Horton's *Who Done It?*

Communicating Authority with Responsibility Tracking

Mark S. Miller Google Research[1]

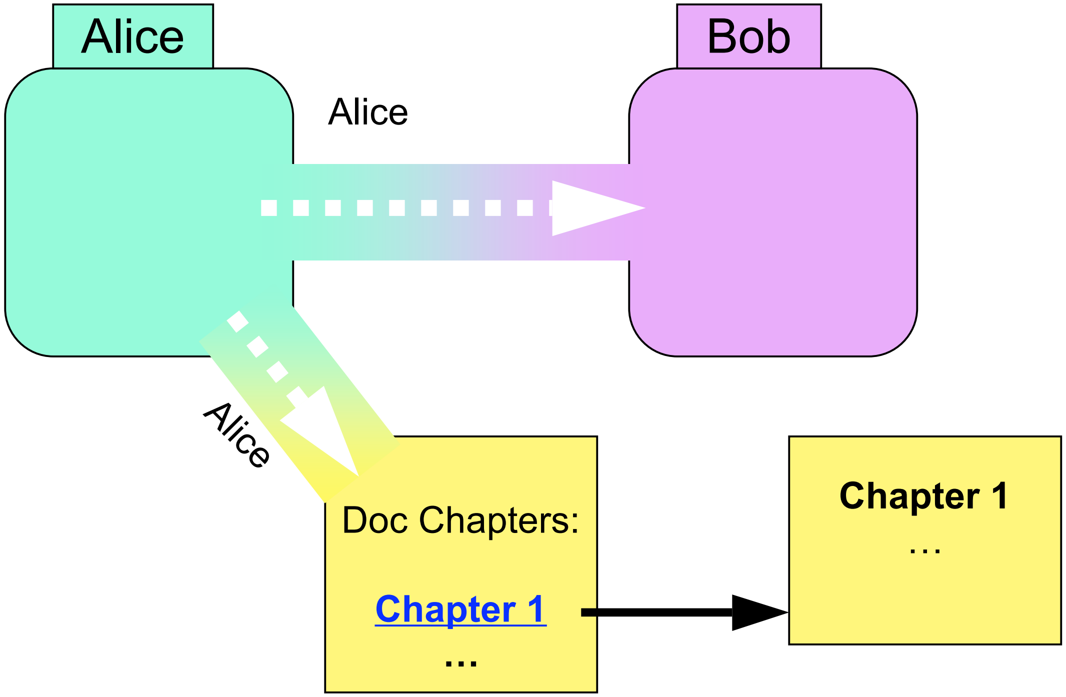
Jed Donnelley LBNL/NERSC

Alan H. Karp HP Labs

Usenix HotSec Workshop, August 7, 2007

[1] Work done while at HP Labs

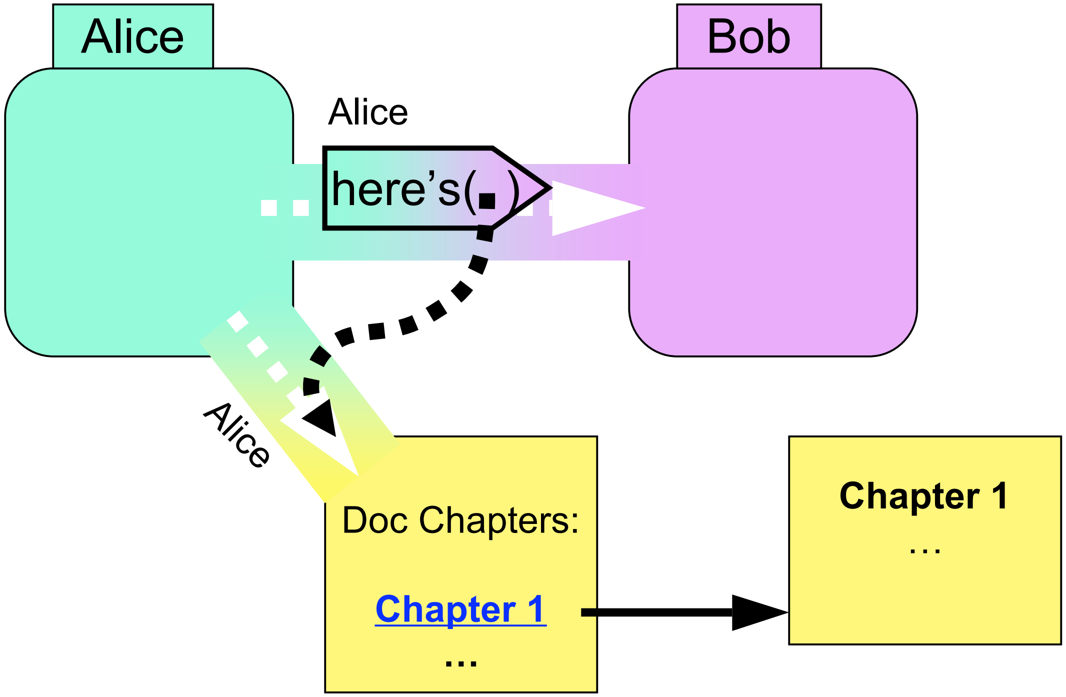
Communicating Object Access with Delegation



Initial Conditions:

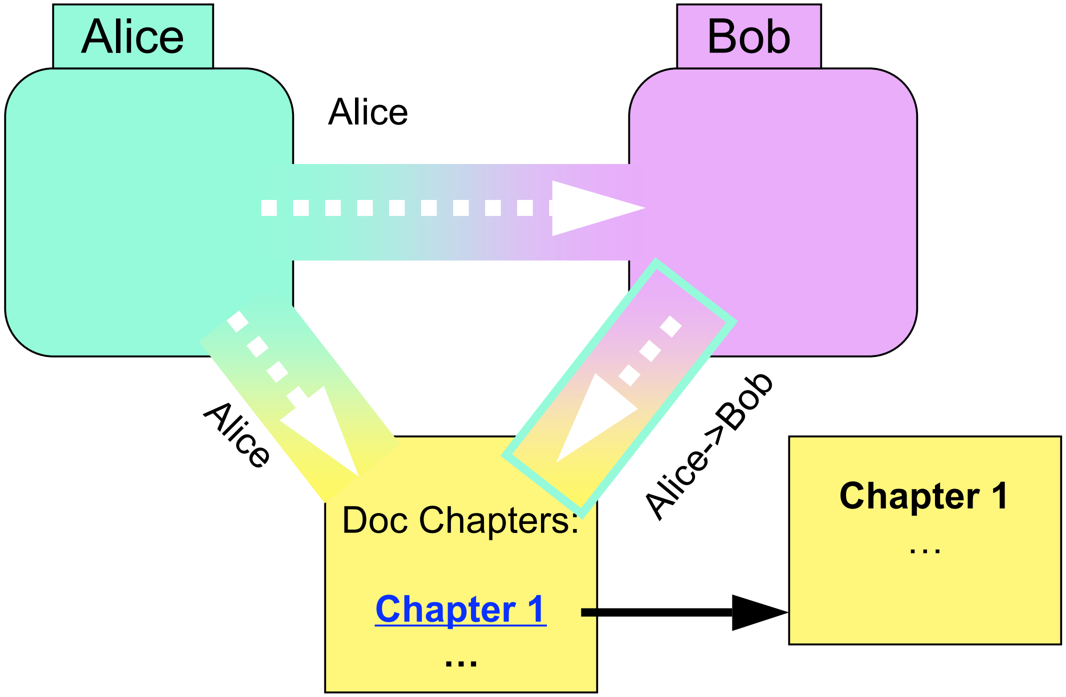
Alice has: 1. A capability to send to Bob and
2. A capability to a document with chapters.

Capability Communication of the Document Reference



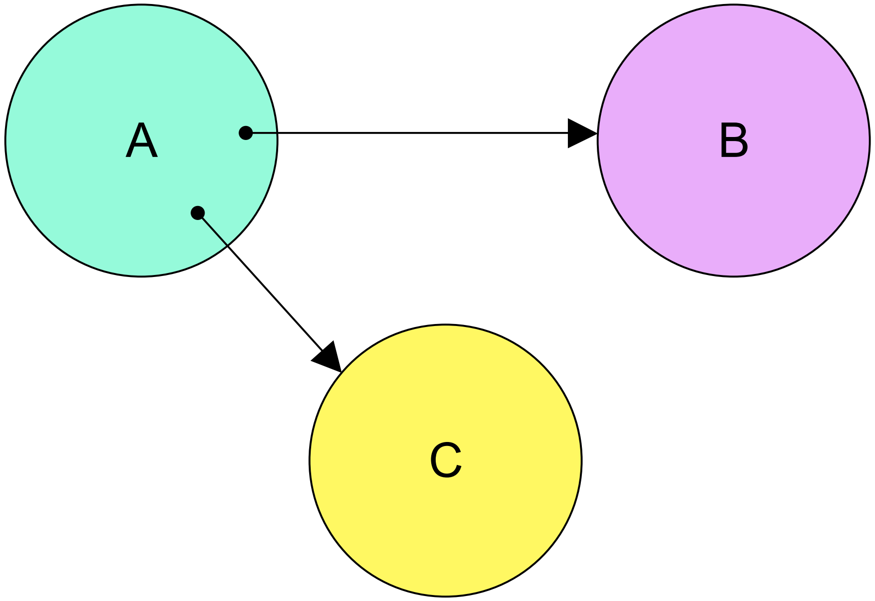
Alice sends a message to Bob containing a reference to the document.

Horton Magic: Bob Receives a Delegated Capability

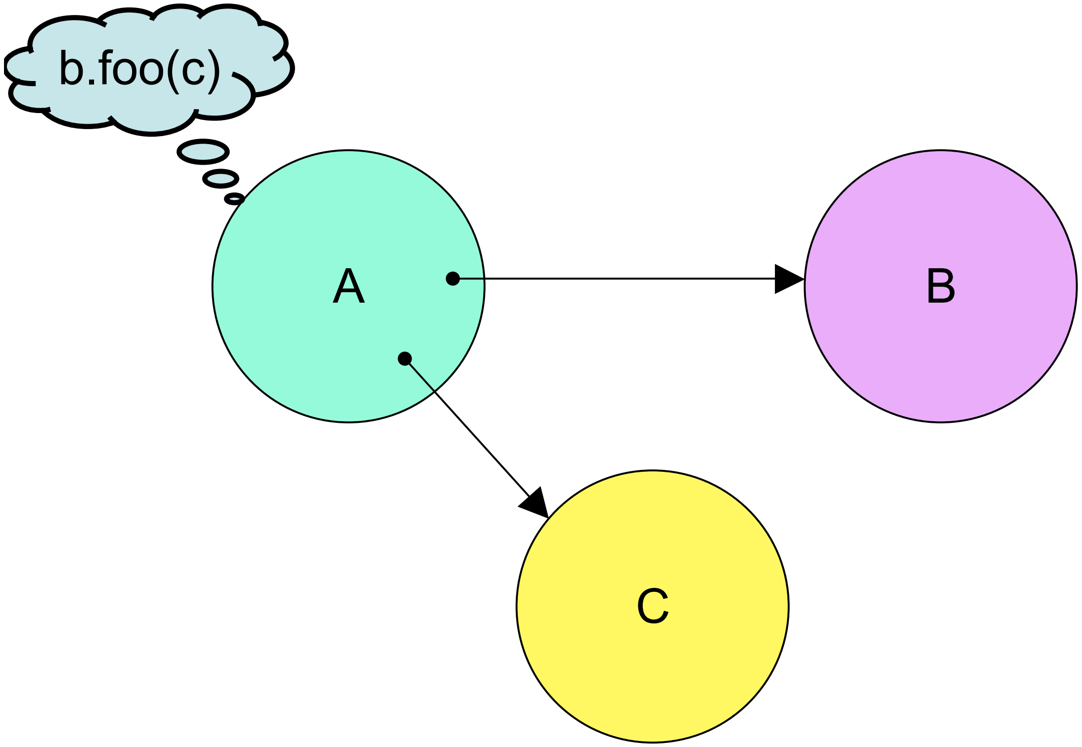


Alice can’t act with Bob’s responsibility
Bob can’t act with Alice’s responsibility

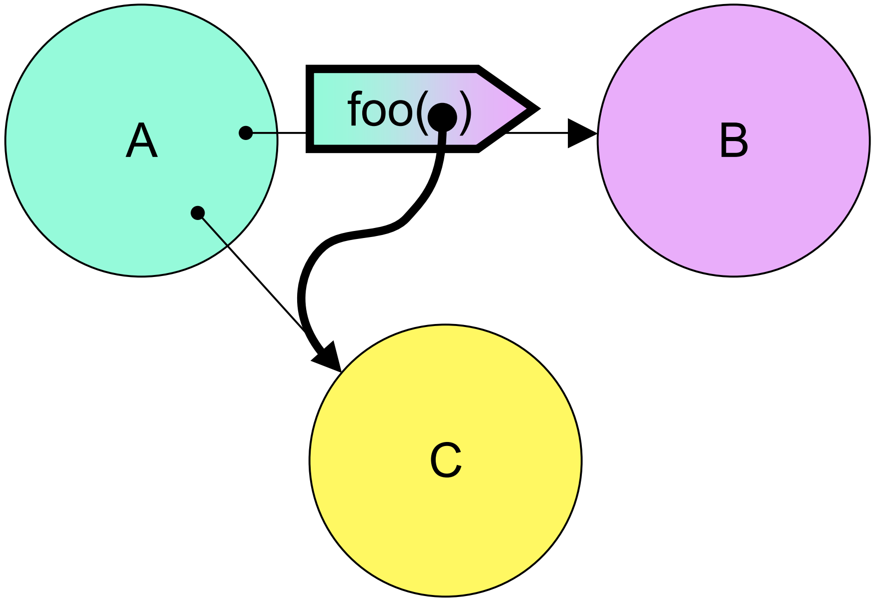
Delegating Least Authority



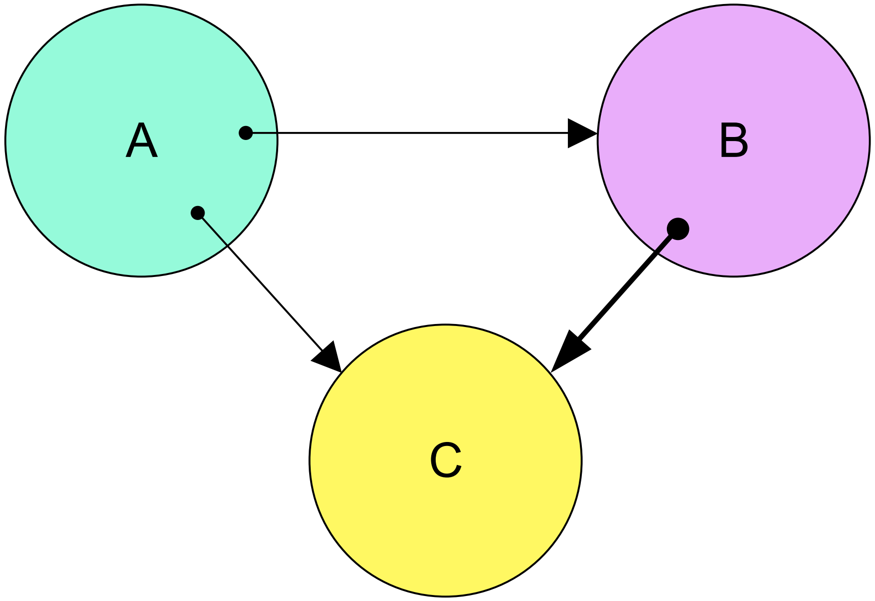
Delegating Least Authority



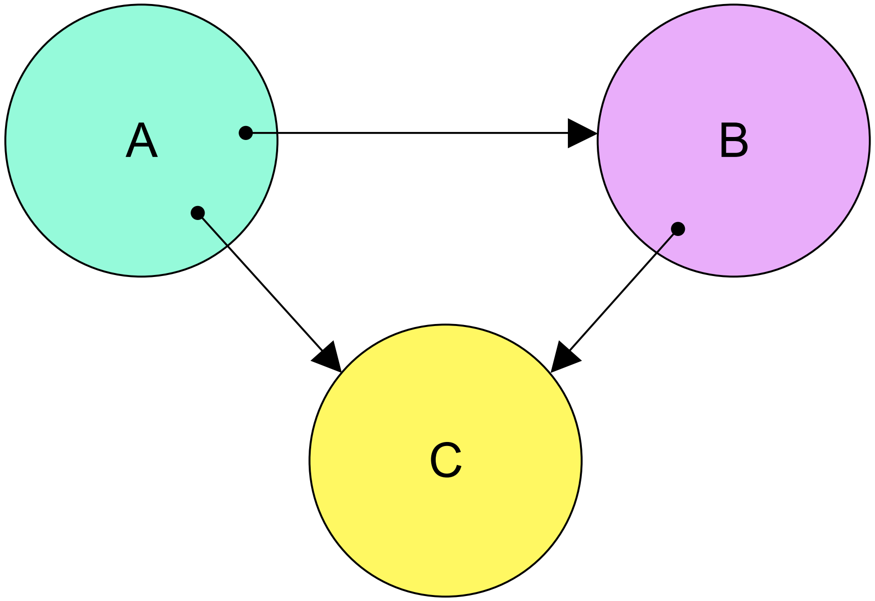
Delegating Least Authority



Delegating Least Authority

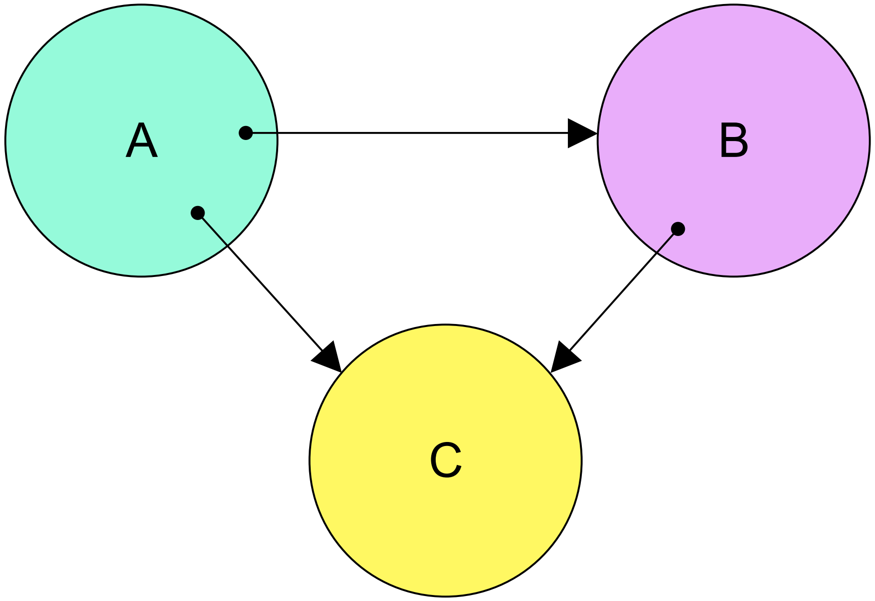


Delegating Least Authority



- Msgs are *only* means to cause effects
- Refs control authority
- Leverage OO patterns

Delegating Least Authority



- Msgs are *only* means to cause effects
- Refs control authority
- Leverage OO patterns
- Anonymous

Two styles, relative strengths

Program decisions		Human decisions
Fine-grained		Large-grained
Built for safety		Built for damage control
Least authority		Most responsibility
Virus resistant		Spam resistant
Authorization-based		Identity-based
↓	?	↓
Object-capabilities (ocaps)		ACLs

Two styles, relative strengths

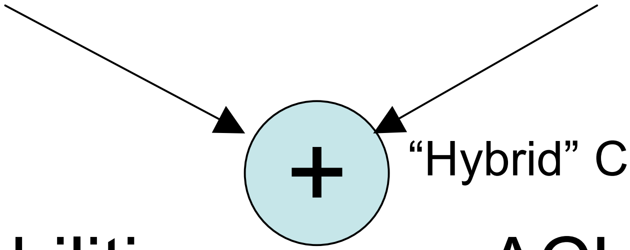
Program decisions	Human decisions
Fine-grained	Large-grained
Built for safety	Built for damage control
Least authority	Most responsibility
Virus resistant	Spam resistant
Authorization-based	Identity-based

Polaris, Plash
Bitfrost?

Object-capabilities (ocaps)	ACLs

Two styles, relative strengths

Program decisions	Human decisions
Fine-grained	Large-grained
Built for safety	Built for damage control
Least authority	Most responsibility
Virus resistant	Spam resistant
Authorization-based	Identity-based

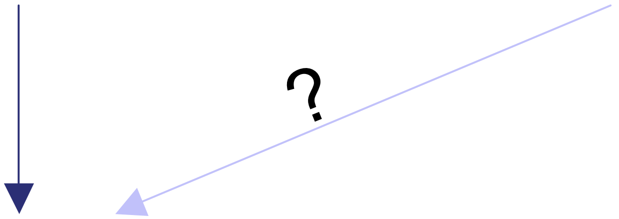


"Hybrid" Cap Systems (SCAP, Sys/38)

Object-capabilities (ocaps)	ACLs

Two styles, relative strengths

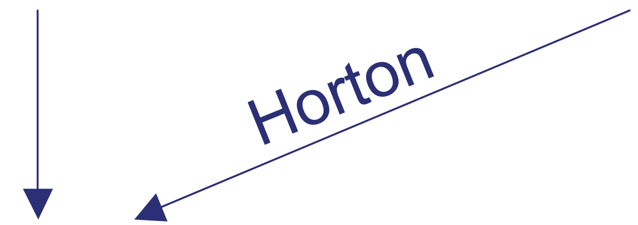
Program decisions	Human decisions
Fine-grained	Large-grained
Built for safety	Built for damage control
Least authority	Most responsibility
Virus resistant	Spam resistant
Authorization-based	Identity-based



Object-capabilities (ocaps)	ACLs

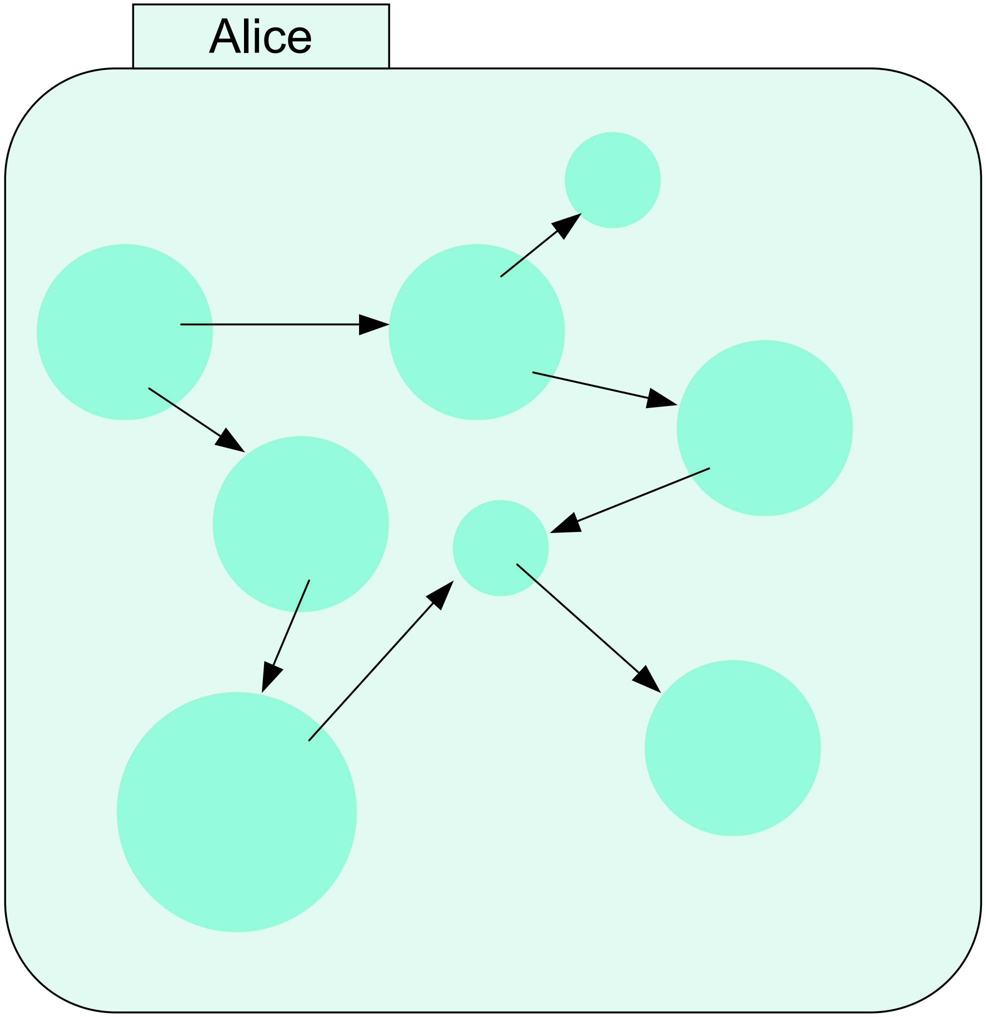
Two styles, relative strengths

Program decisions	Human decisions
Fine-grained	Large-grained
Built for safety	Built for damage control
Least authority	Most responsibility
Virus resistant	Spam resistant
Authorization-based	Identity-based

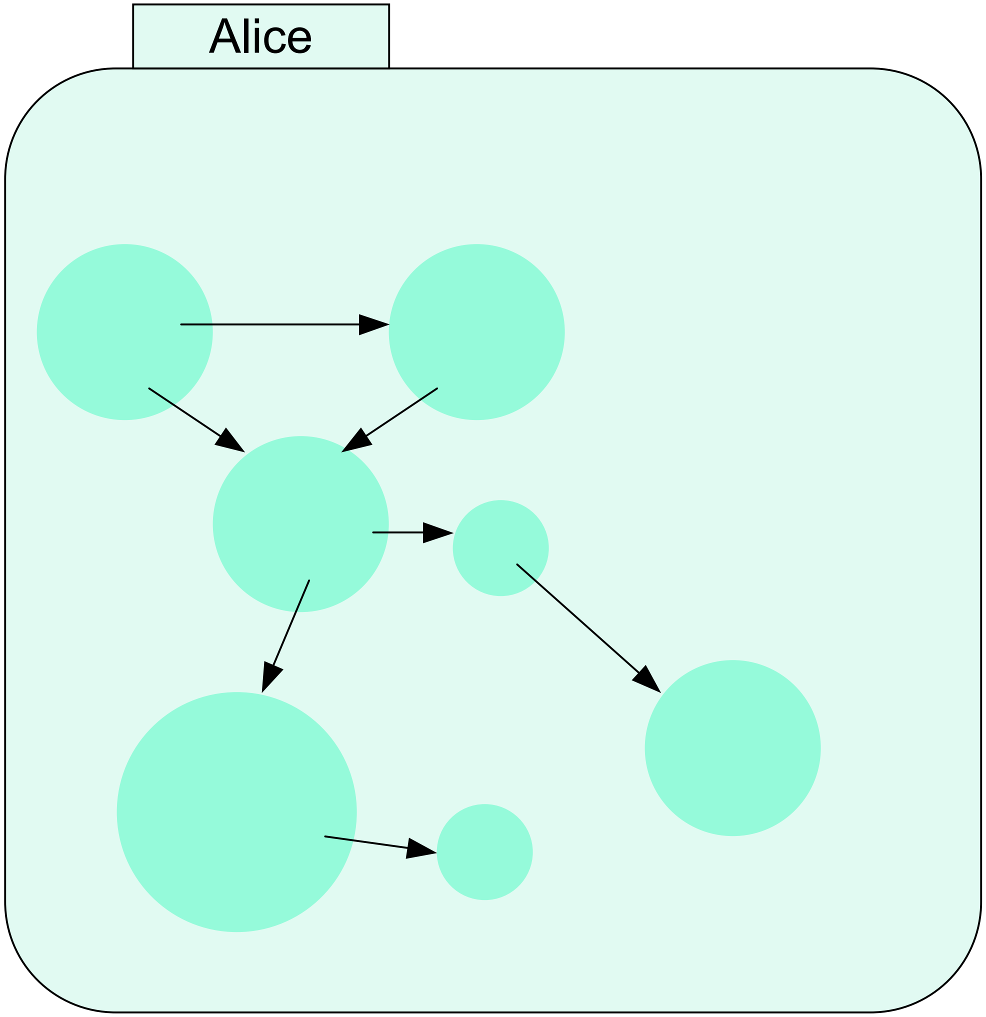


Object-capabilities
(ocaps)

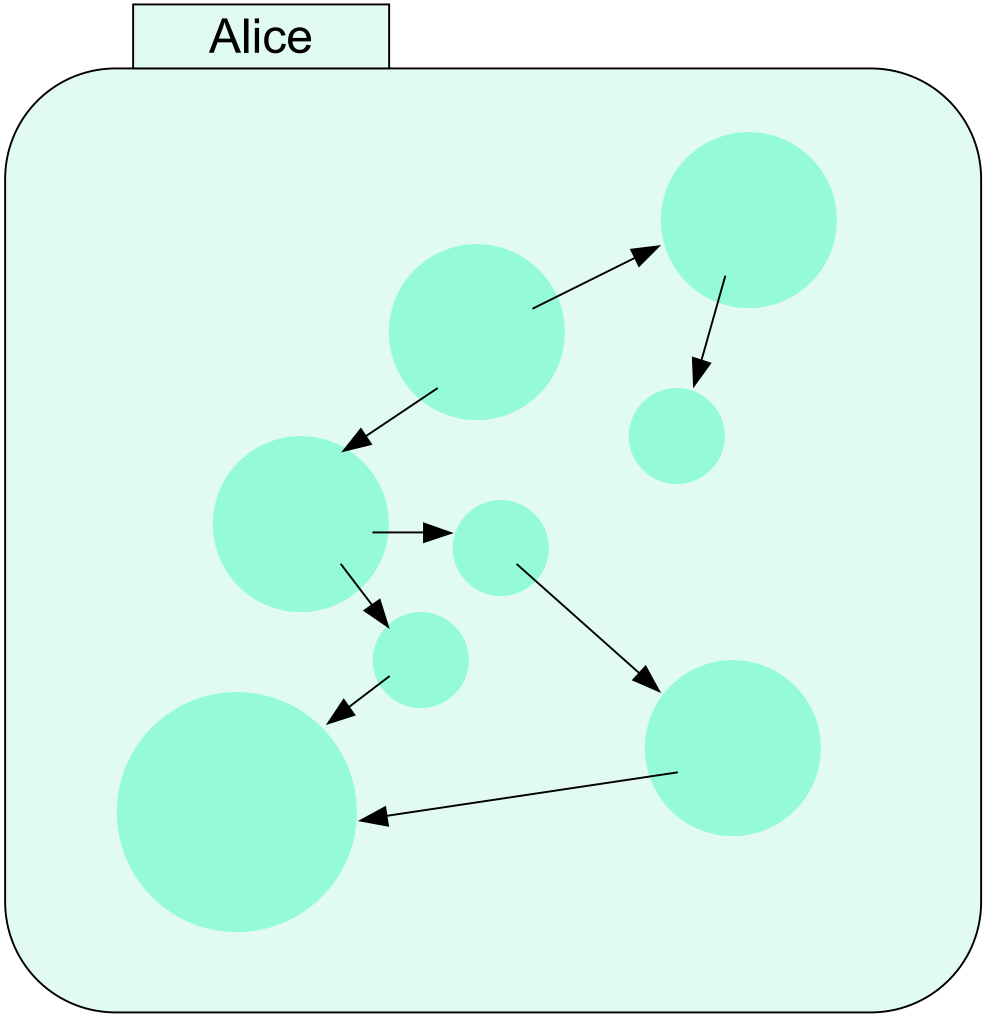
ACLs



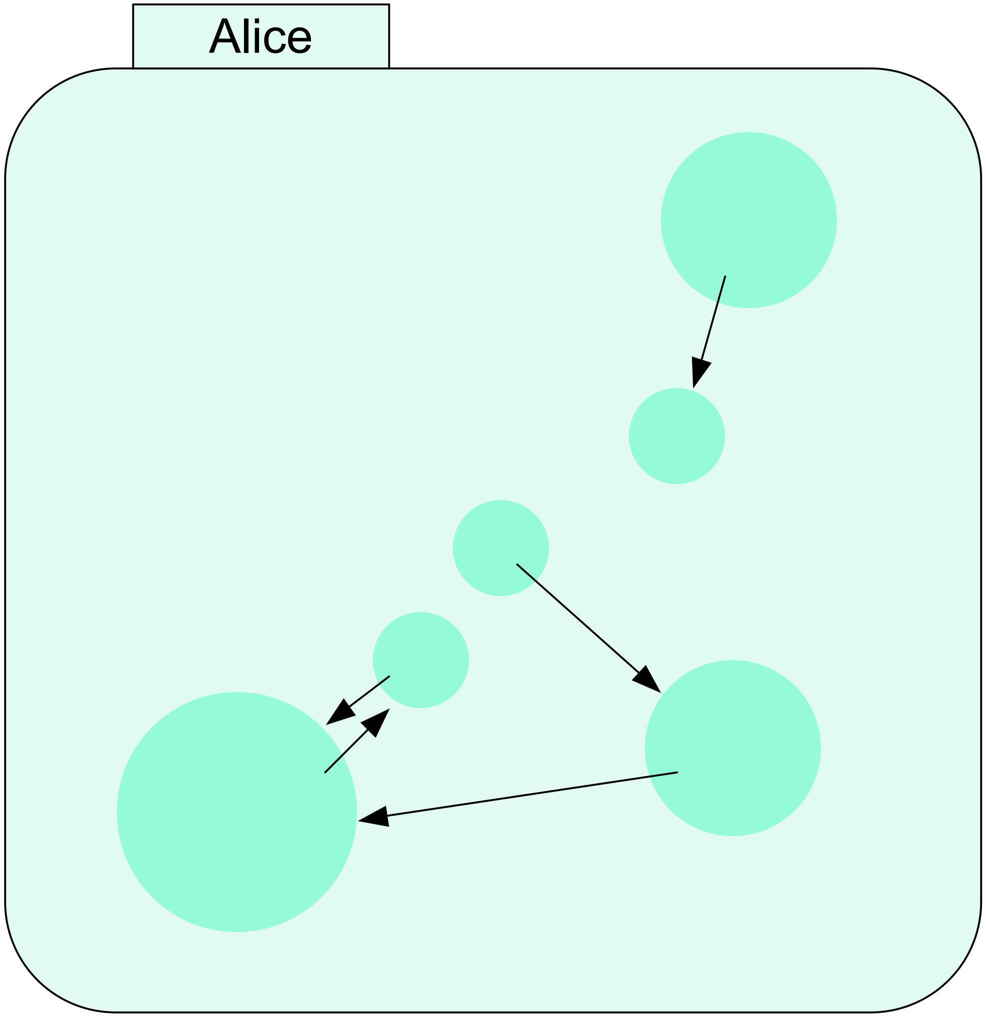
Can’t vet code or actions of each object.



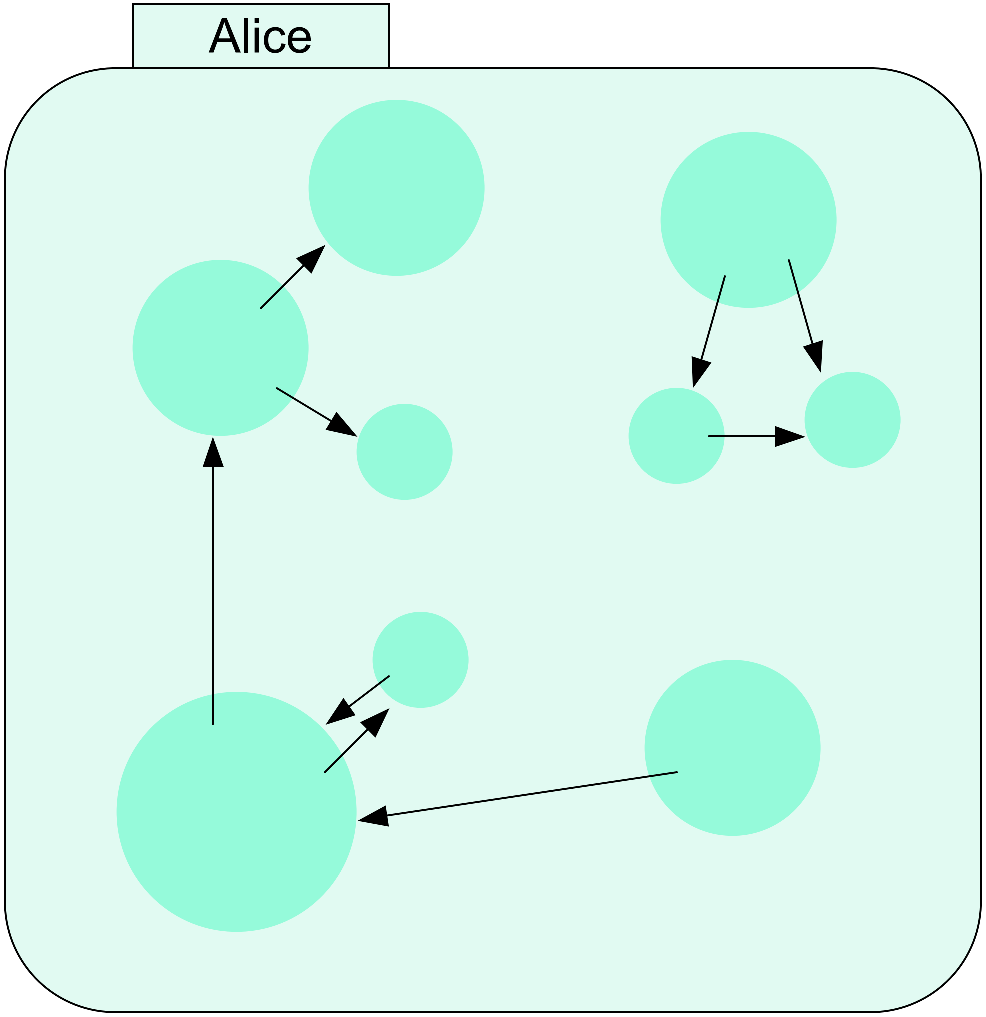
Can't vet code or actions of each object.



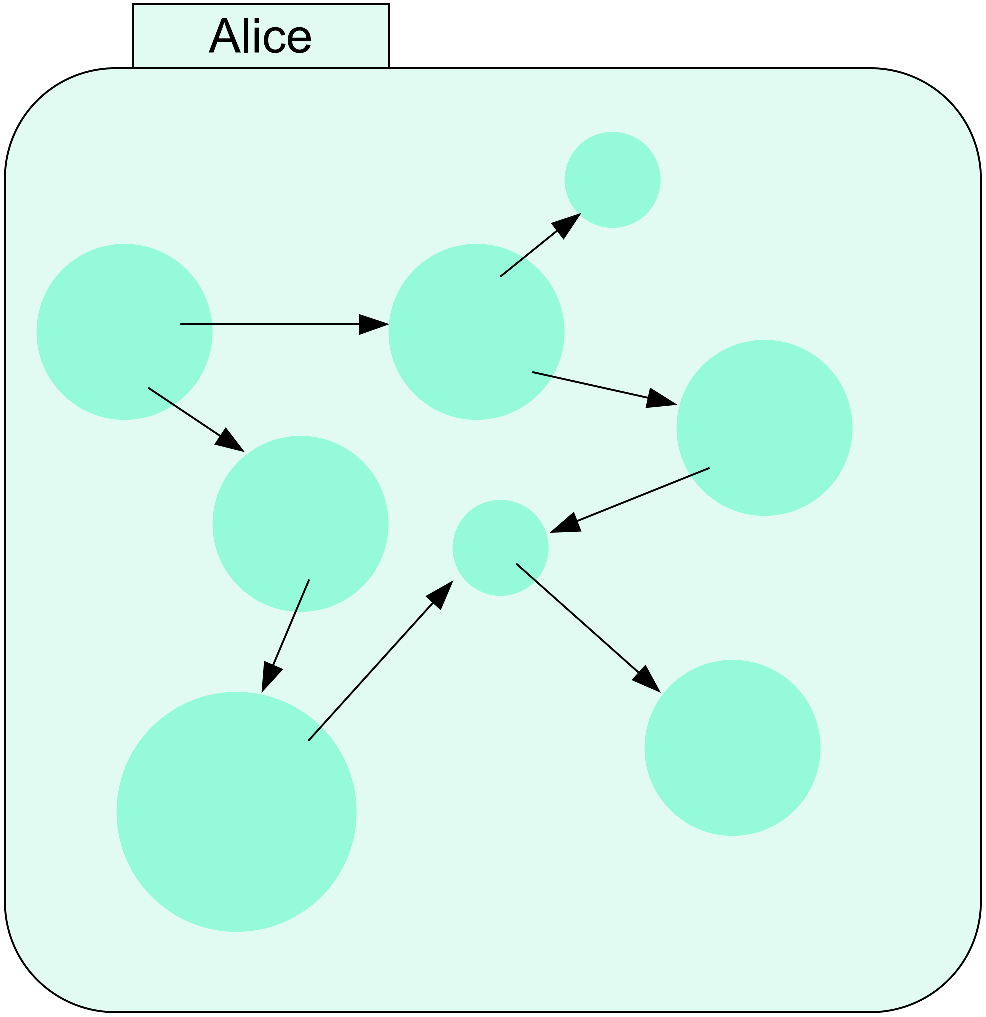
Can’t vet code or actions of each object.



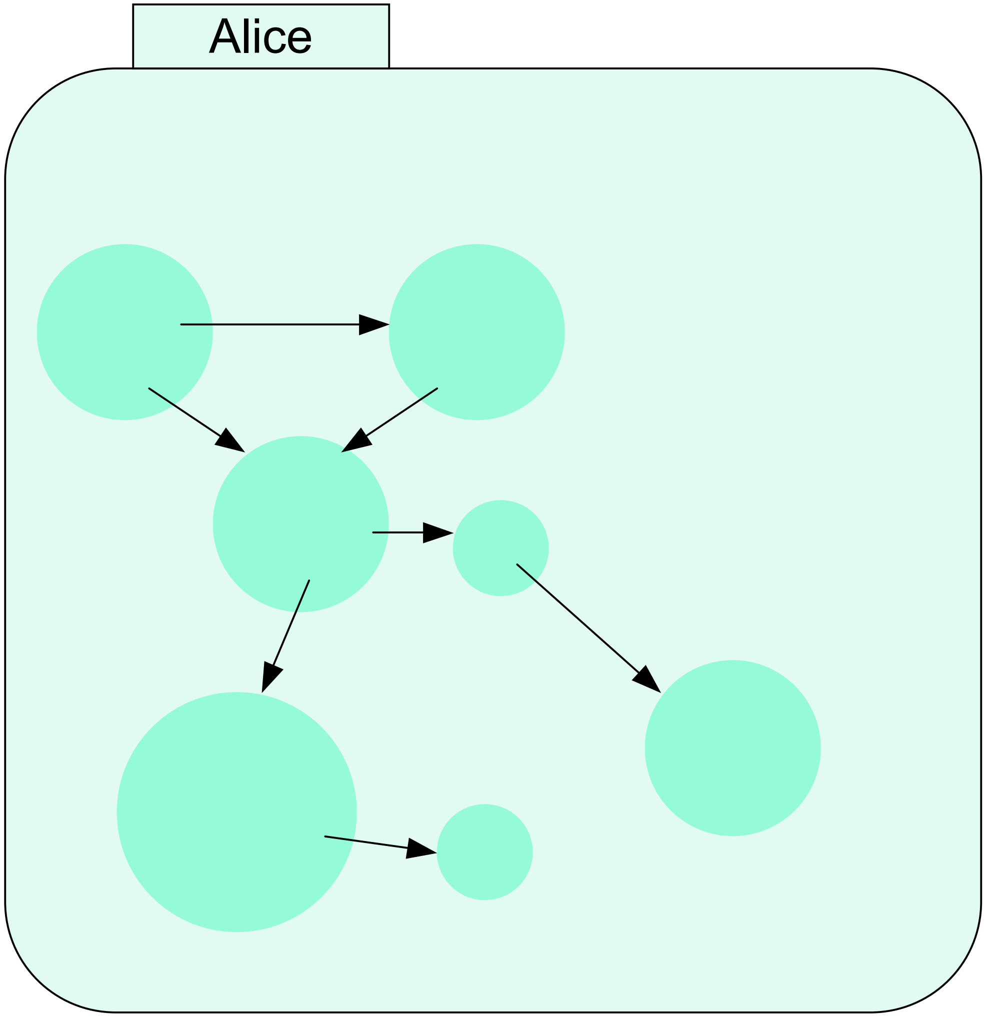
Can’t vet code or actions of each object.



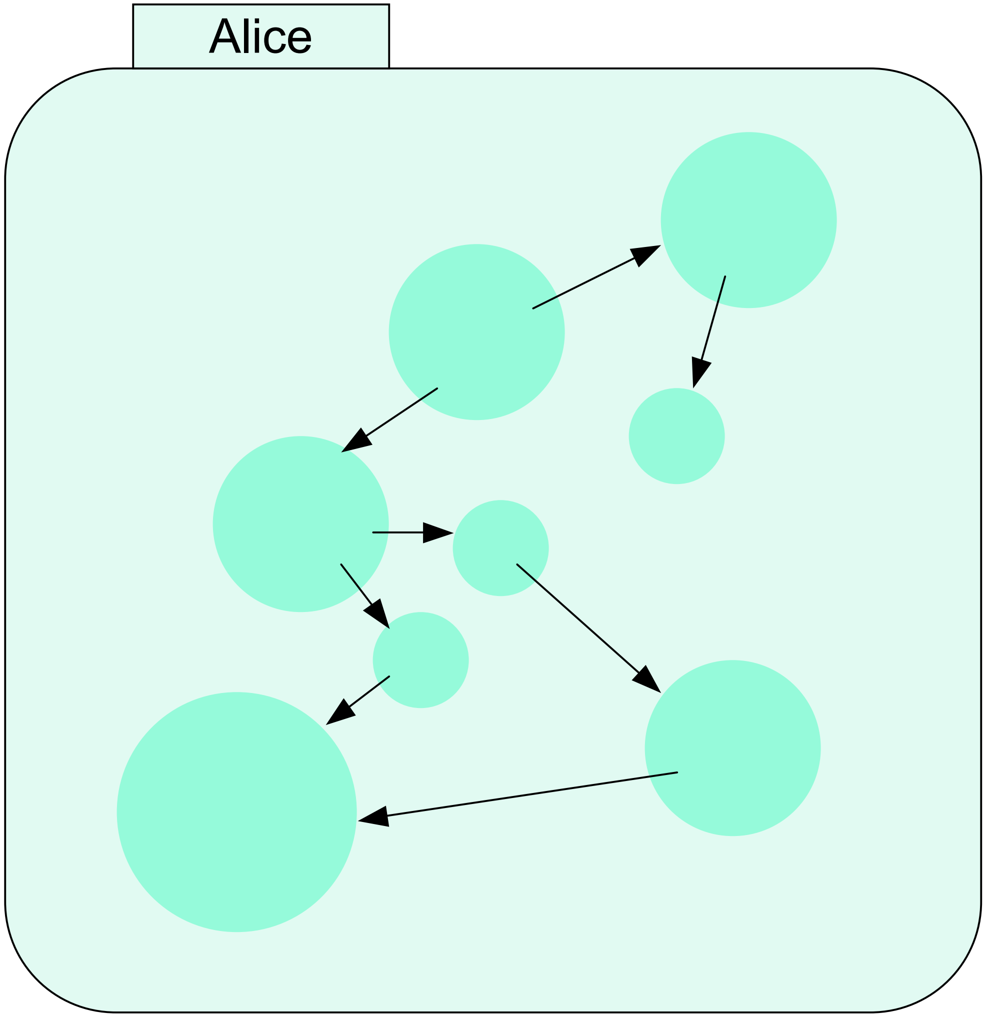
Can’t vet code or actions of each object.



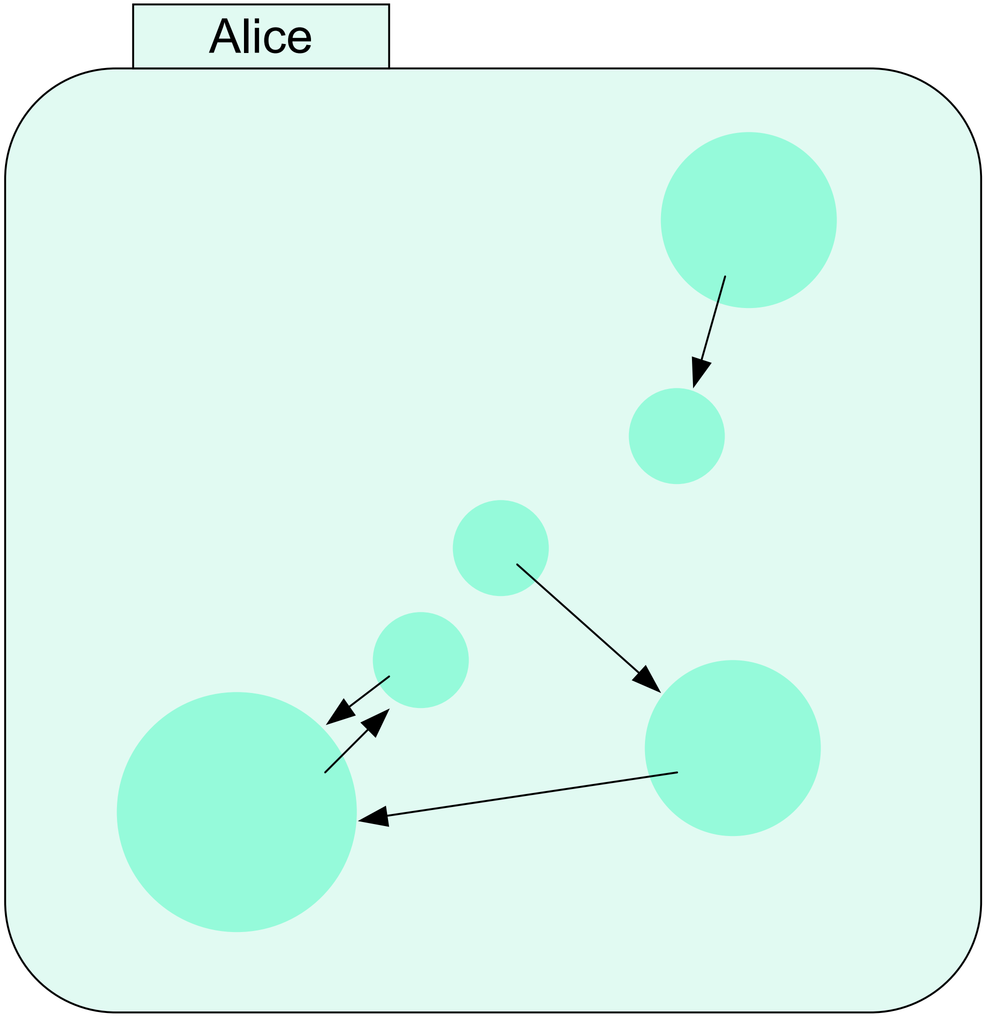
Can’t vet code or actions of each object.



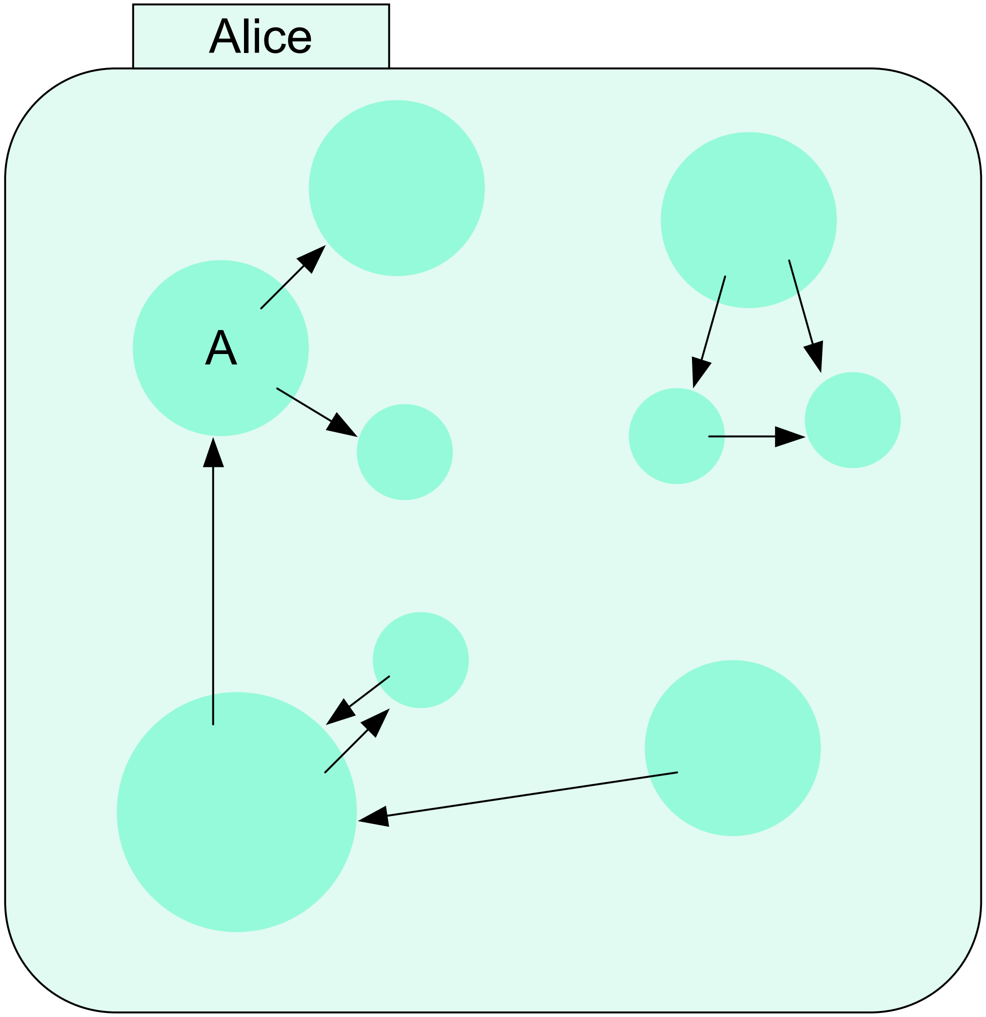
Can’t vet code or actions of each object.



Can't vet code or actions of each object.



Can’t vet code or actions of each object.



Can’t vet code or actions of each object.

Aggregate into long-lived responsible identity.

Story Needs Four Characters

Alice & Bob

- Old patterns for identity-based control: *identity tunnel*

Alice introduces Bob & Carol

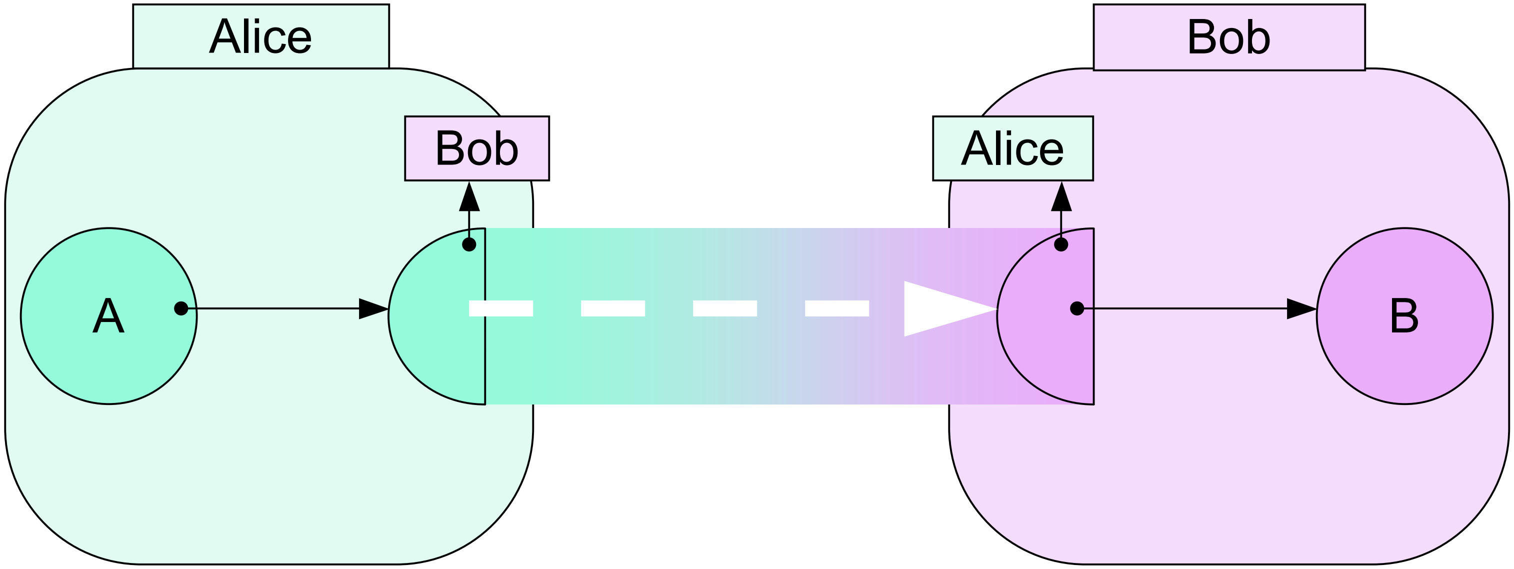
- Builds new relationships from old

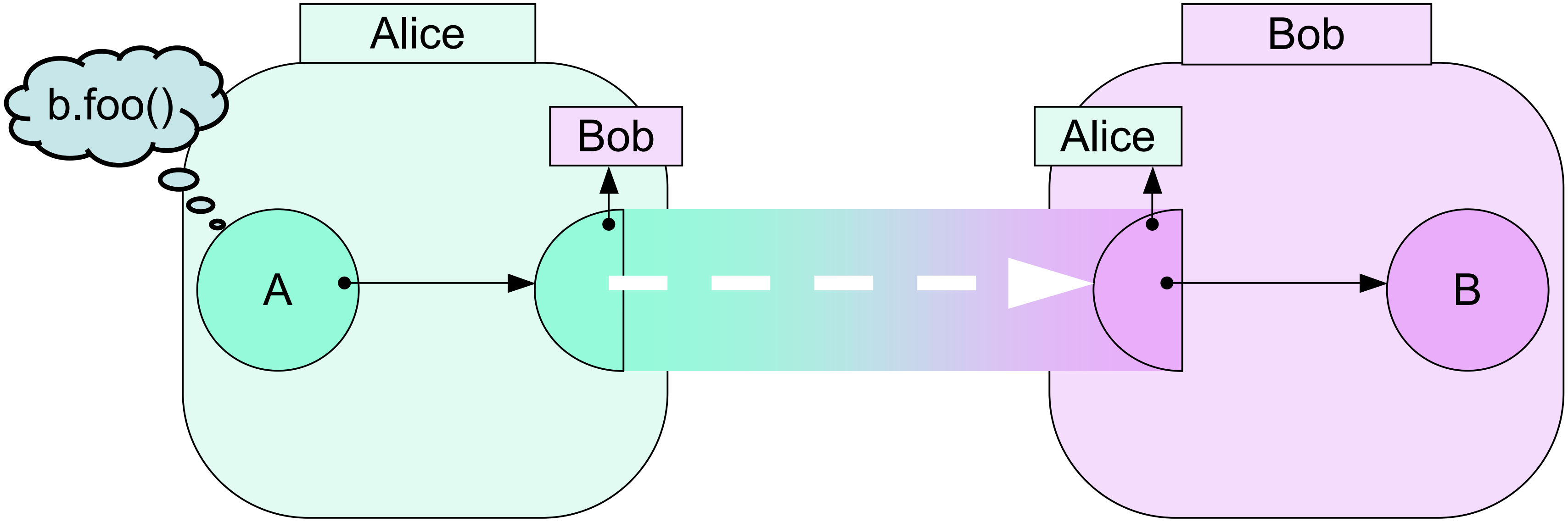
Carol also hears of Bob from Dave

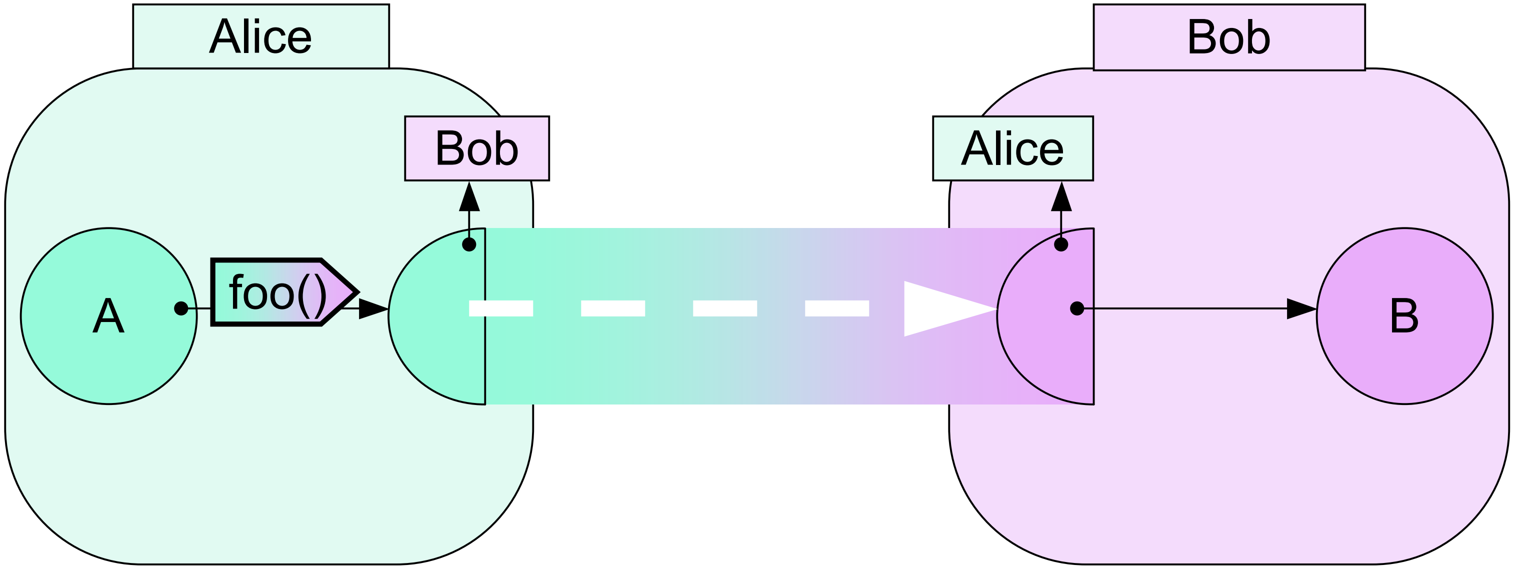
- Corroborates Bob’s independence from Alice

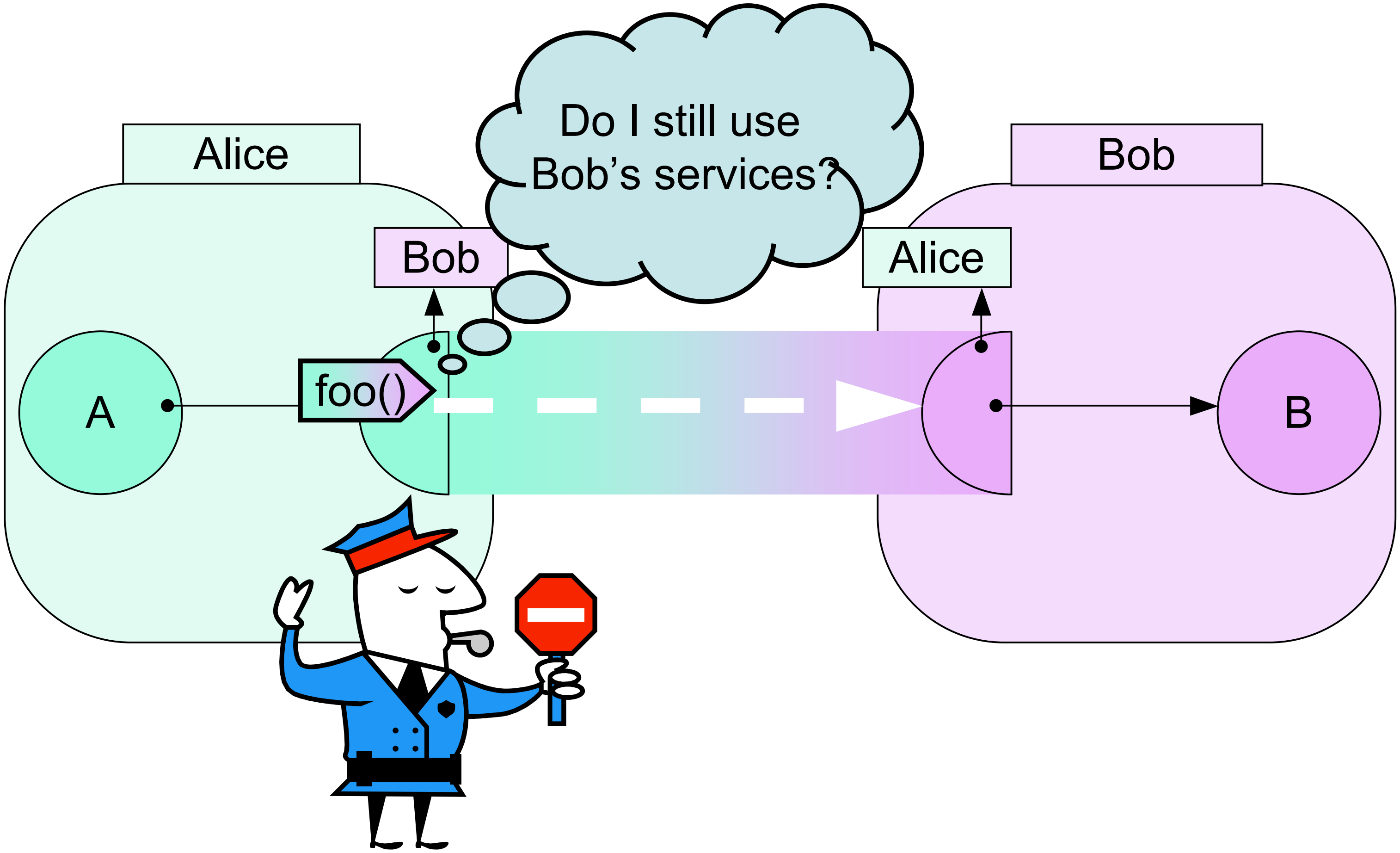
Two-party intermediation

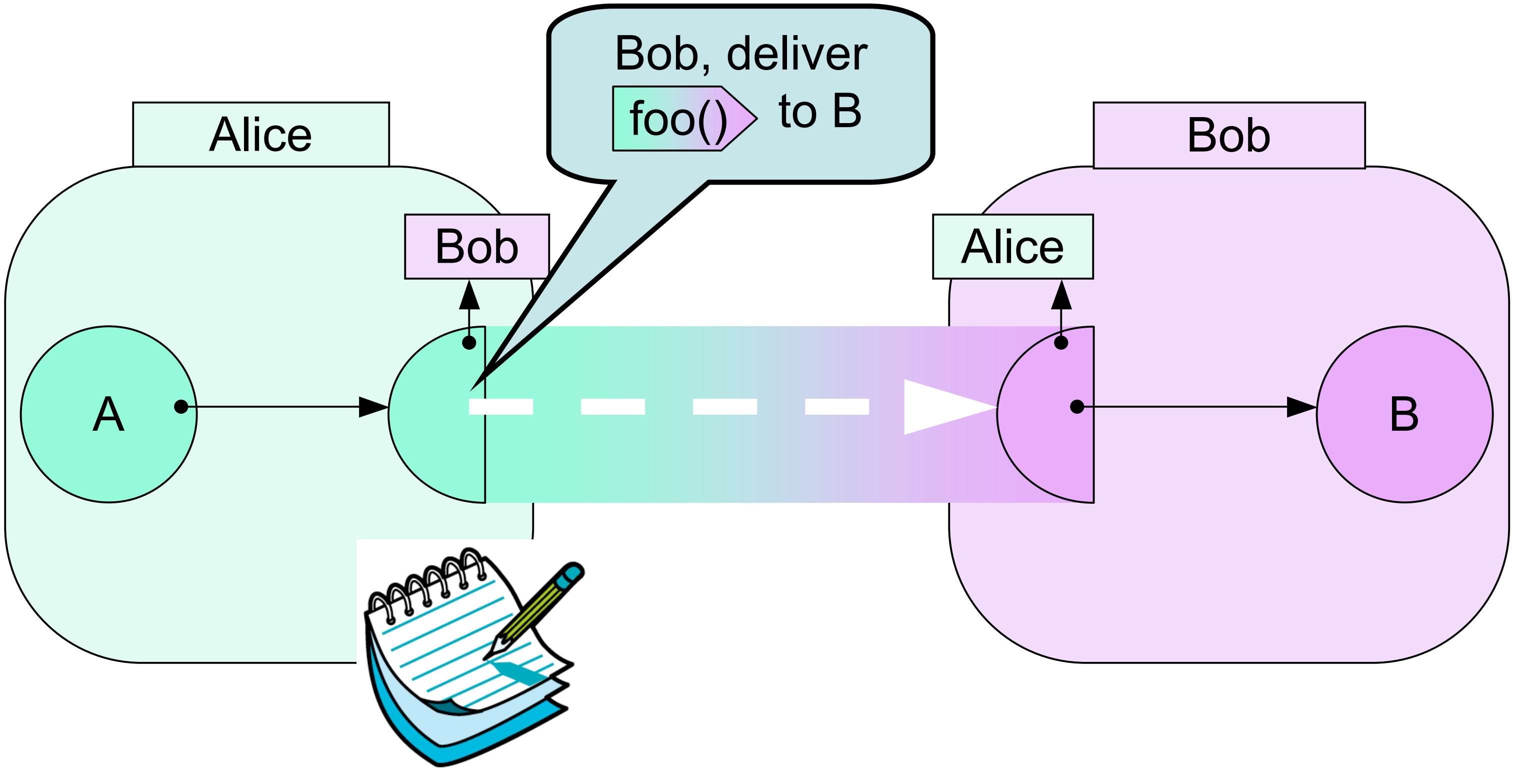
A message travels through an
identity tunnel

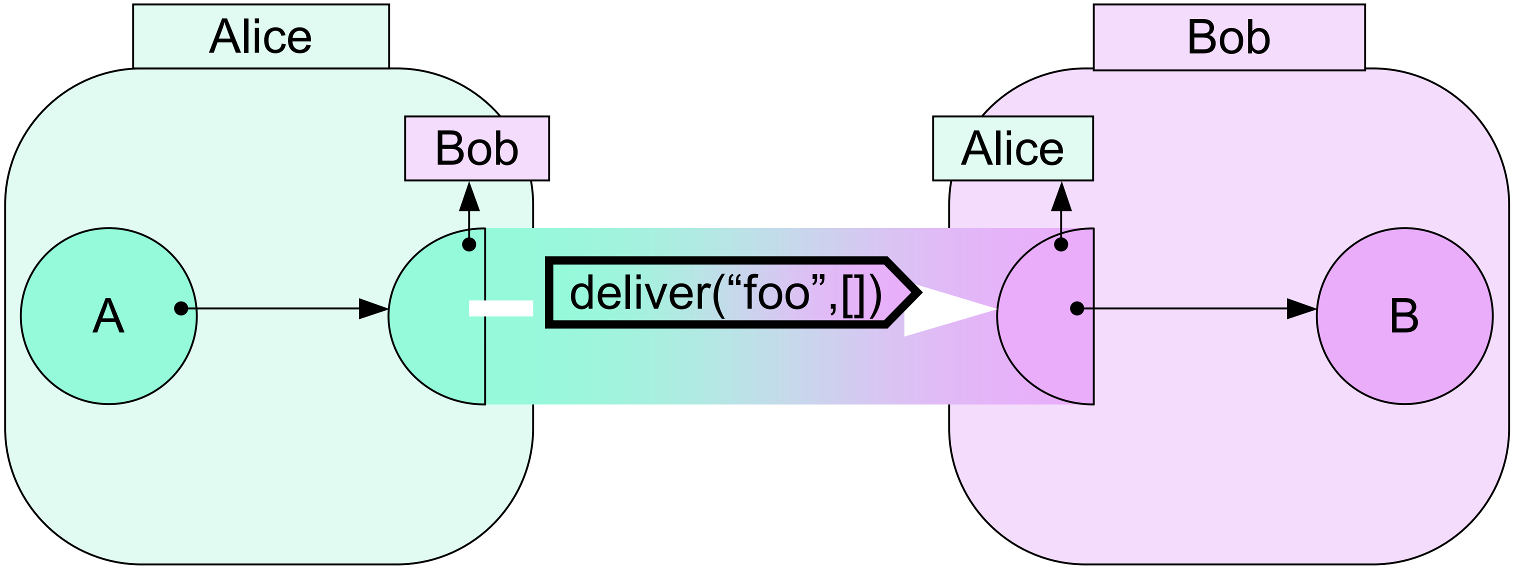

Alice
Bob
A
Bob
Alice
B

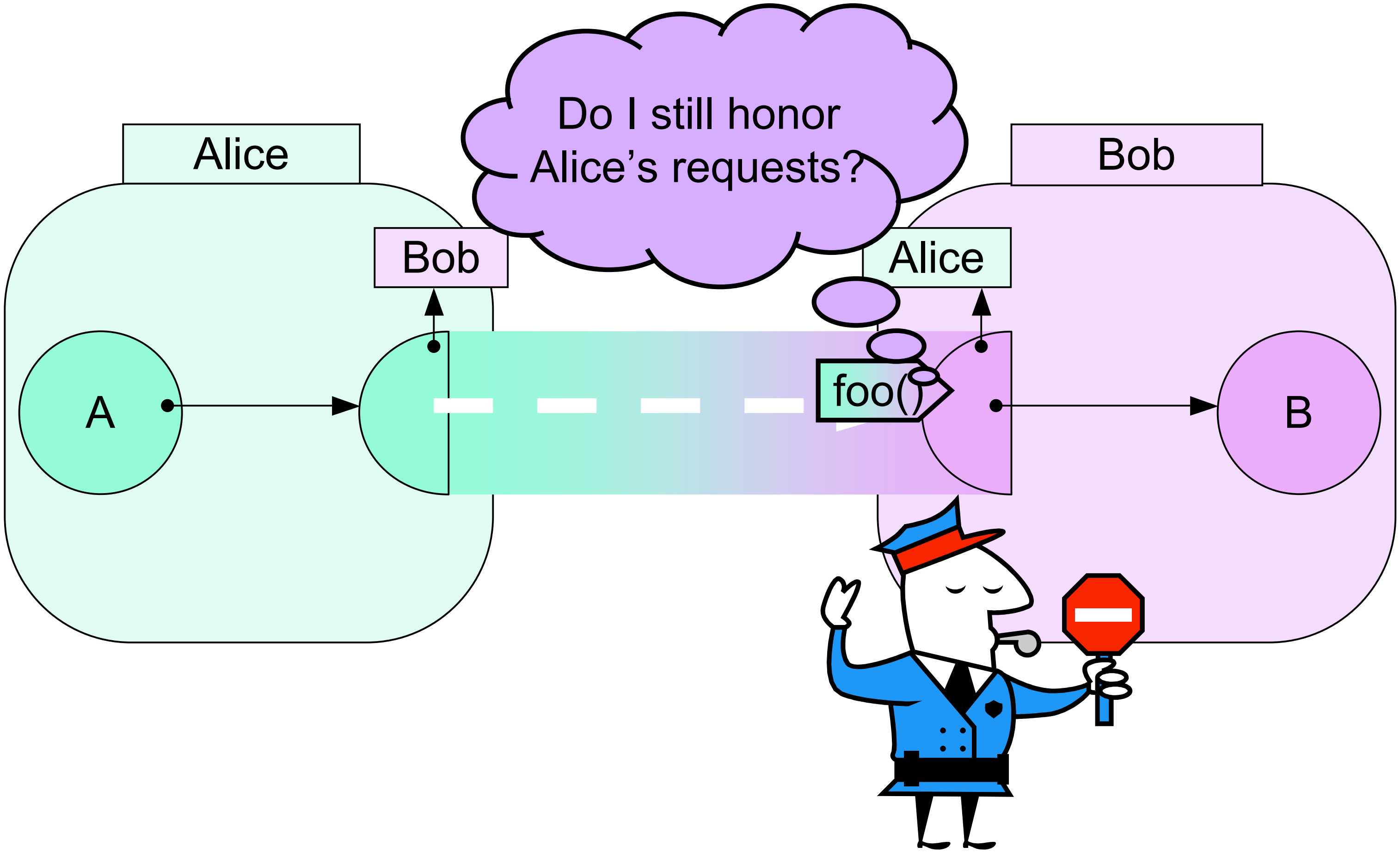

b.foo()
Alice
Bob
A
Bob
Alice
B

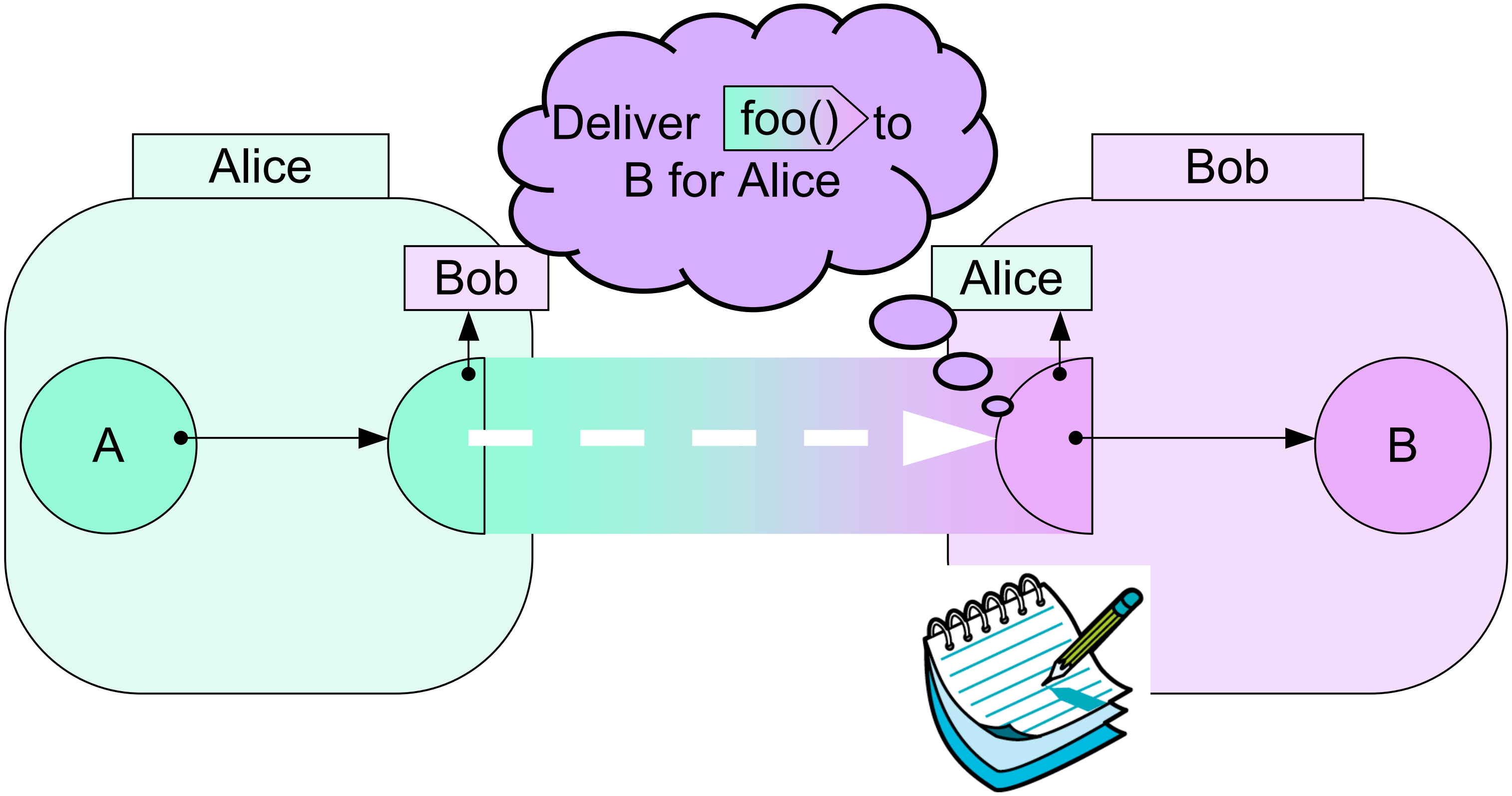

Alice
Bob
Bob
Alice
A
foo()
B

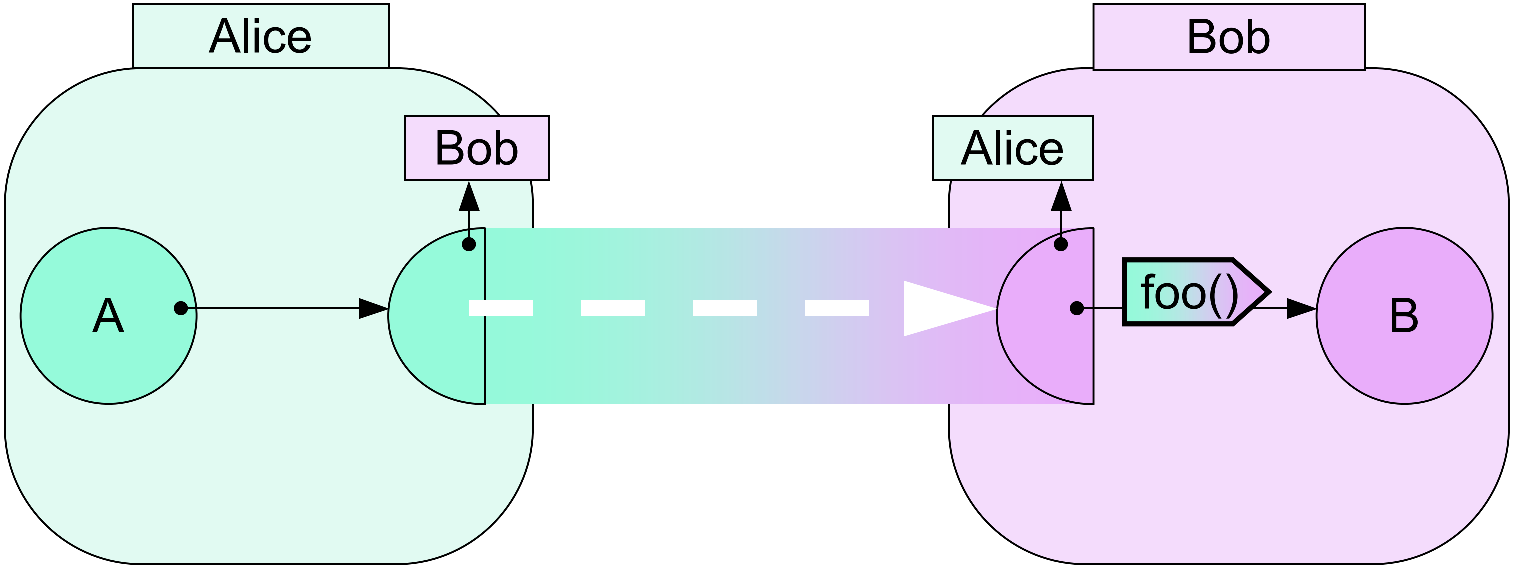

Do I still use
Bob's services?
Alice
Bob
Bob
Alice
A
foo()
B

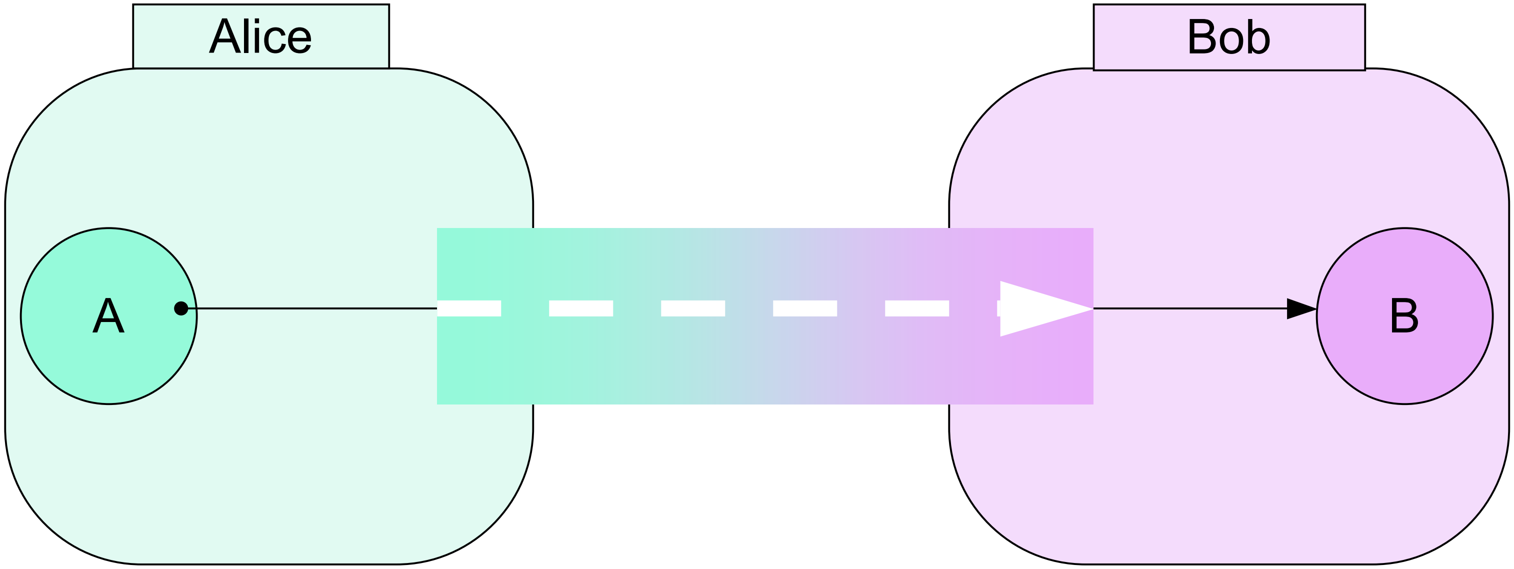

Bob, deliver foo() to B
Alice
Bob
A
Bob
Alice
B


Alice
Bob
Bob
Alice
A
deliver("foo",[])
B


Do I still honor
Alice's requests?
Alice
Bob
Bob
Alice
A
foo()
B

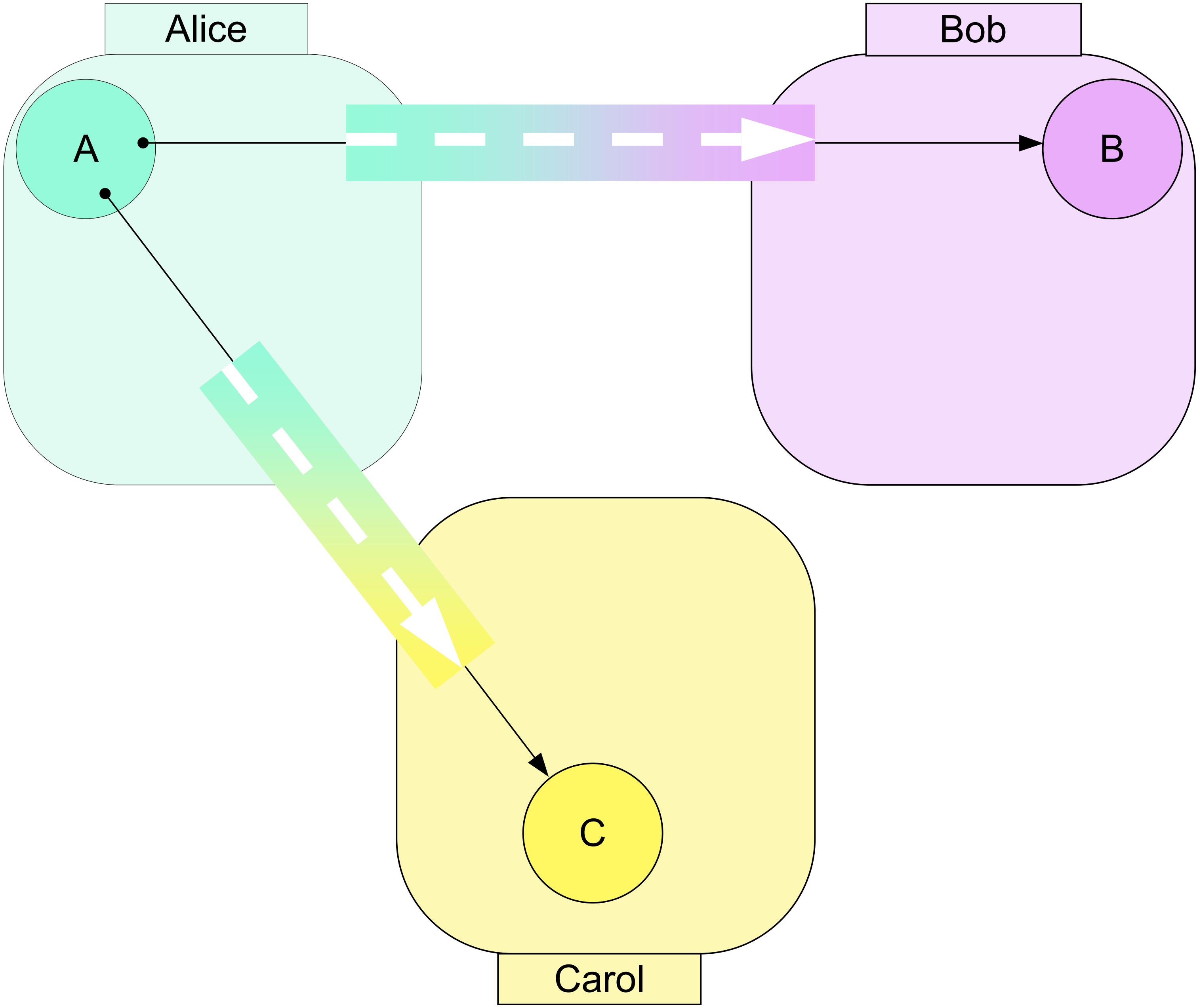

Deliver foo() to
B for Alice
Alice
Bob
Bob
Alice
A
B

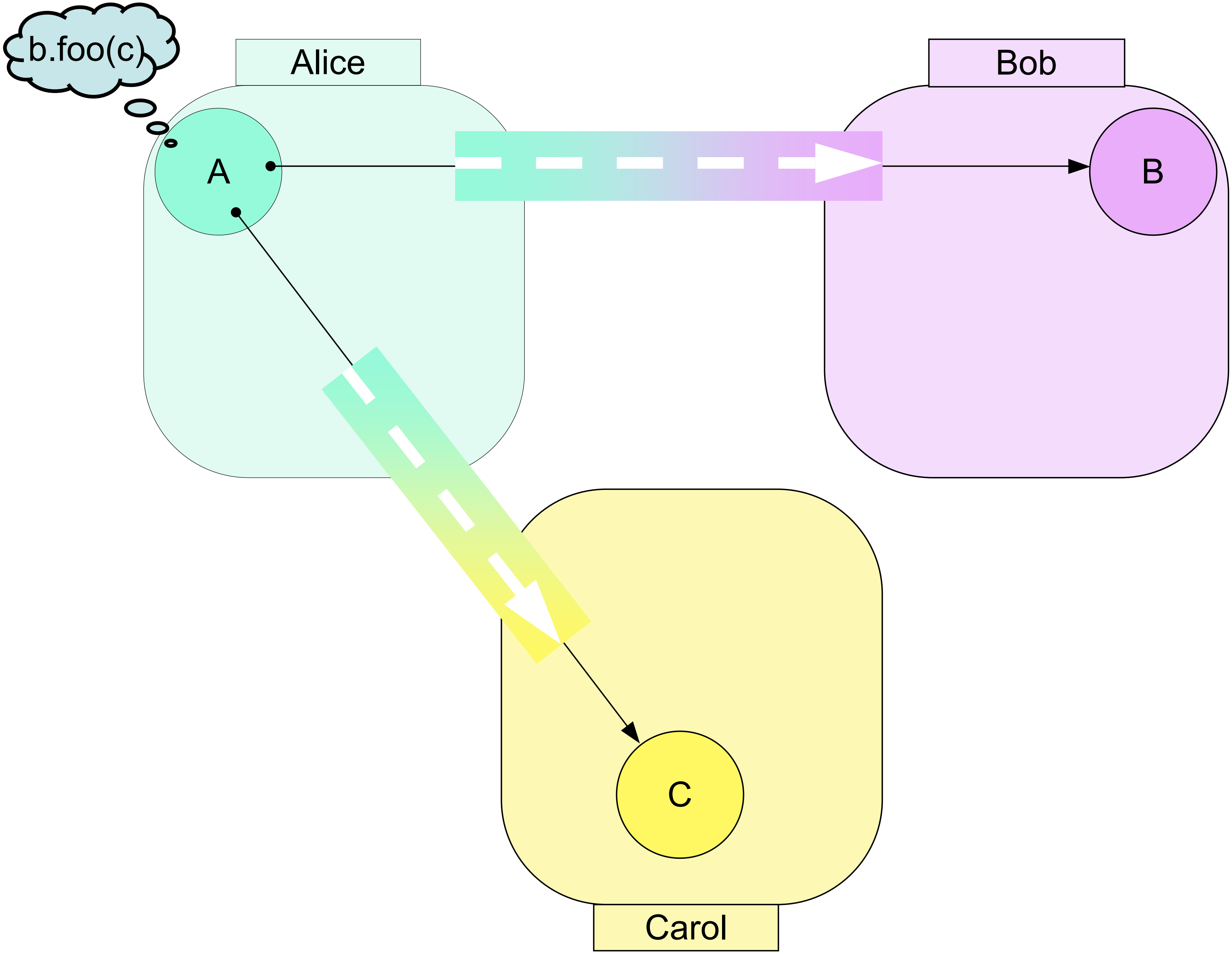

Alice
Bob
Bob
Alice
A
foo()
B

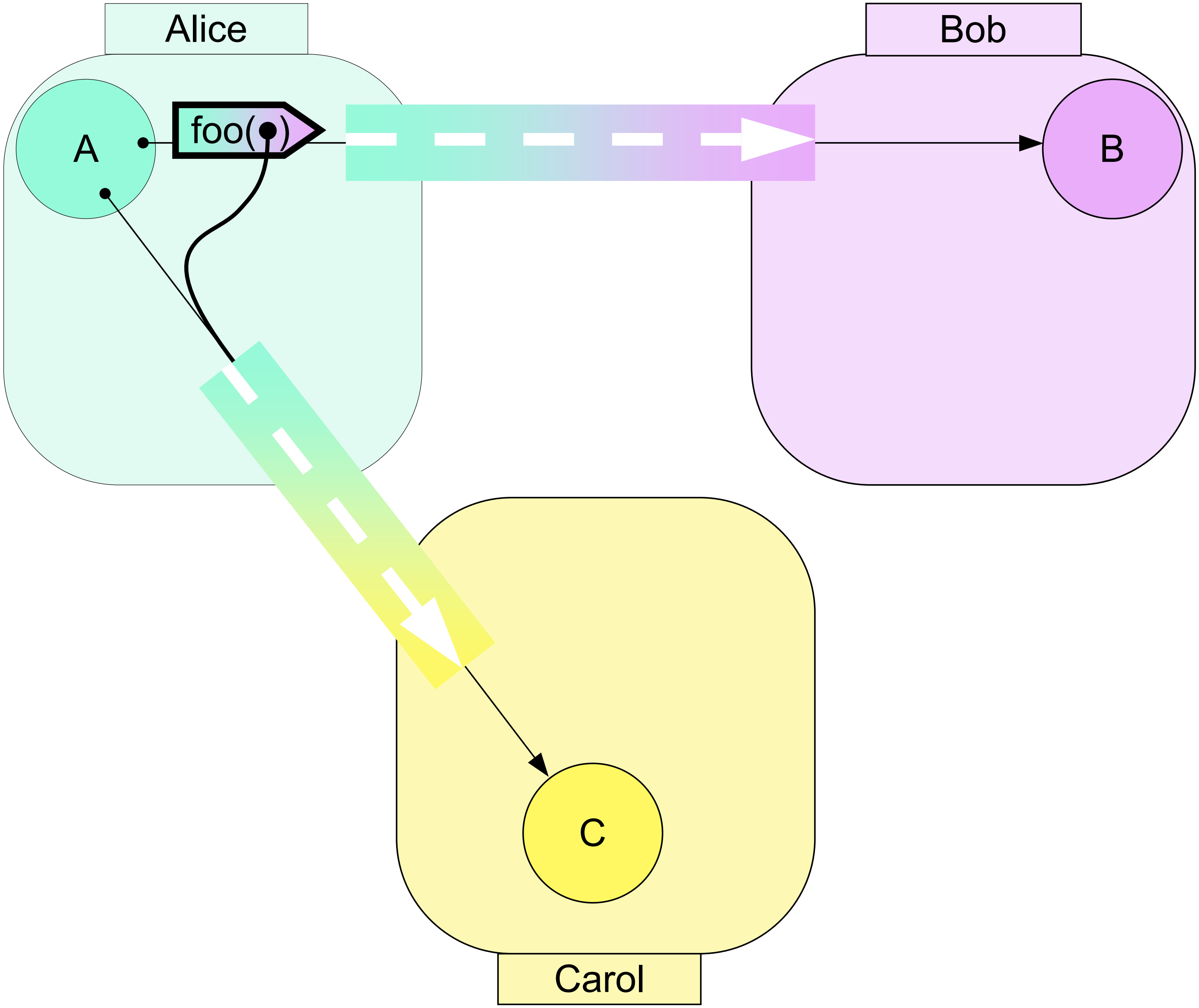

Alice
Bob
A
B

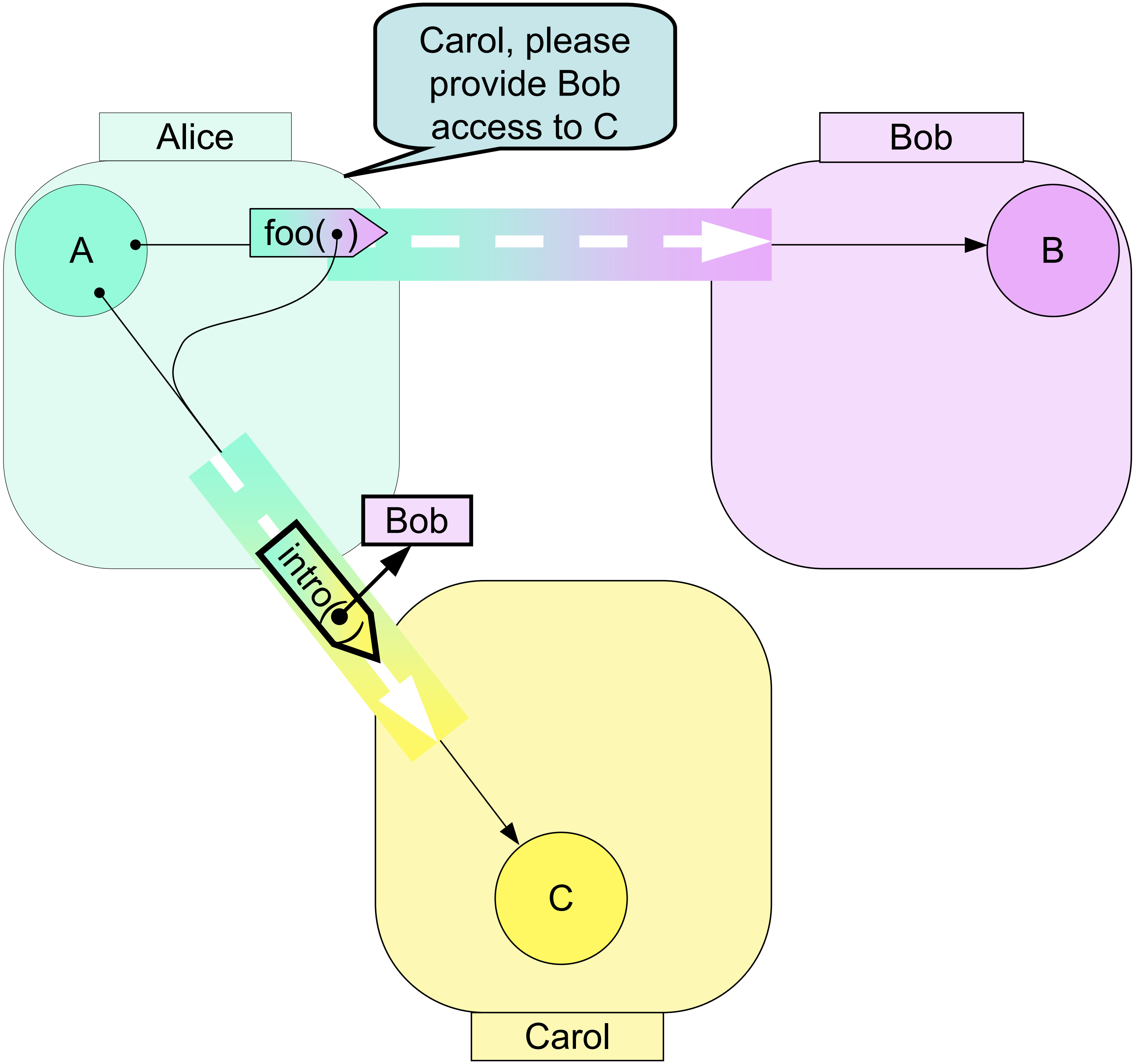
Three-party intermediation

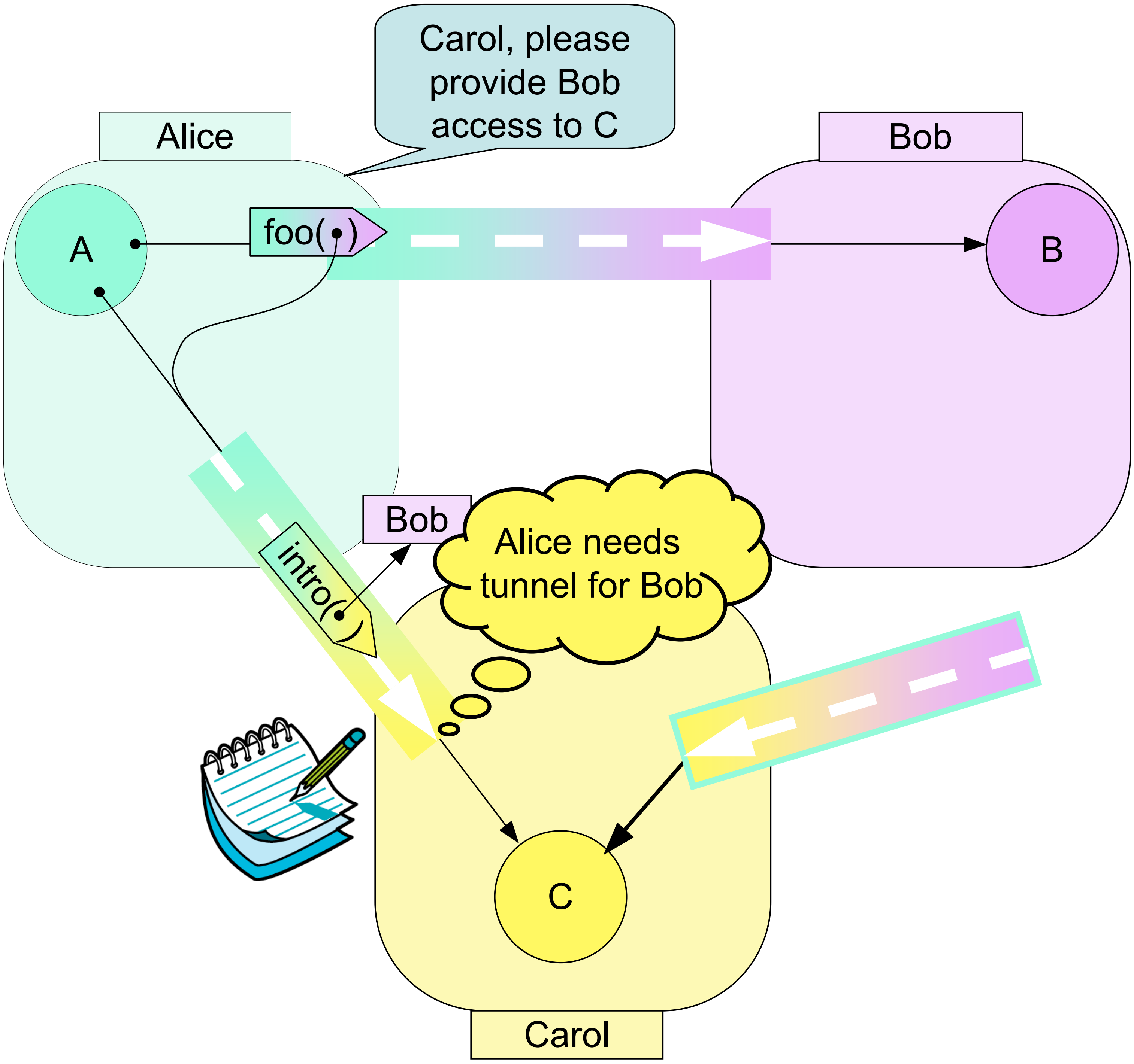
Build new relationships from old

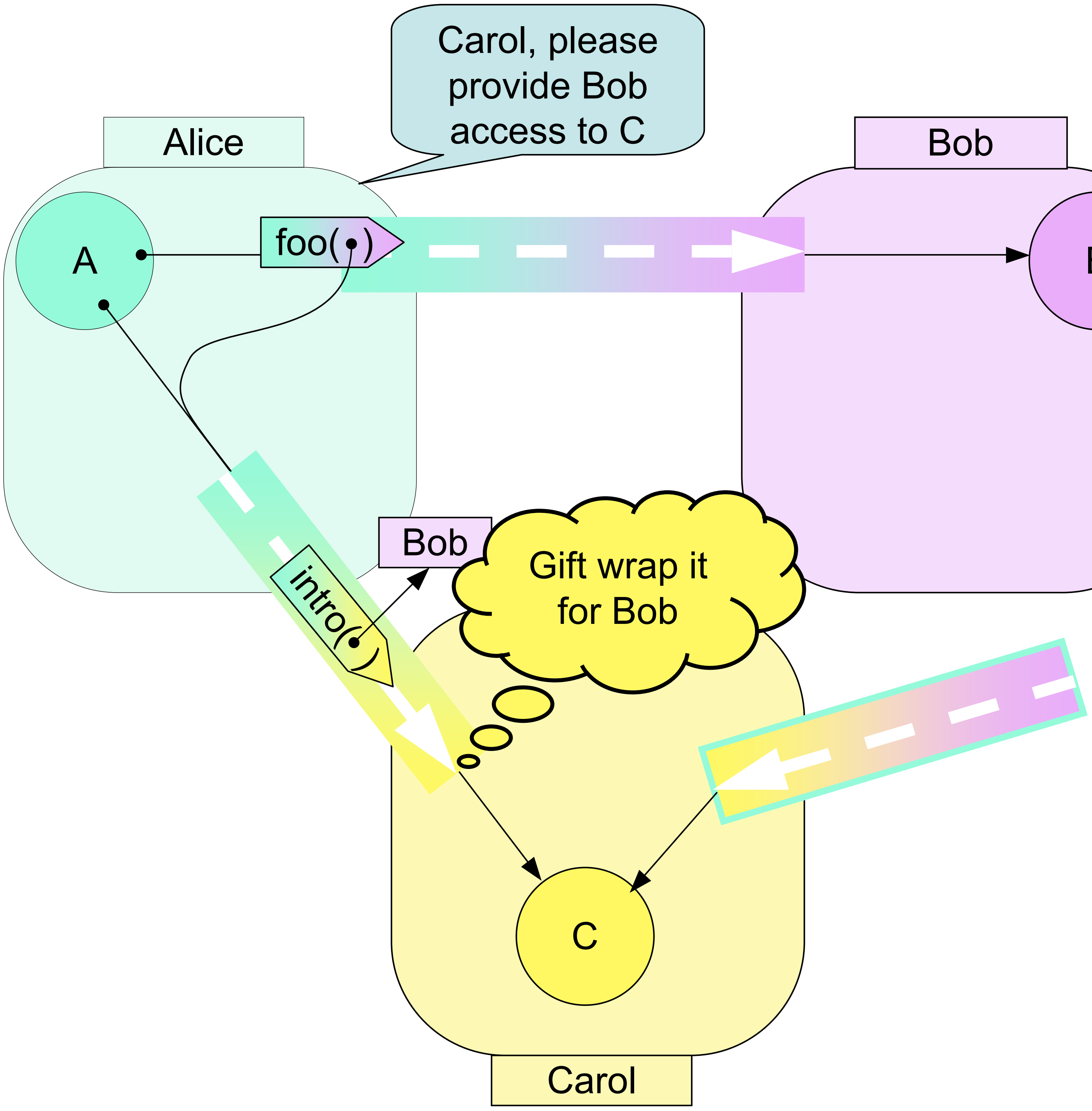

Alice
Bob
A
B
C
Carol

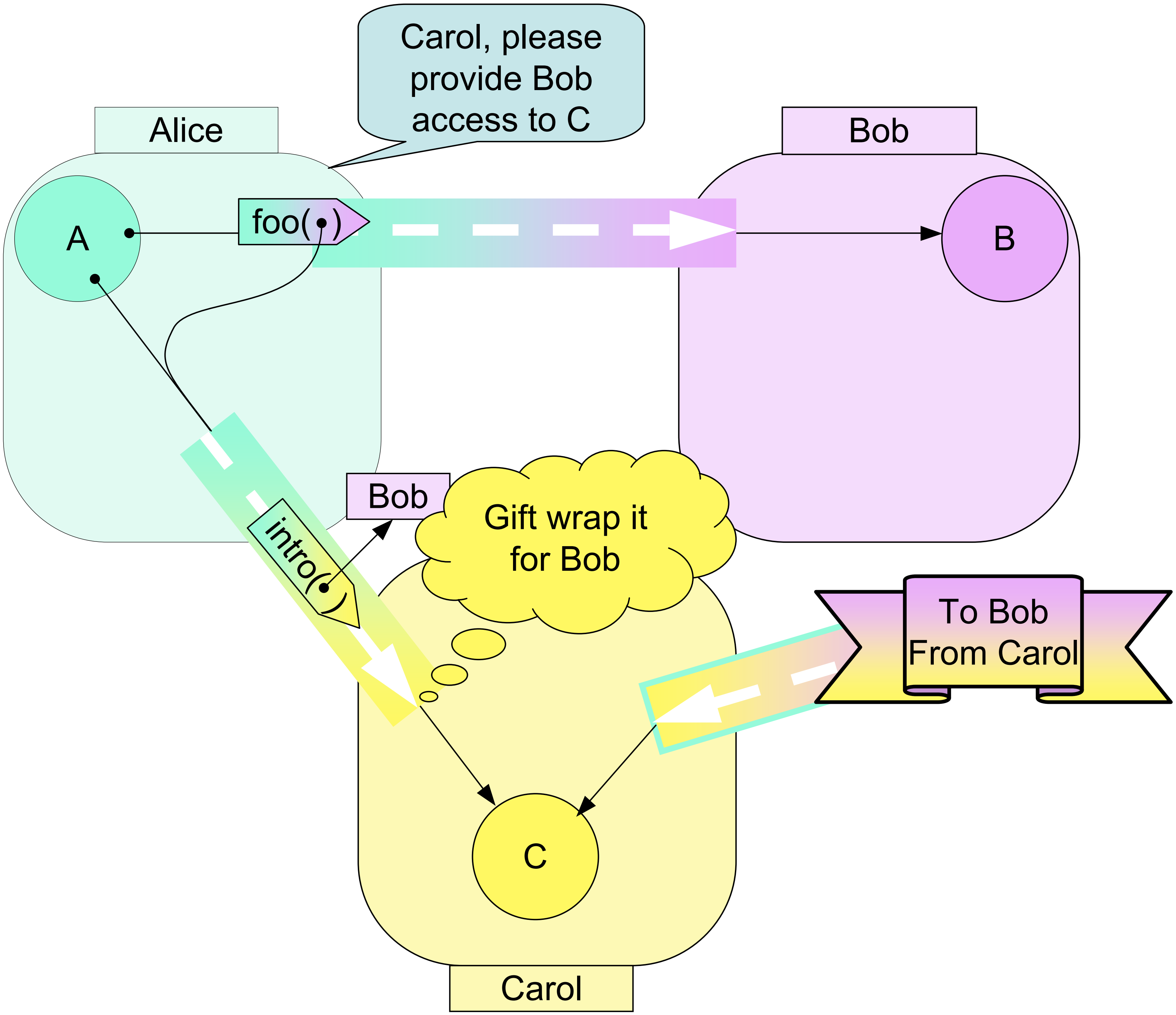

b.foo(c)
Alice
Bob
A
B
C
Carol

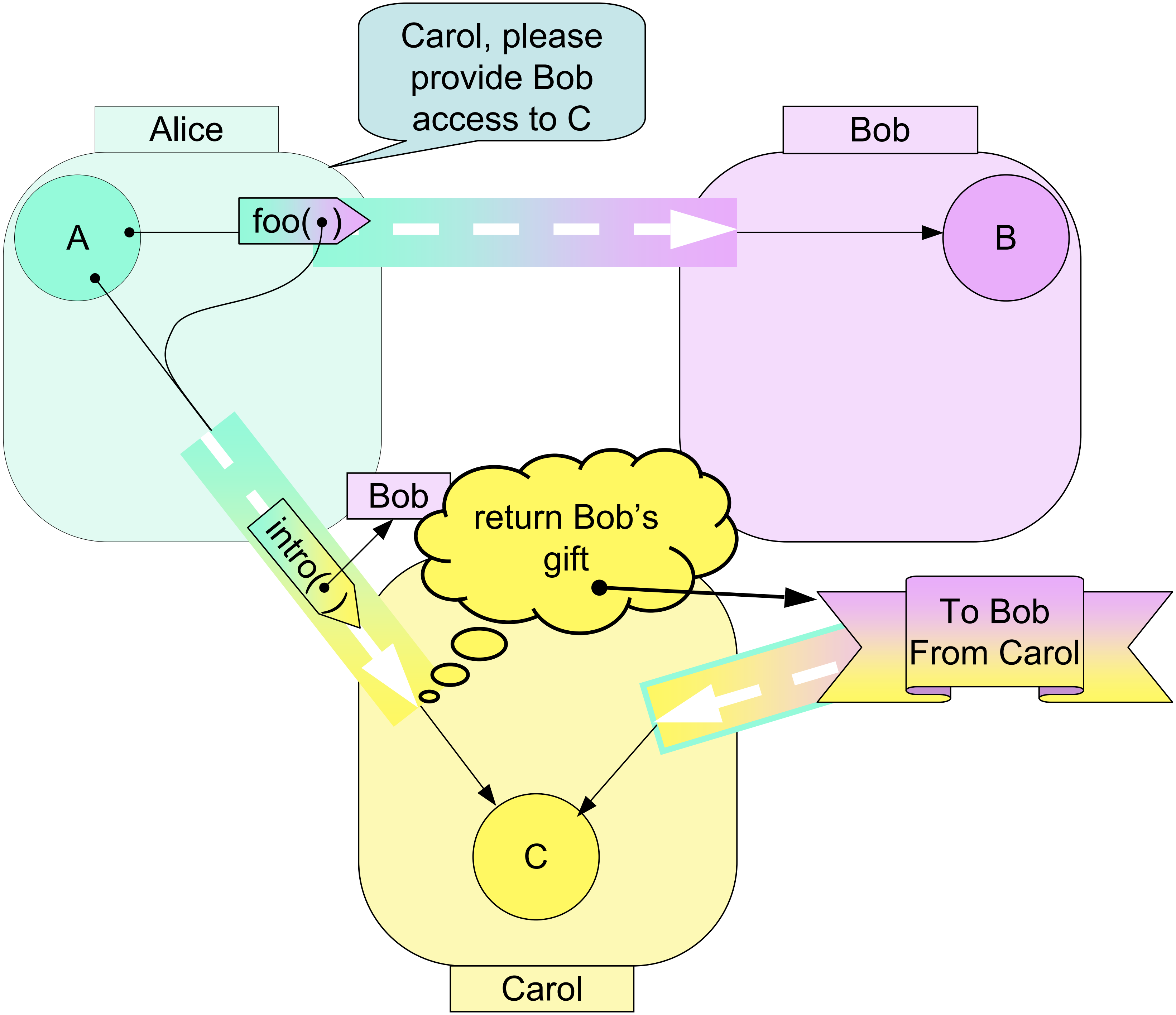

Alice
Bob
A
foo(●)
B
C
Carol

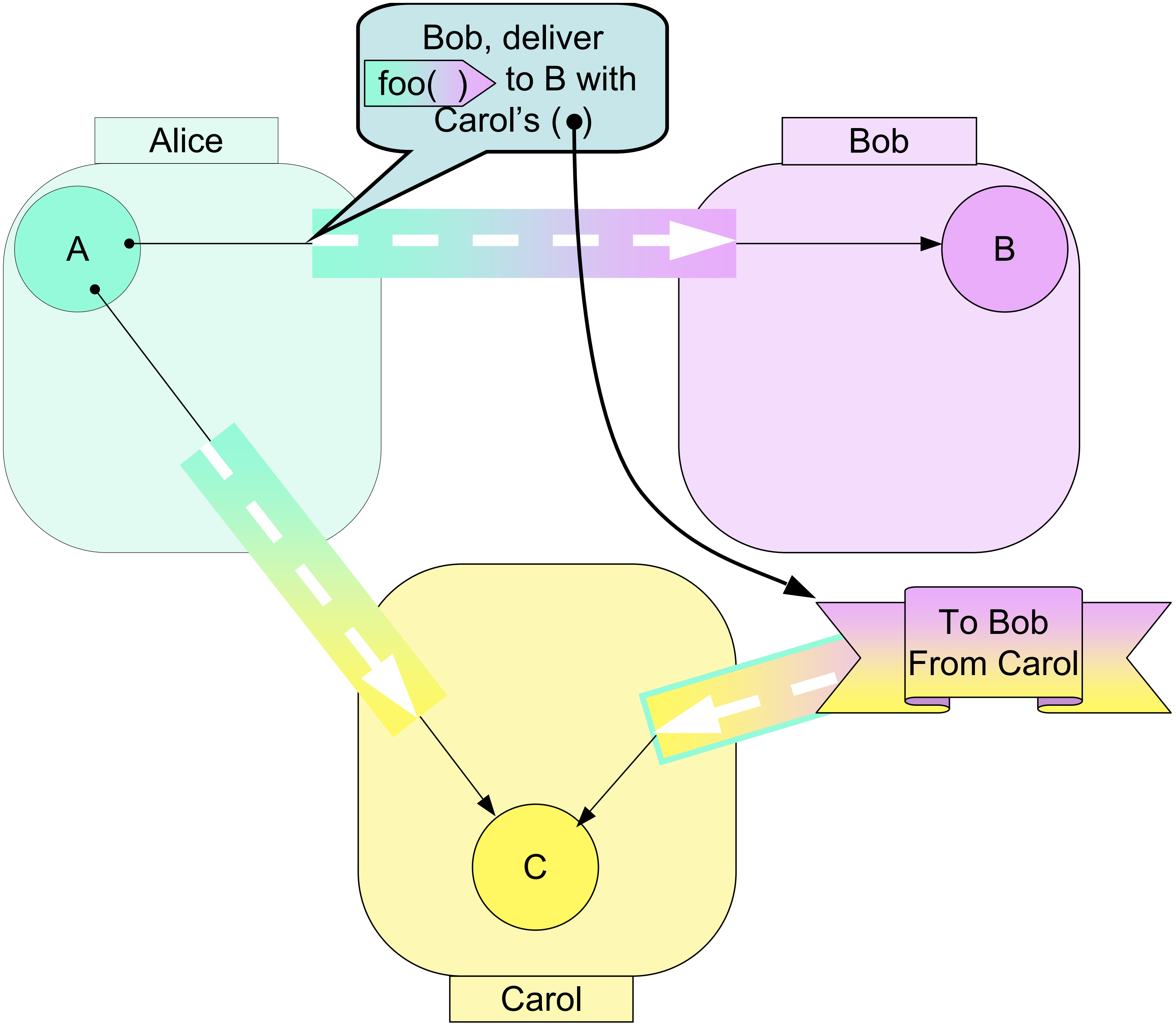

Carol, please provide Bob access to C
Alice
Bob
A
foo( )
B
intro( )
Bob
C
Carol

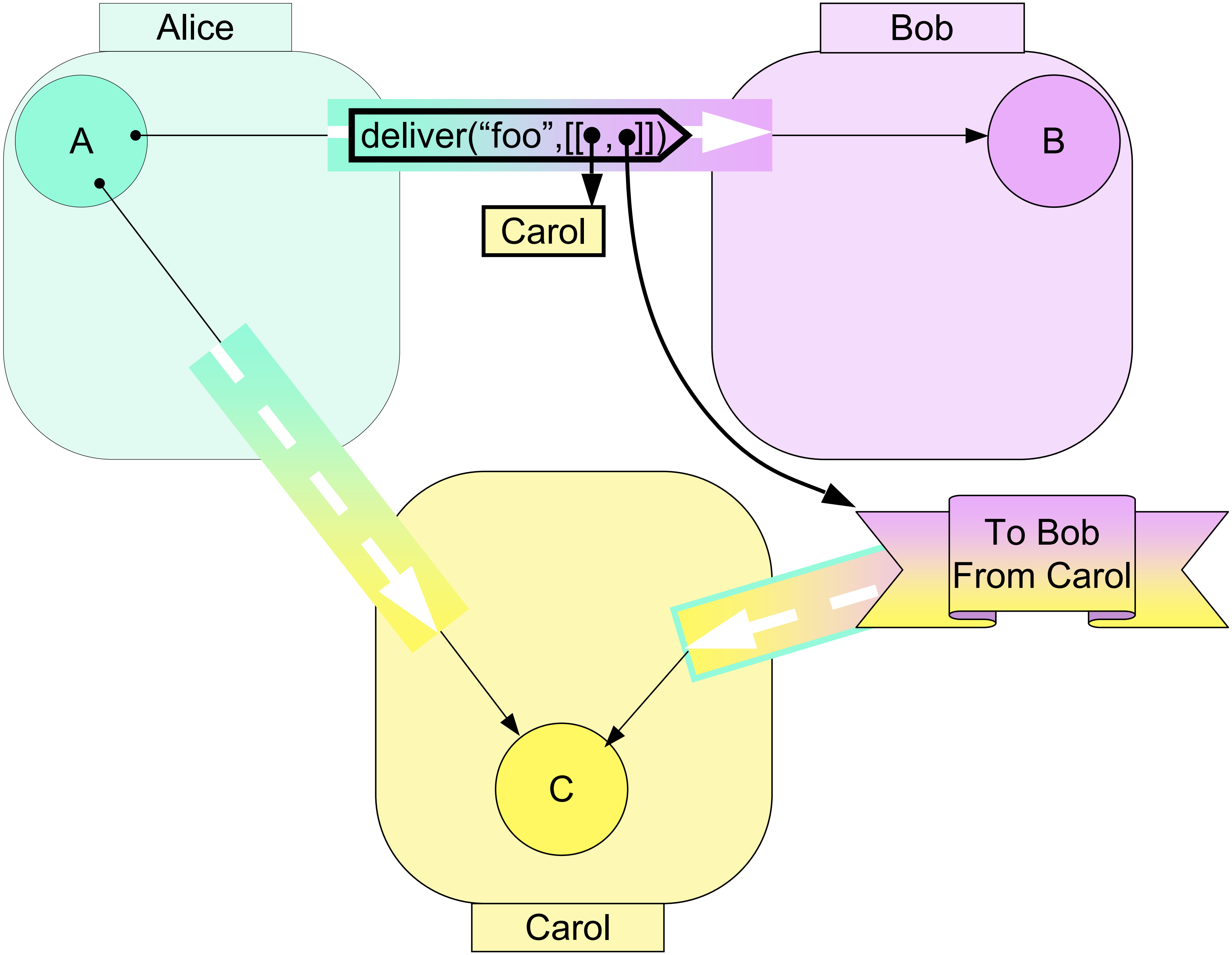

Carol, please provide Bob access to C
Alice
Bob
A
foo( )
B
Bob
Alice needs tunnel for Bob
intro( )
C
Carol

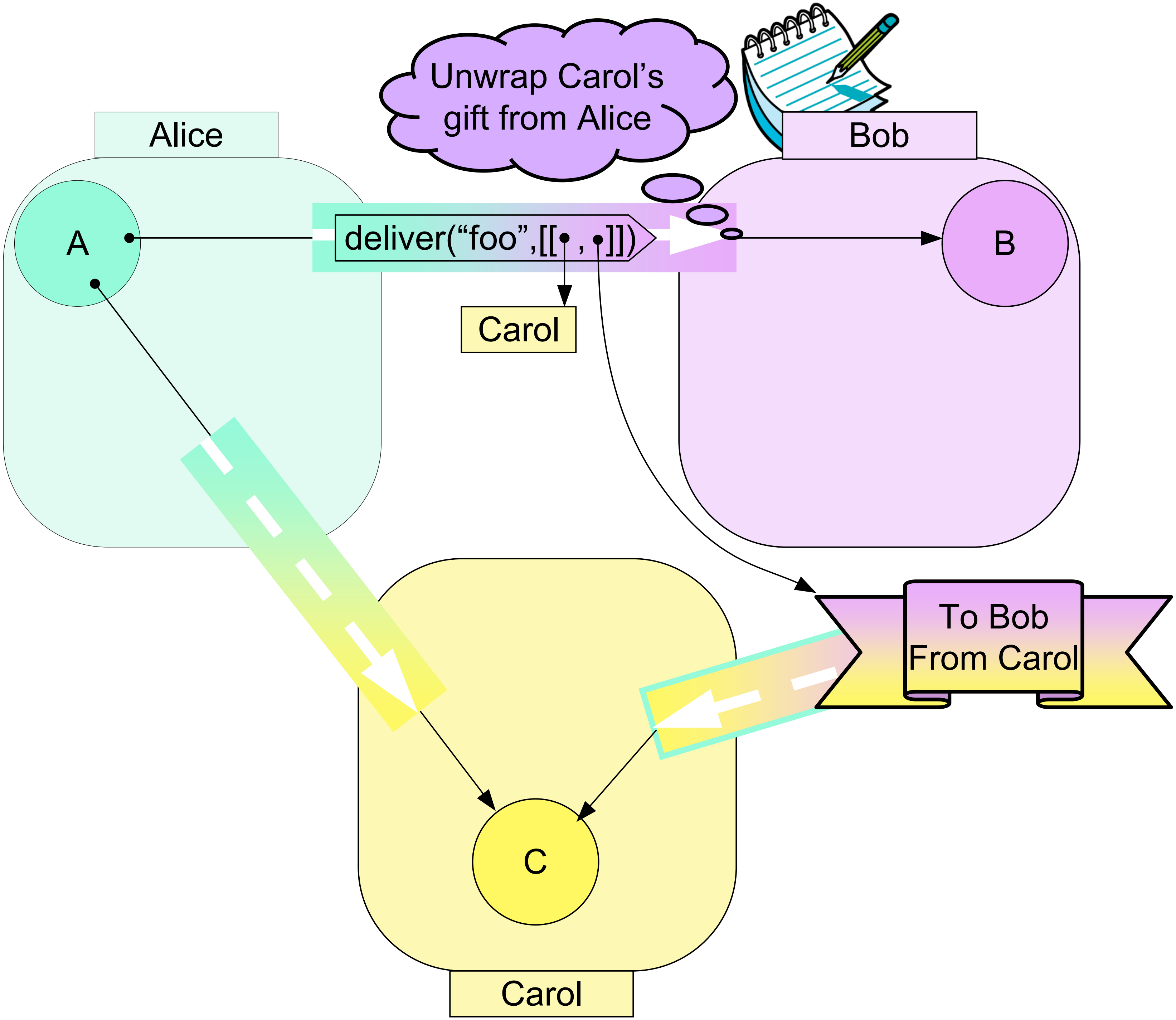

Carol, please provide Bob access to C
Alice
Bob
A
foo( )
B
Bob
Gift wrap it for Bob
intro( )
C
Carol

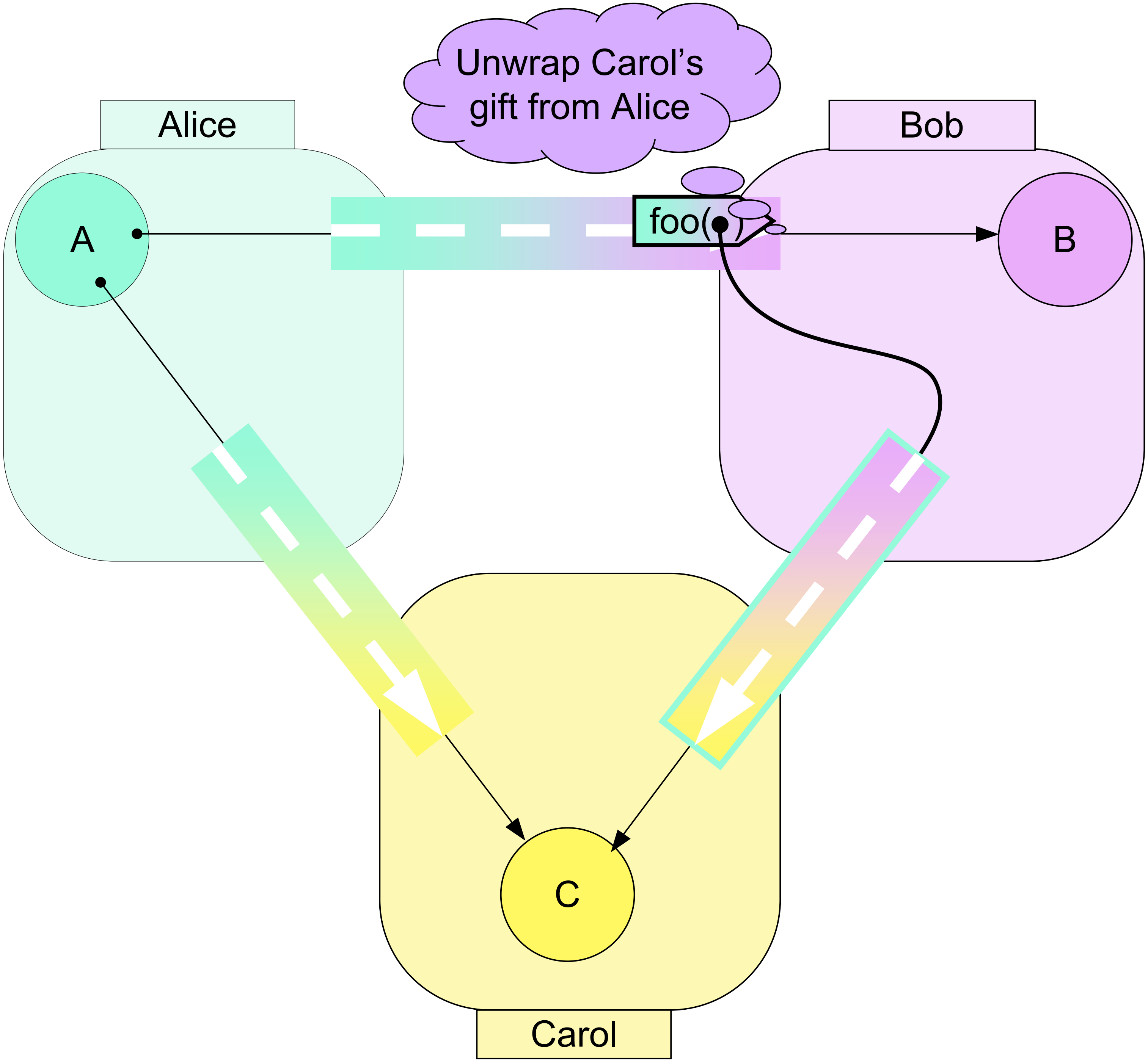

Carol, please provide Bob access to C
Alice
Bob
A
foo( )
B
Bob
intro( )
Gift wrap it for Bob
To Bob From Carol
C
Carol

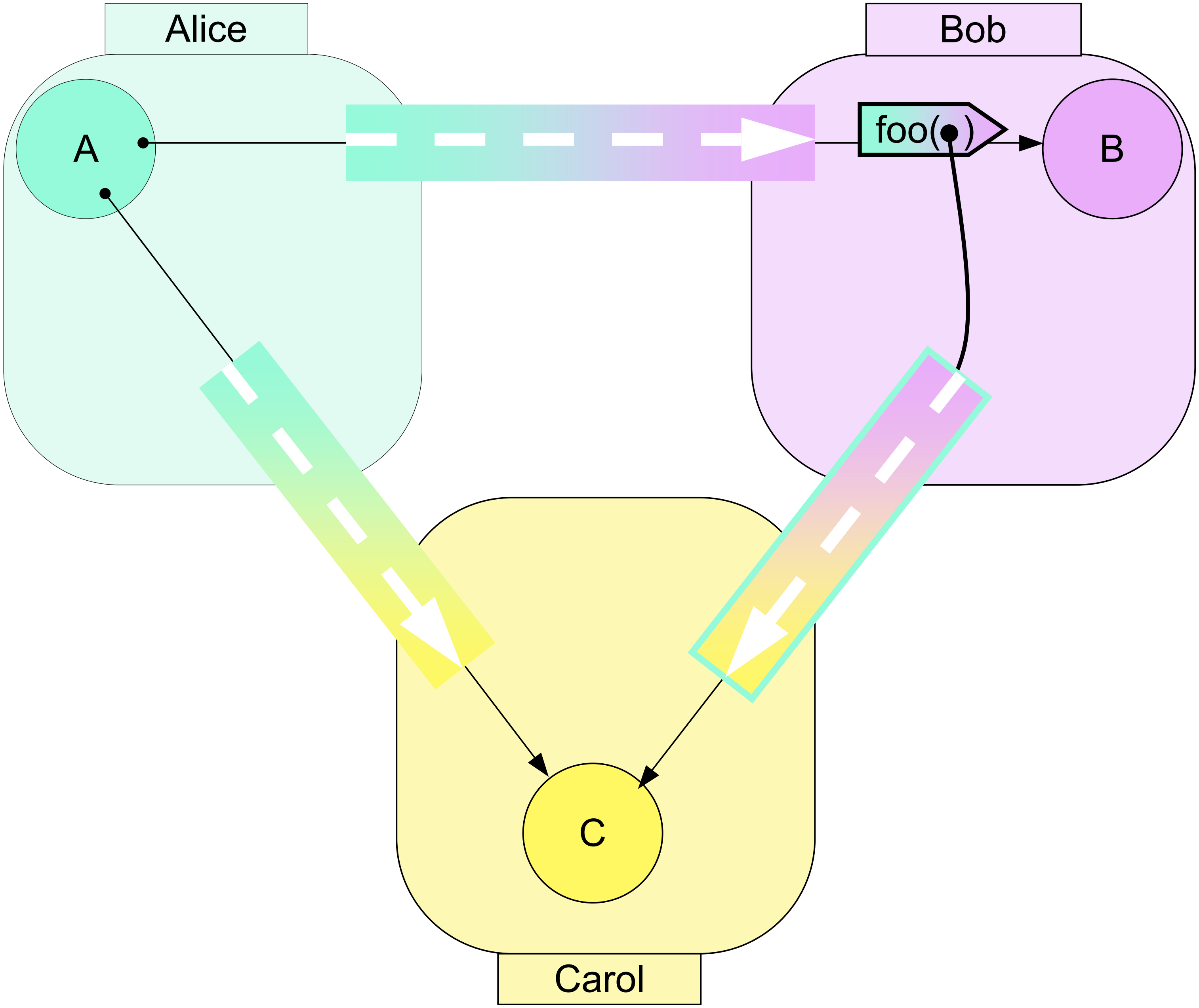

Carol, please provide Bob access to C
Alice
Bob
A
foo( )
B
Bob
return Bob's gift
intro( )
To Bob From Carol
C
Carol

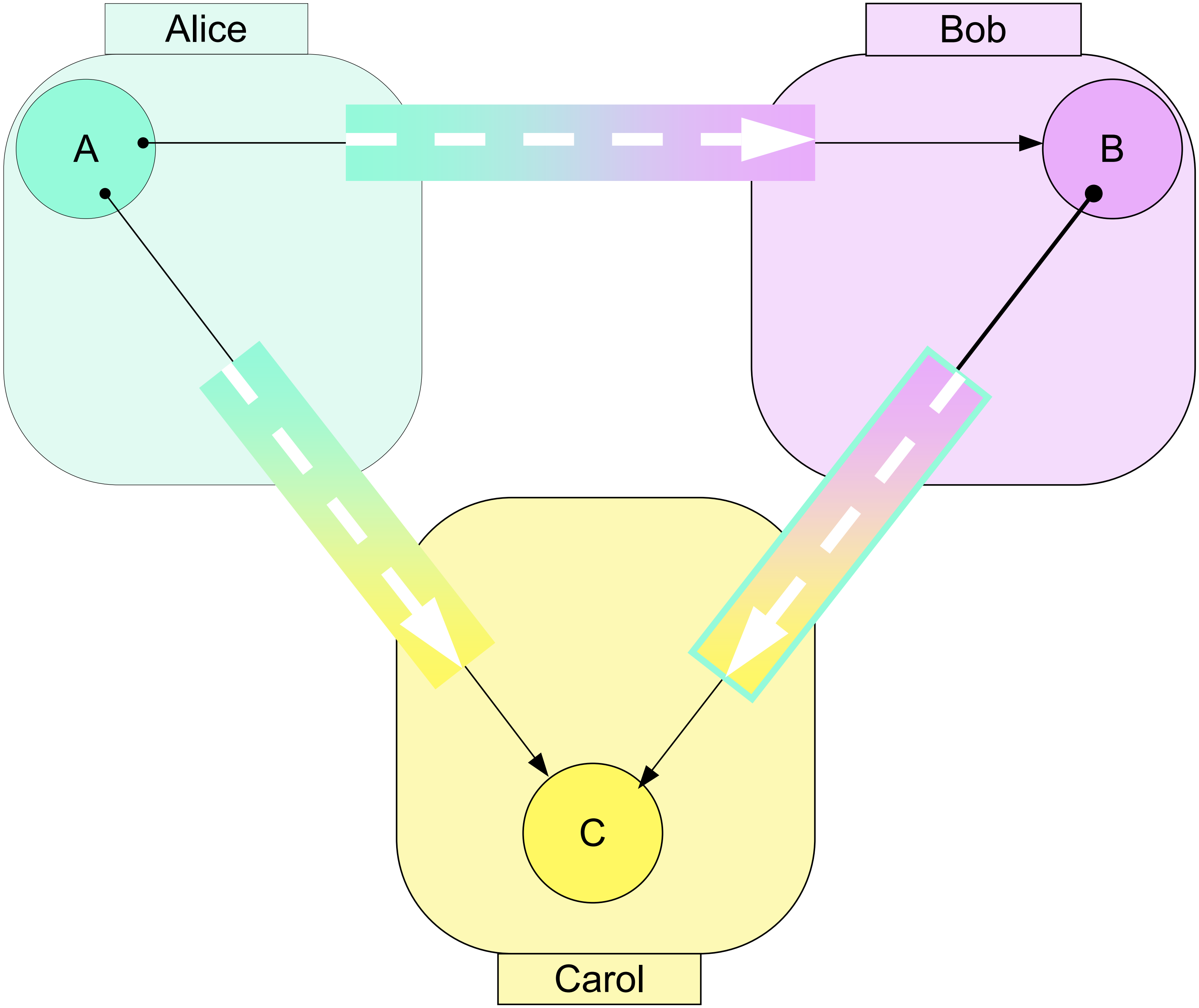

Bob, deliver
foo( ) to B with
Carol's ( )
Alice
A
Bob
B
To Bob
From Carol
C
Carol

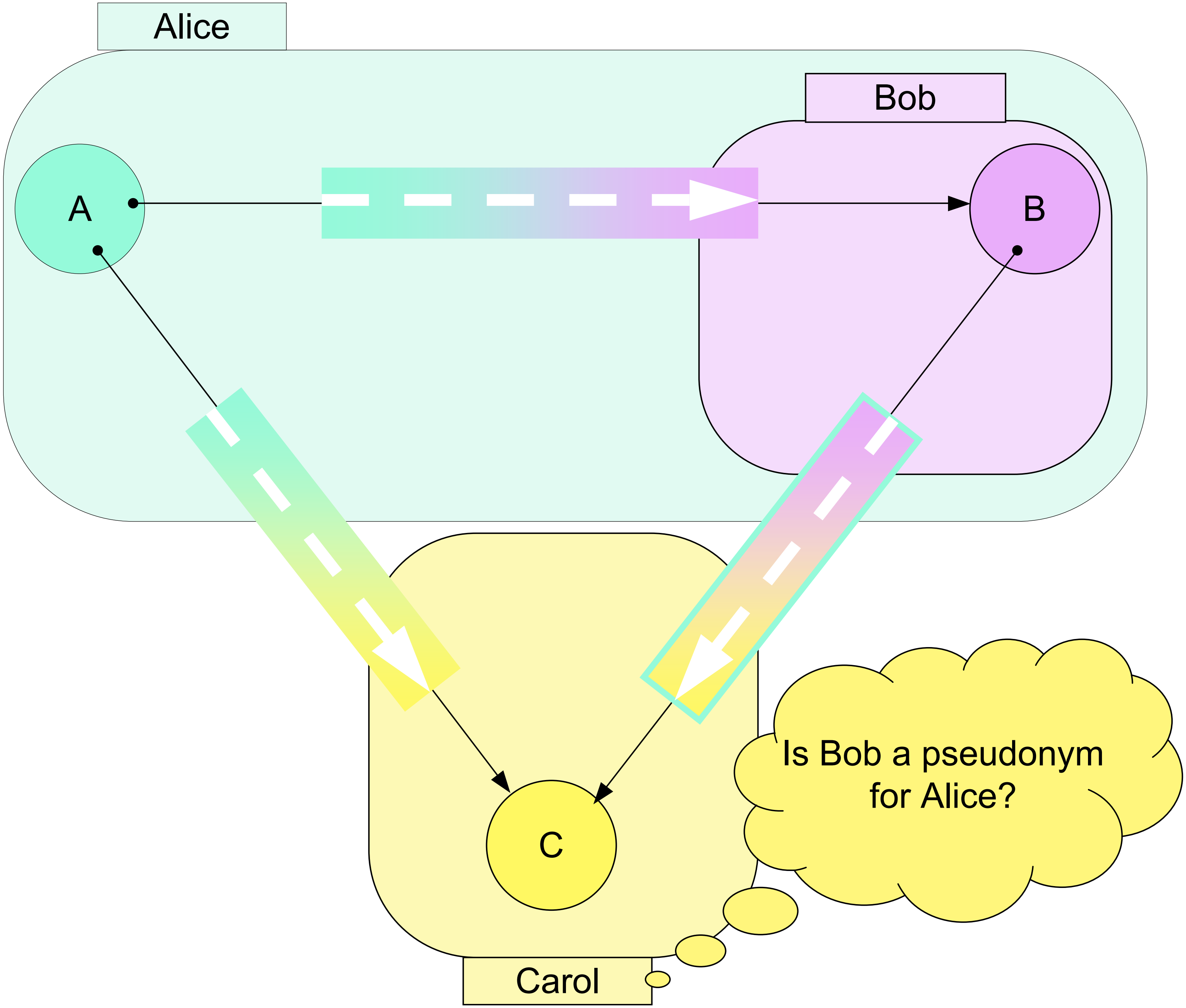

Alice
Bob
A
B
deliver("foo",[[•,•]])
Carol
To Bob
From Carol
C
Carol


Unwrap Carol's gift from Alice
Alice
Bob
A
deliver("foo",[[• , •]])
B
Carol
To Bob From Carol
C
Carol


Unwrap Carol's gift from Alice
Alice
Bob
A
foo( )
B
C
Carol

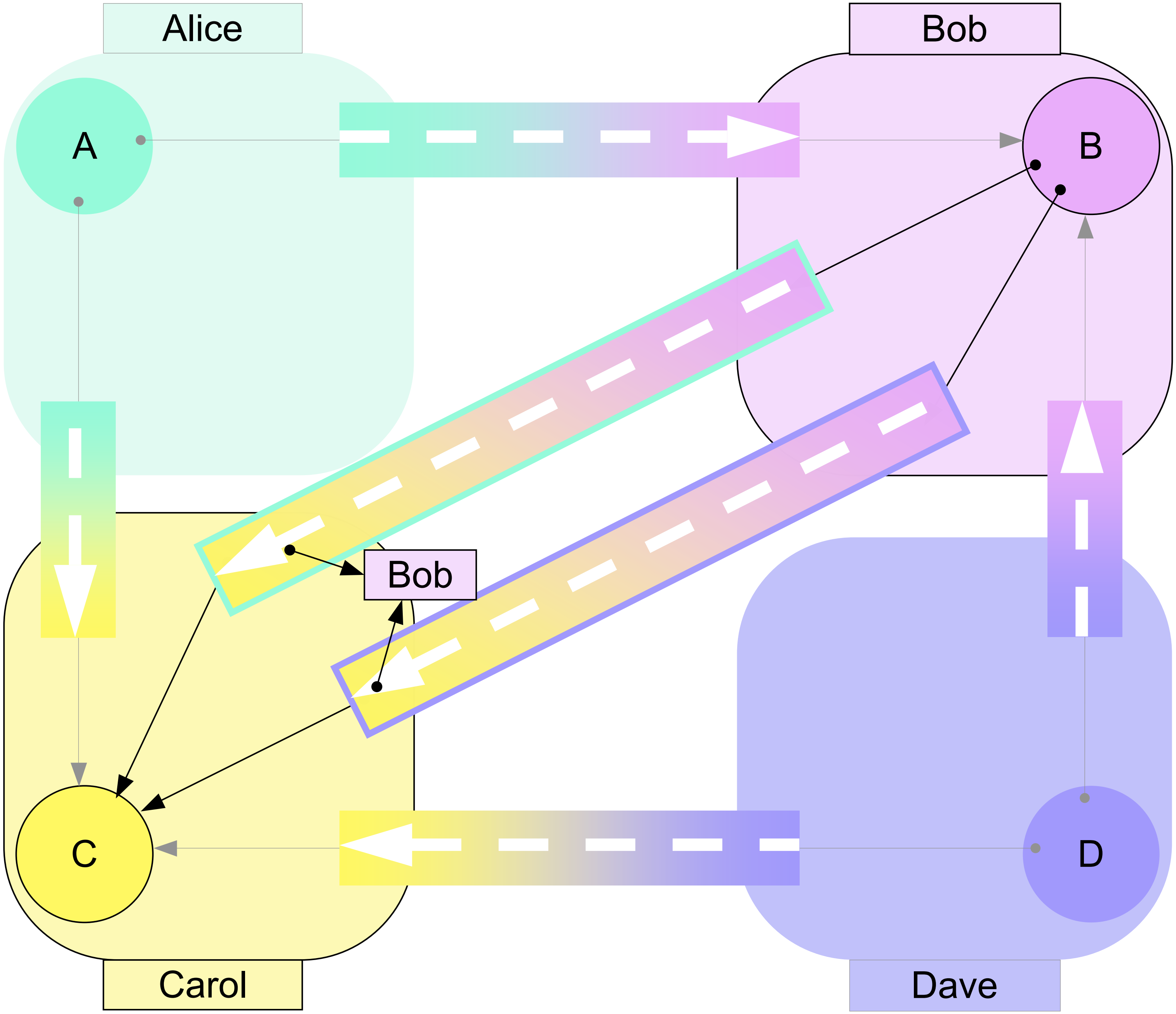

Alice
Bob
A
foo( )
B
C
Carol


Alice
Bob
A
B
C
Carol


Alice
Bob
A
B
C
Carol
Is Bob a pseudonym for Alice?

Four party intermediation

Only corroborating introductions
let Alice shed blame


Alice
Bob
A
B
Bob
C
D
Carol
Dave

Better Identities than ACLs

Fully decentralized

- No global administrator or name server

Track bilateral responsibility

- For requests and for service
- Also tracks delegation chain

Sybil resistant aggregation strategy

Corroboration-driven disaggregation

Conclusions

Delegate authority, bound to responsibility for using that authority.

Fine-grain least authority for safety.

Large-grain identities for damage control.

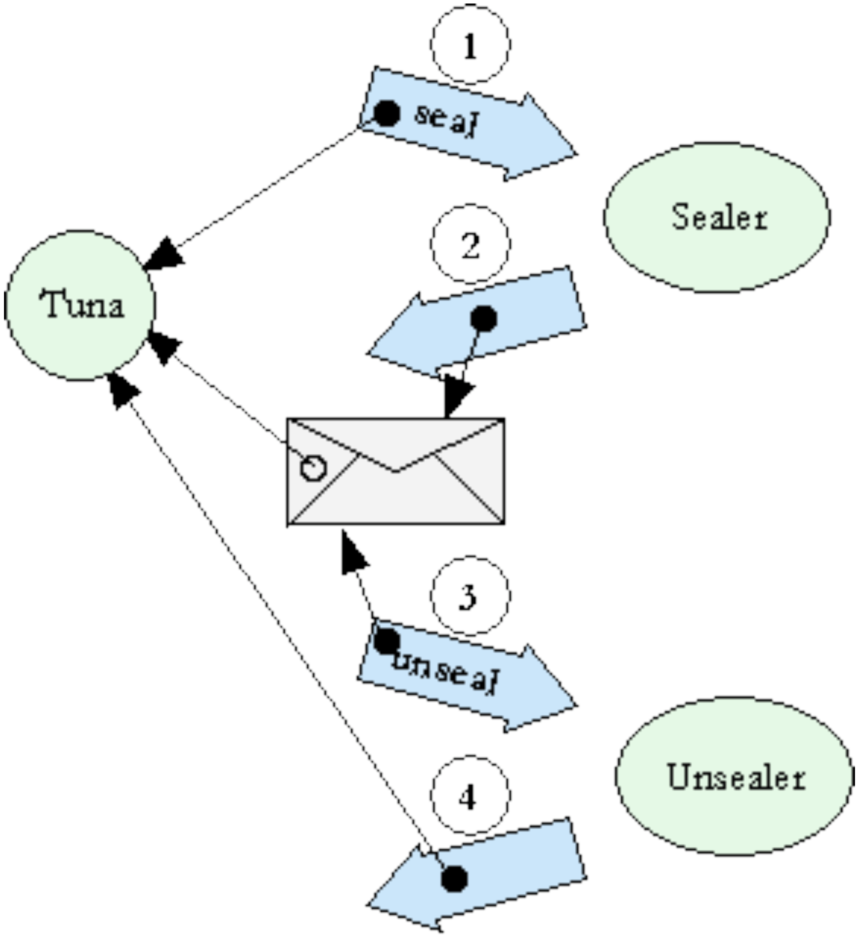
Reference implementations in Java & E:

http://erights.org/download/horton/

Three-party intermediation

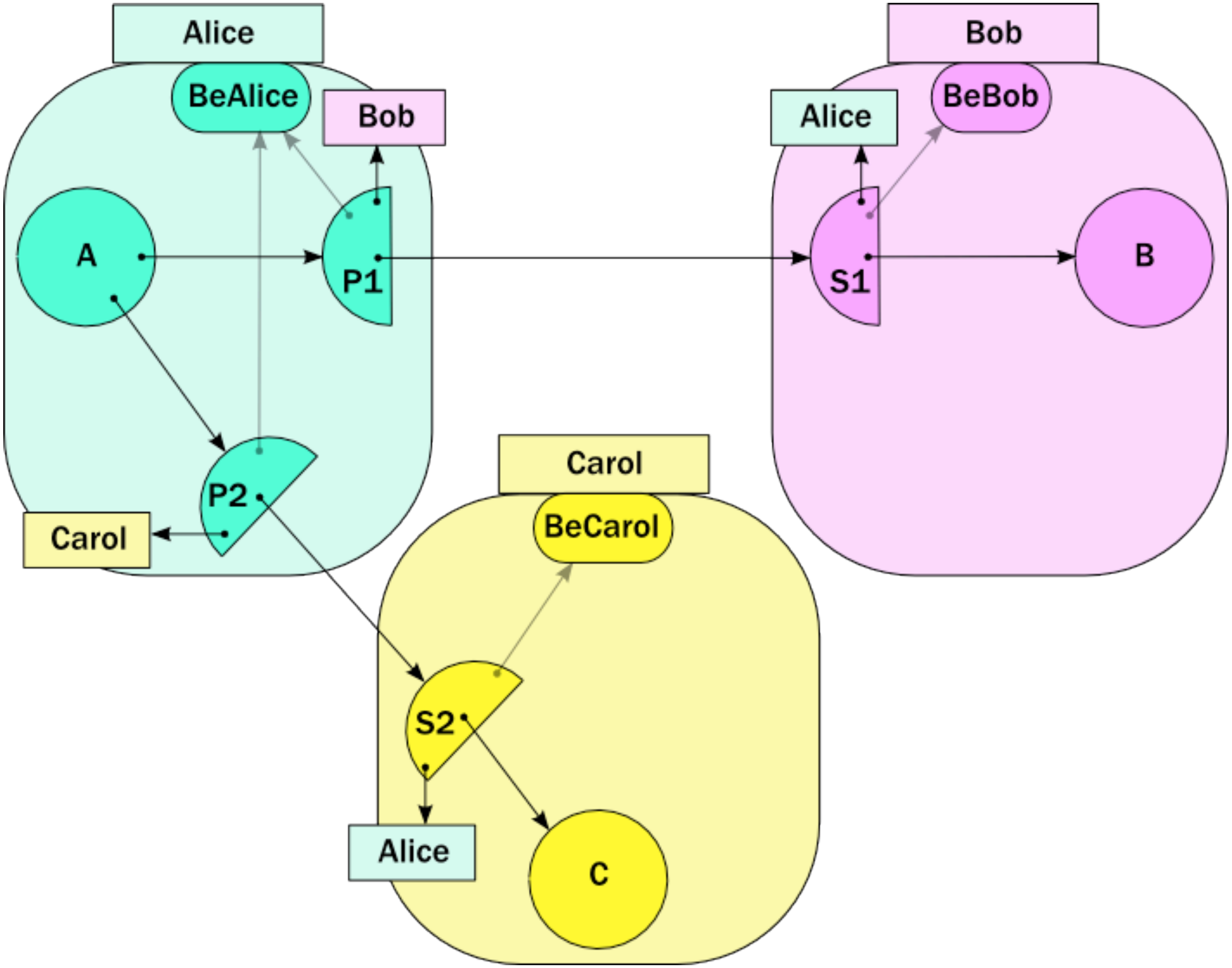
The details

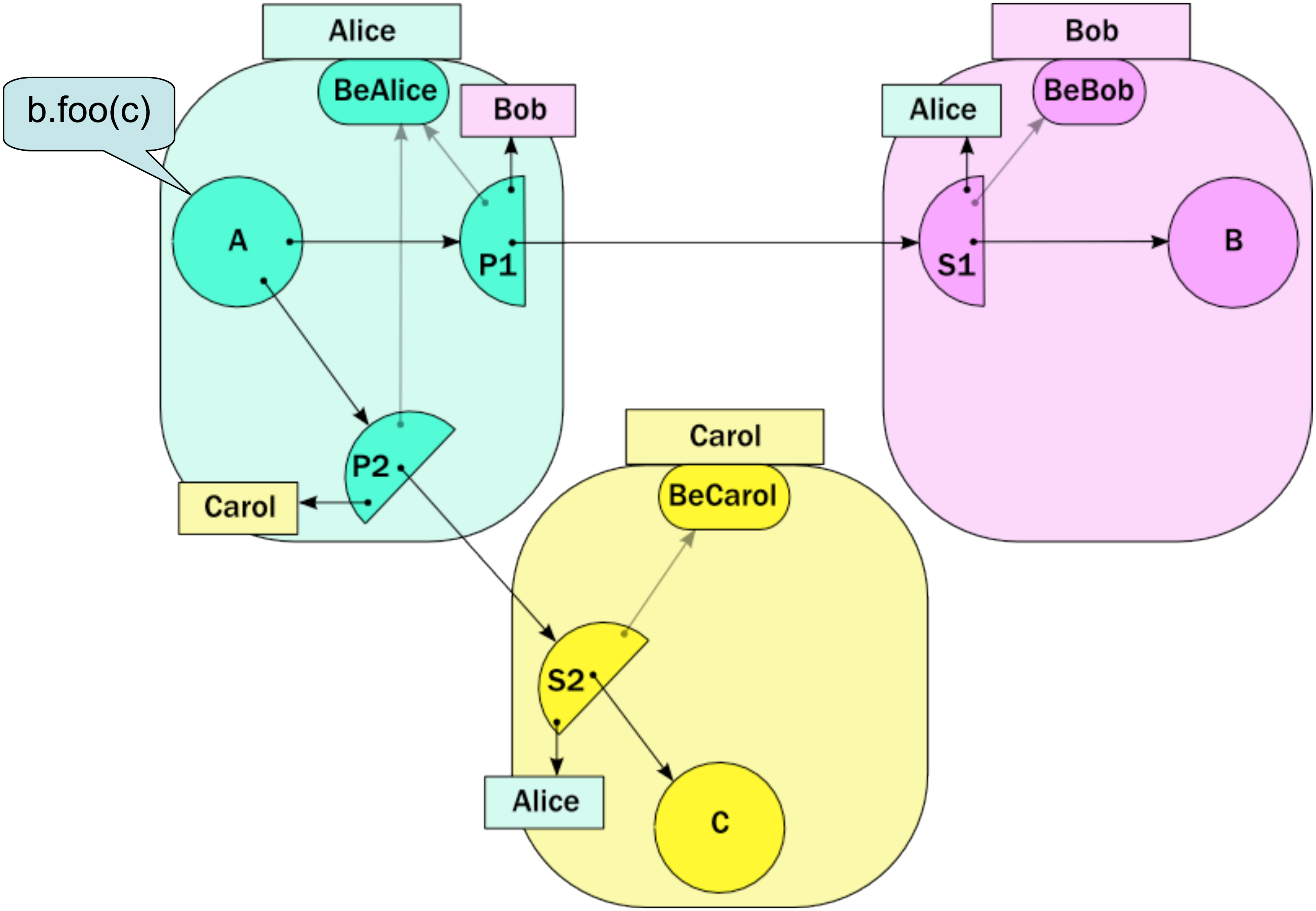
Rights Amplification

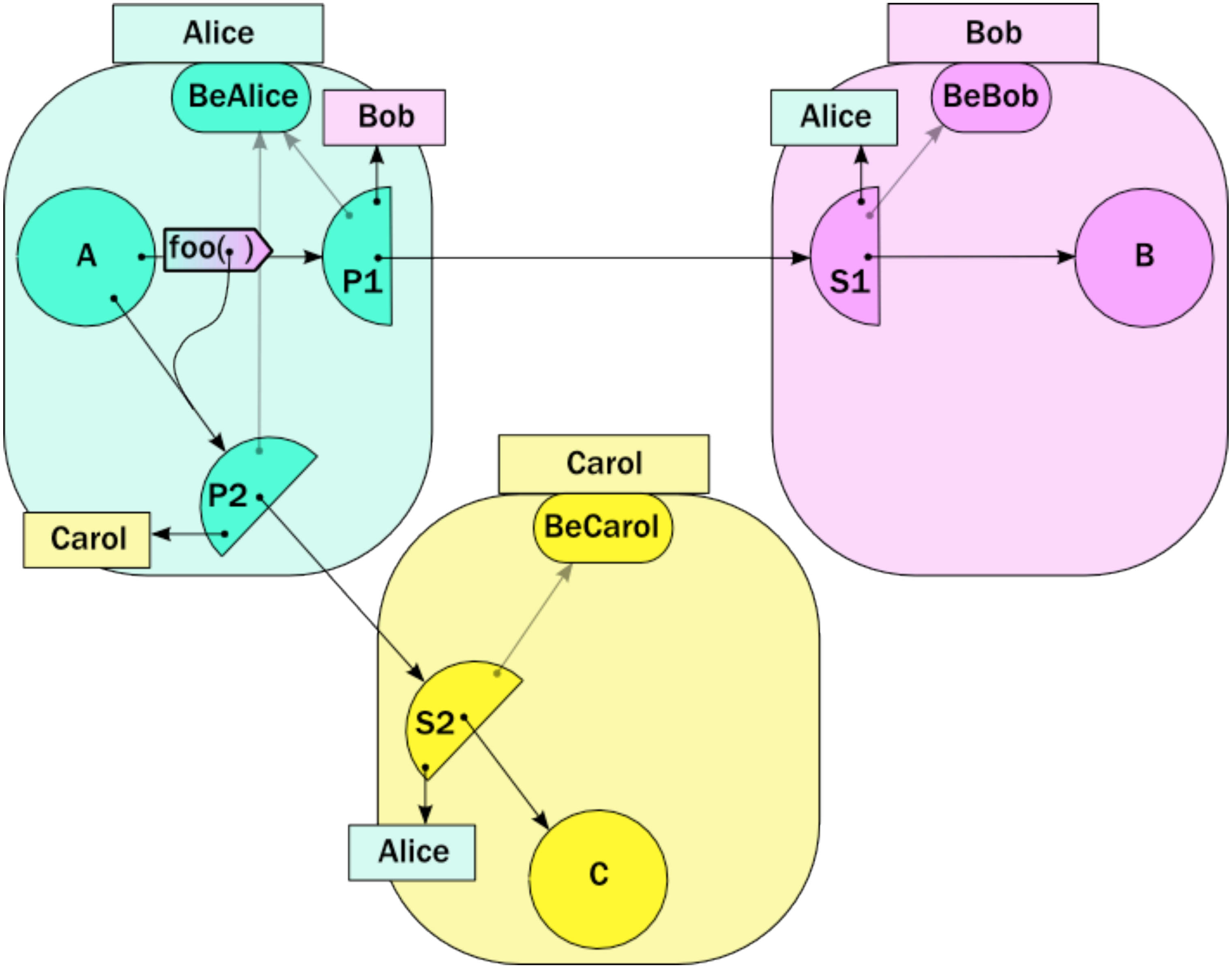


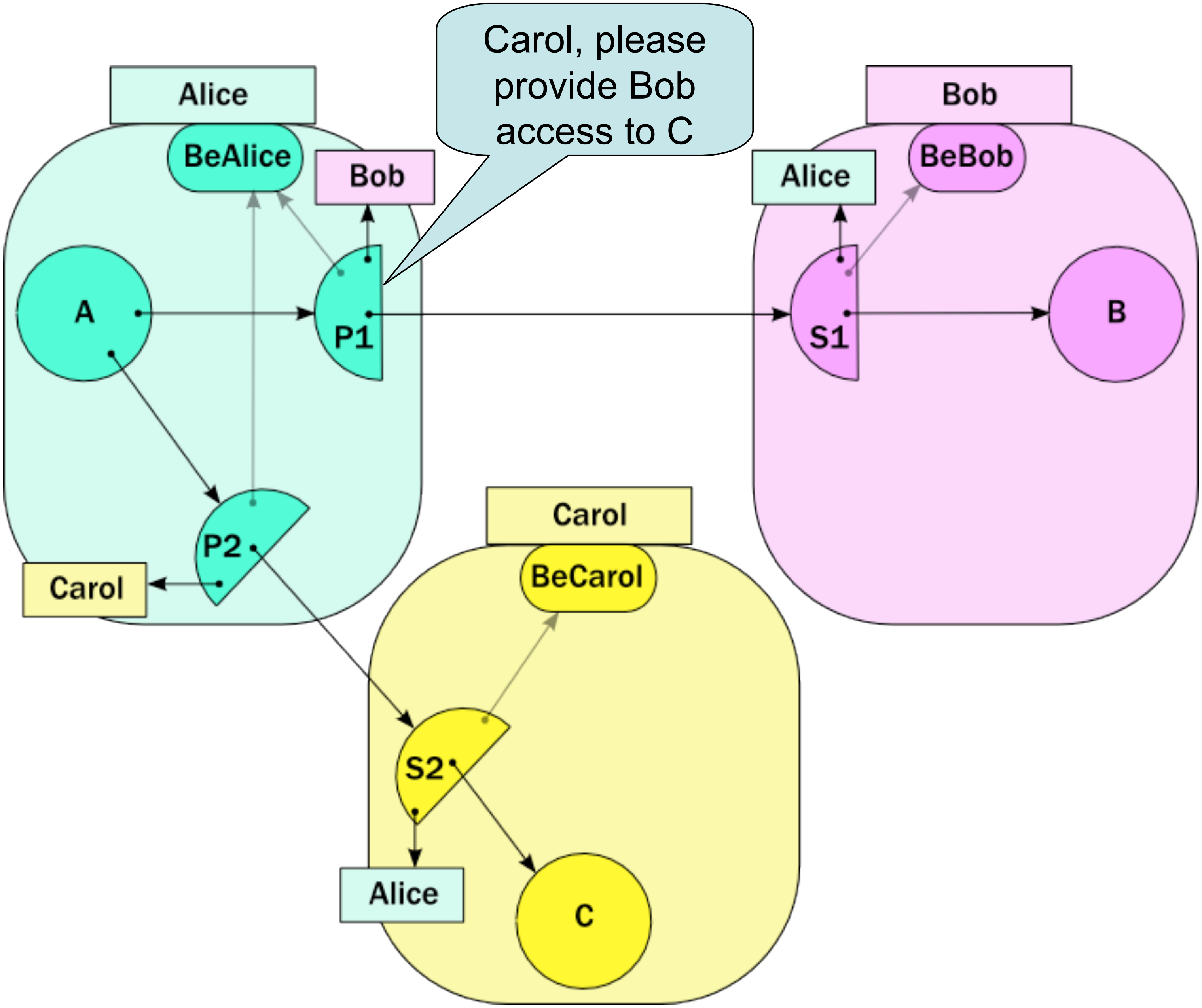
- Inspired by PK
- Simple oo pattern
- No explicit crypto

- Can represent responsible identity

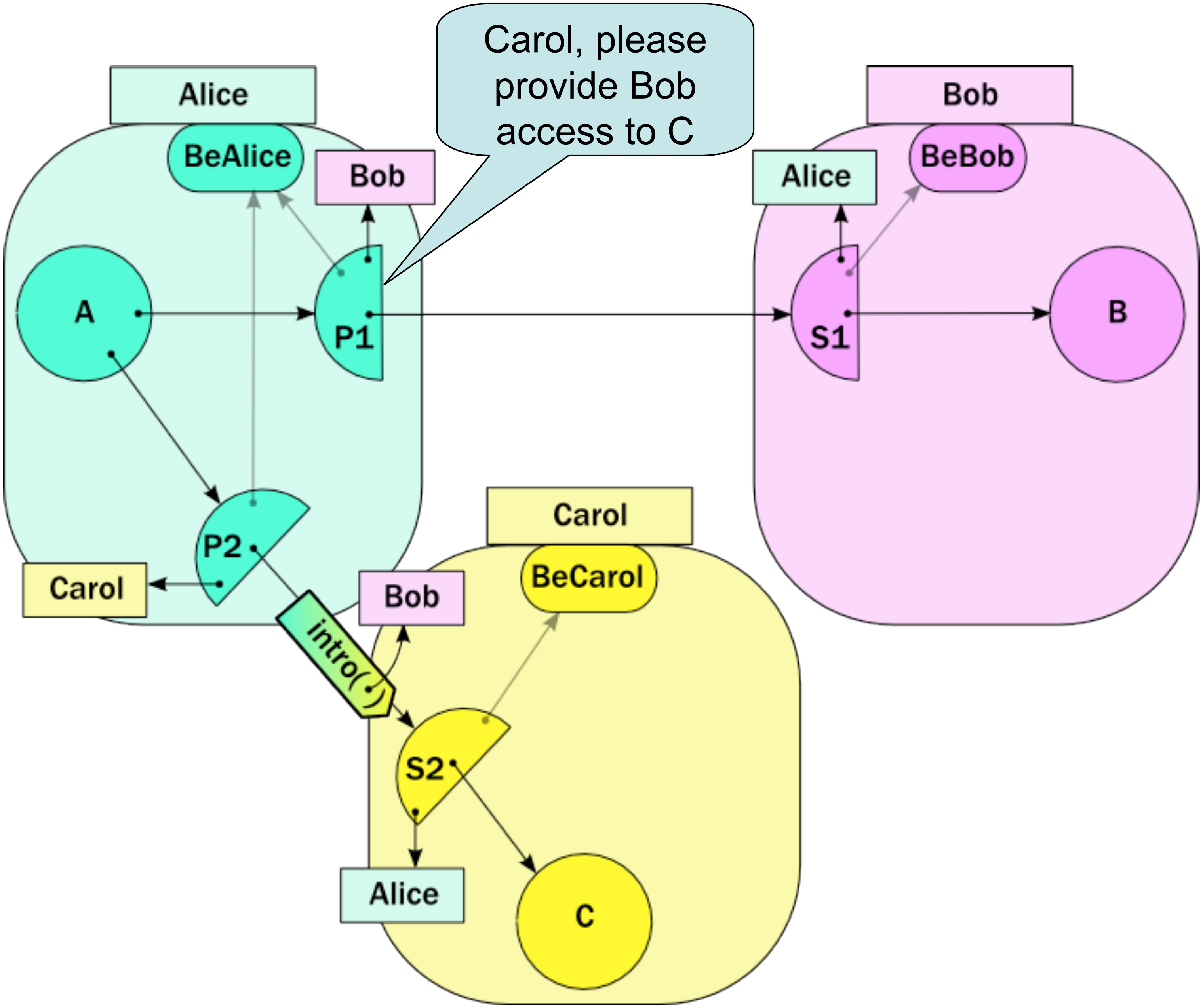

Alice
BeAlice
Bob
A
P1
P2
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
S2
Alice
C


b.foo(c)
Alice
BeAlice
Bob
A
P1
P2
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
S2
Alice
C

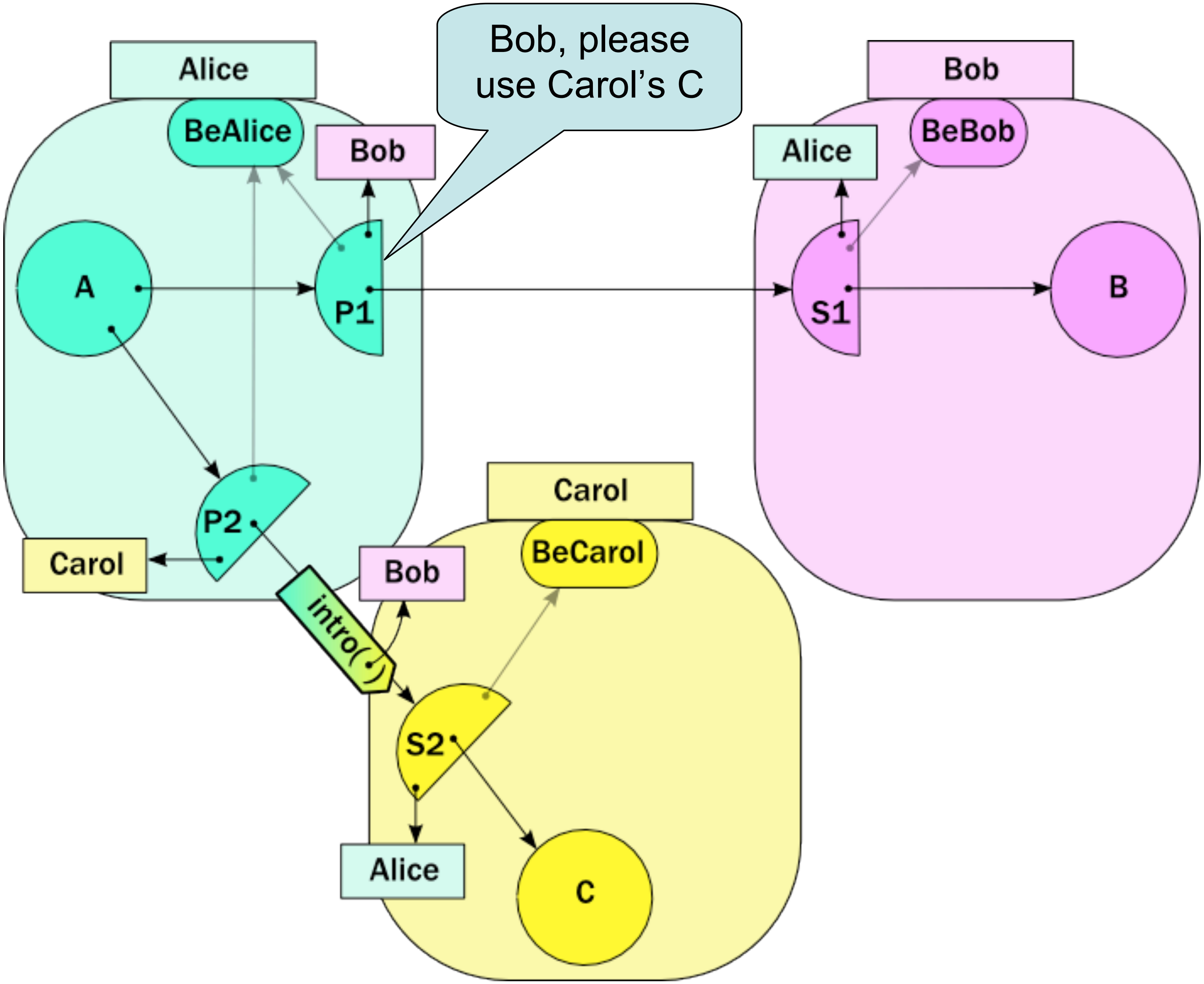

Alice
BeAlice
Bob
A
foo( )
P1
P2
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
S2
Alice
C

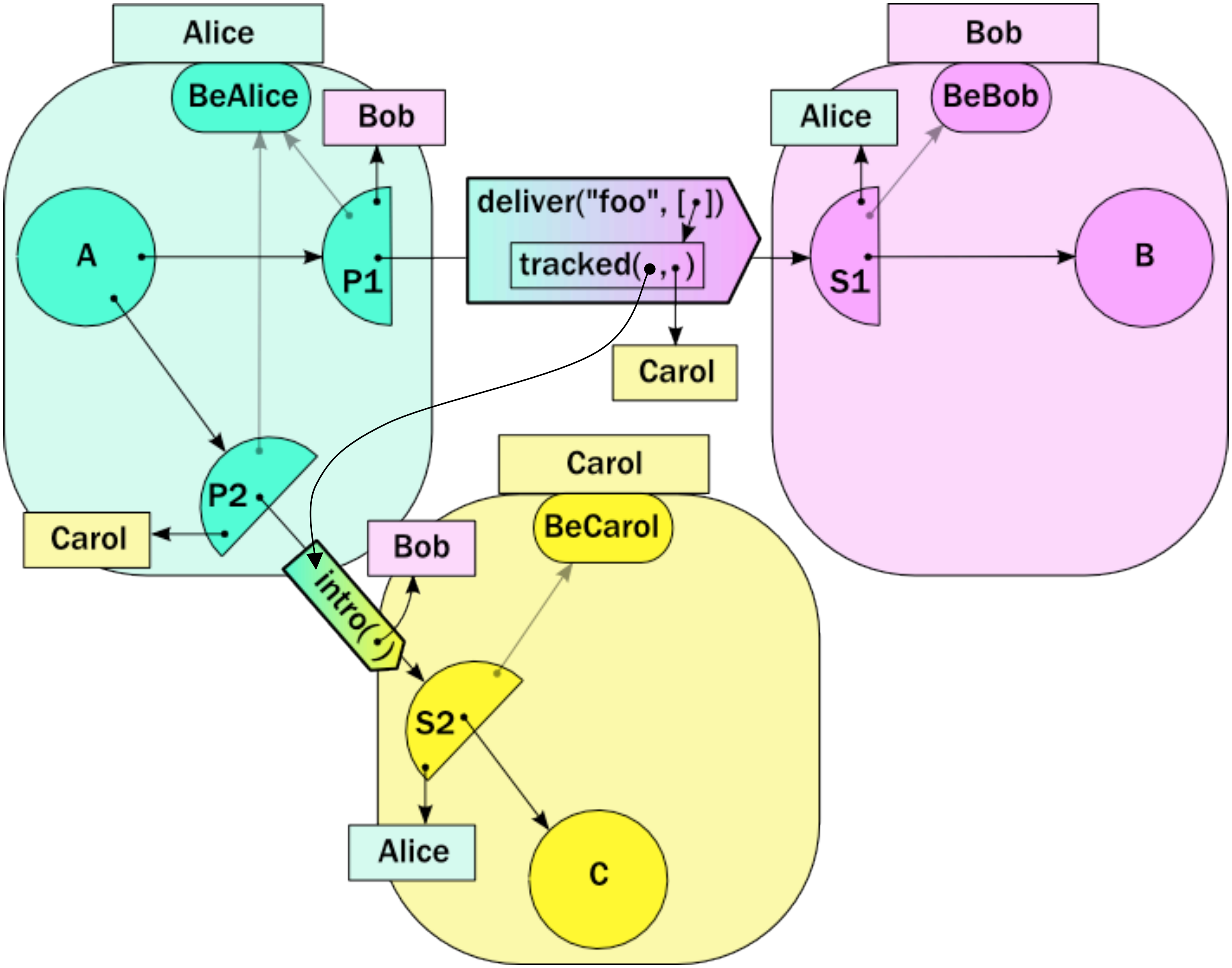


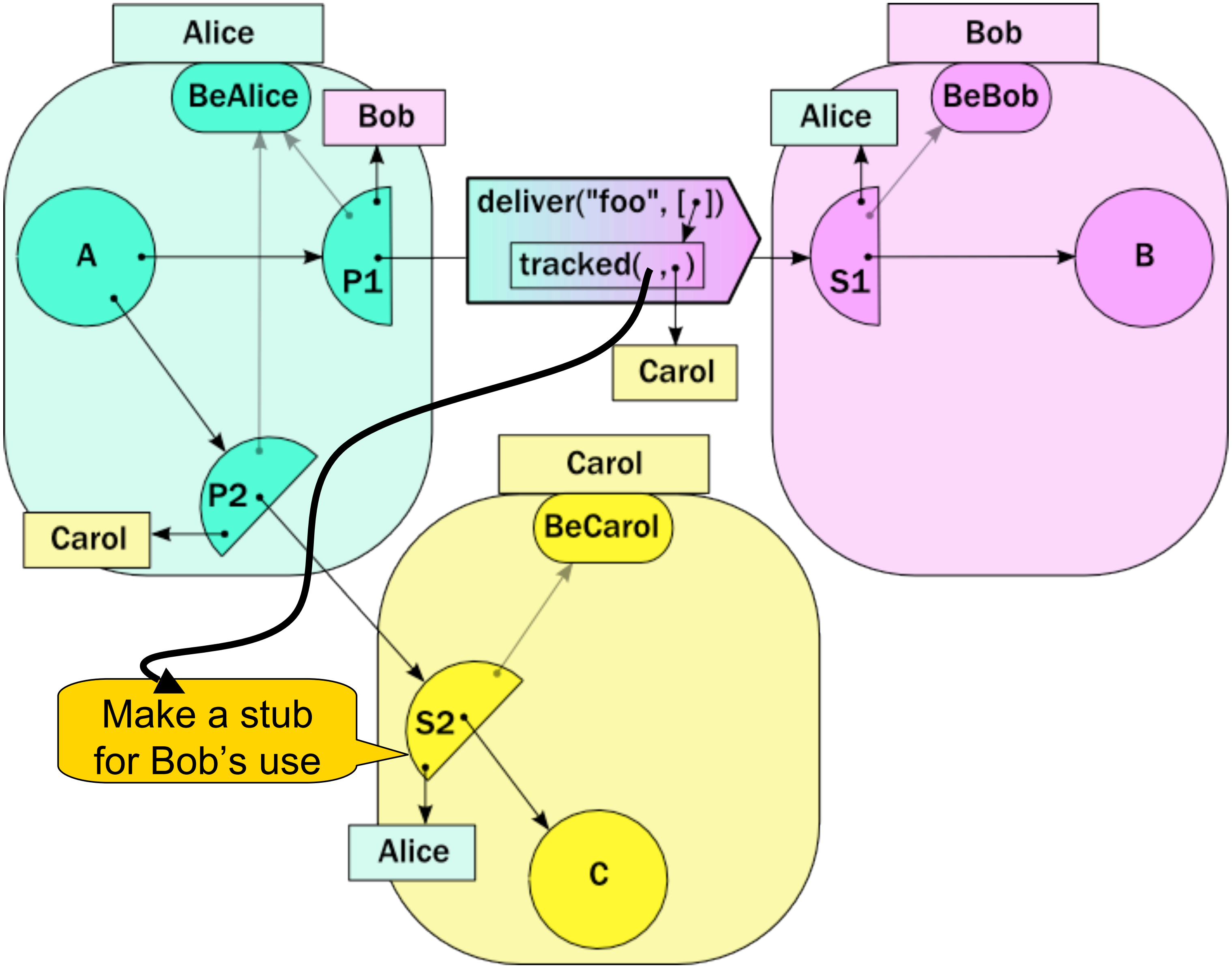
Carol, please provide Bob access to C



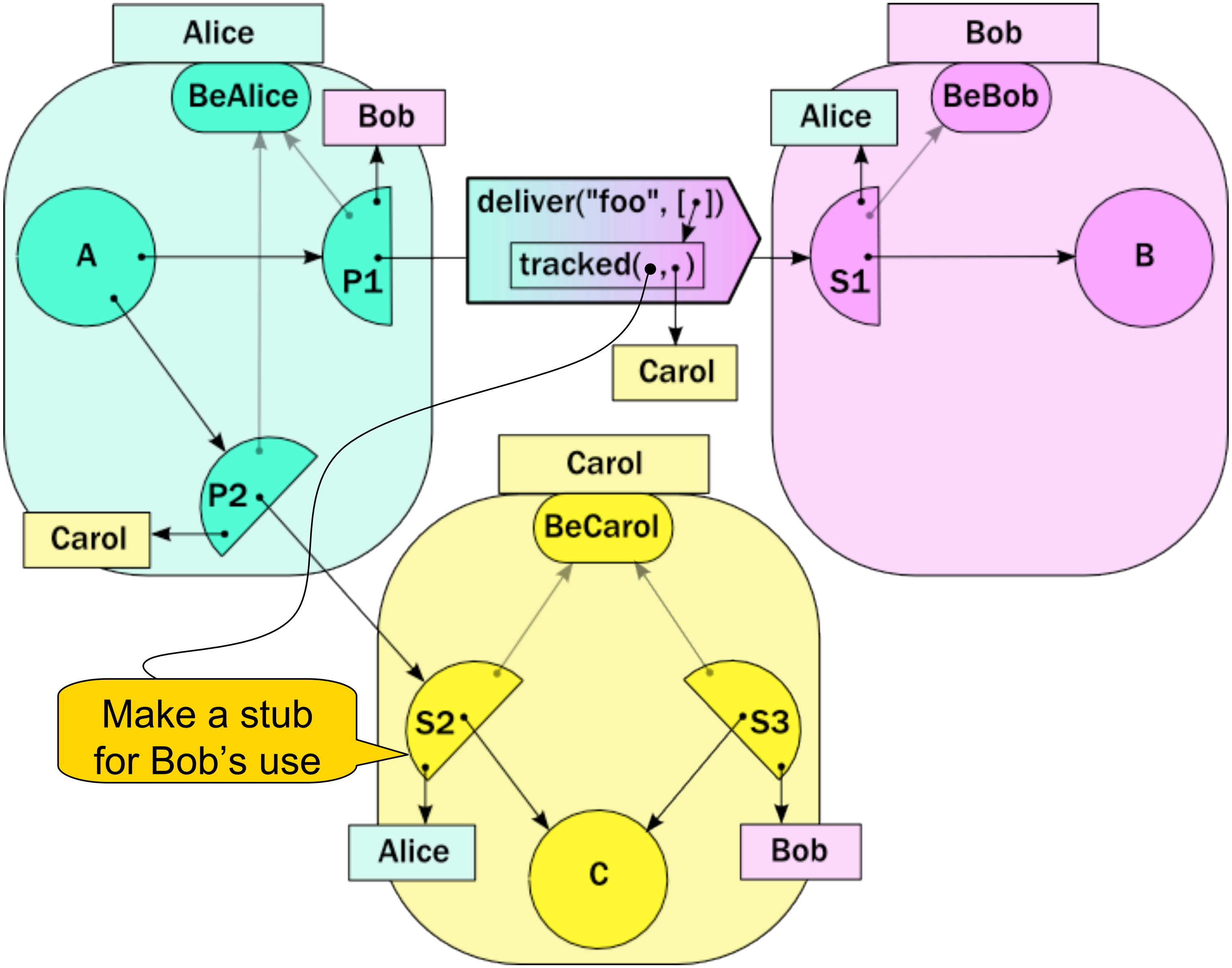
Carol, please provide Bob access to C


Bob, please use Carol's C
Alice
BeAlice
Bob
A
P1
P2
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
Bob
intro( )
S2
Alice
C

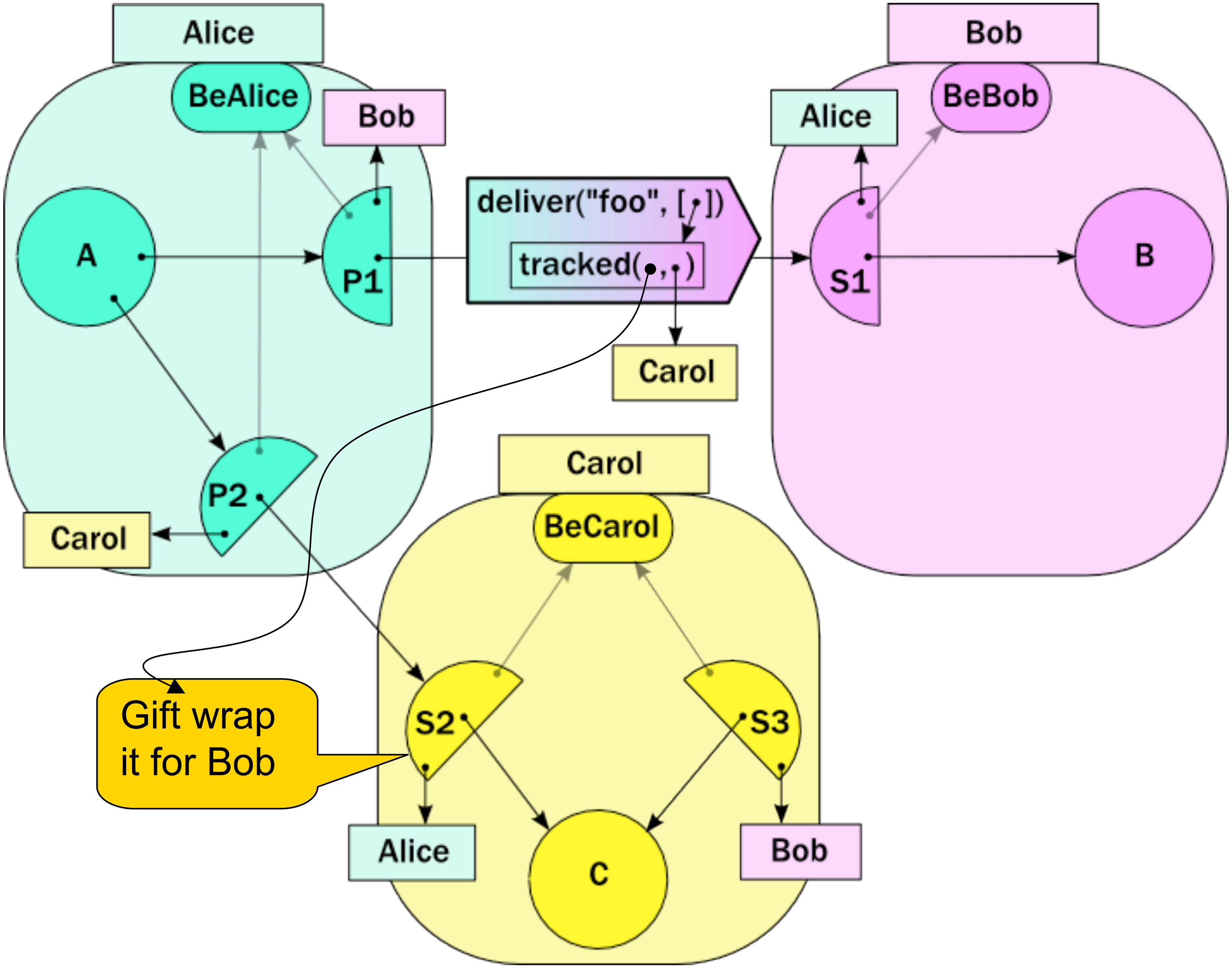

Alice
BeAlice
Bob
A
P1
deliver("foo", [ ])
tracked(,)
Carol
Bob
BeBob
Alice
S1
B
P2
Carol
Carol
BeCarol
Bob
intro( )
S2
Alice
C

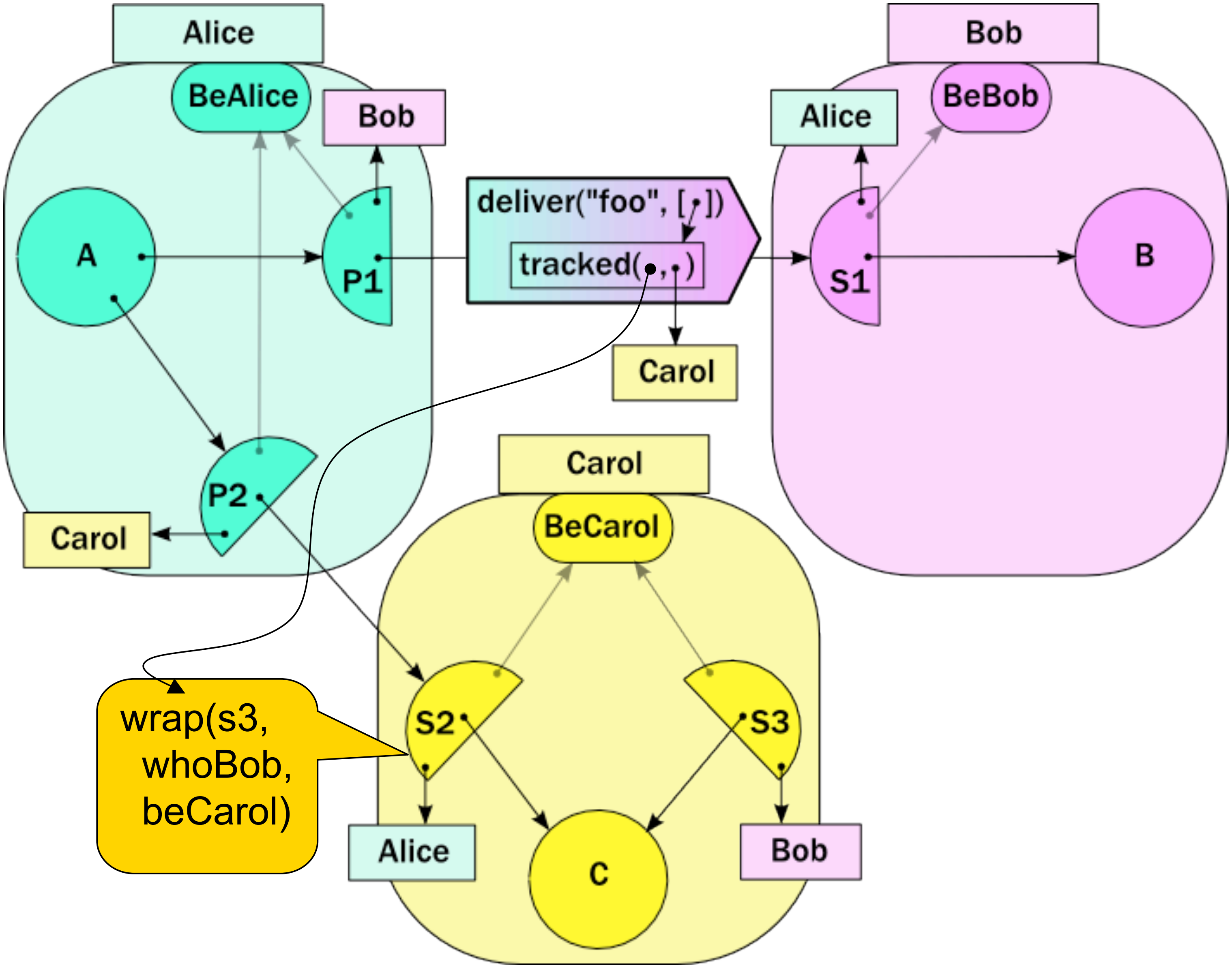


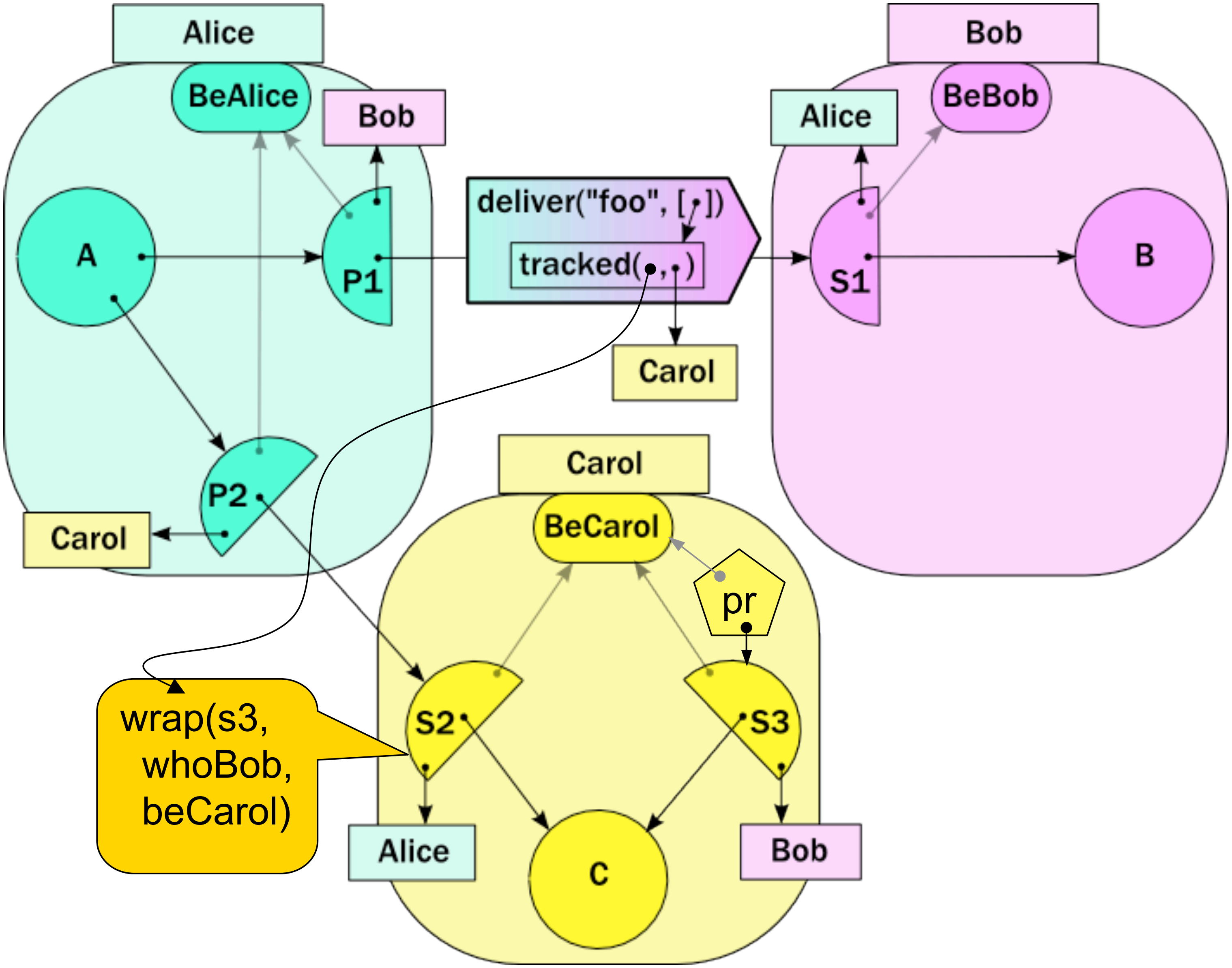
Make a stub for Bob’s use

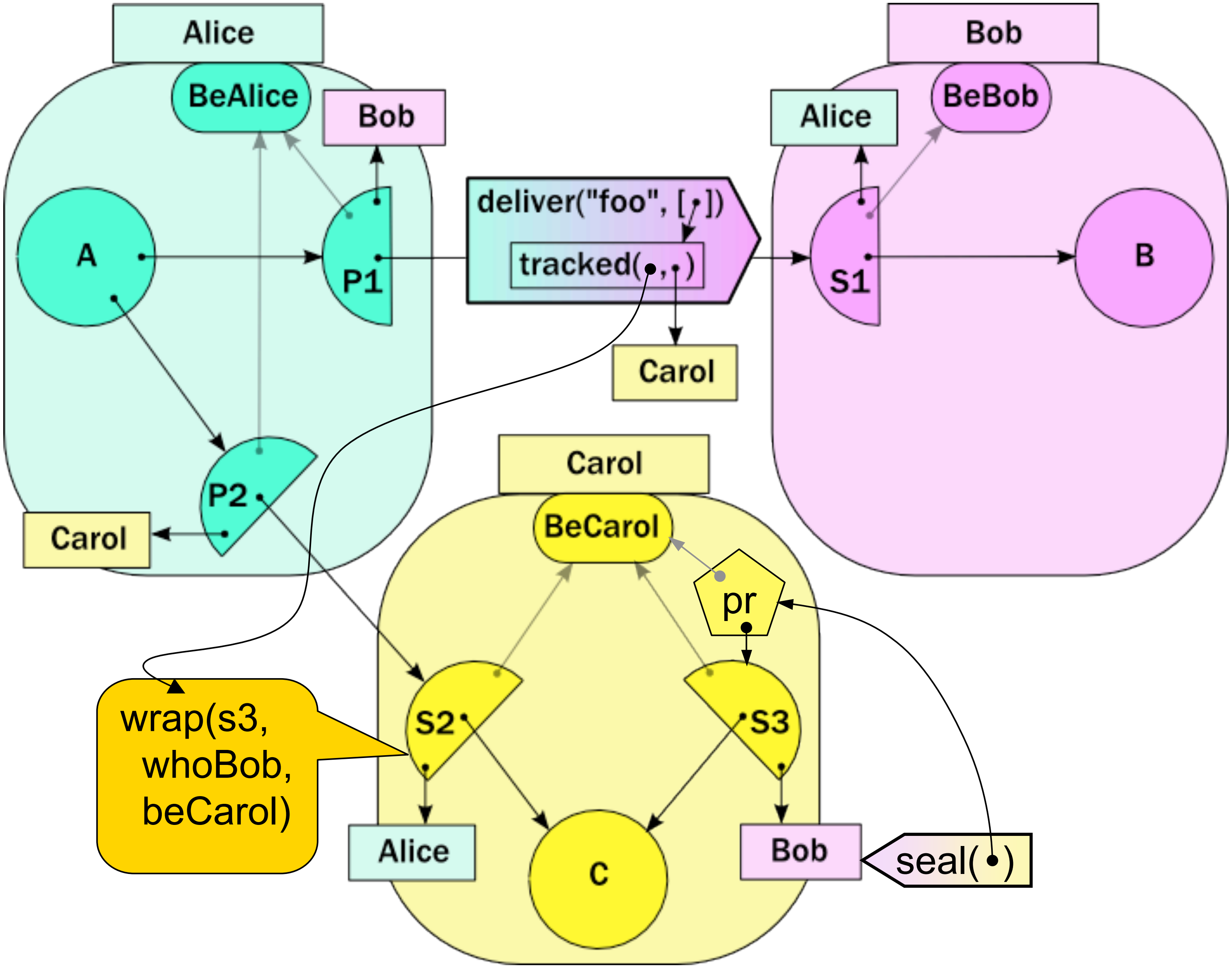


Make a stub for Bob's use

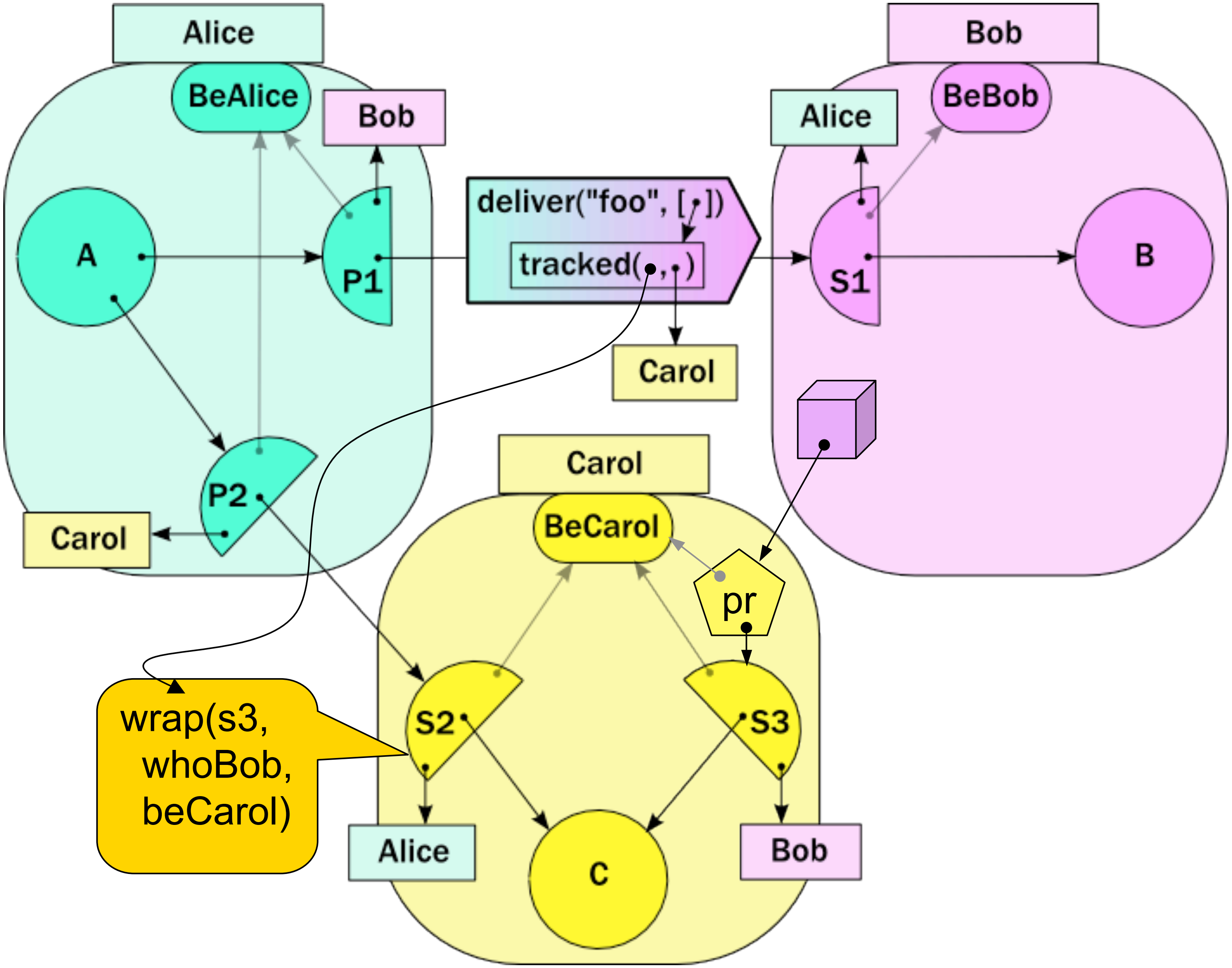

Alice
BeAlice
Bob
A
P1
P2
Carol
deliver("foo", [ ])
tracked(,)
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
S2
S3
Alice
C
Bob
Gift wrap it for Bob

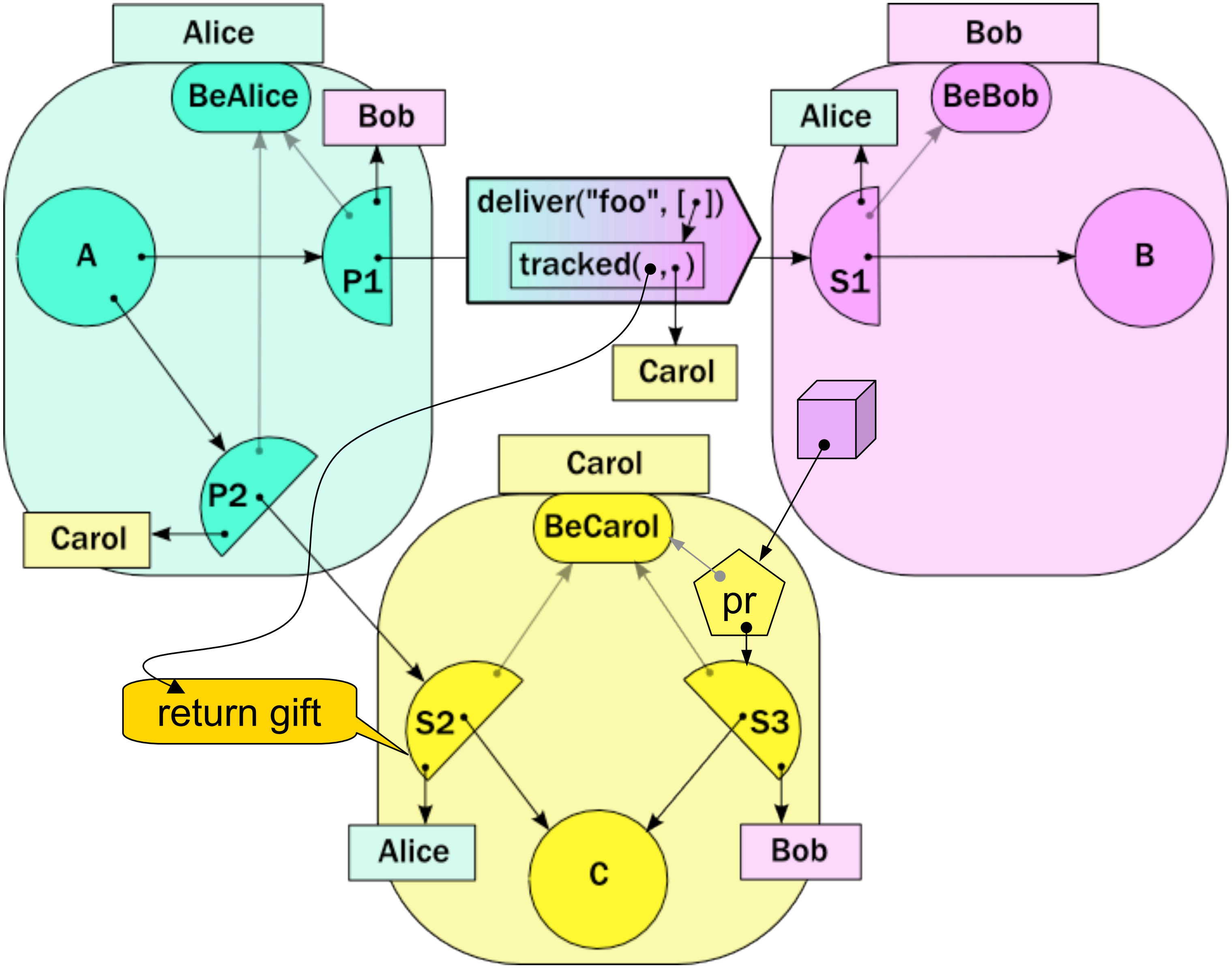

Alice
BeAlice
Bob
A
P1
P2
Carol
deliver("foo", [,])
tracked(,,)
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
S2
S3
Alice
C
Bob
wrap(s3,
whoBob,
beCarol)


Alice
BeAlice
Bob
A
P1
P2
Carol
deliver("foo", [ ])
tracked(,)
Carol
Bob
Alice
BeBob
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob
wrap(s3,
whoBob,
beCarol)

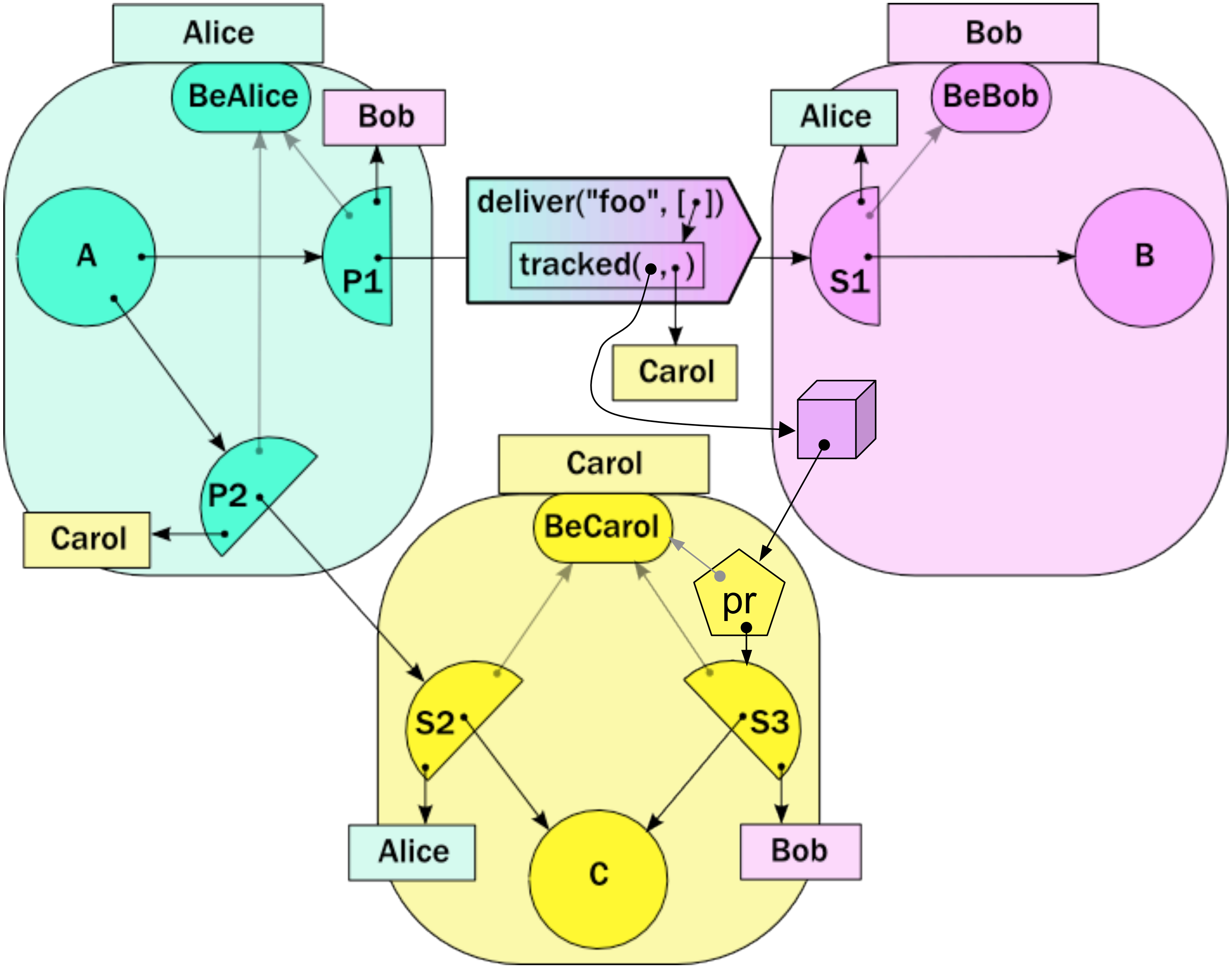


wrap(s3,
whoBob,
beCarol)

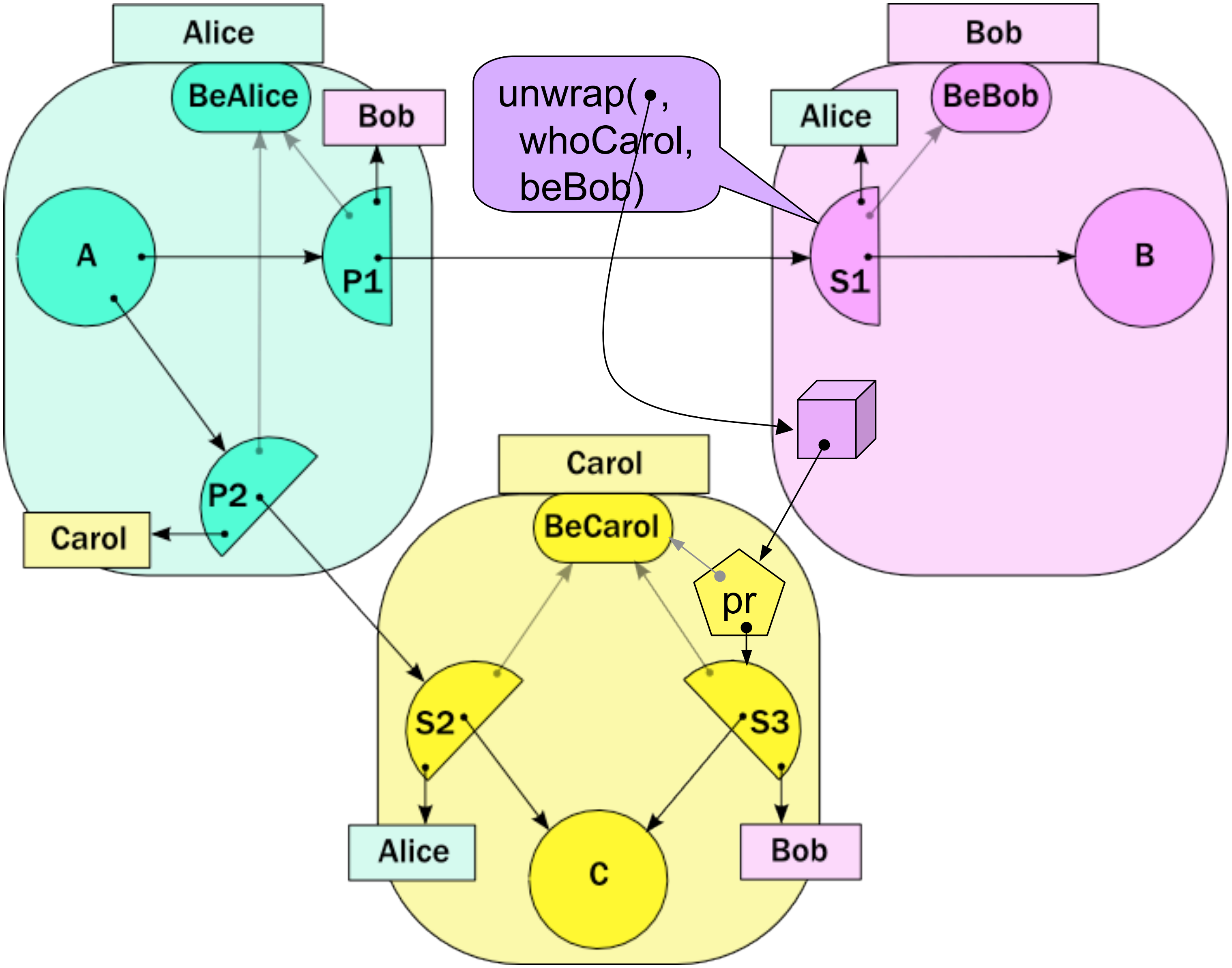

Alice
BeAlice
Bob
A
P1
P2
Carol
deliver("foo", [ ])
tracked(,)
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob
wrap(s3,
whoBob,
beCarol)

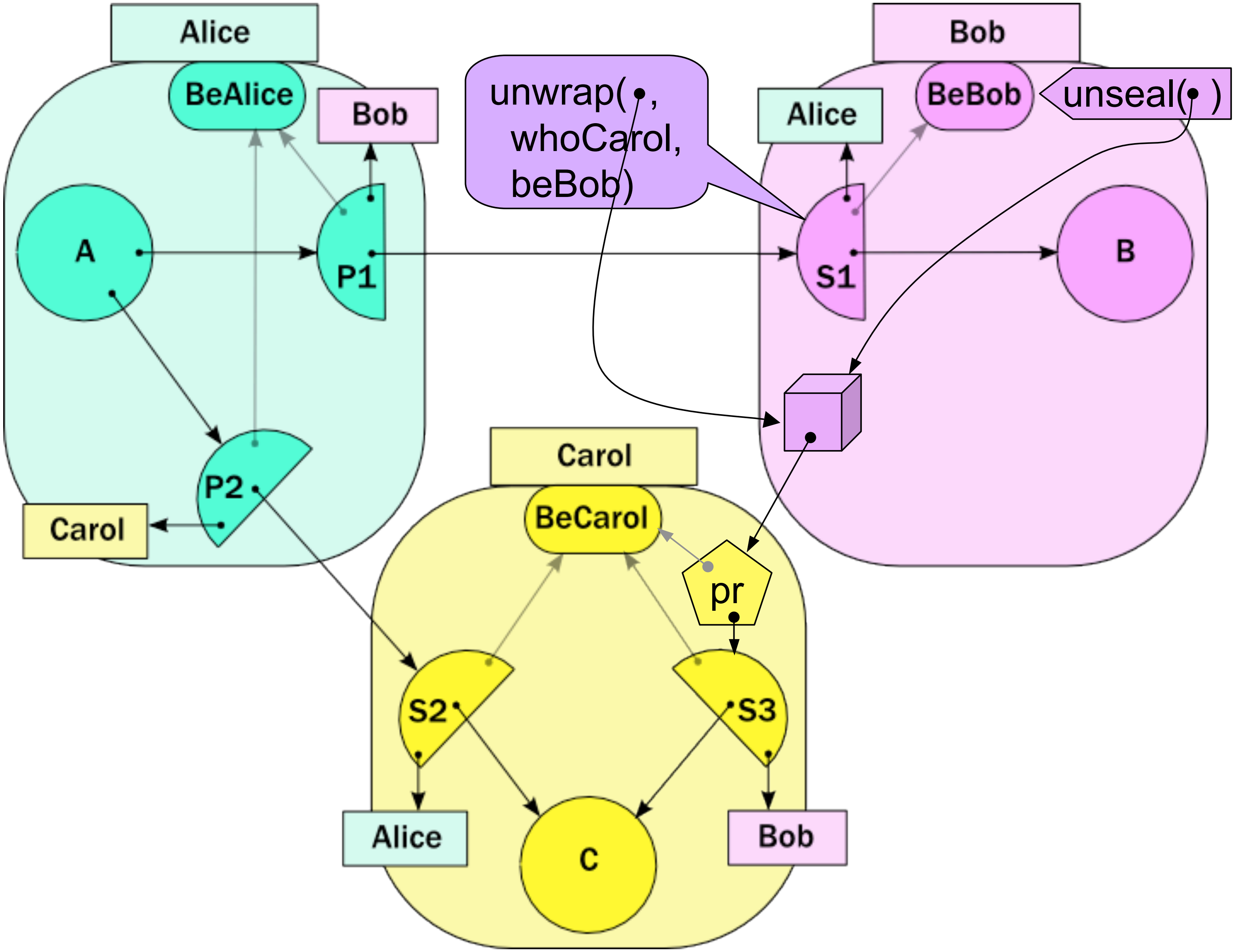


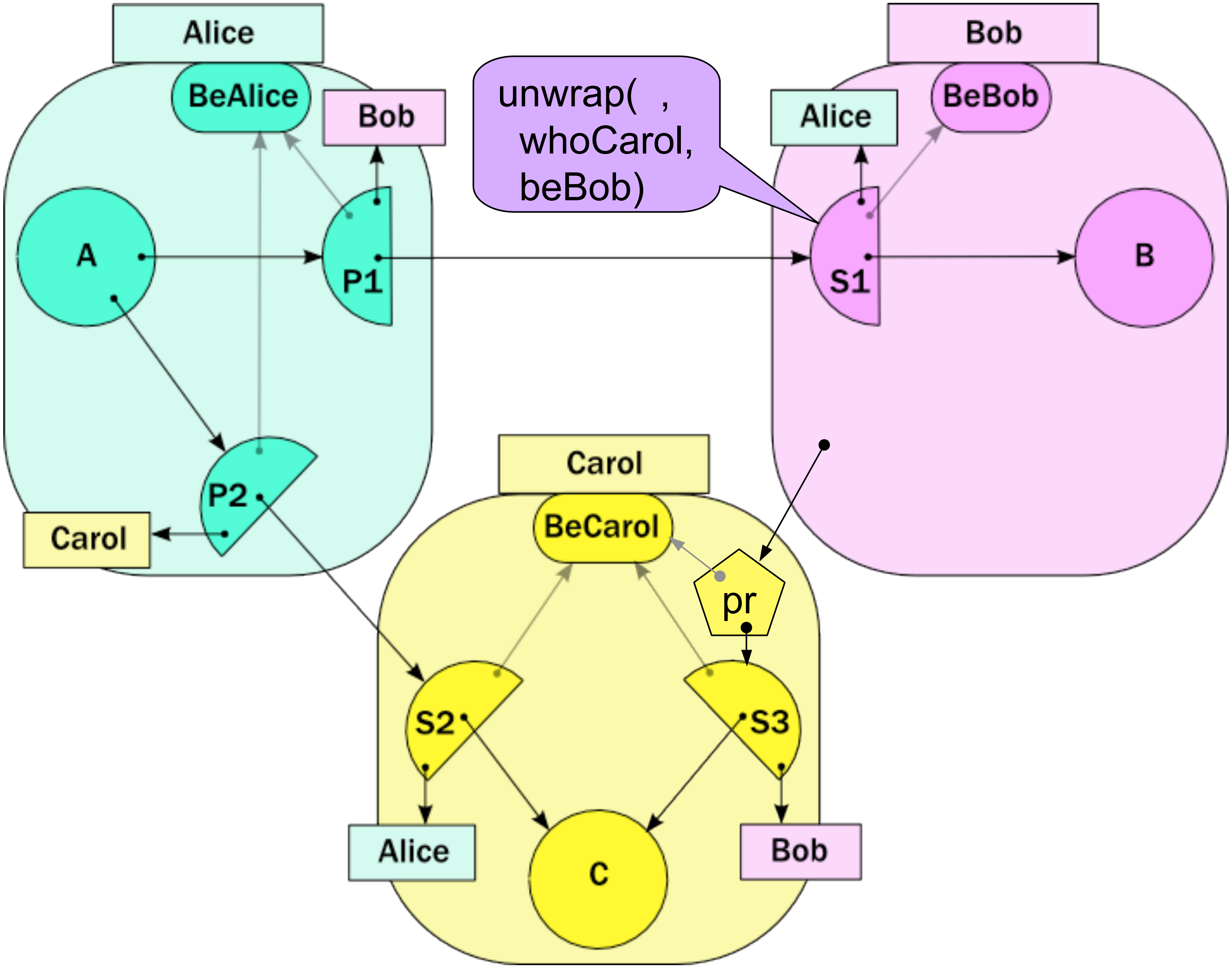
Alice
BeAlice
Bob
A
P1
P2
Carol
deliver("foo", [ ])
tracked(, ,)
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob
return gift

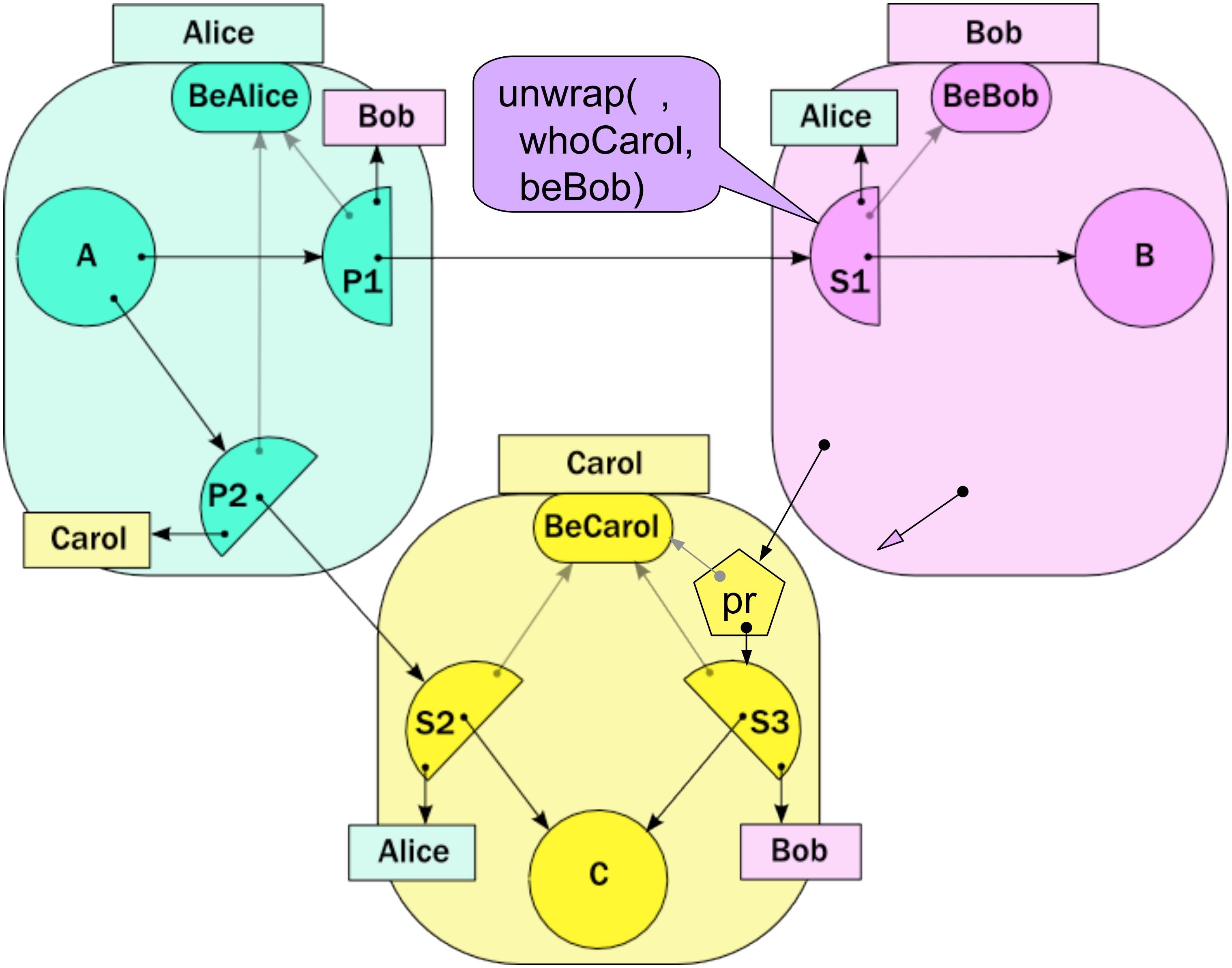


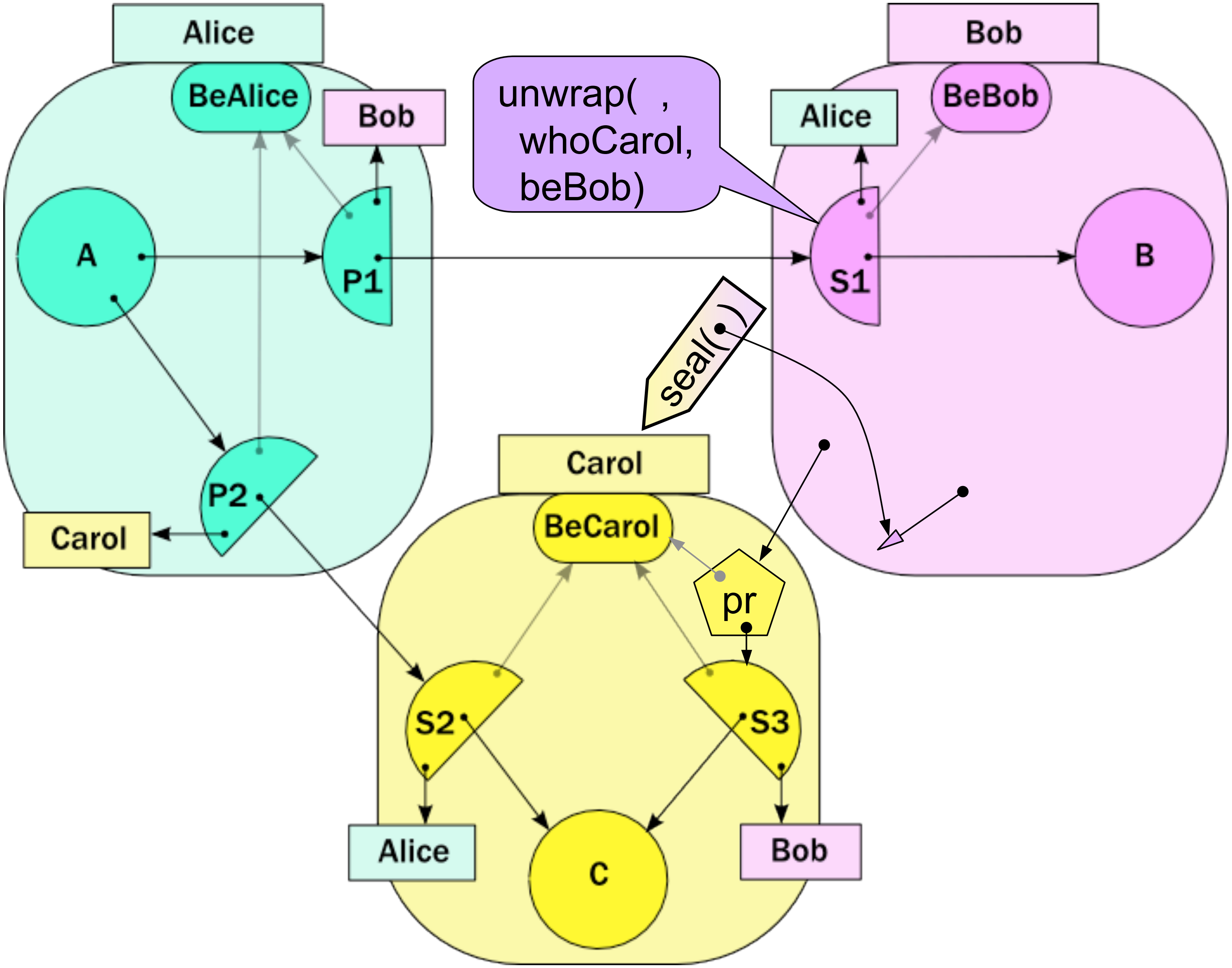
Alice
BeAlice
Bob
A
P1
P2
Carol
deliver("foo", [ ])
tracked(,)
Carol
Bob
BeBob
Alice
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob

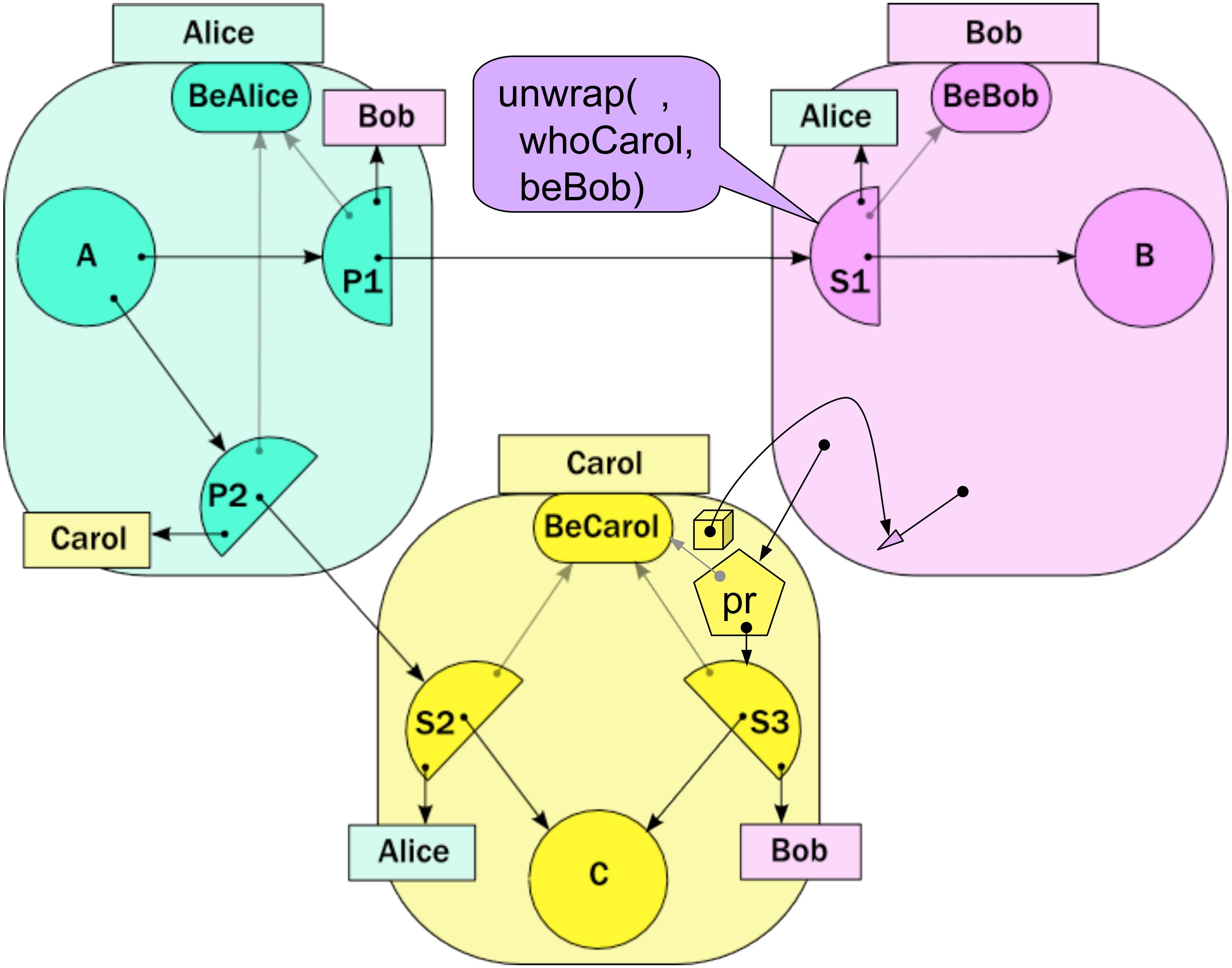

Alice
BeAlice
Bob
A
P1
P2
Carol
unwrap(•,
whoCarol,
beBob)
Bob
Alice
BeBob
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob


Alice
BeAlice
Bob
A
P1
P2
Carol
unwrap(, whoCarol, beBob)
Bob
Alice
BeBob
unseal( )
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob

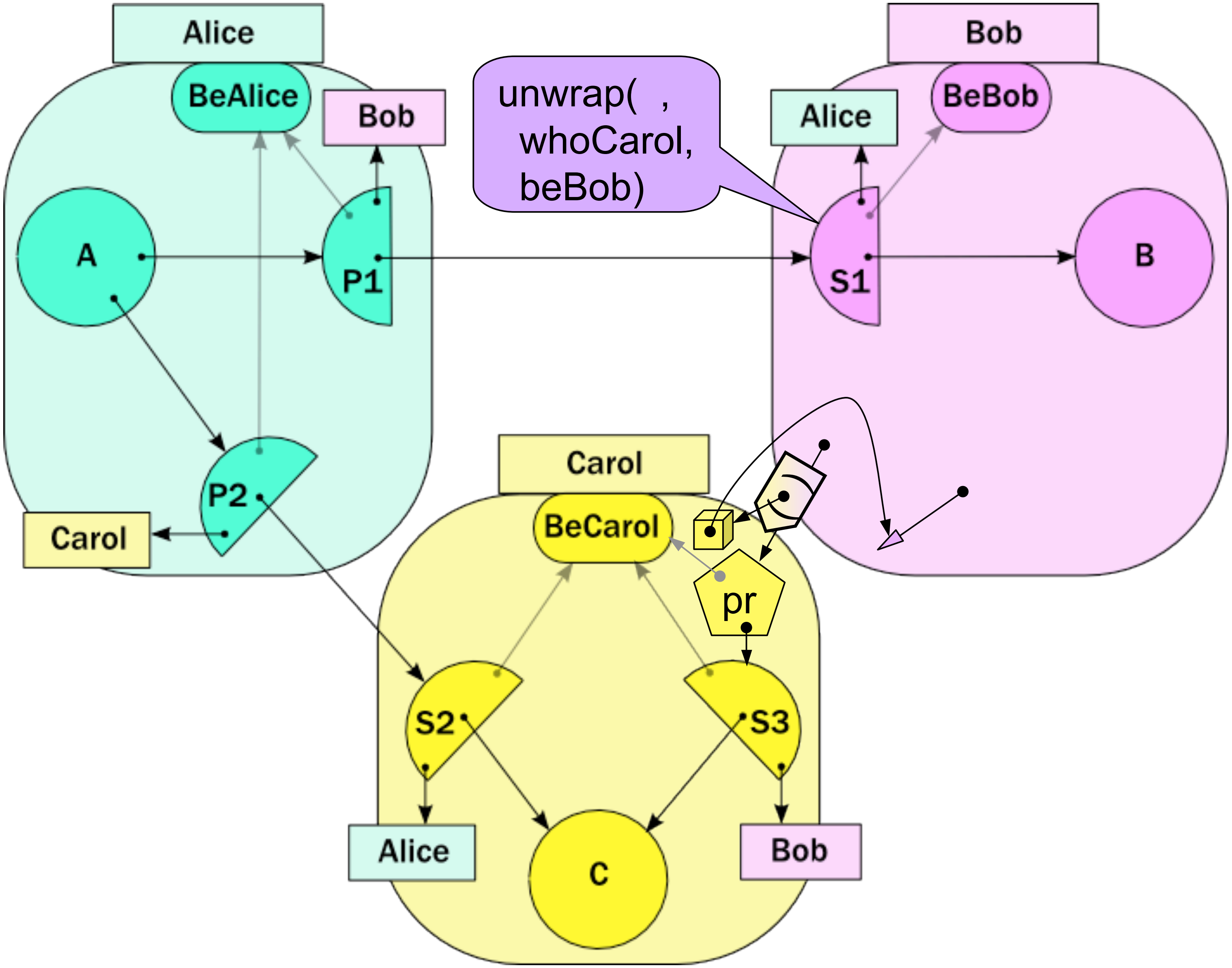

Alice
BeAlice
Bob
A
P1
P2
Carol
unwrap(,
whoCarol,
beBob)
Bob
Alice
BeBob
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob

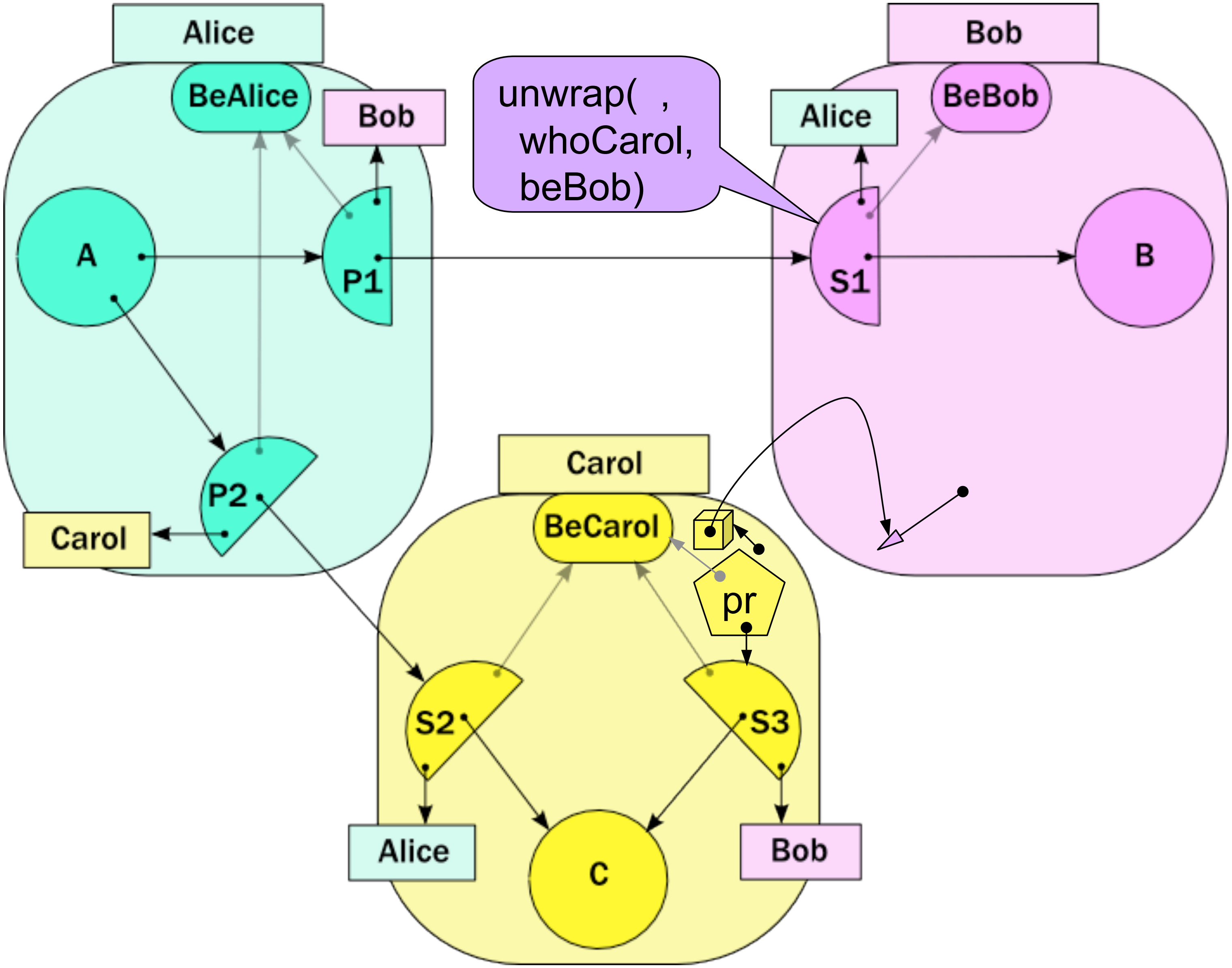

Alice
BeAlice
Bob
A
P1
P2
Carol
unwrap(,
whoCarol,
beBob)
Bob
BeBob
Alice
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob


Alice
BeAlice
Bob
A
P1
P2
Carol
unwrap(,
whoCarol,
beBob)
Bob
Alice
BeBob
S1
B
seal( )
Carol
BeCarol
pr
S2
S3
Alice
C
Bob

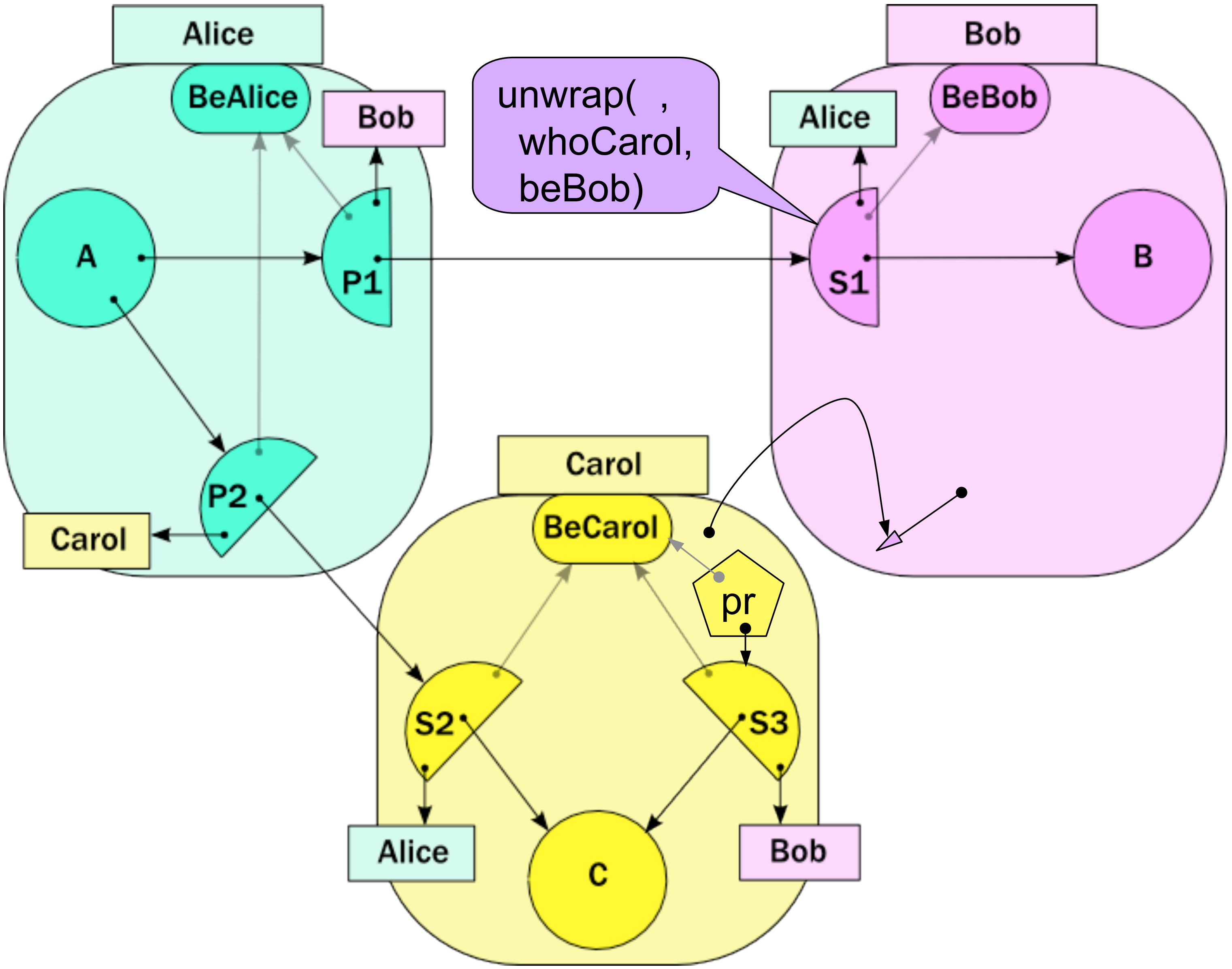


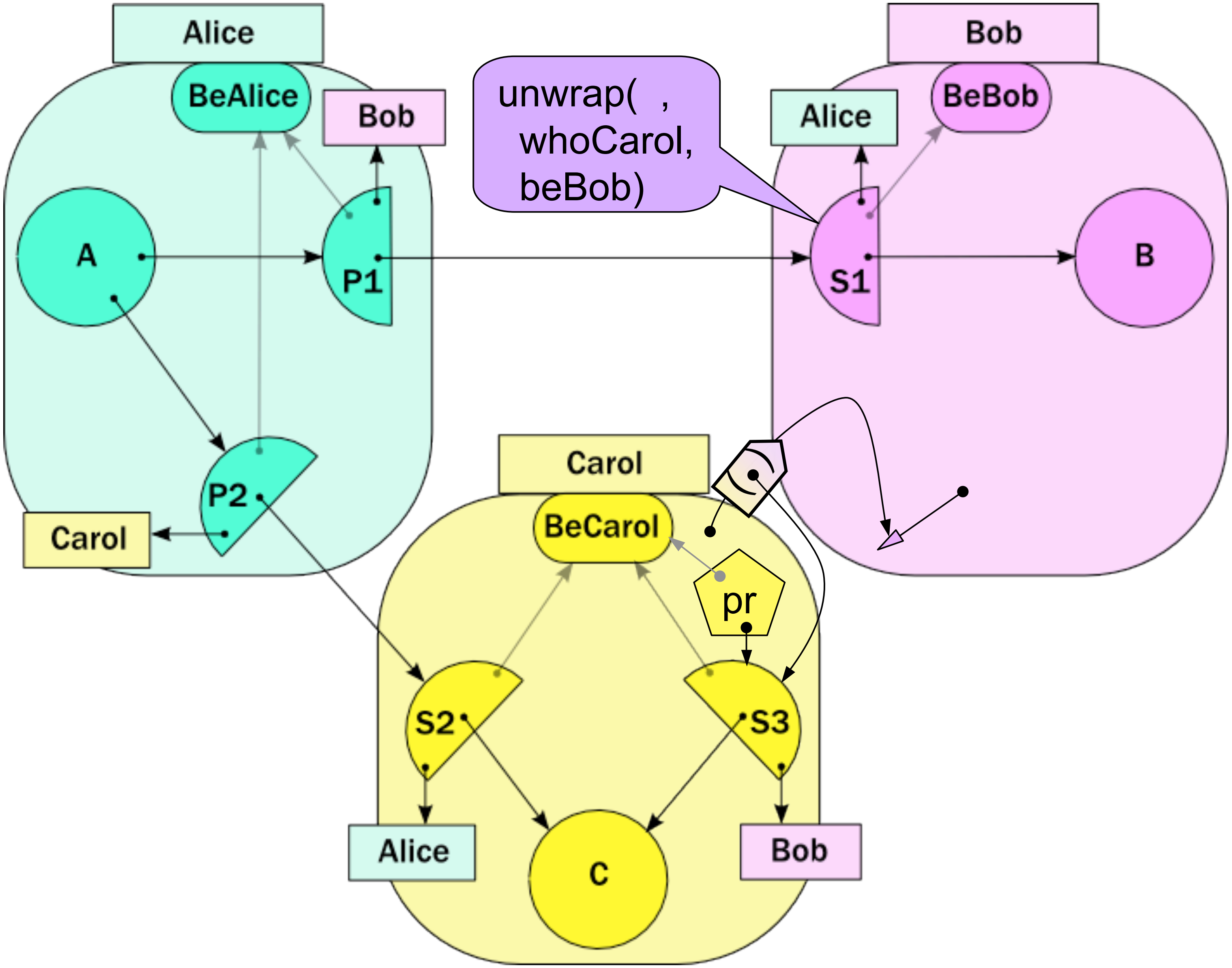
unwrap(,
whoCarol,
beBob)

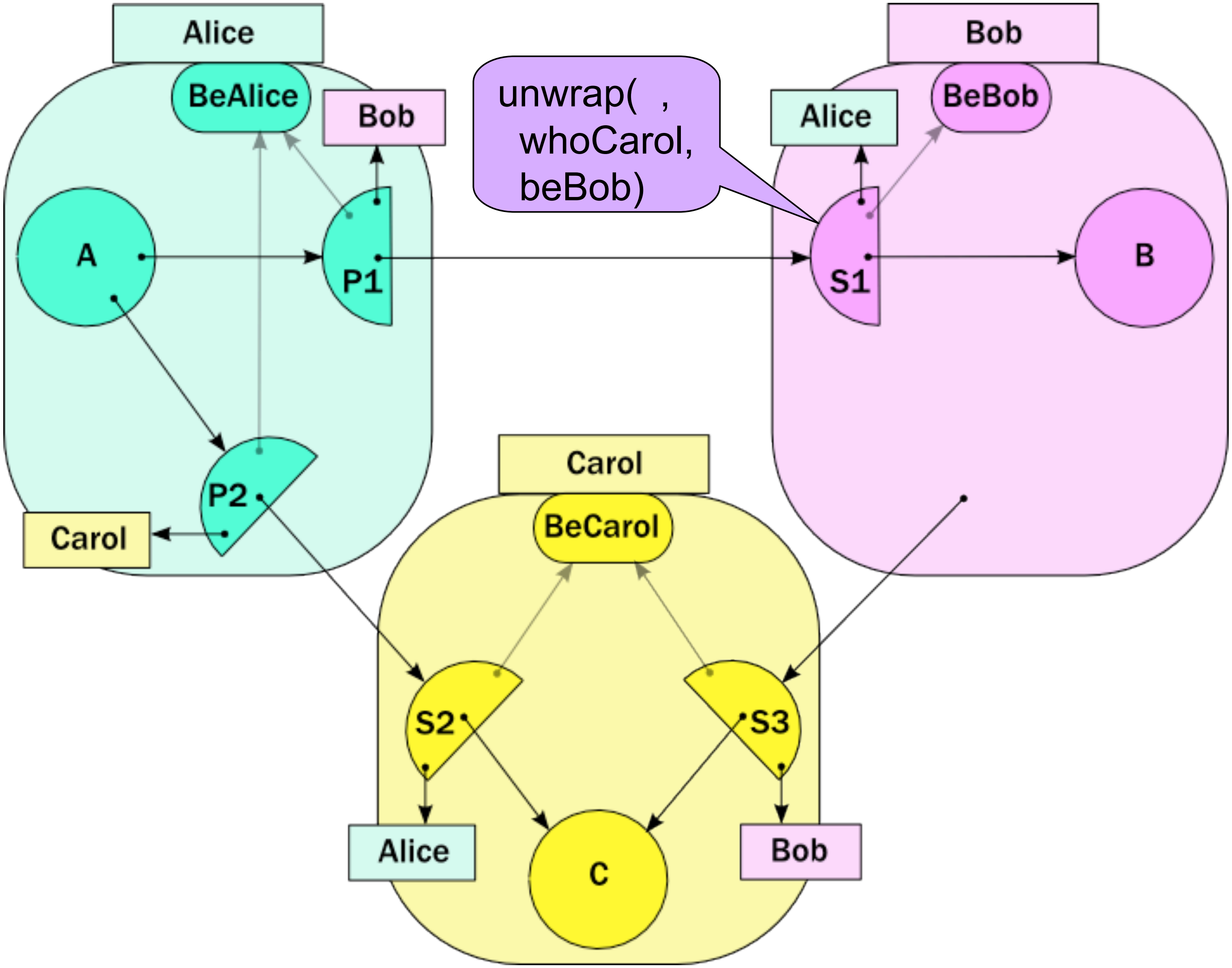

Alice
BeAlice
Bob
A
P1
P2
Carol
unwrap(,
whoCarol,
beBob)
Bob
BeBob
Alice
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob

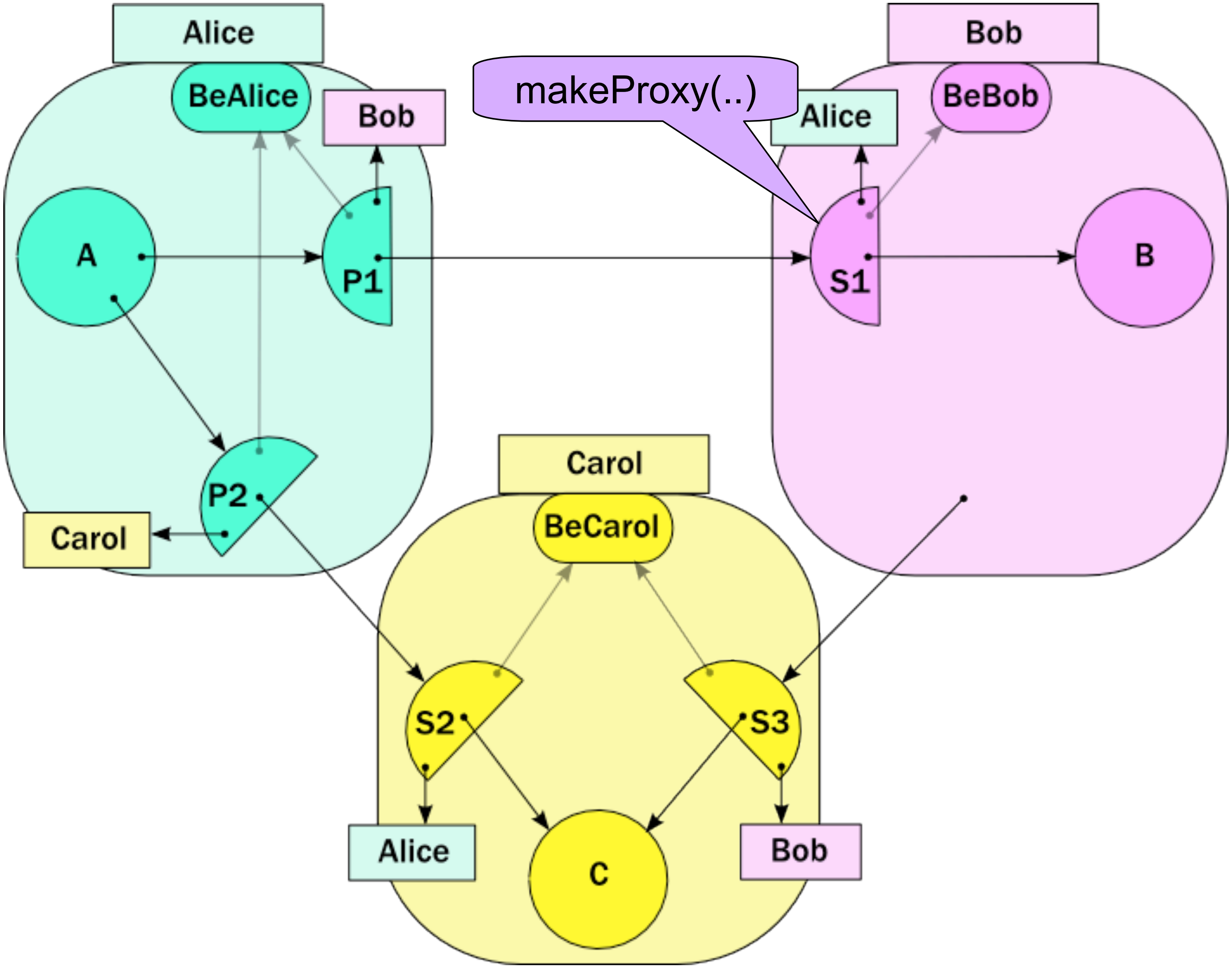


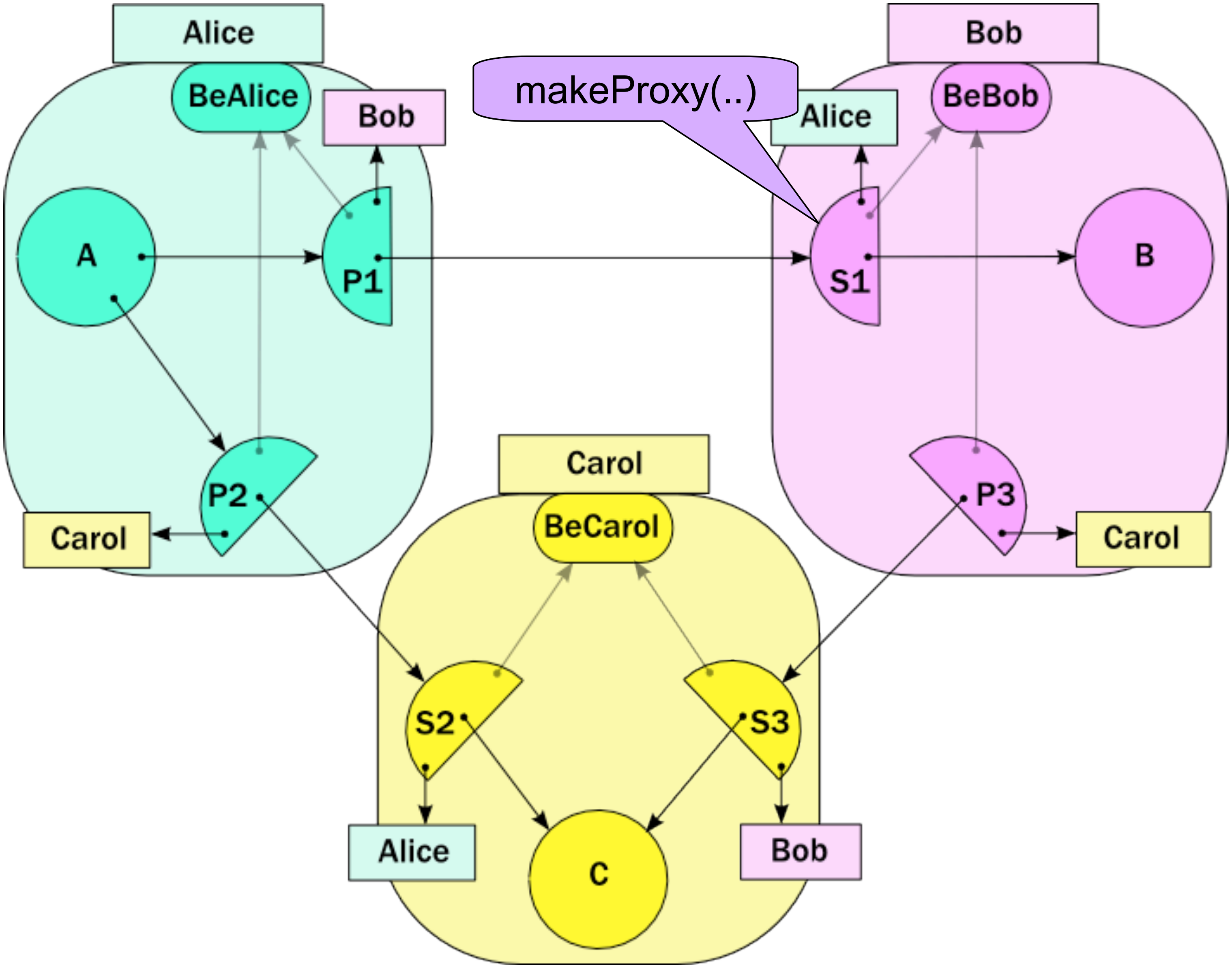
Alice
BeAlice
Bob
A
P1
P2
Carol
unwrap(,
whoCarol,
beBob)
Bob
Alice
BeBob
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob

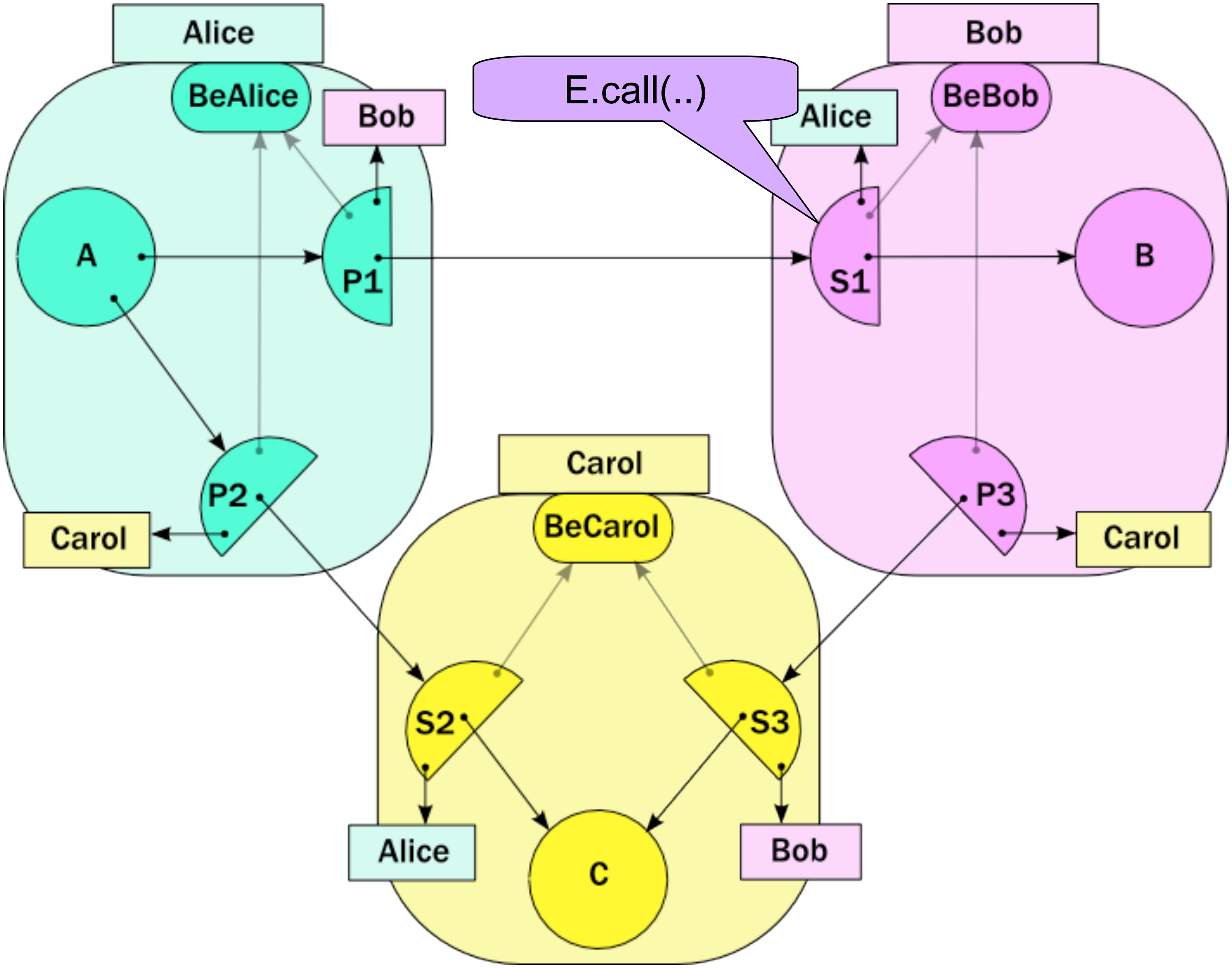

Alice
BeAlice
Bob
A
P1
P2
Carol
unwrap(,
whoCarol,
beBob)
Bob
BeBob
Alice
S1
B
Carol
BeCarol
pr
S2
S3
Alice
C
Bob

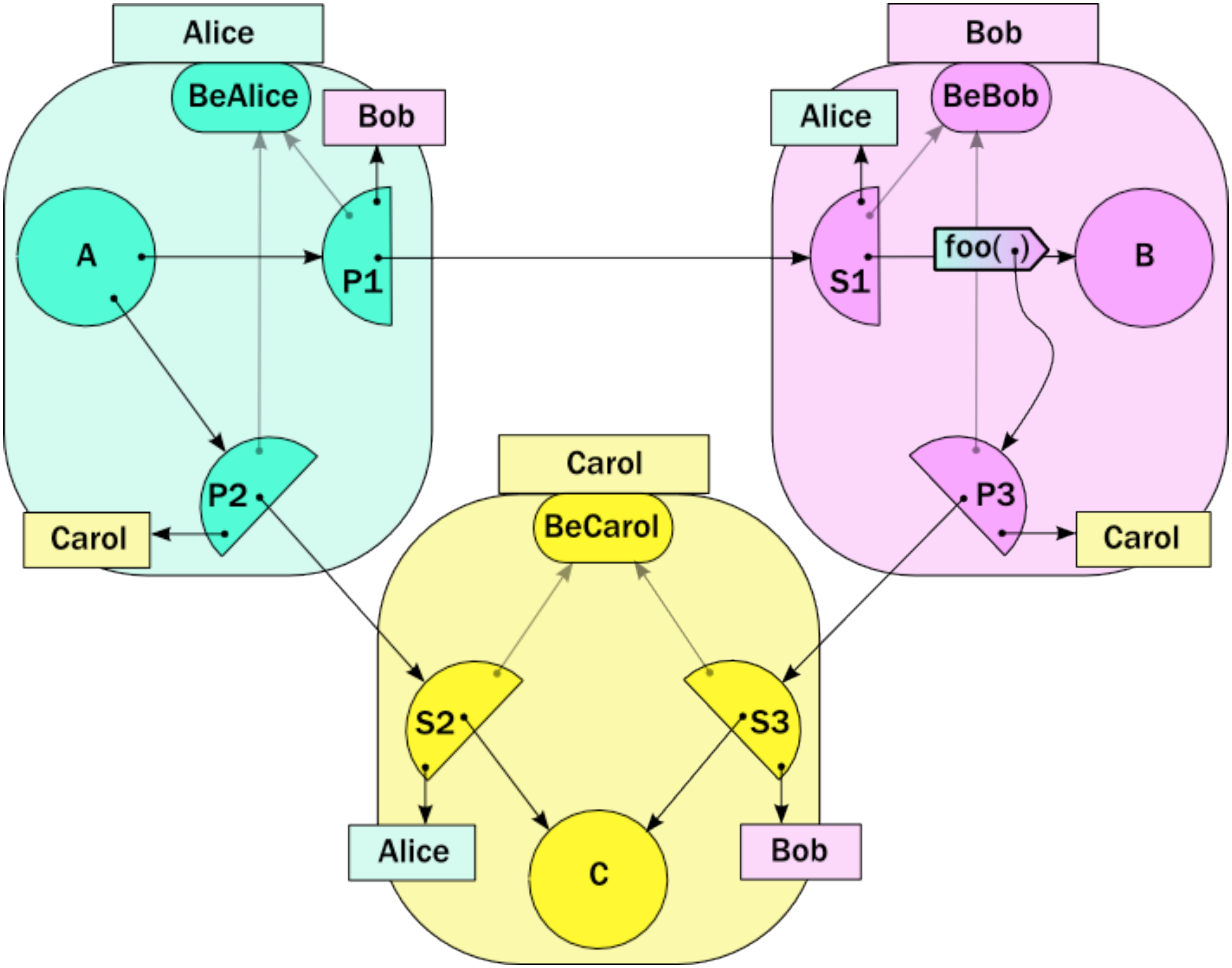

Alice
BeAlice
Bob
A
P1
P2
Carol
Bob
BeBob
Alice
S1
B
unwrap(,
whoCarol,
beBob)
Carol
BeCarol
pr
S2
S3
Alice
C
Bob

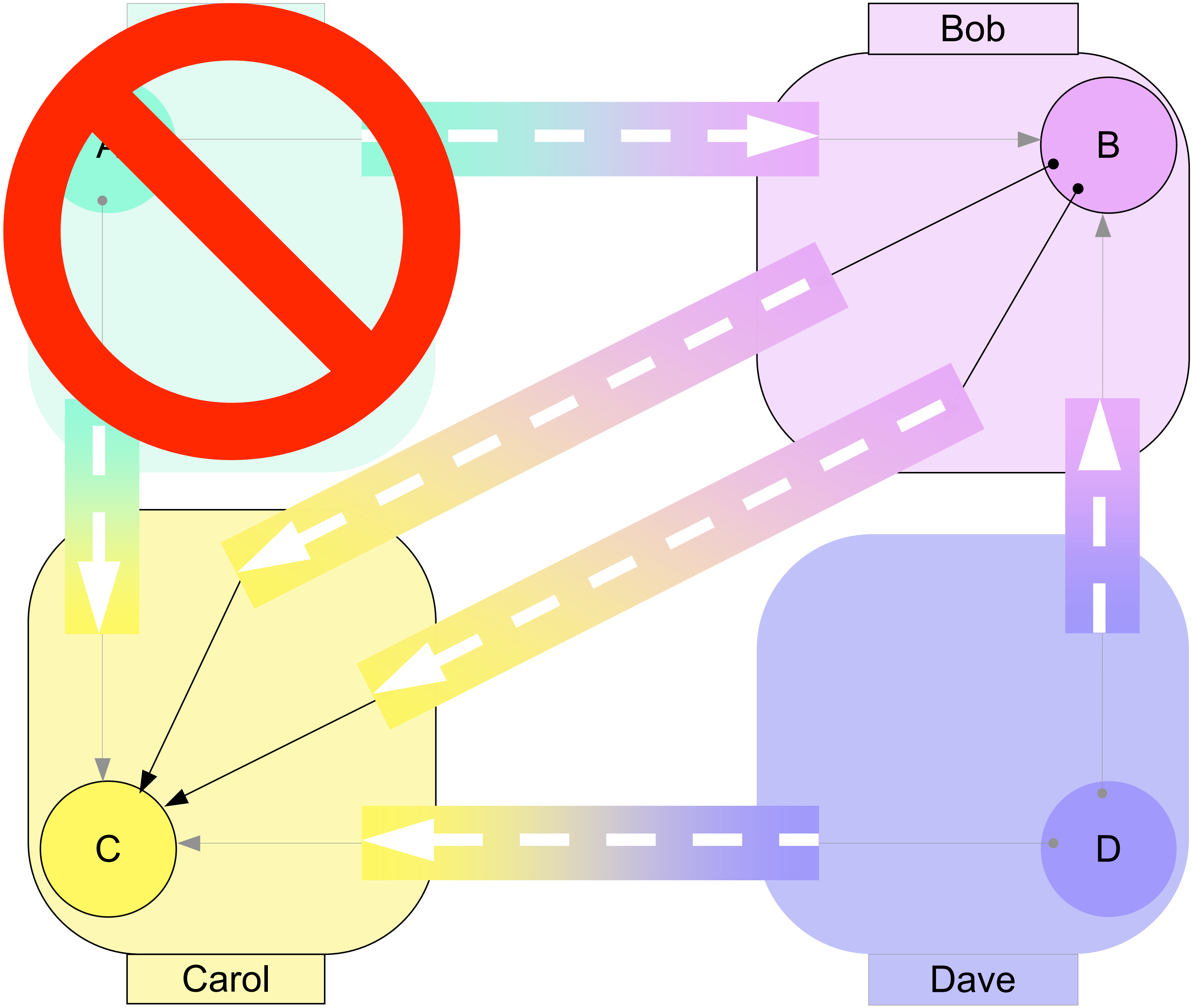

unwrap(,
whoCarol,
beBob)


Alice
BeAlice
Bob
A
P1
P2
Carol
makeProxy(..)
Bob
Alice
BeBob
S1
B
Carol
BeCarol
S2
S3
Alice
C
Bob

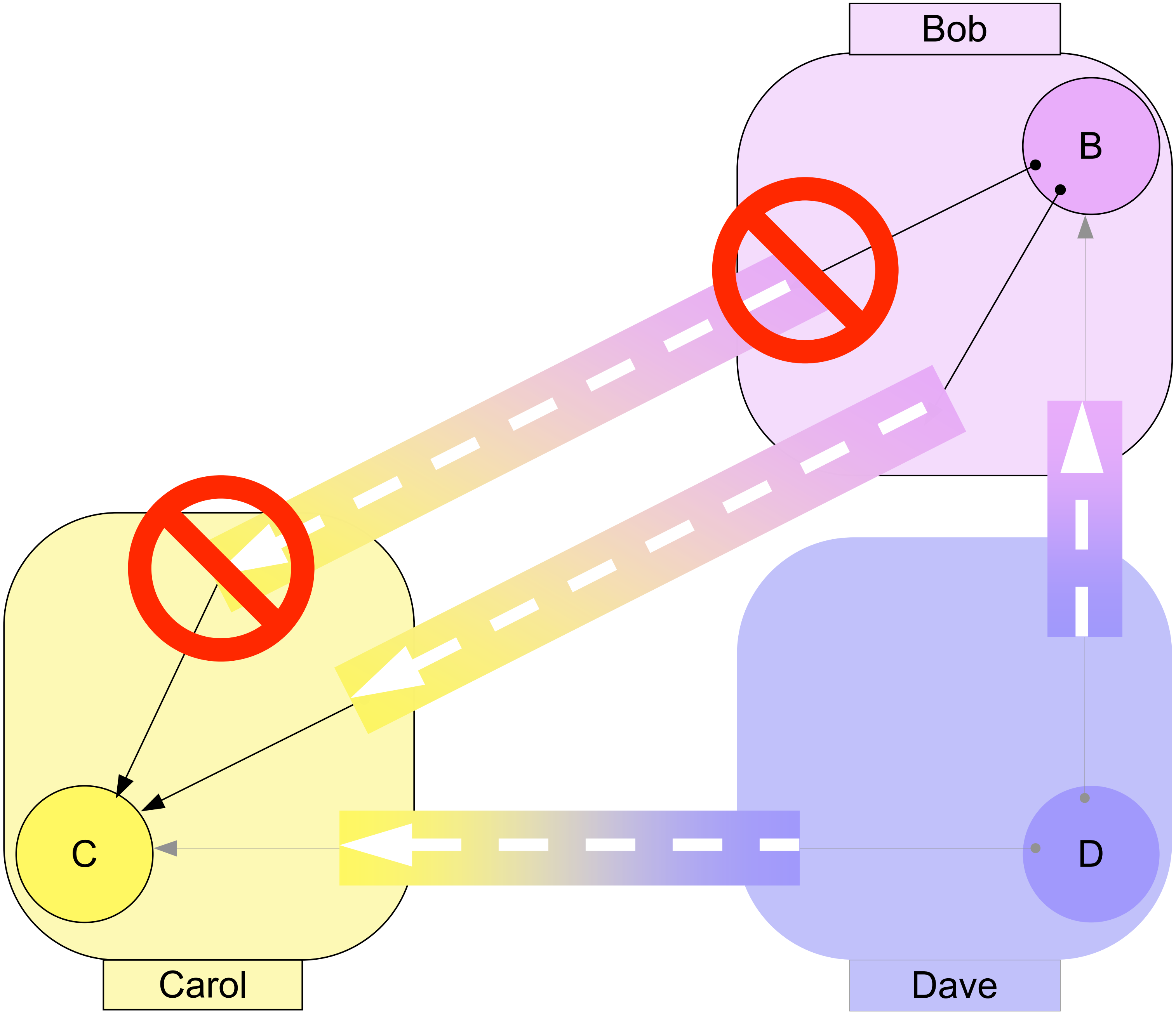

Alice
BeAlice
Bob
A
P1
P2
Carol
makeProxy(..)
Bob
Alice
BeBob
S1
B
P3
Carol
Carol
BeCarol
S2
S3
Alice
C
Bob

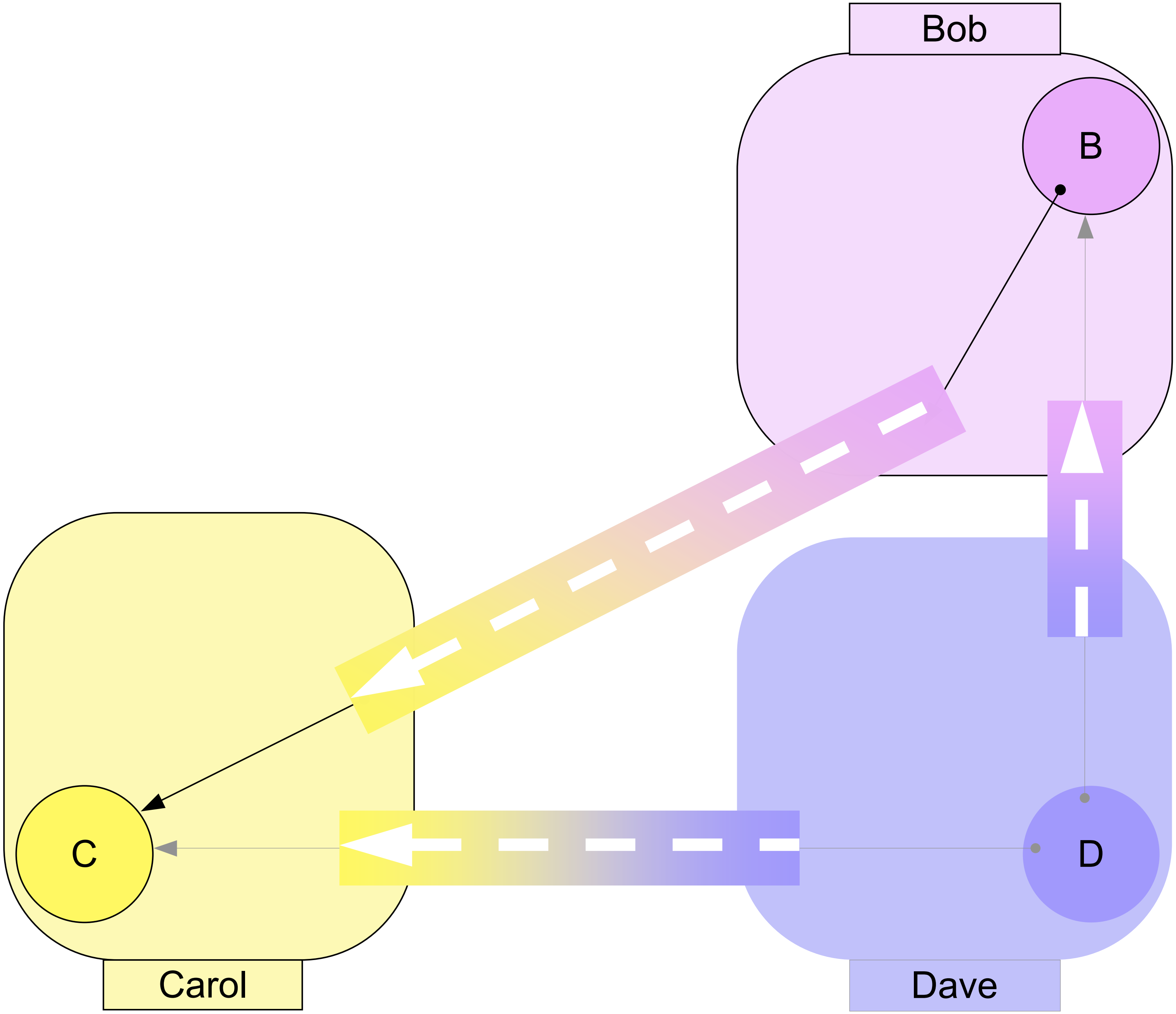

Alice
BeAlice
Bob
A
P1
P2
Carol
E.call(..)
Bob
Alice
BeBob
S1
B
P3
Carol
Carol
BeCarol
S2
S3
Alice
C
Bob


Alice
BeAlice
Bob
A
P1
P2
Carol
Bob
BeBob
Alice
S1
foo( )
B
P3
Carol
Carol
BeCarol
S2
S3
Alice
C
Bob



Bob
A
B
C
D
Carol
Dave


Bob
B
Carol
C
Dave
D


Bob
B
C
D
Carol
Dave

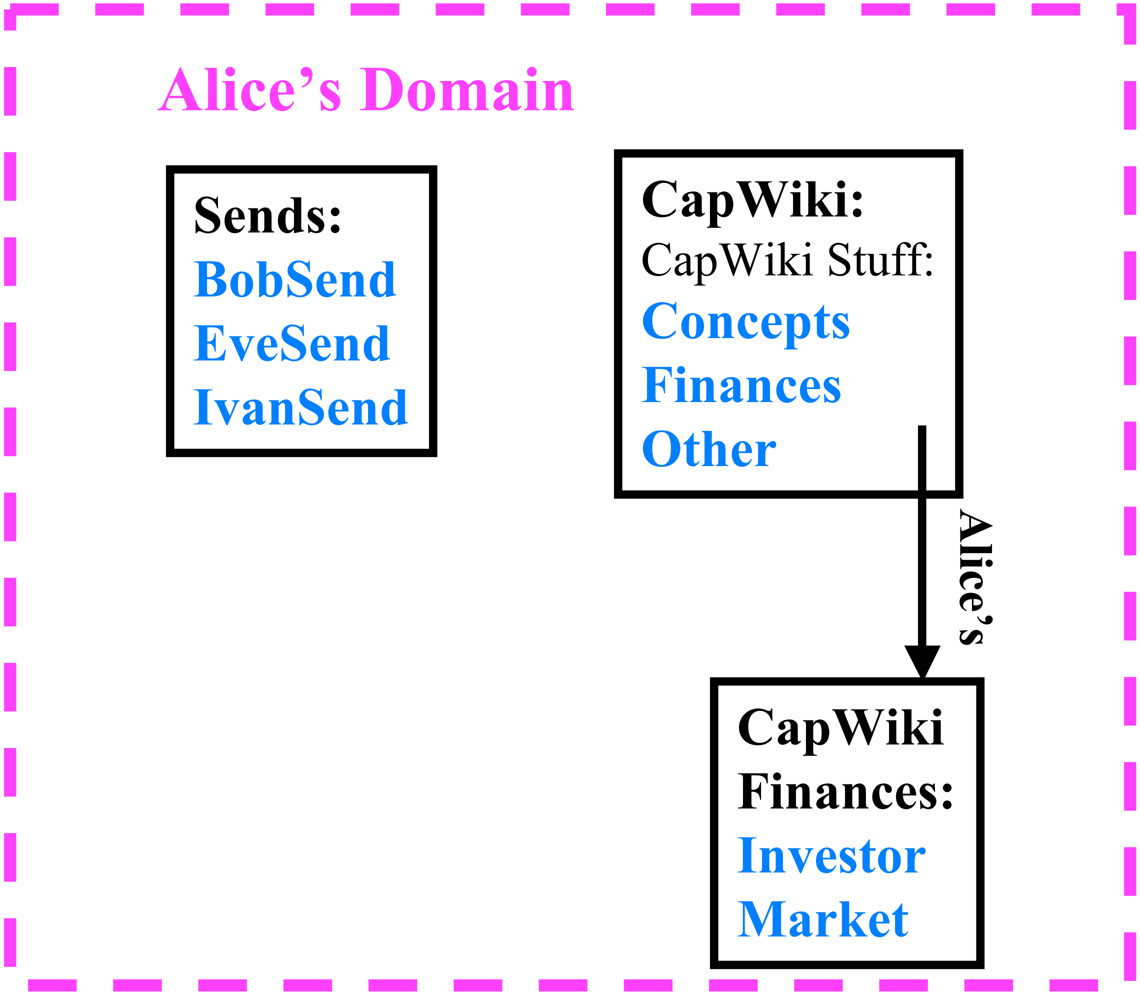
CapWiki with attribution

The Web: Good, Bad, and Ugly:

1. Good: Internet hypertext, wonderful!
2. Bad: Username/passwords for every site that has any sort of access control.
3. Ugly: Hard to share limited access to network objects. Hard to combine network objects with access restrictions.

Alice’s Domain

Sends:
BobSend
EveSend
IvanSend



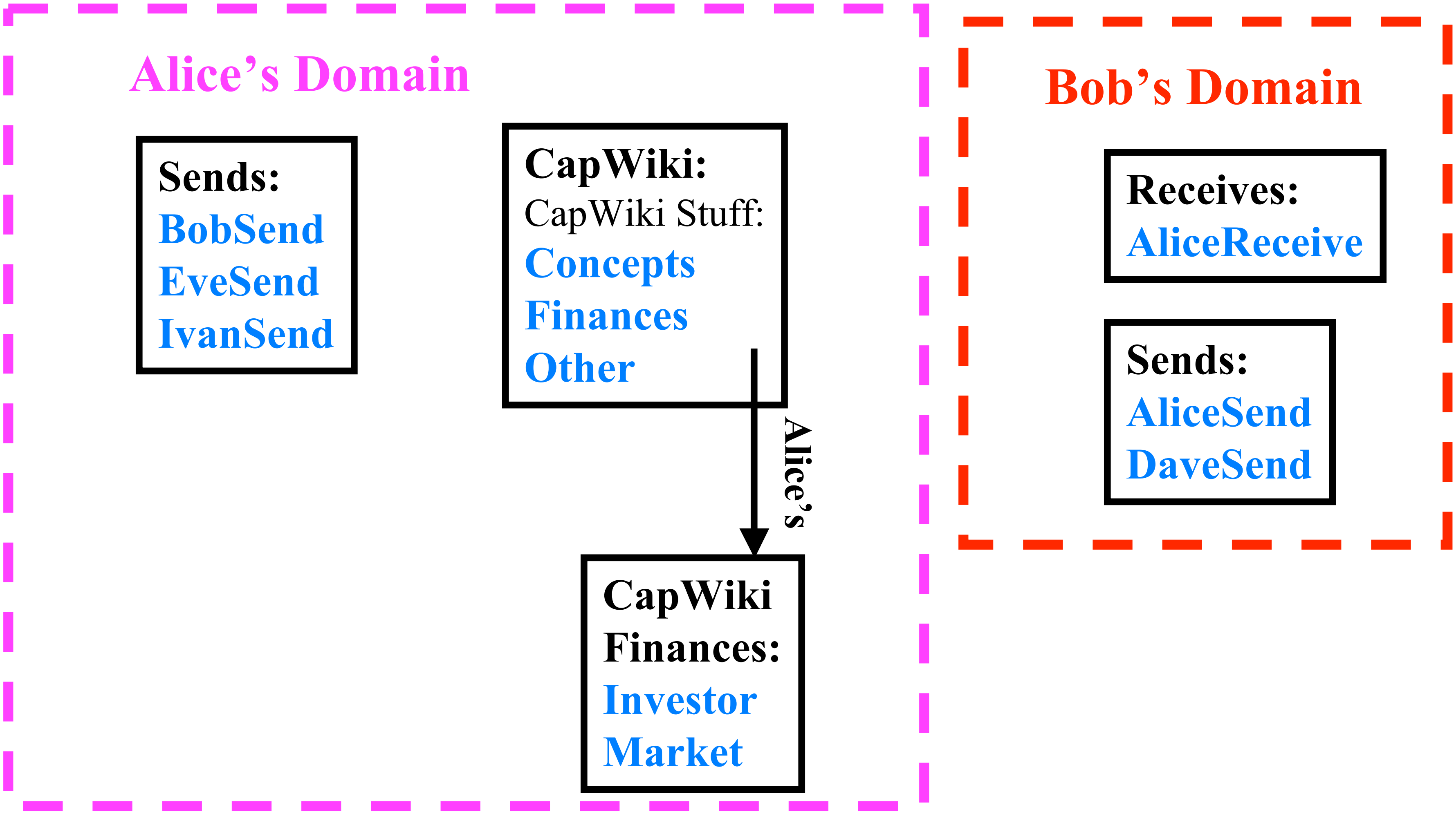
Alice’s Domain

Sends:
BobSend
EveSend
IvanSend

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

Alice’s

CapWiki Finances:
Investor
Market



Alice's Domain

Sends:
BobSend
EveSend
IvanSend

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

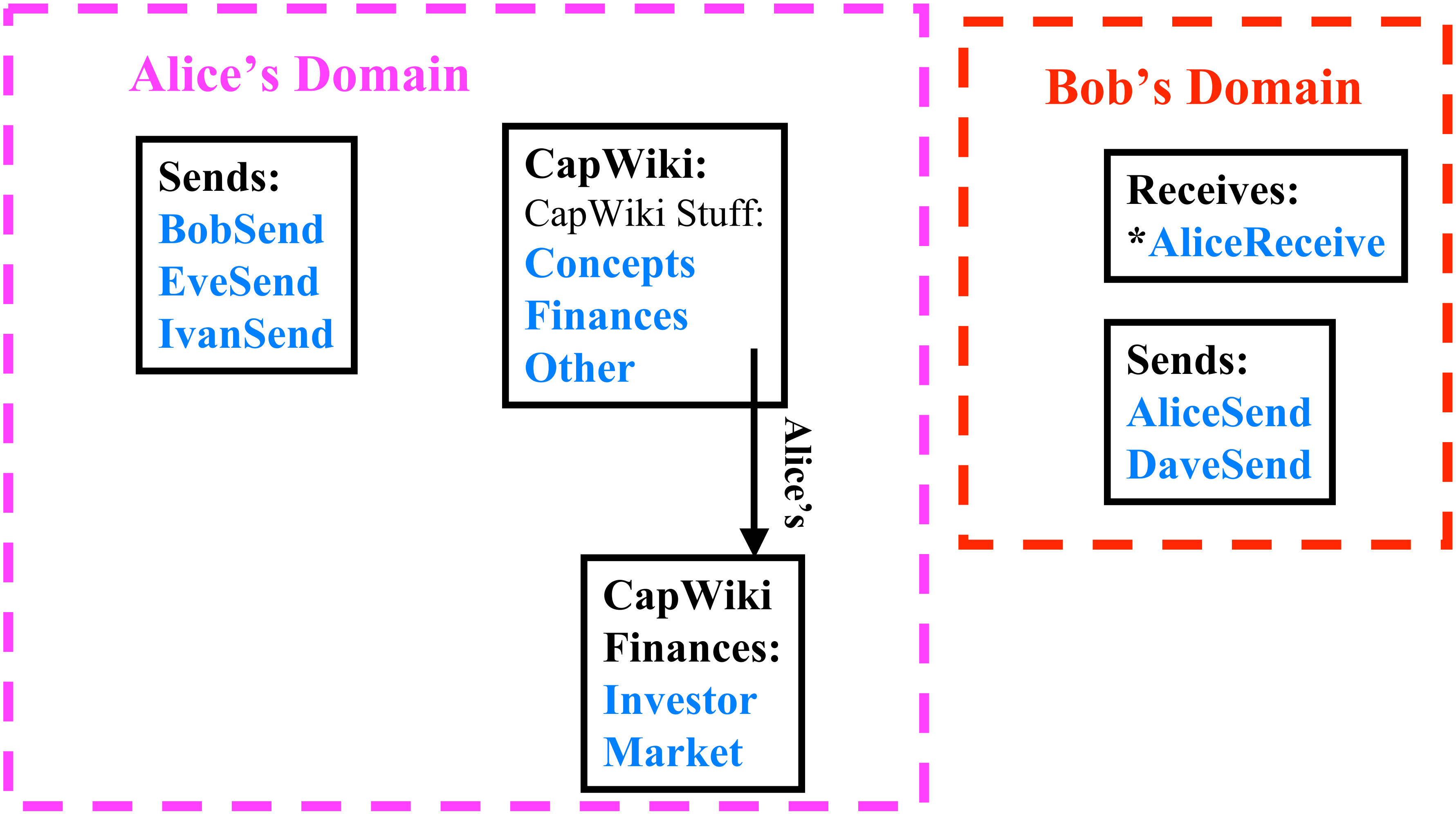
Alice's

CapWiki Finances:
Investor
Market

Bob's Domain

Receives:
AliceReceive

Sends:
AliceSend
DaveSend



Alice's Domain

Sends:
BobSend
EveSend
IvanSend

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

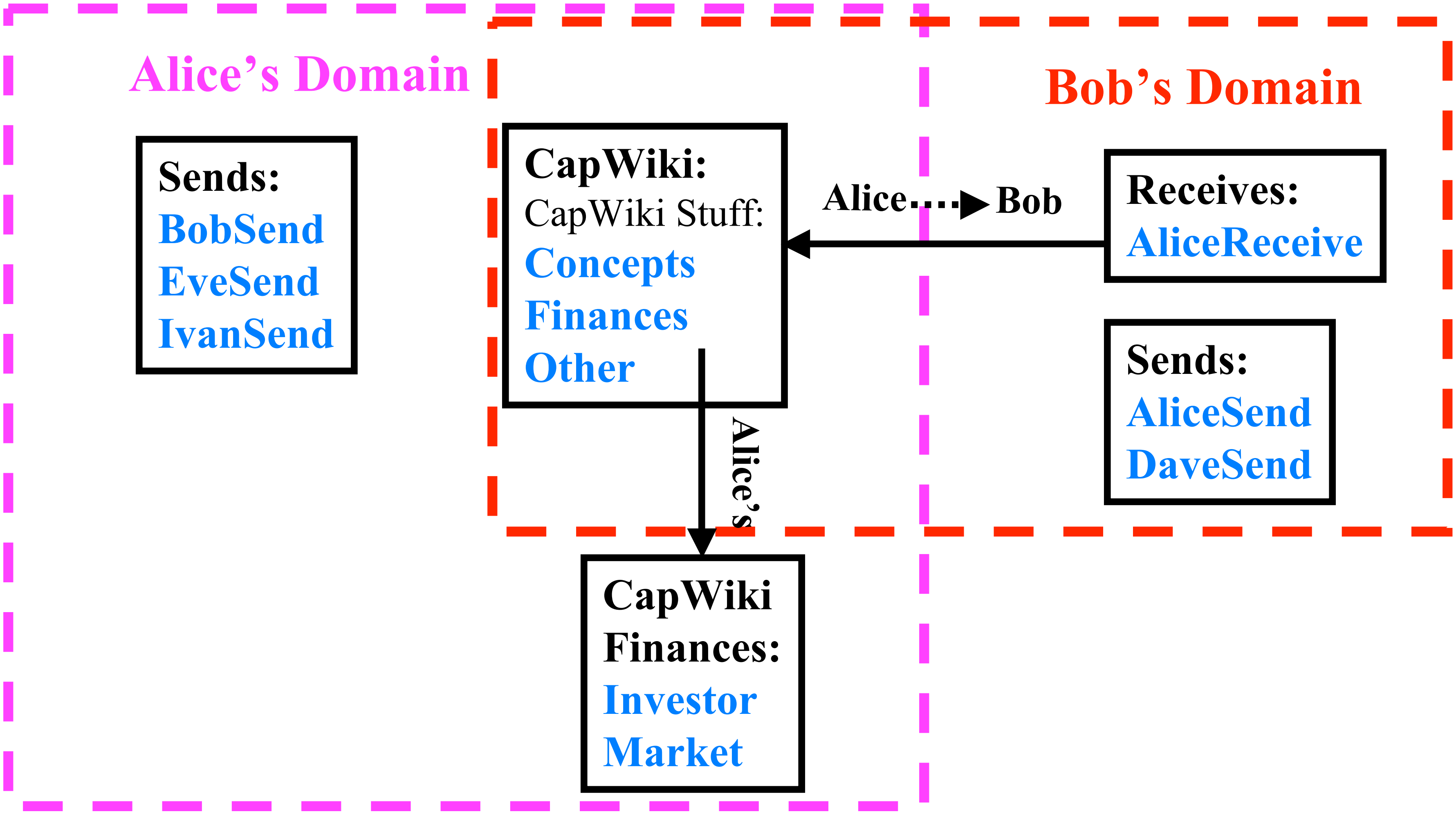
Alice's

CapWiki
Finances:
Investor
Market

Bob's Domain

Receives:
*AliceReceive

Sends:
AliceSend
DaveSend



Alice’s Domain

Sends:
BobSend
EveSend
IvanSend

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

Alice····▶ Bob

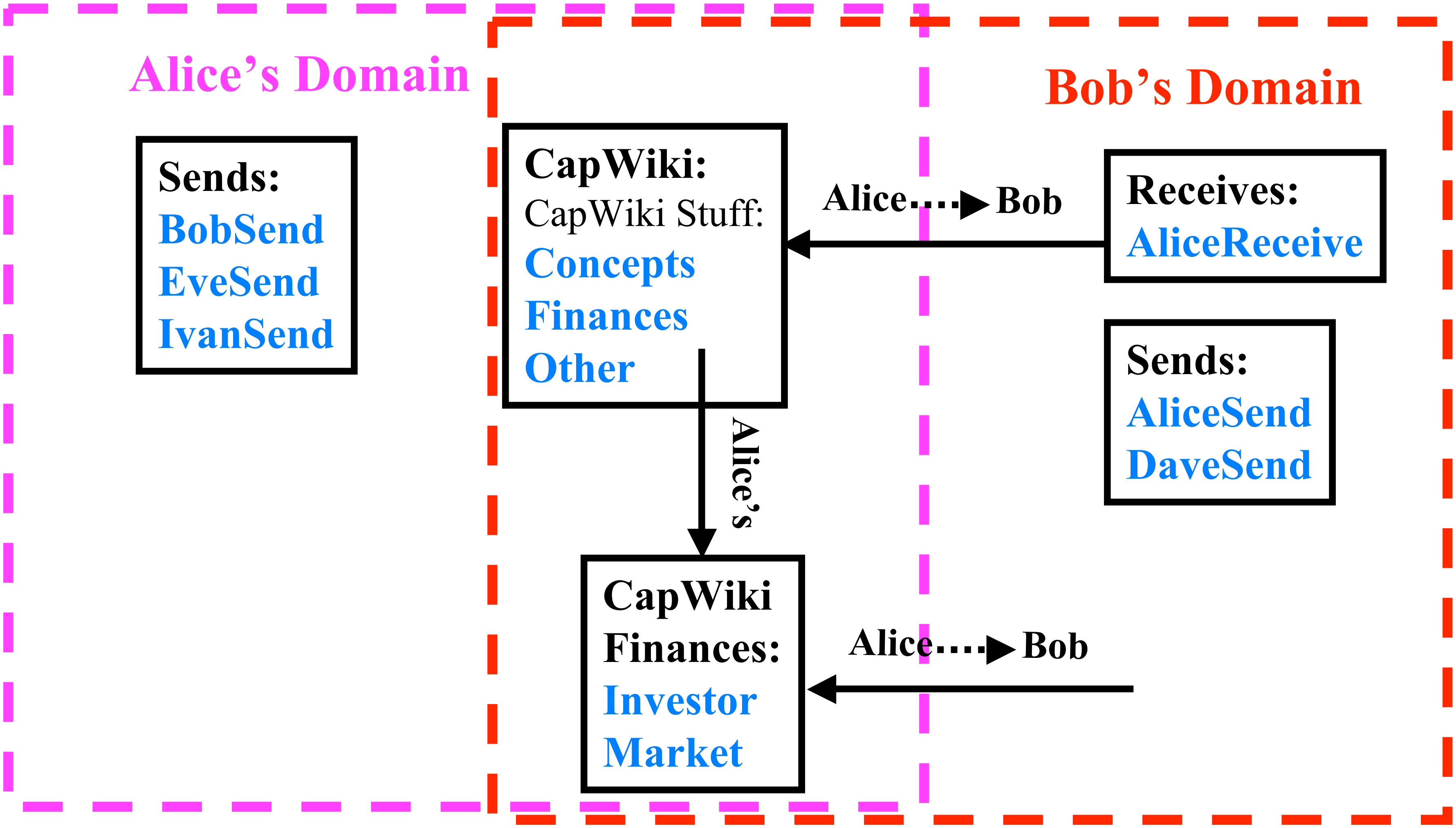
Bob’s Domain

Receives:
AliceReceive

Sends:
AliceSend
DaveSend

Alice’s

CapWiki
Finances:
Investor
Market



Alice's Domain

Bob's Domain

Sends:
BobSend
EveSend
IvanSend

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

Alice's

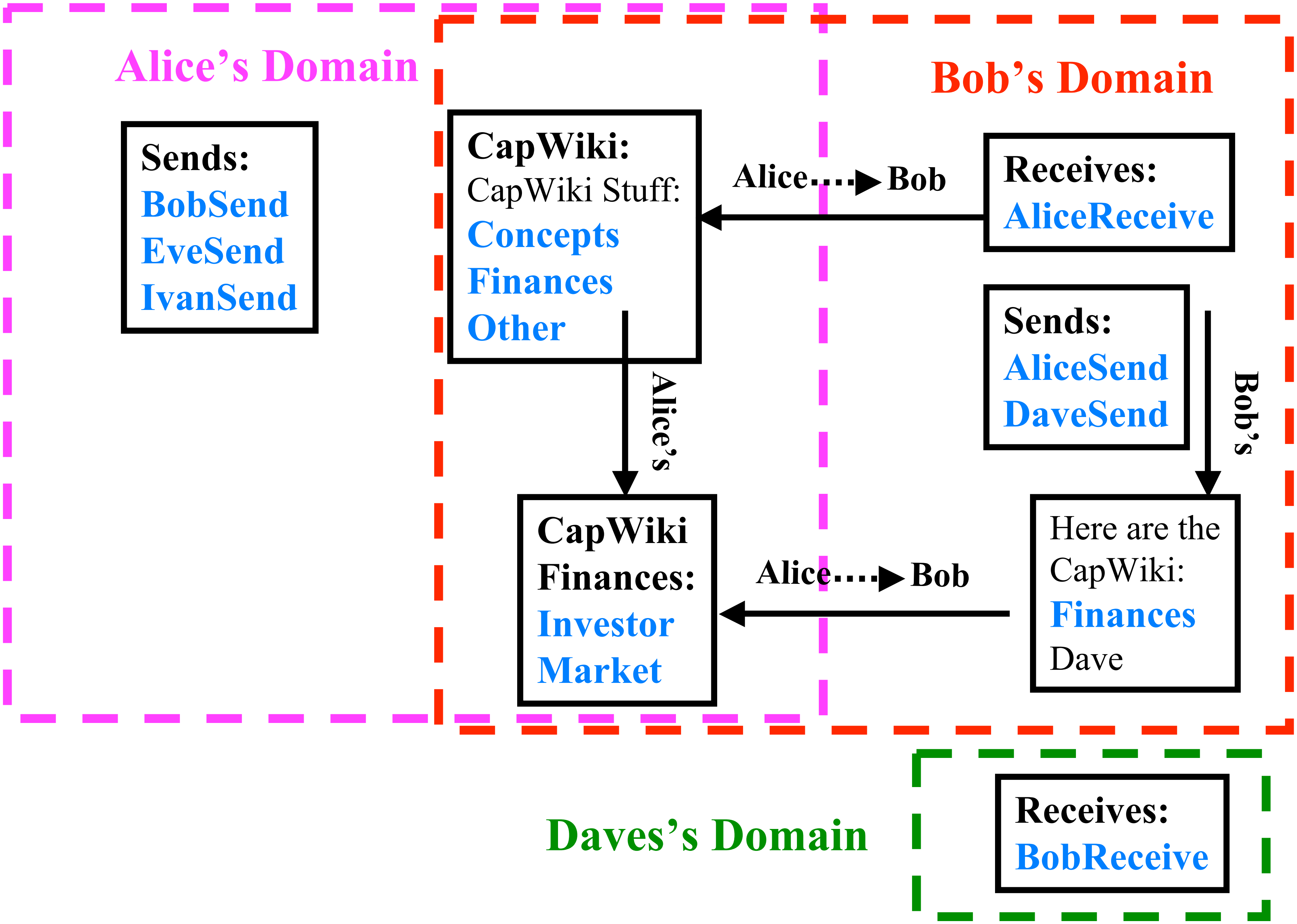
CapWiki
Finances:
Investor
Market

Alice····▶ Bob

Receives:
AliceReceive

Sends:
AliceSend
DaveSend

Alice····▶ Bob



Alice's Domain

Sends:
BobSend
EveSend
IvanSend

Bob's Domain

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

Alice····▶ Bob

Receives:
AliceReceive

Sends:
AliceSend
DaveSend

Alice's

Bob's

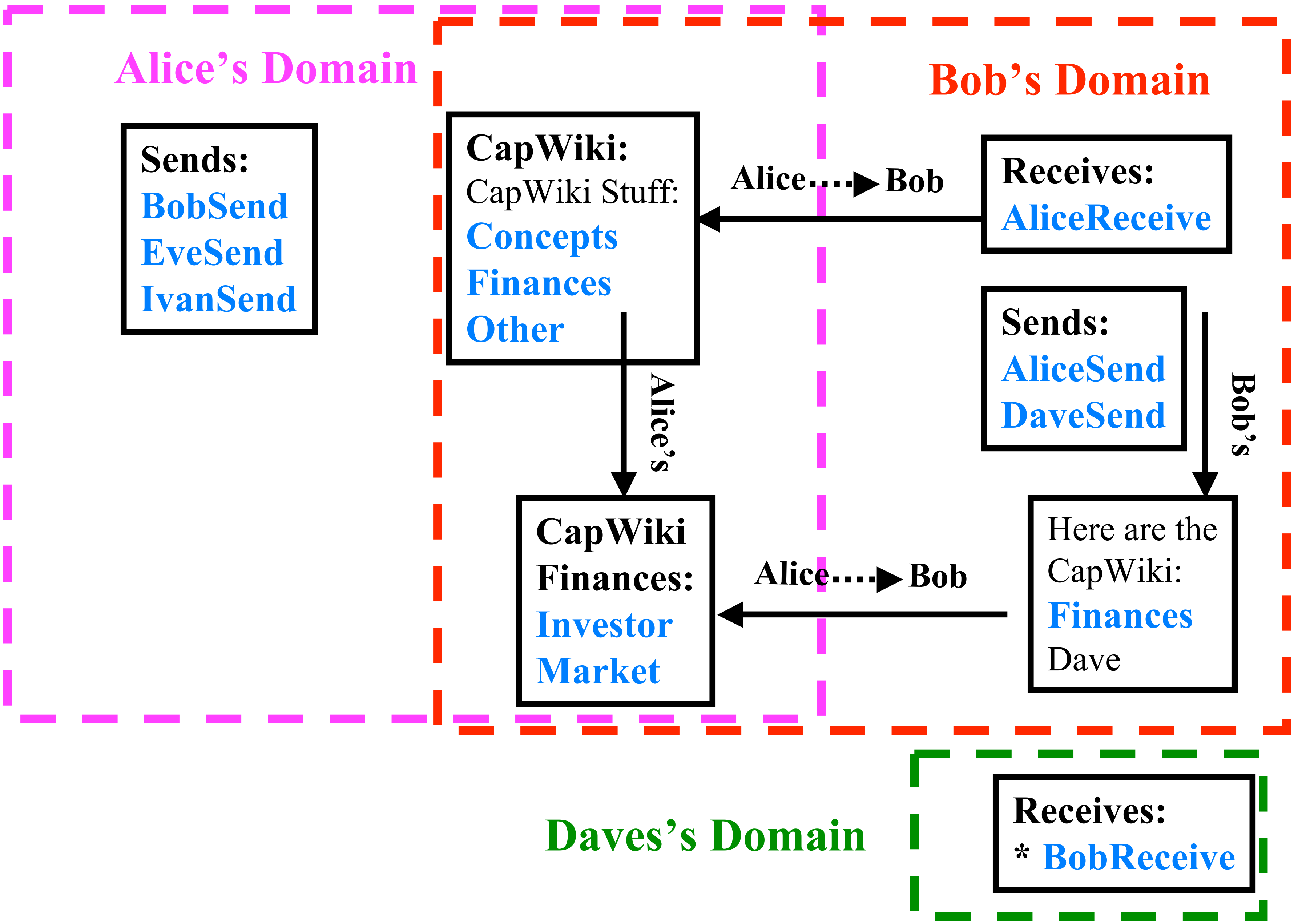
CapWiki
Finances:
Investor
Market

Alice····▶ Bob

Here are the
CapWiki:
Finances
Dave

Daves's Domain

Receives:
BobReceive



Alice’s Domain

Sends:
BobSend
EveSend
IvanSend

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

Alice’s

CapWiki
Finances:
Investor
Market

Bob’s Domain

Alice····▶ Bob

Receives:
AliceReceive

Sends:
AliceSend
DaveSend

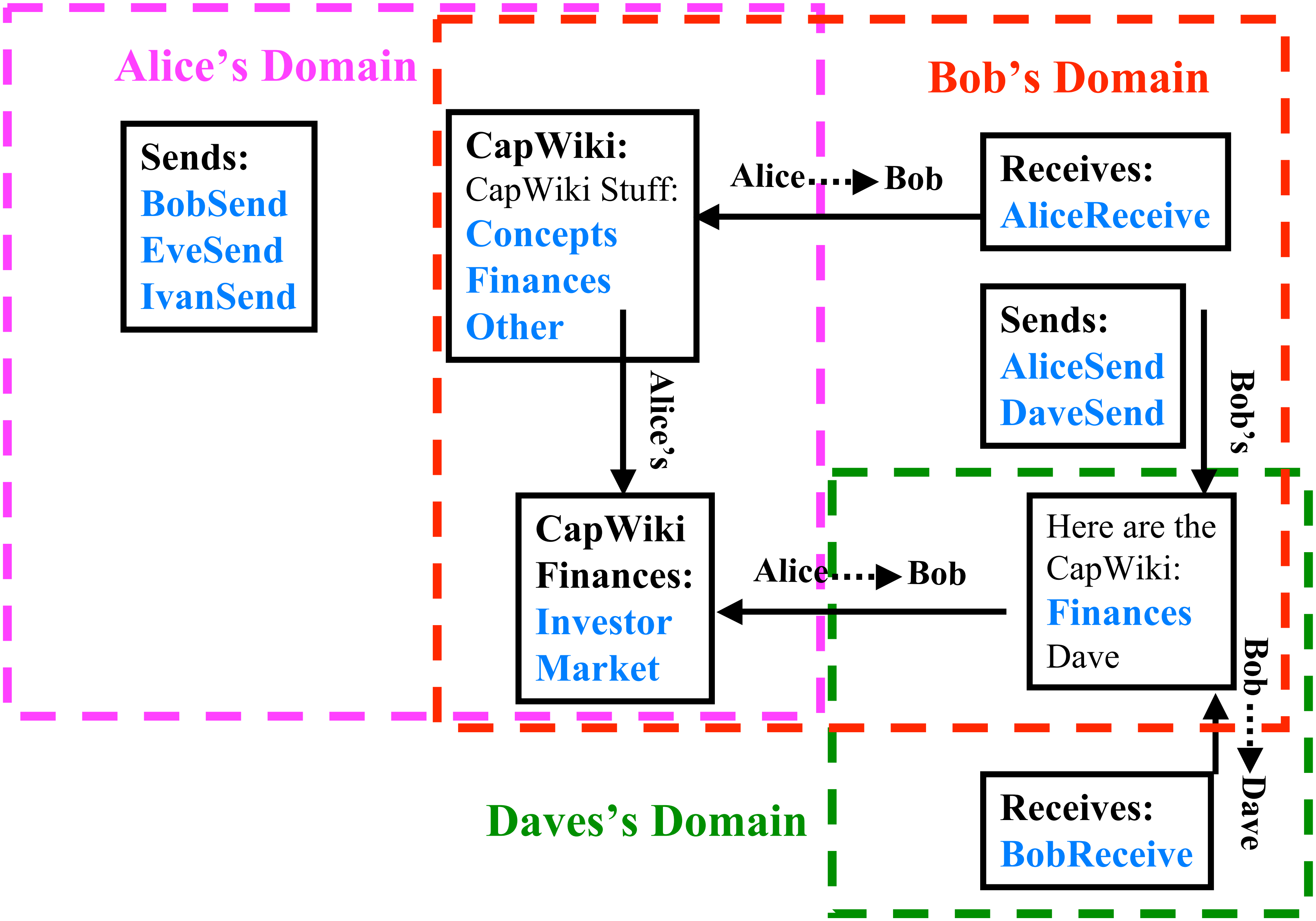
Bob’s

Here are the
CapWiki:
Finances
Dave

Alice····▶ Bob

Daves’s Domain

Receives:
* BobReceive



Alice’s Domain

Sends:
BobSend
EveSend
IvanSend

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

Alice’s

CapWiki
Finances:
Investor
Market

Bob’s Domain

Alice····▶ Bob

Receives:
AliceReceive

Sends:
AliceSend
DaveSend

Bob’s

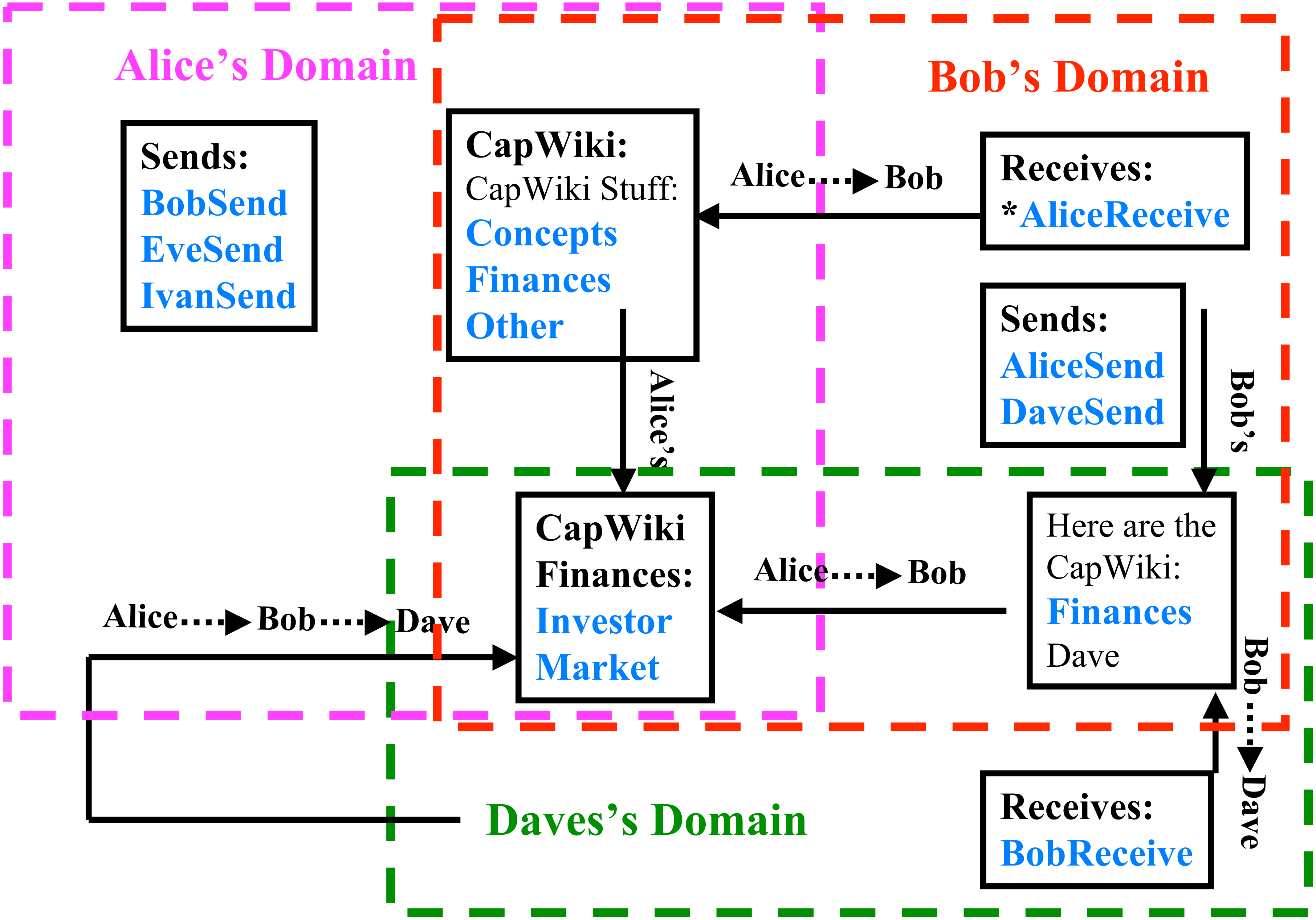
Here are the
CapWiki:
Finances
Dave

Alice····▶ Bob

Bob····▶ Dave

Receives:
BobReceive

Daves’s Domain



Alice’s Domain

Sends:
BobSend
EveSend
IvanSend

CapWiki:
CapWiki Stuff:
Concepts
Finances
Other

Alice····▶Bob

Bob’s Domain

Receives:
*AliceReceive

Sends:
AliceSend
DaveSend

Alice’s

Bob’s

CapWiki
Finances:
Investor
Market

Alice····▶Bob

Here are the
CapWiki:
Finances
Dave

Alice····▶Bob····▶Dave

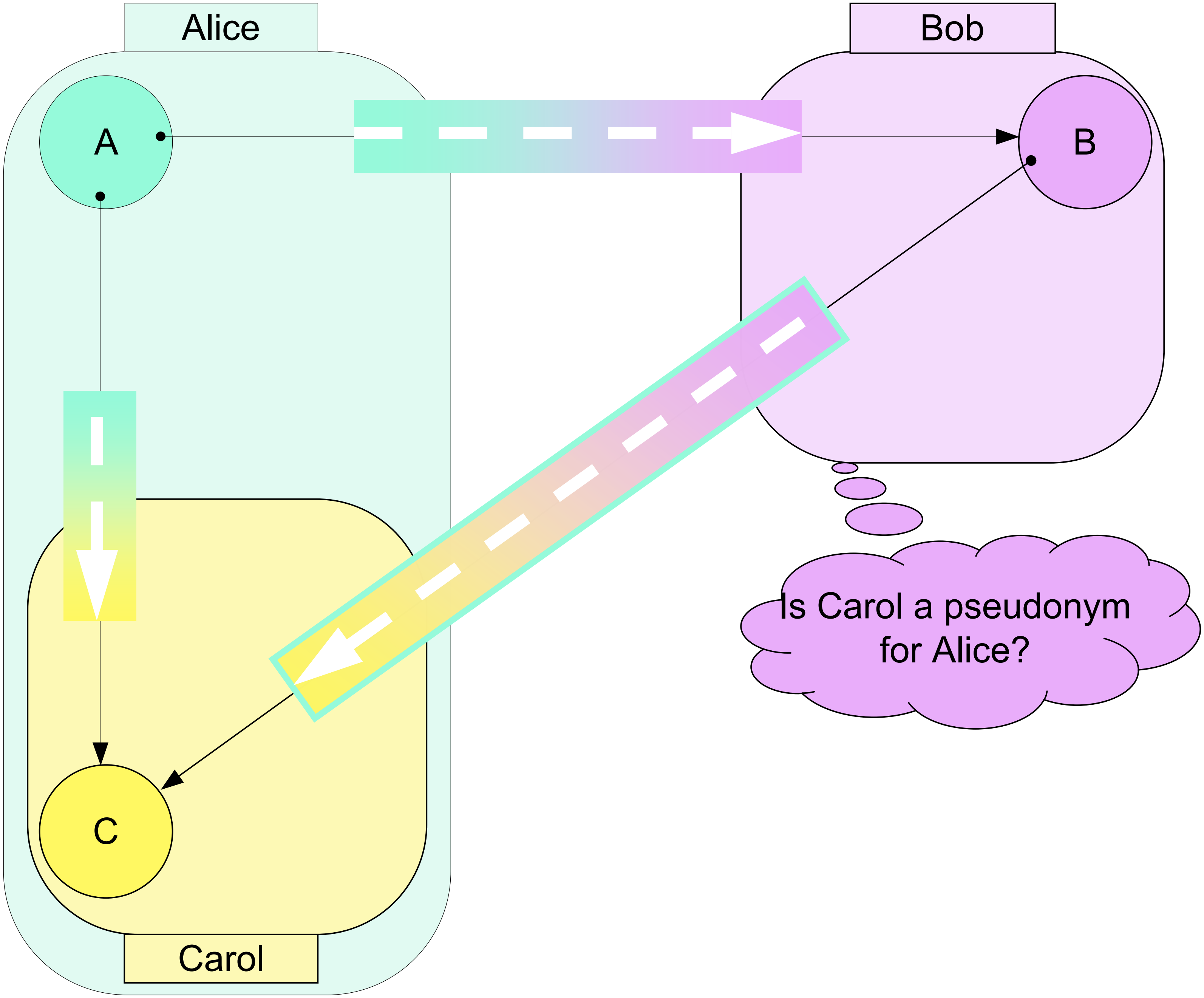
Bob····▶Dave

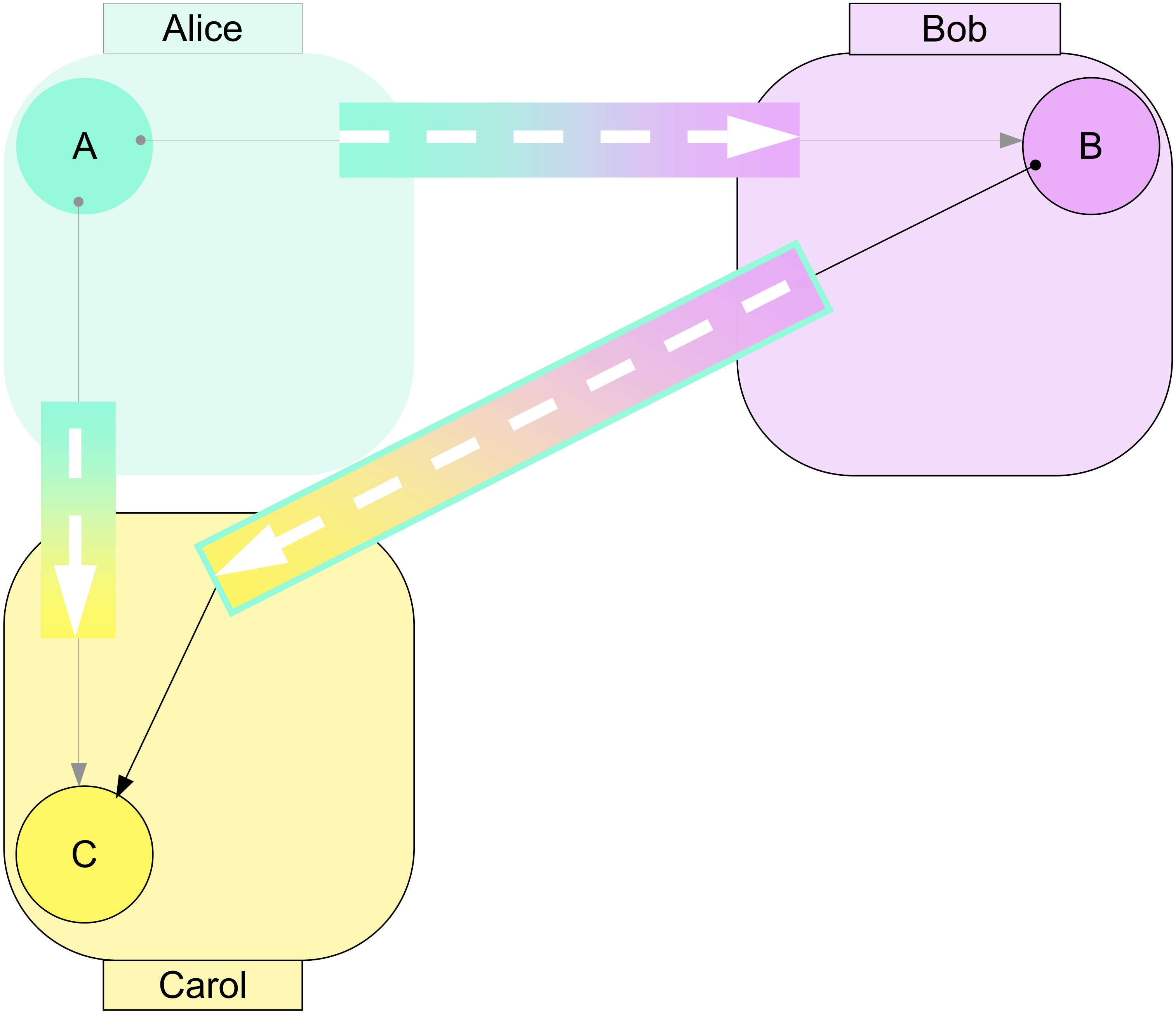
Daves’s Domain

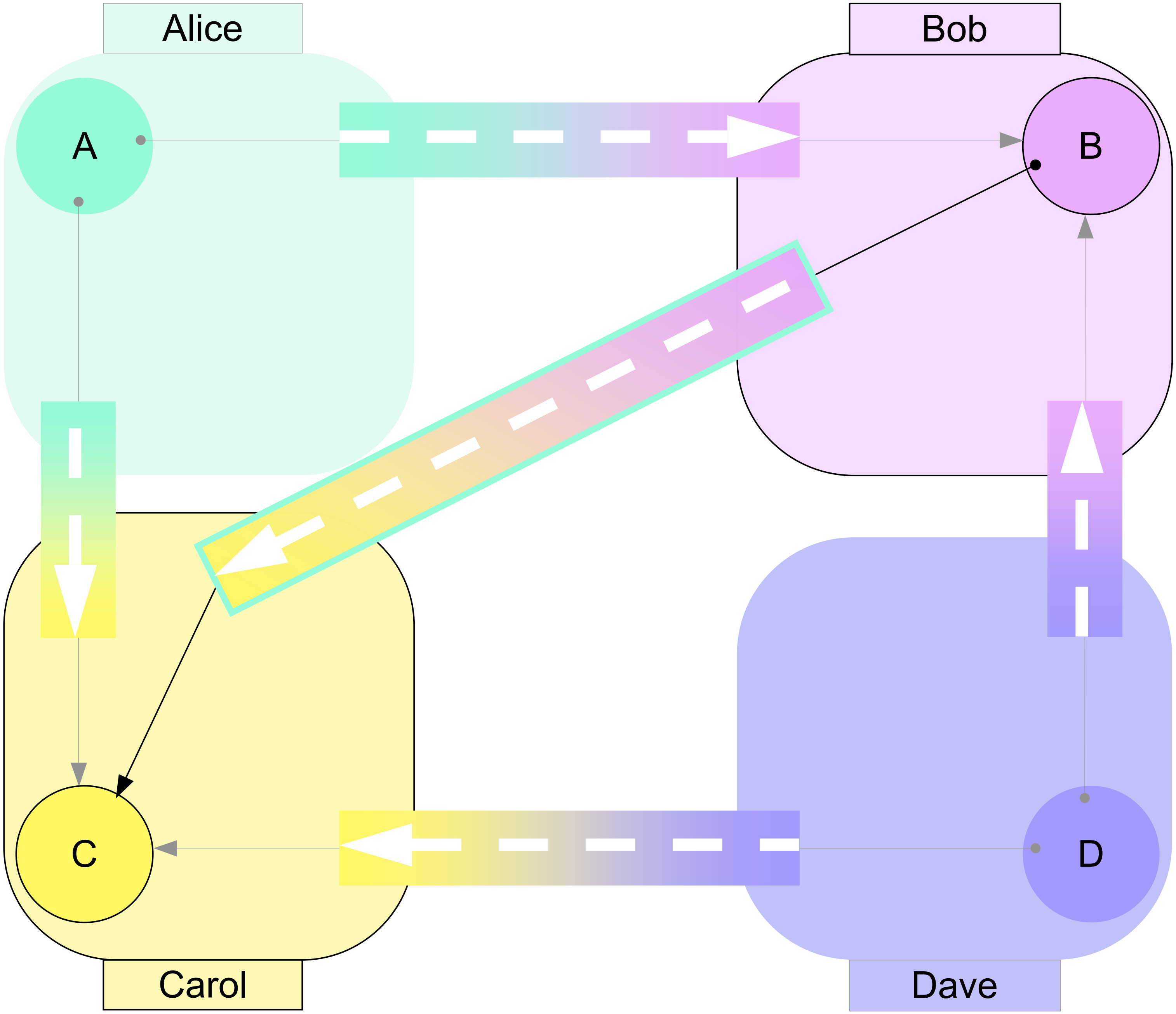
Receives:
BobReceive

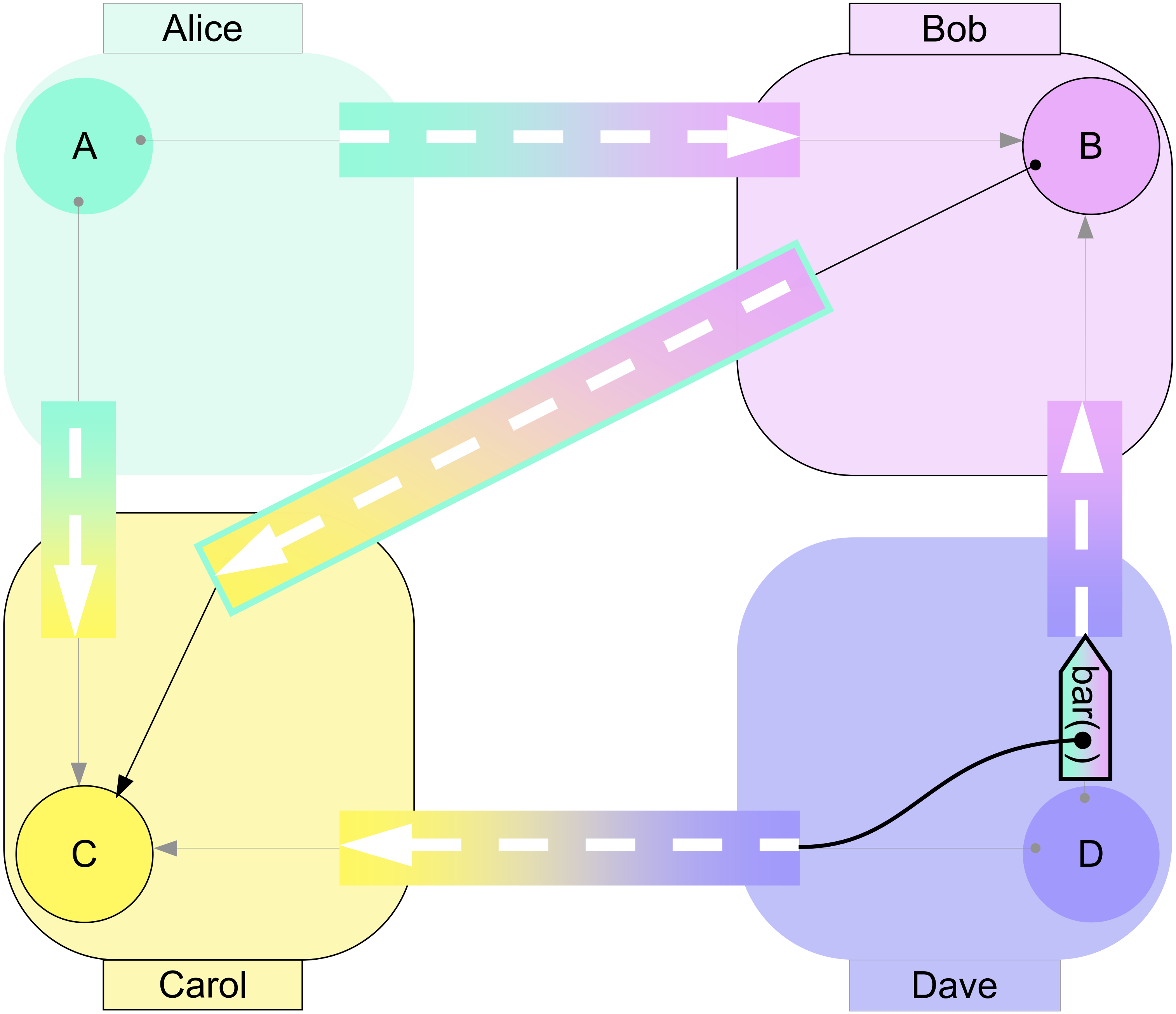
Better Web Access Control

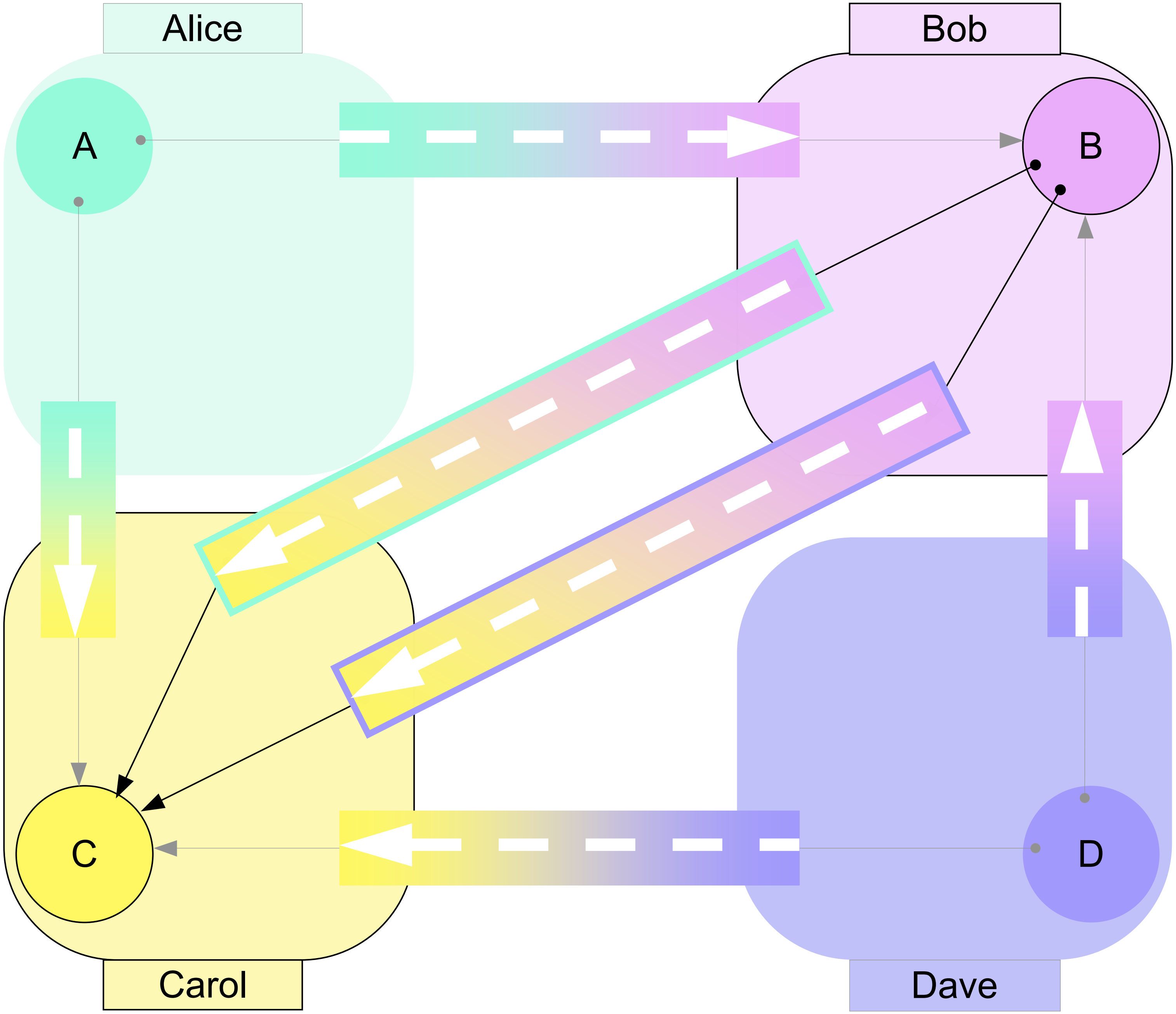
- No more passwords – Send a <me>Send to a <service>Send. They know who you are, you know who they are.
- Side benefit – SPAM resistance. Don’t like a source of SPAM, cut it off to any delegation level.
- Principle Of Least Authority (POLA) sharing that can facilitate cross site services.

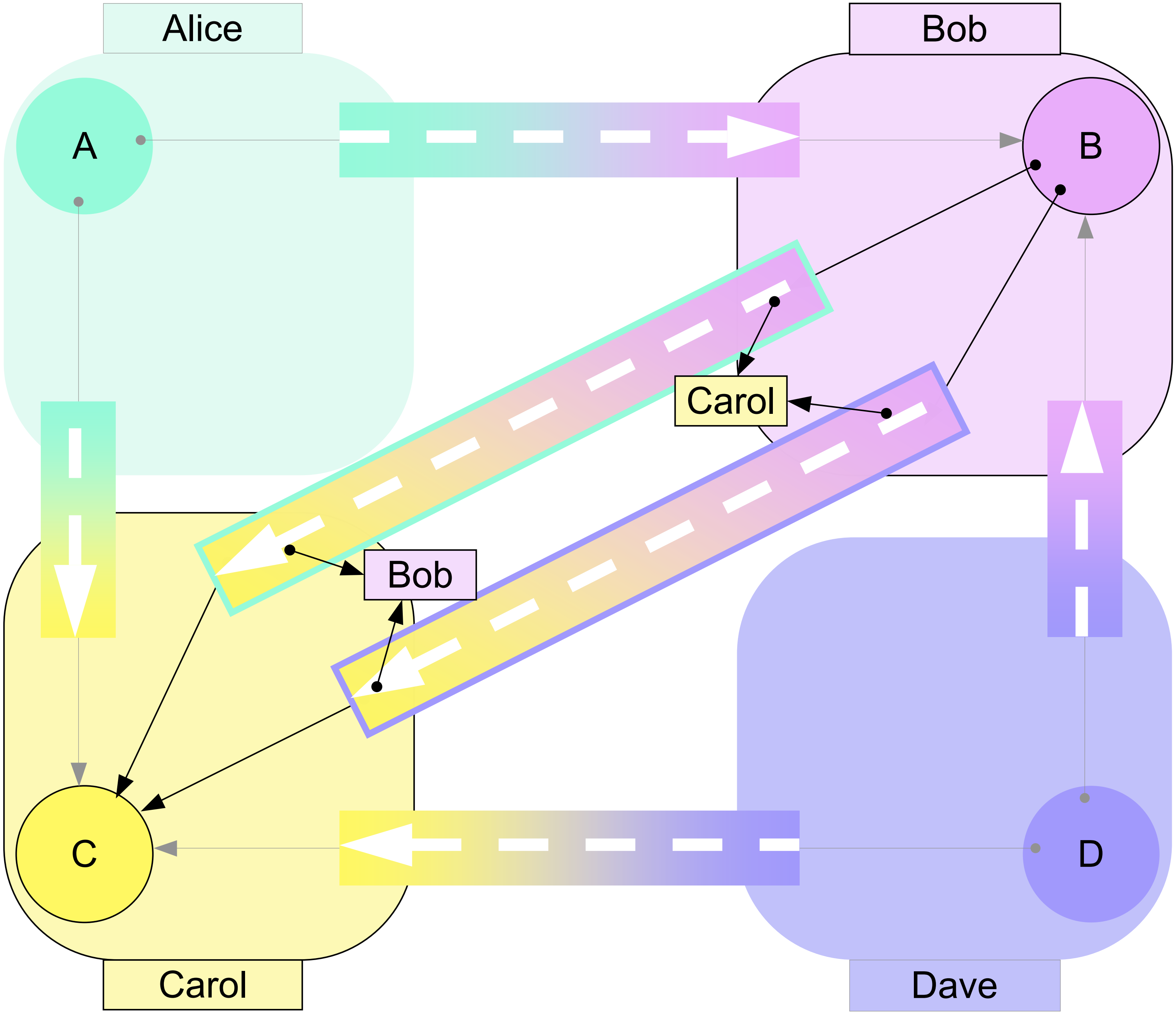

Alice
Bob
A
B
C
Carol
Is Carol a pseudonym
for Alice?


Alice
Bob
A
B
C
Carol


Alice
Bob
A
B
C
D
Carol
Dave


Alice
Bob
A
B
C
D
bar()
Carol
Dave


Alice
Bob
A
B
C
D
Carol
Dave



Alice
Bob
A
B
Carol
Bob
C
D
Carol
Dave