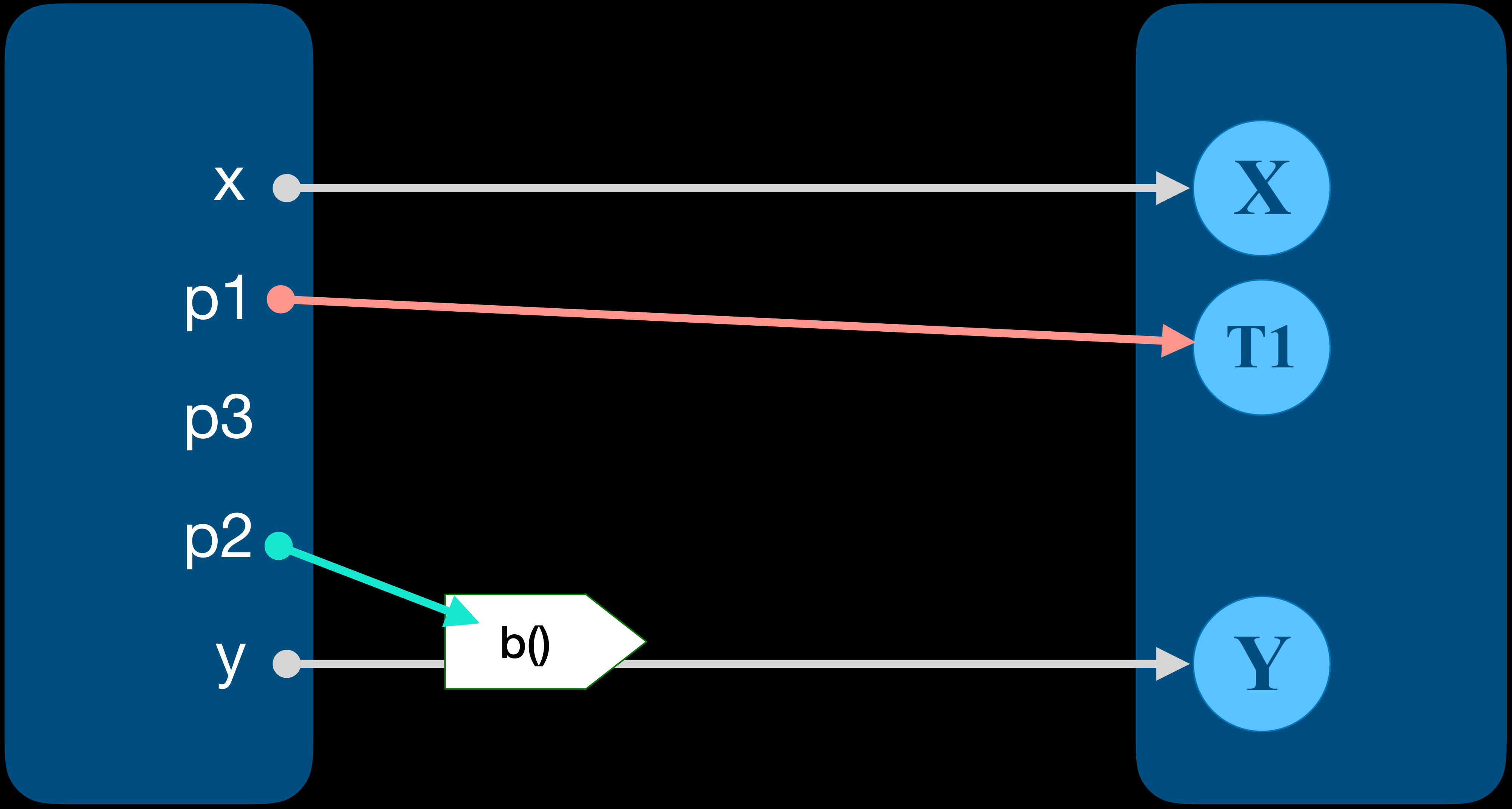


```
const t3 = (x.a()).c(y.b());

const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

```
const p3 = (await (await x).a()).c((await y).b());

const p1 = (await x).a();
const p2 = (await y).b();
const p3 = (await p1).c((await p2));
```

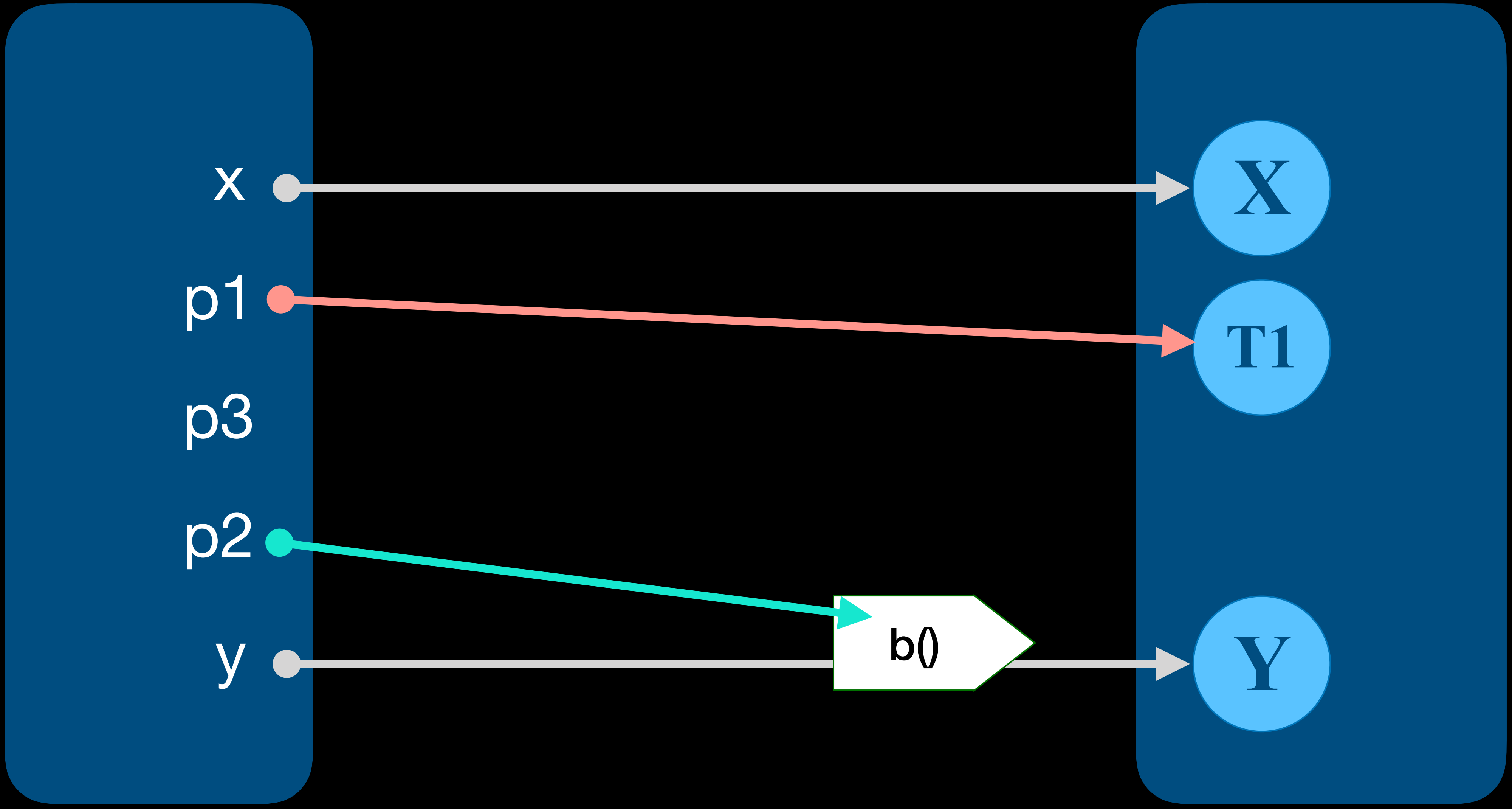


```
const t3 = (x.a()).c(y.b());
```

```
const p3 = (await (await x).a()).c((await y).b());
```

```
const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

```
const p1 = (await x).a();
const p2 = (await y).b();
const p3 = (await p1).c((await p2));
```

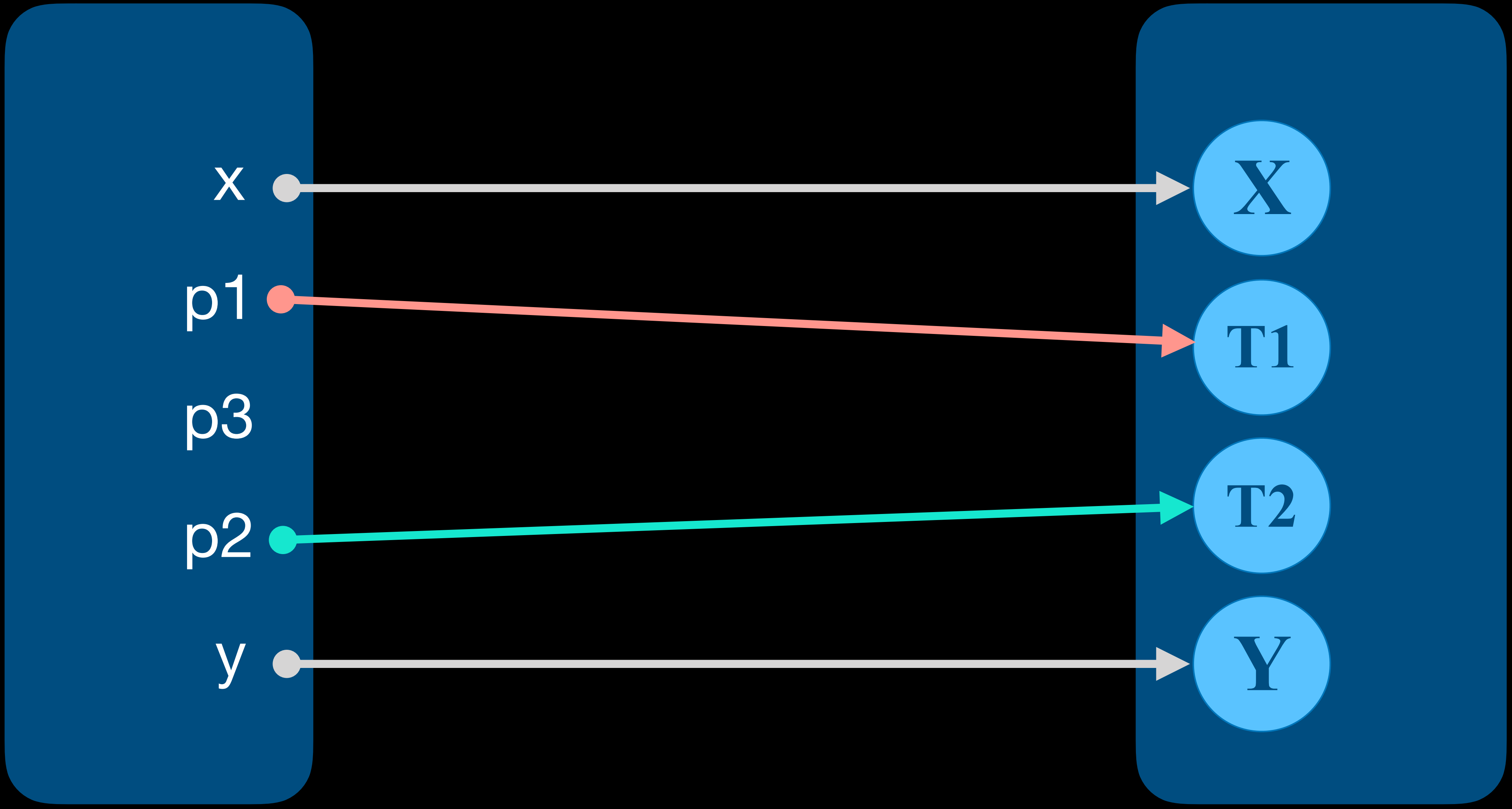


```
const t3 = (x.a()).c(y.b());

const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

```
const p3 = (await (await x).a()).c((await y).b());

const p1 = (await x).a();
const p2 = (await y).b();
const p3 = (await p1).c((await p2));
```

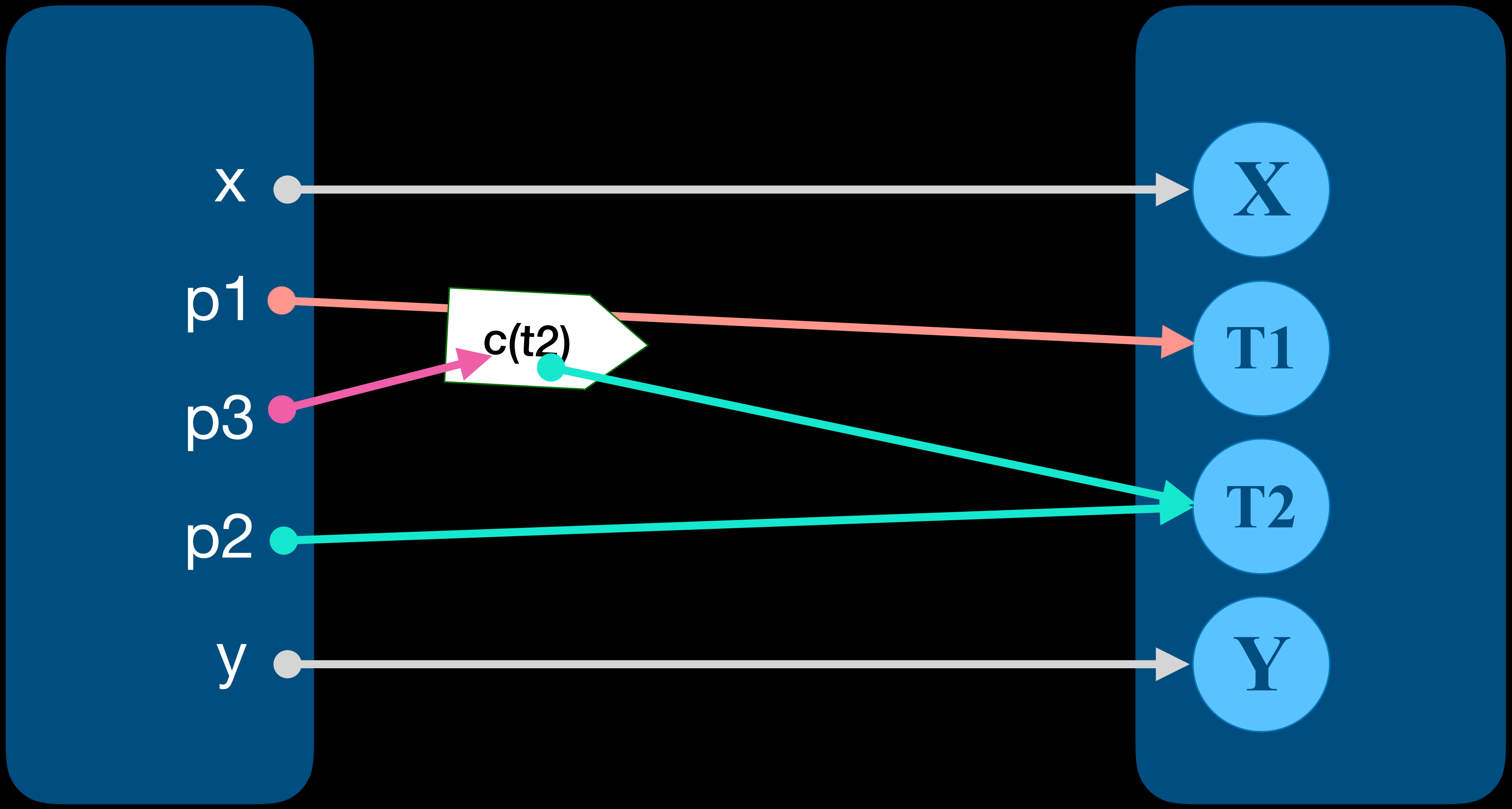


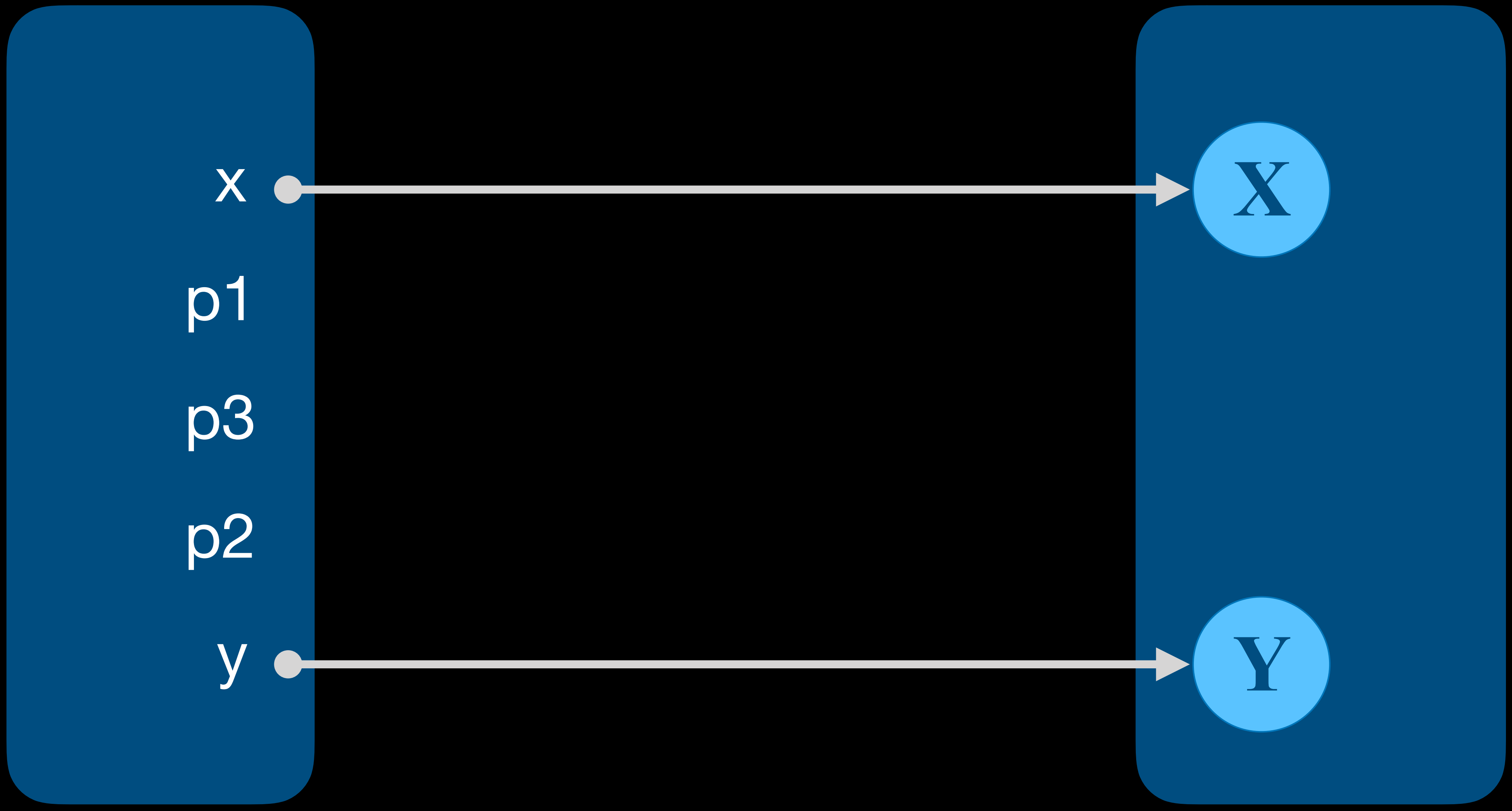
```
const t3 = (x.a()).c(y.b());
```

```
const p3 = (await (await x).a()).c((await y).b());
```

```
const t1 = x.a();
const t2 = y.b();
const t3 = t1.c(t2);
```

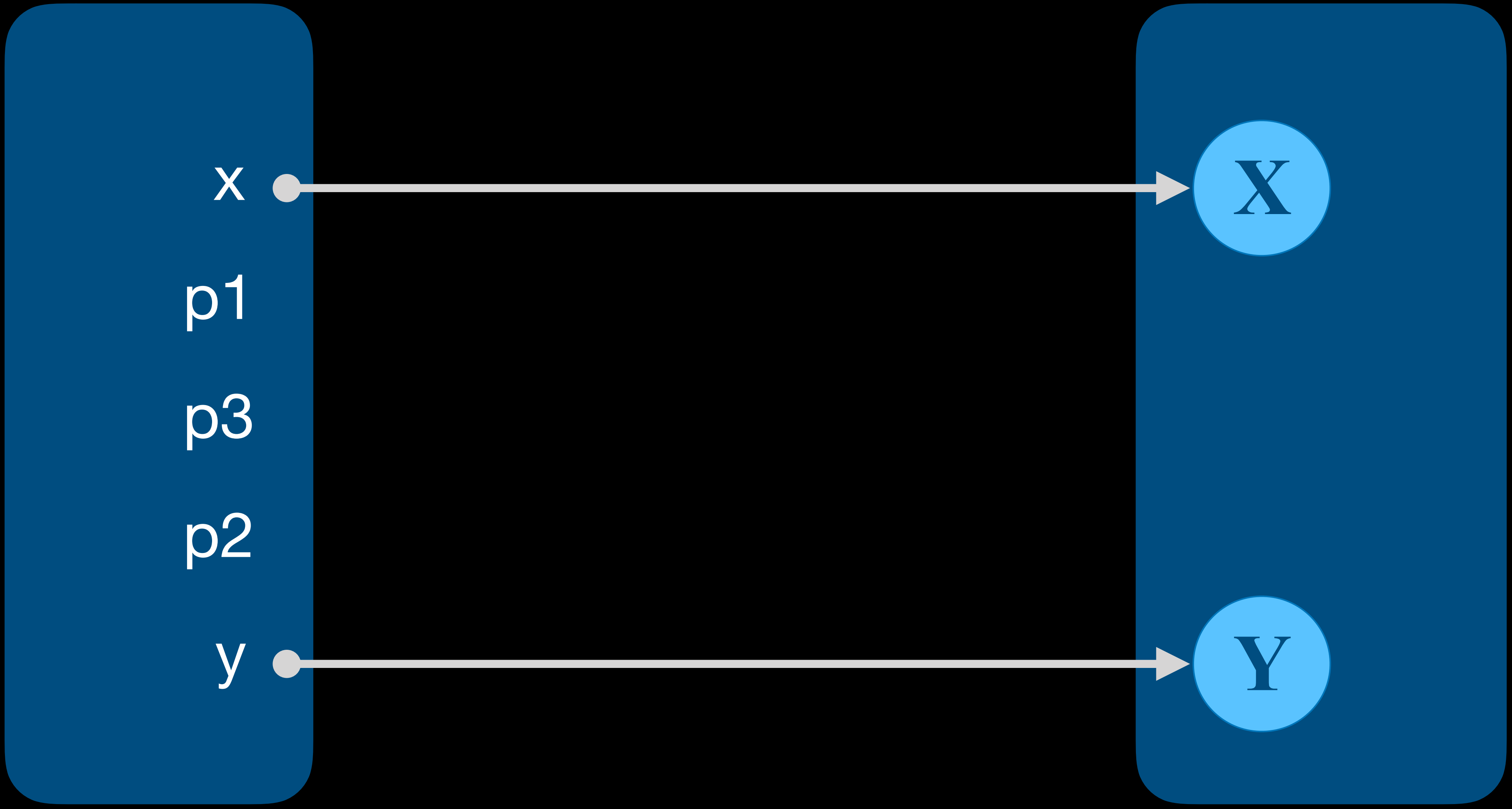
```
const p1 = (await x).a();
const p2 = (await y).b();
const p3 = (await p1).c((await p2));
```




x
p1
p3
p2
y
X
Y

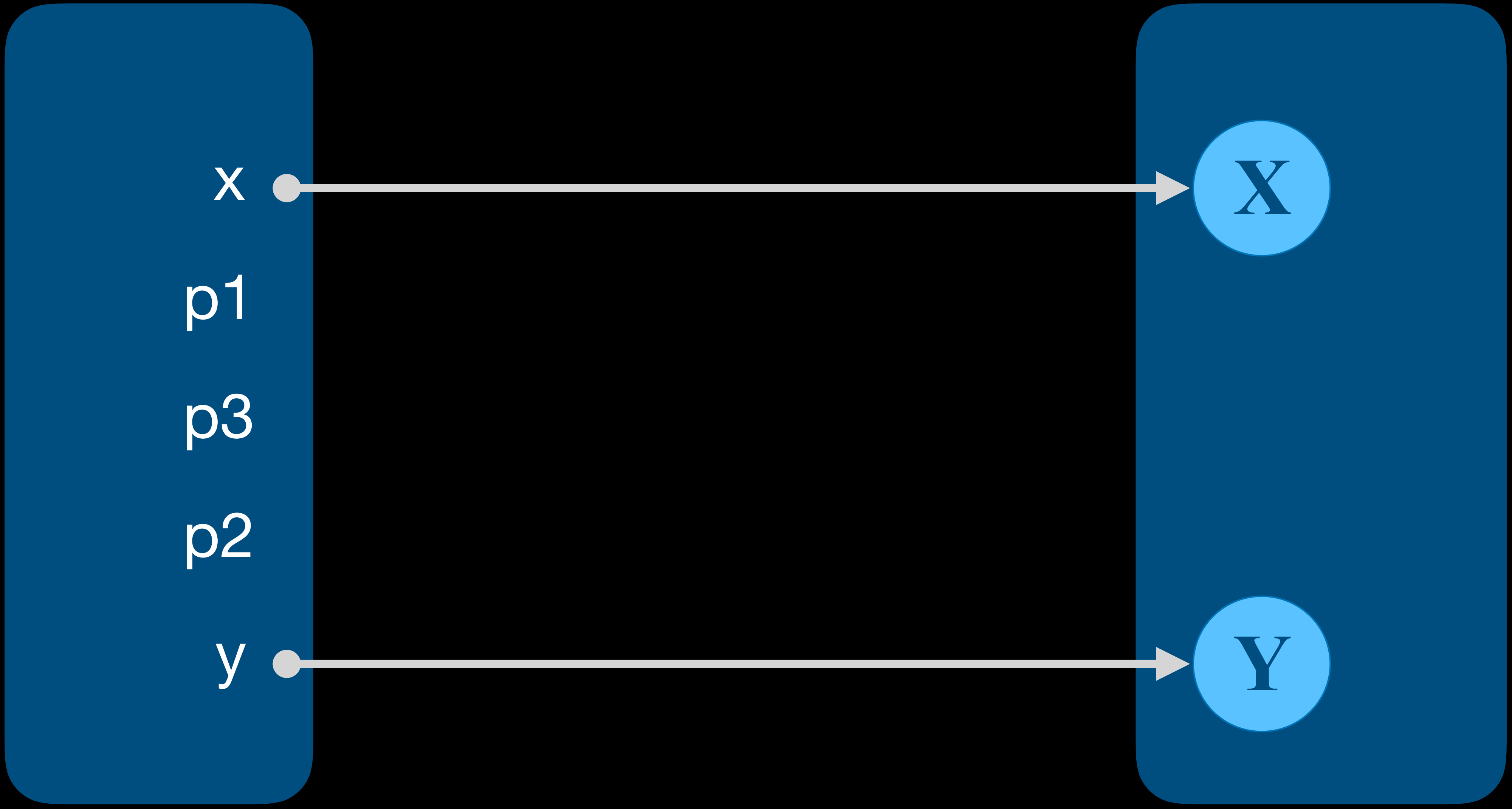
```
const p3 = E(E(x).a()).c(E(y).b());
```

```
const p1 = E(x).a();
const p2 = E(y).b();
const p3 = E(p1).c(p2);
```



```
const p3 = E(E(x).a()).c(E(y).b());        const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = E(x).a();                       const p1 = x ~. a();
const p2 = E(y).b();                       const p2 = y ~. b();
const p3 = E(p1).c(p2);                    const p3 = p1 ~. c(p2);
```

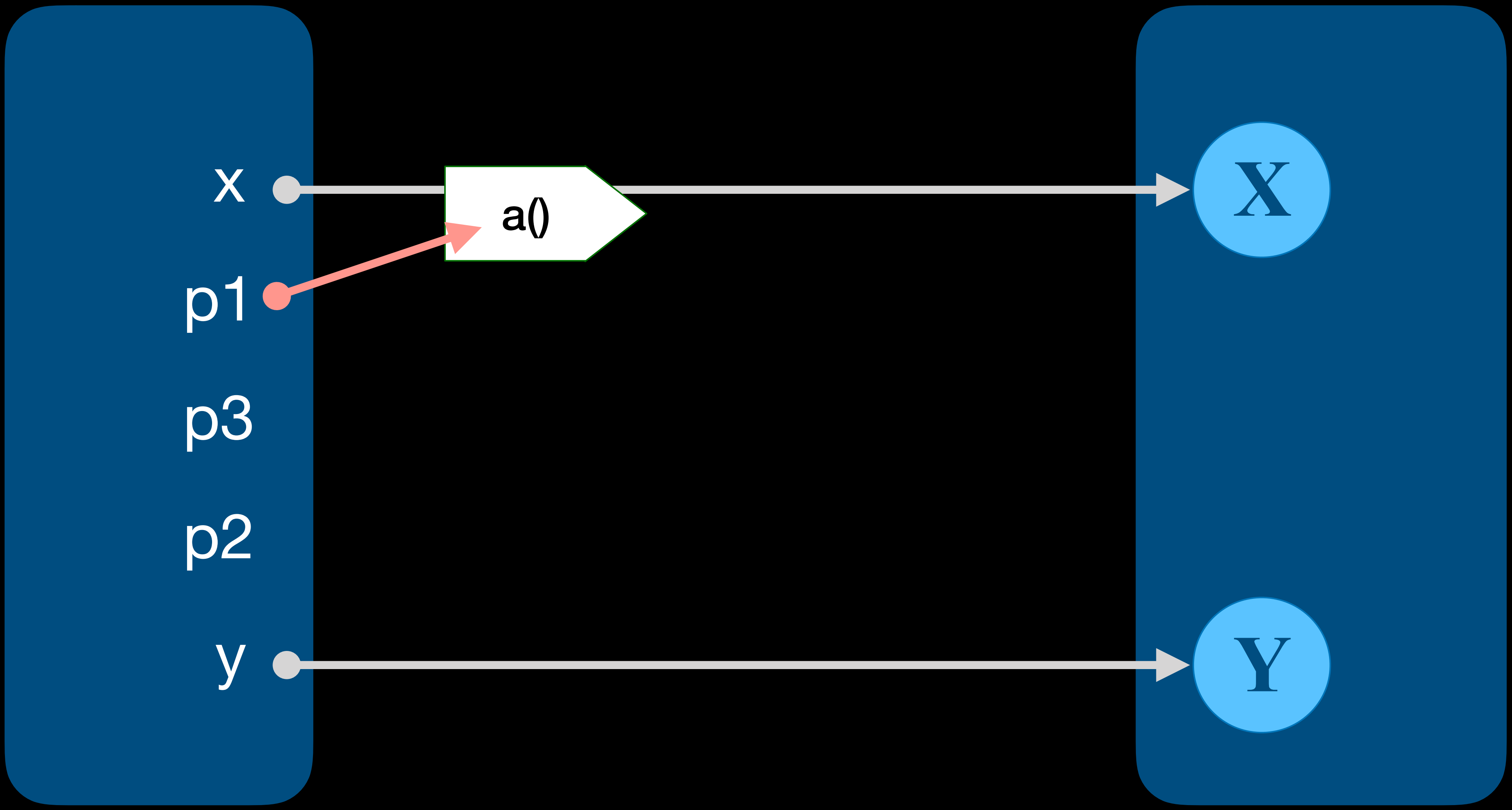


```
const p3 = E(E(x).a()).c(E(y).b());

const p1 = E(x).a();
const p2 = E(y).b();
const p3 = E(p1).c(p2);
```

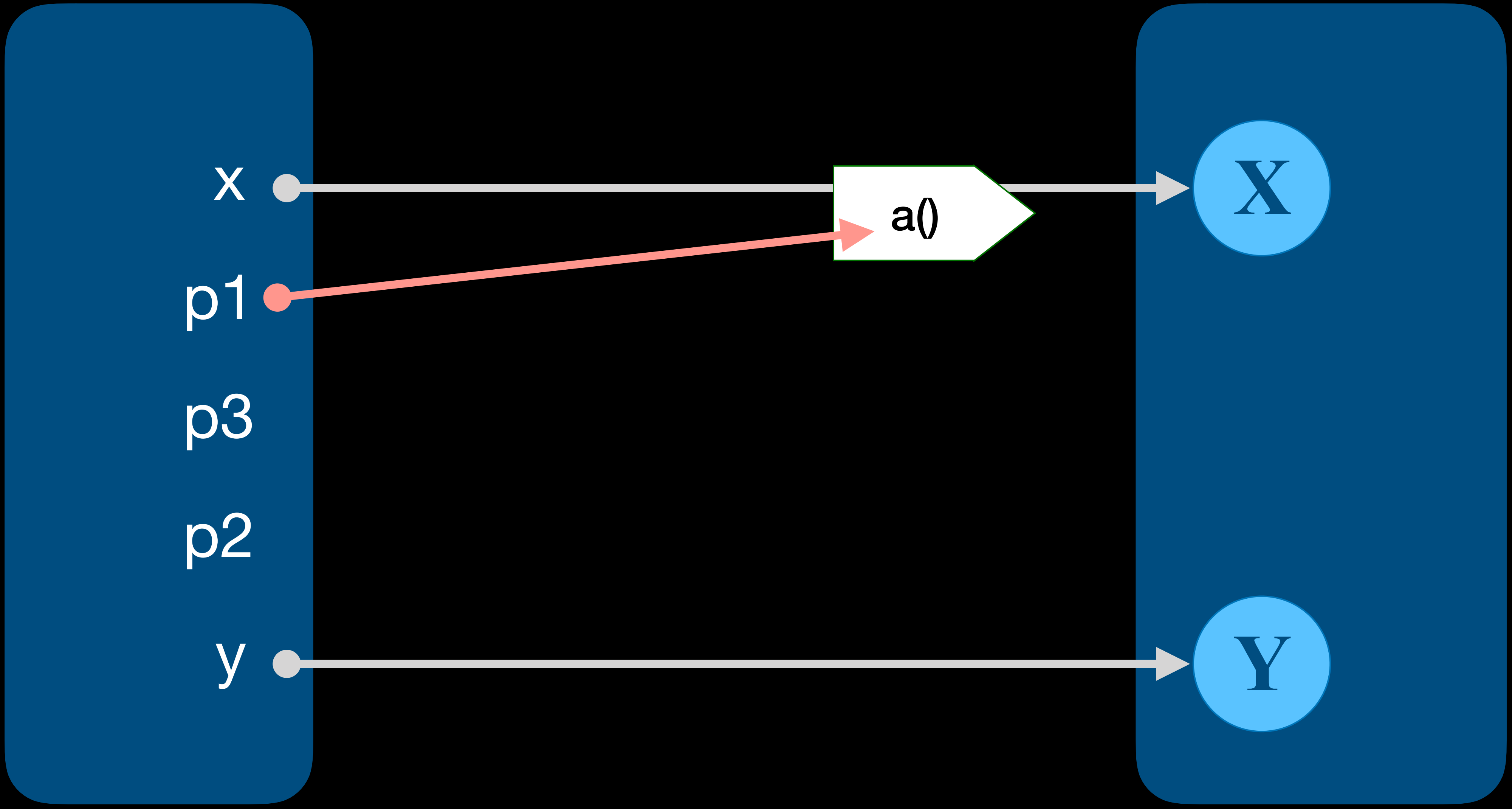
```
const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = x ~. a();
const p2 = y ~. b();
const p3 = p1 ~. c(p2);
```



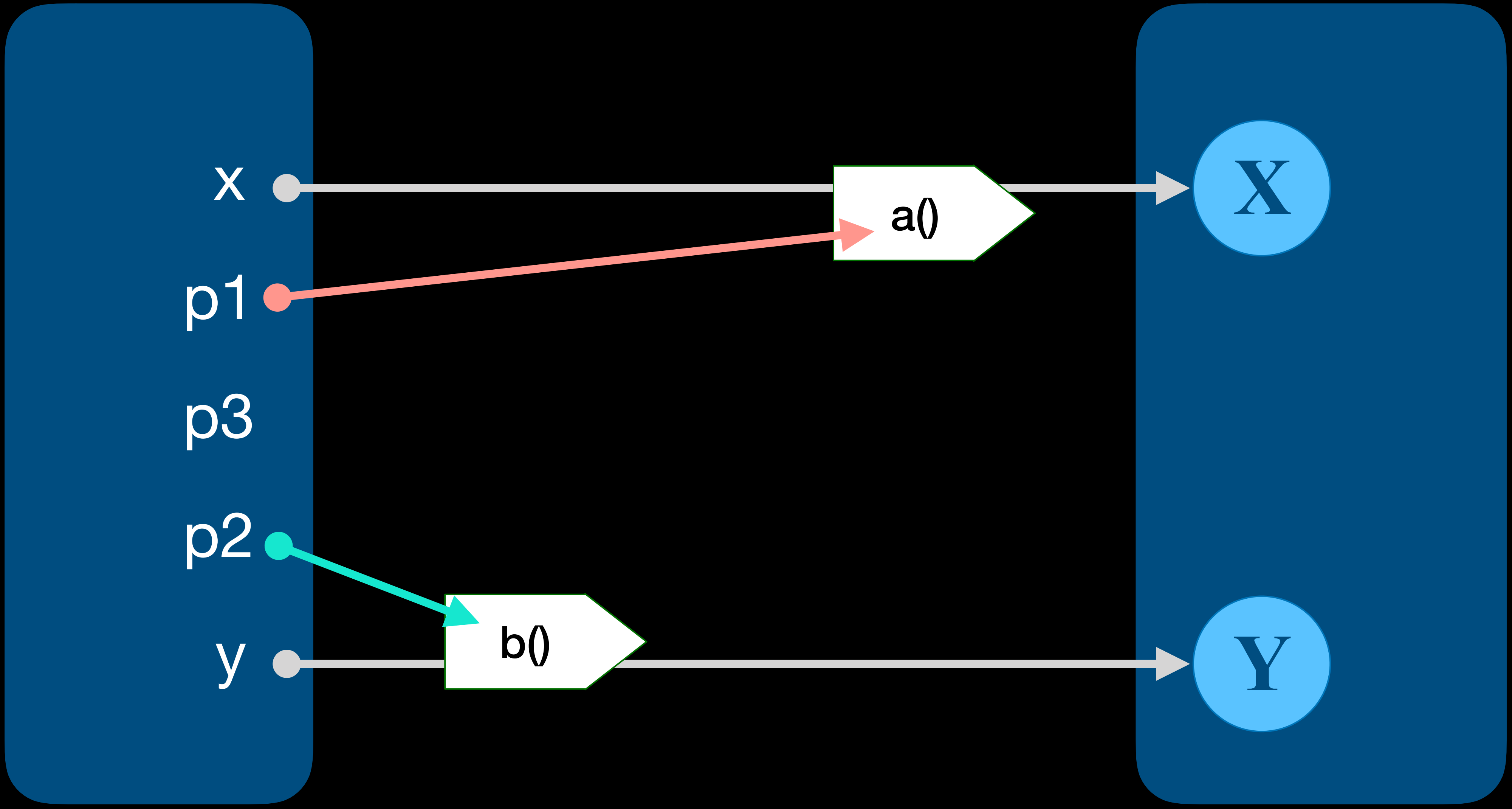
```
const p3 = E(E(x).a()).c(E(y).b());        const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = E(x).a();                       const p1 = x ~. a();
const p2 = E(y).b();                       const p2 = y ~. b();
const p3 = E(p1).c(p2);                    const p3 = p1 ~. c(p2);
```



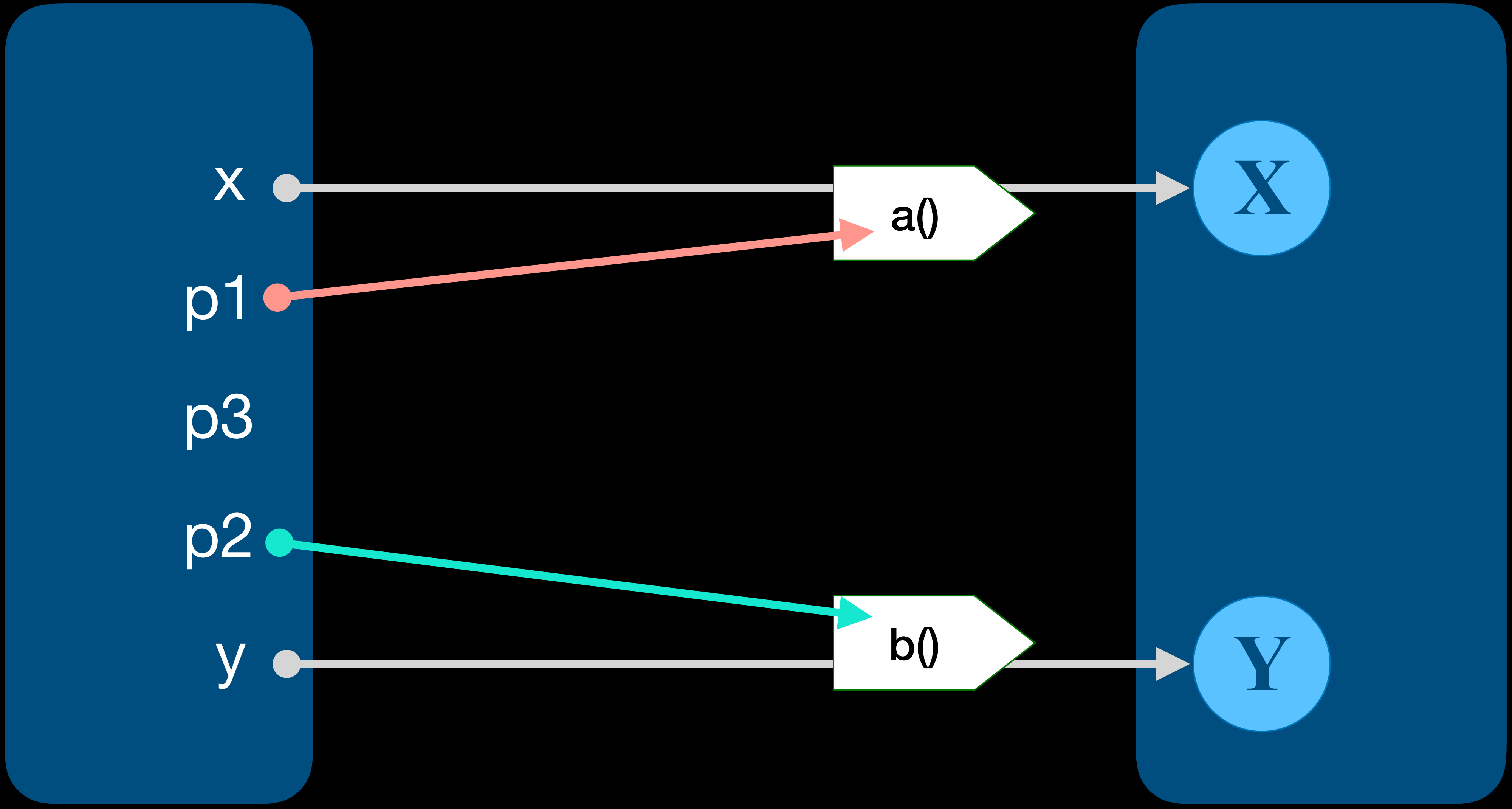
```
const p3 = E(E(x).a()).c(E(y).b());       const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = E(x).a();                      const p1 = x ~. a();
const p2 = E(y).b();                      const p2 = y ~. b();
const p3 = E(p1).c(p2);                   const p3 = p1 ~. c(p2);
```



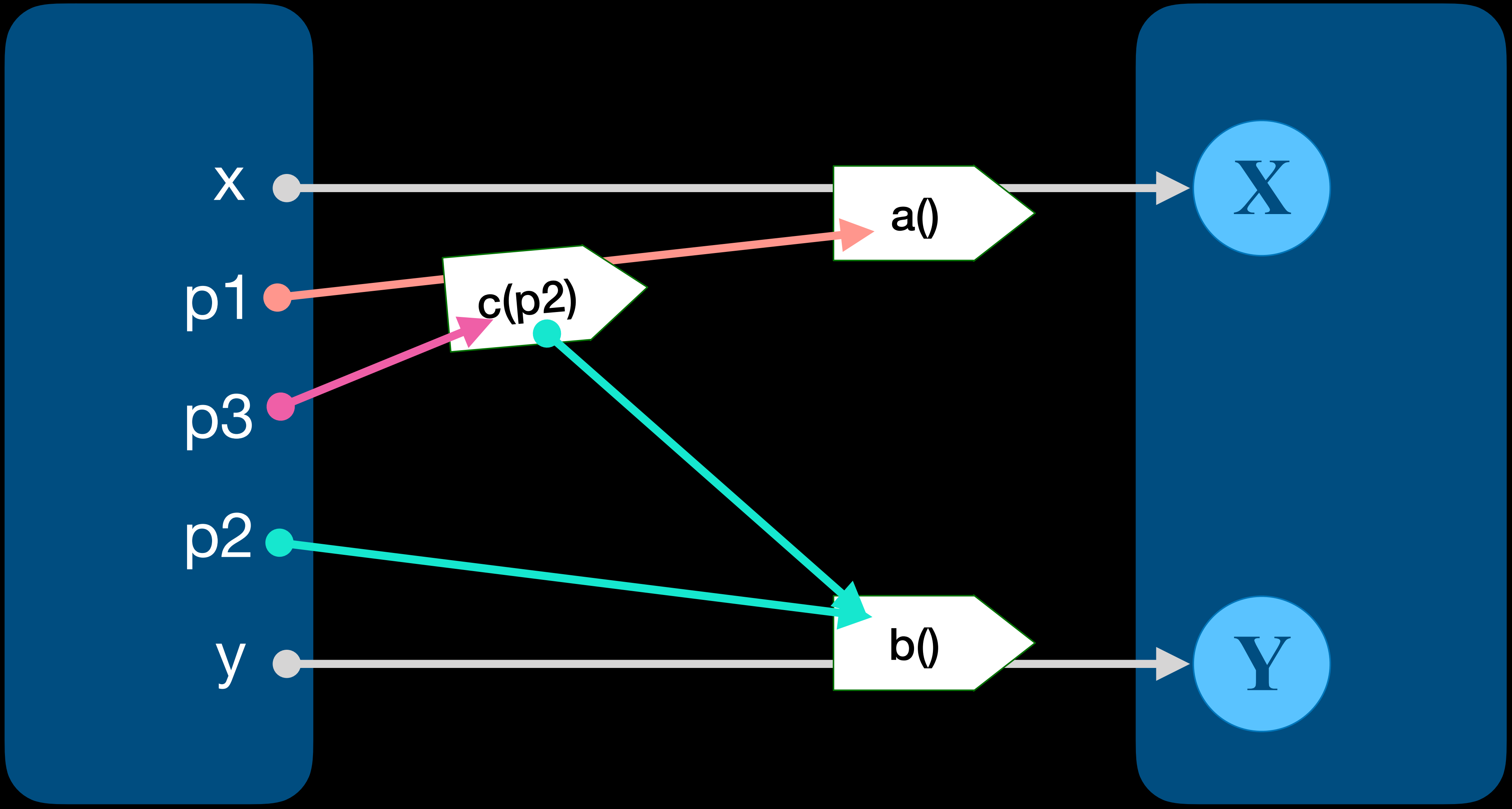
```
const p3 = E(E(x).a()).c(E(y).b());        const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = E(x).a();                       const p1 = x ~. a();
const p2 = E(y).b();                       const p2 = y ~. b();
const p3 = E(p1).c(p2);                    const p3 = p1 ~. c(p2);
```



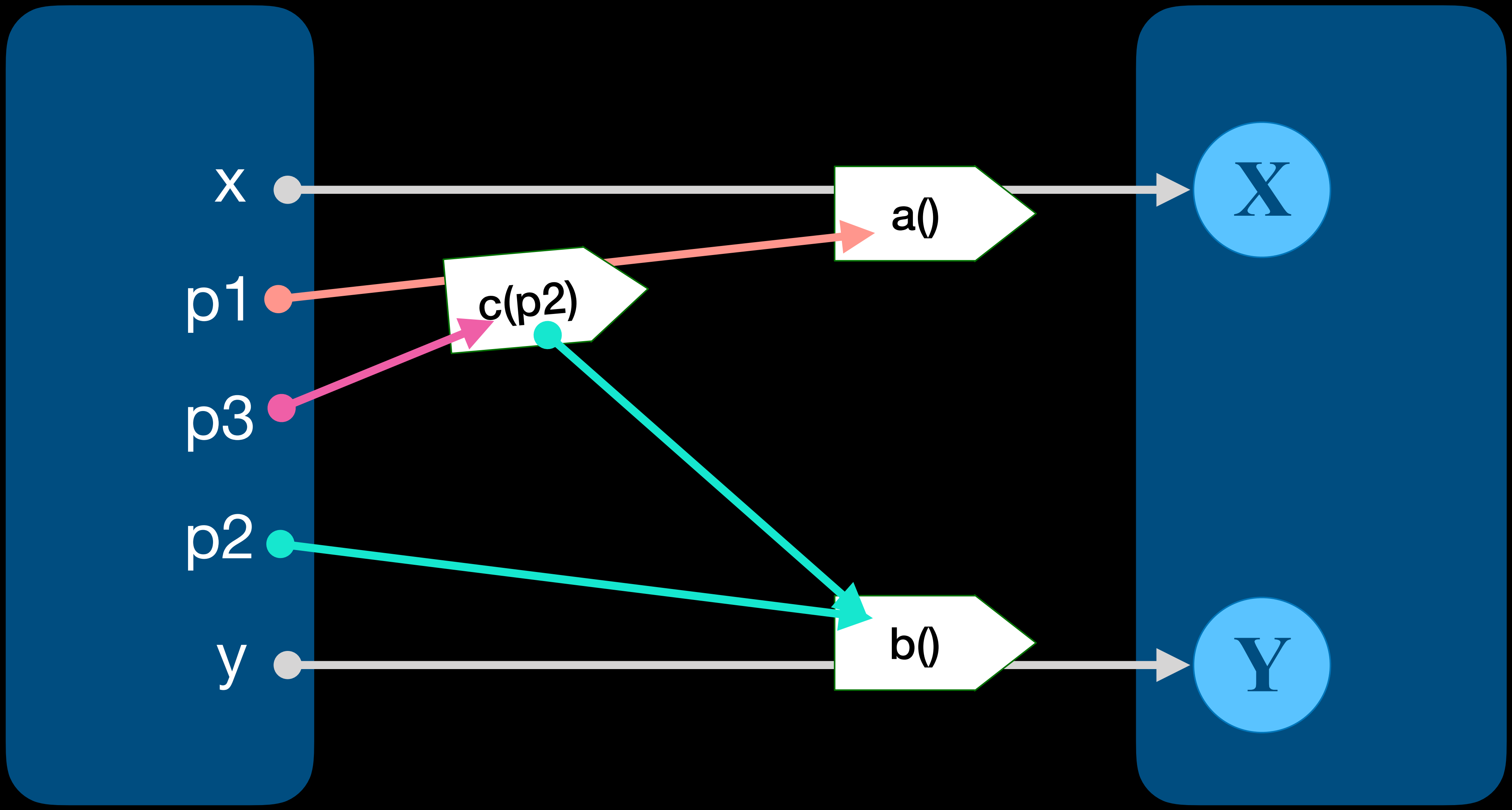
```
const p3 = E(E(x).a()).c(E(y).b());      const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = E(x).a();                     const p1 = x ~. a();
const p2 = E(y).b();                     const p2 = y ~. b();
const p3 = E(p1).c(p2);                  const p3 = p1 ~. c(p2);
```



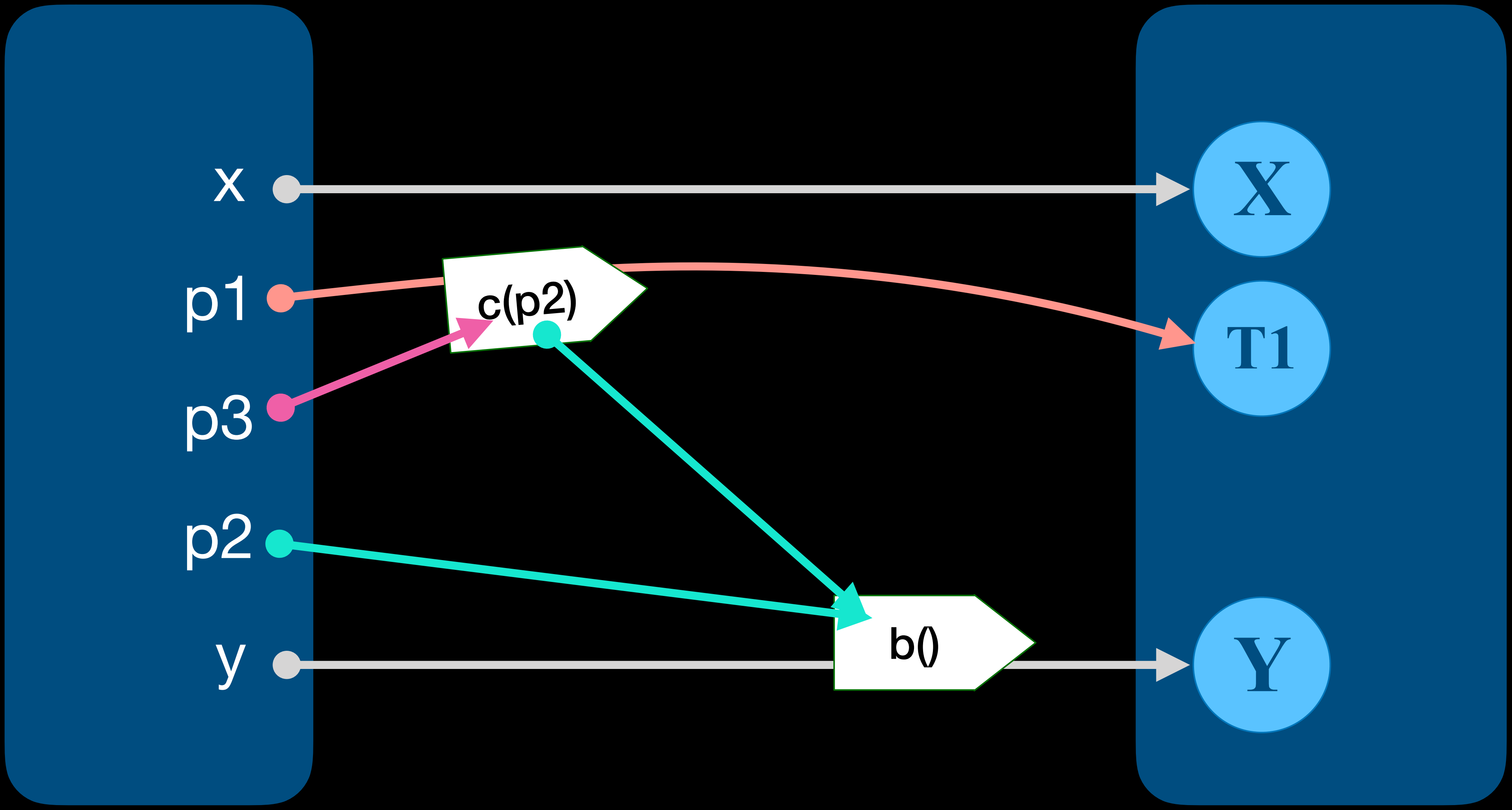
```
const p3 = E(E(x).a()).c(E(y).b());        const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = E(x).a();                       const p1 = x ~. a();
const p2 = E(y).b();                       const p2 = y ~. b();
const p3 = E(p1).c(p2);                    const p3 = p1 ~. c(p2);
```



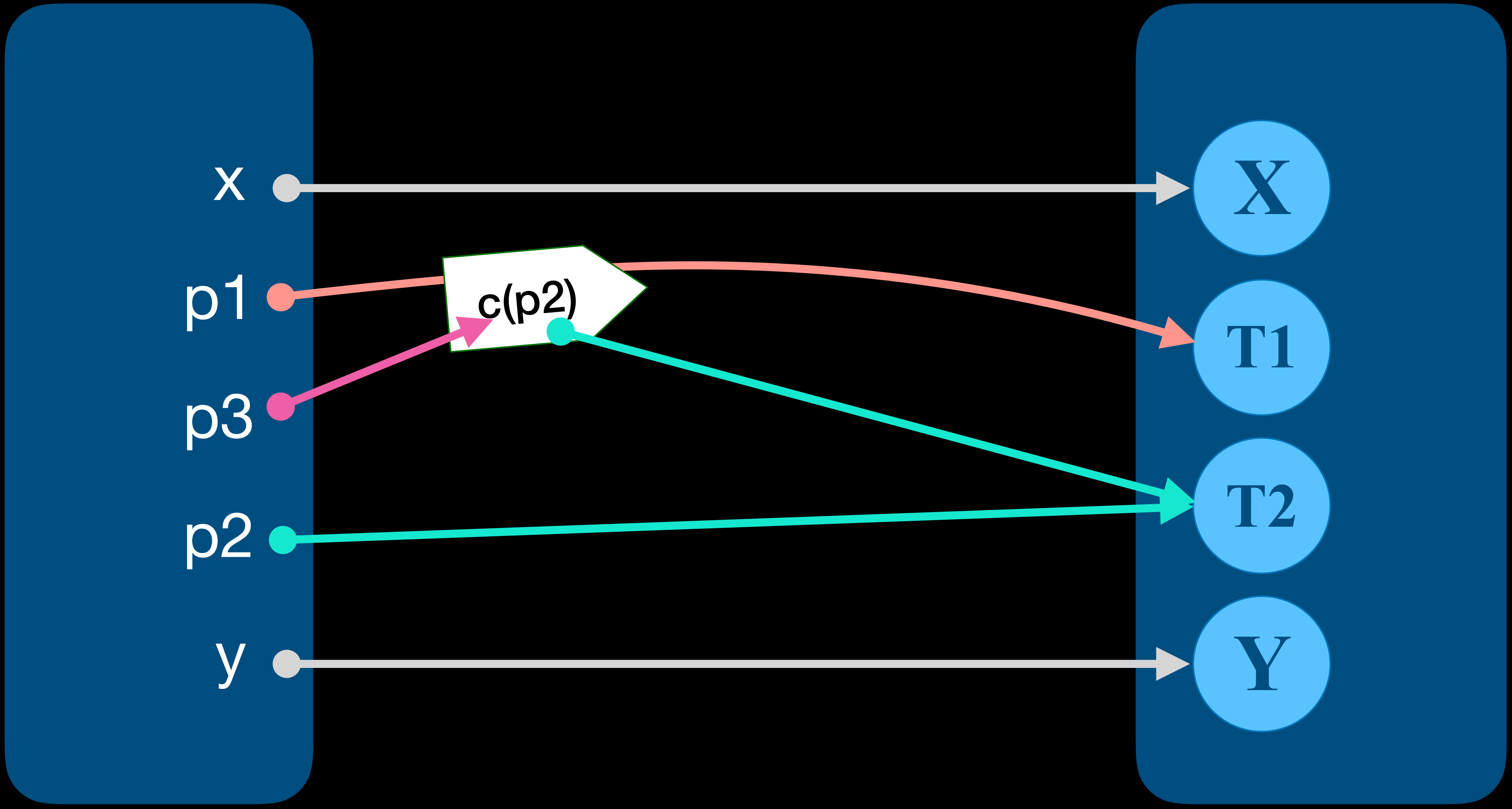
```
const p3 = E(E(x).a()).c(E(y).b());        const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = E(x).a();                       const p1 = x ~. a();
const p2 = E(y).b();                       const p2 = y ~. b();
const p3 = E(p1).c(p2);                    const p3 = p1 ~. c(p2);
```



```
const p3 = E(E(x).a()).c(E(y).b());      const p3 = (x ~. a()) ~. c(y ~. b());

const p1 = E(x).a();                     const p1 = x ~. a();
const p2 = E(y).b();                     const p2 = y ~. b();
const p3 = E(p1).c(p2);                  const p3 = p1 ~. c(p2);
```



Internal Method	Static Method
`p.[[GetSend]](prop)`	`get(p, prop)`
`p.[[HasSend]](prop)`	`has(p, prop)`
`p.[[SetSend]](prop, value)`	`set(p, prop, value)`
`p.[[DeleteSend]](prop)`	`delete(p, prop)`
`p.[[ApplyFunctionSend]](args)`	`applyFunction(p, args)`
`p.[[ApplyMethodSend]](prop, args)`	`applyMethod(p, prop, args)`

Static Method	Default Behavior	Handler trap
`get(p, prop)`	`p.then(t => t[prop])`	`h.get(t, prop)`
`has(p, prop)`	`p.then(t => prop in t)`	`h.has(t, prop)`
`set(p, prop, value)`	`p.then(t => (t[prop] = value))`	`h.set(t, prop, value)`
`delete(p, prop)`	`p.then(t => delete t[prop])`	`h.delete(t, prop)`
`applyFunction(p, args)`	`p.then(t => t(...args))`	`h.applyFunction(t, args)`
`applyMethod(p, prop, args)`	`p.then(t => t[prop](...args))`	`h.applyMethod(t, prop, args)`

Internal Method	Static Method
`p.[[GetSendOnly]](prop)`	`getSendOnly(p, prop)`
`p.[[HasSendOnly]](prop)`	`hasSendOnly(p, prop)`
`p.[[SetSendOnly]](prop, value)`	`setSendOnly(p, prop, value)`
`p.[[DeleteSendOnly]](prop)`	`deleteSendOnly(p, prop)`
`p.[[ApplyFunctionSendOnly]](args)`	`applyFunctionSendOnly(p, args)`
`p.[[ApplyMethodSendOnly]](prop, args)`	`applyMethodSendOnly(p, prop, args)`

Static Method	Handler trap
`getSendOnly(p, prop)`	`h.getSendOnly(t, prop)`
`hasSendOnly(p, prop)`	`h.hasSendOnly(t, prop)`
`setSendOnly(p, prop, value)`	`h.setSendOnly(t, prop, value)`
`deleteSendOnly(p, prop)`	`h.deleteSendOnly(t, prop)`
`applyFunctionSendOnly(p, args)`	`h.applyFunctionSendOnly(t, args)`
`applyMethodSendOnly(p, prop, args)`	`h.applyMethodSendOnly(t, prop, args)`

```
get                    (target, prop):        Promise<result>,
getSendOnly            (target, prop):        void,
has                    (target, prop):        Promise<boolean>,
hasSendOnly            (target, prop):        void,
set                    (target, prop, value): Promise<boolean>,
setSendOnly            (target, prop, value): void,
delete                 (target, prop):        Promise<boolean>,
deleteSendOnly         (target, prop):        void,
applyFunction          (target, args):        Promise<result>,
applyFunctionSendOnly  (target, args):        void,
applyMethod            (target, prop, args):  Promise<result>,
applyMethodSendOnly    (target, prop, args):  void,
new Promise((resolve, reject) => {...}
           ) -> unhandled promise
resolve(resolution) -> void
reject(reason) -> void
```

```
new Promise((resolve, reject) => {...}
           ) -> unhandled promise
resolve(resolution) -> void
reject(reason) -> void


new HandledPromise((resolve, reject, resolveWithPresence) => {...},
                   unfulfilledHandler)
                  ) -> handled promise
resolve(resolution) -> void
reject(reason) -> void
resolveWithPresence(presenceHandler) -> fresh presence
```

Cannot be shimmed!

```
let pr;
const p = new Promise(r => pr = r);
E(p).foo();
let qr;
const q = new HandledPromise(r => qr = r,
                             unfulfilledHandler);
pr.resolve(q);
```

Questions?