ECOOP'12

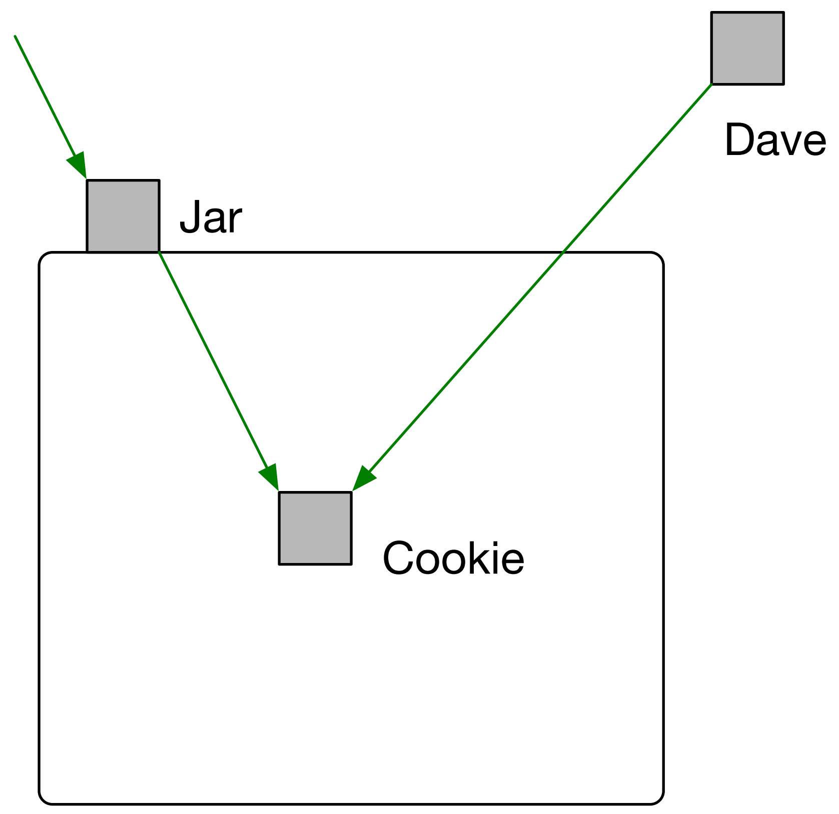
Multiple Aggregate Entry Points for Ownership Types

Johan Östlund

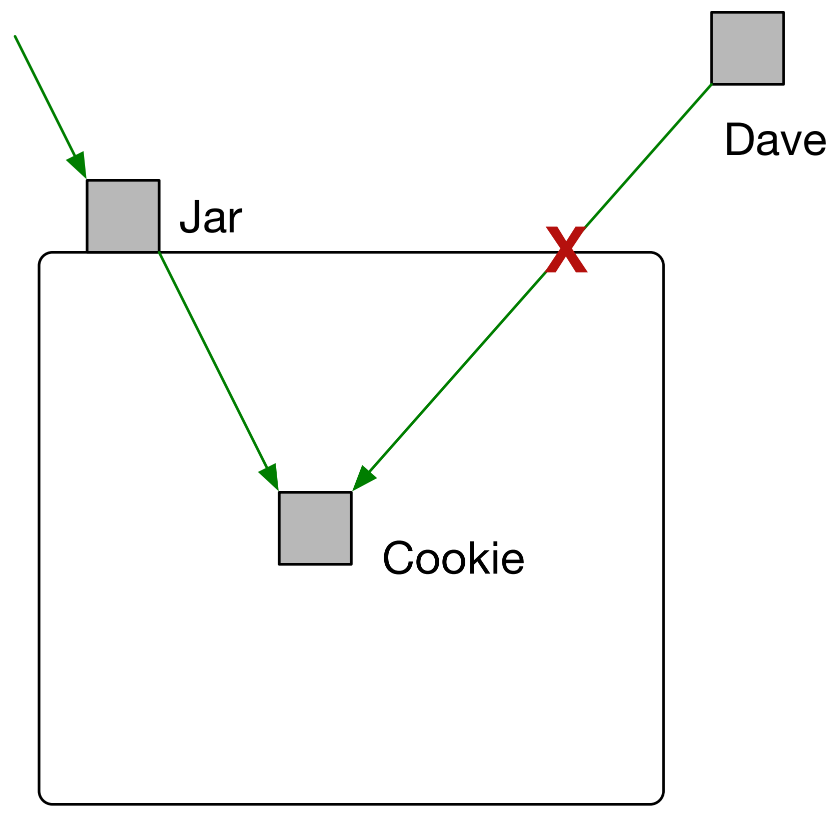
Tobias Wrigstad

Department of
Information Technology
Uppsala University
Sweden

The Perils of Life

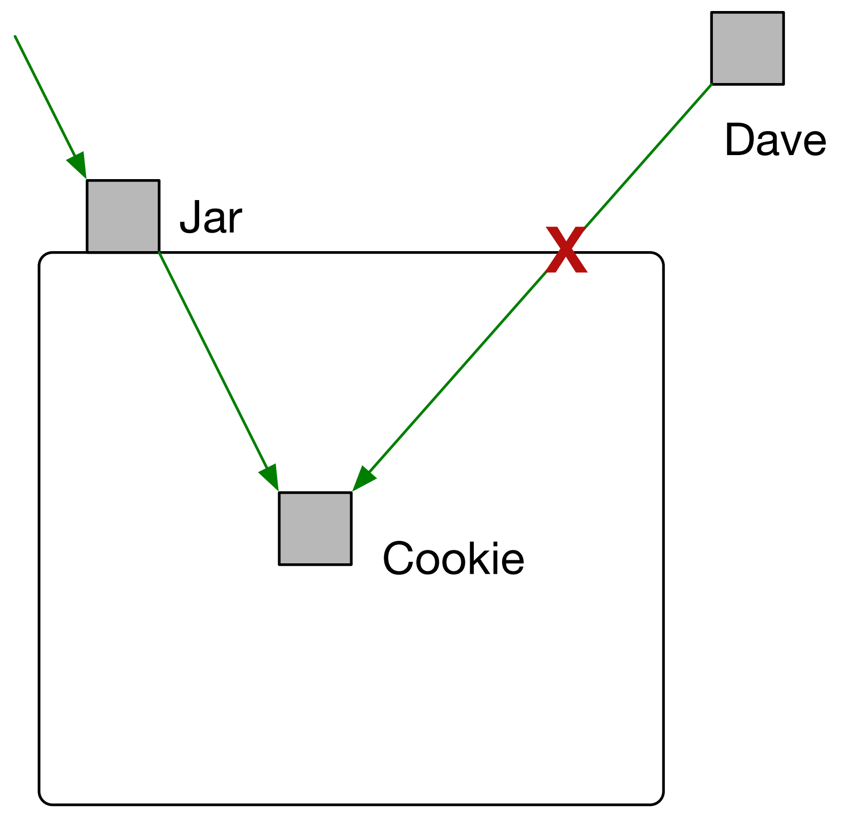


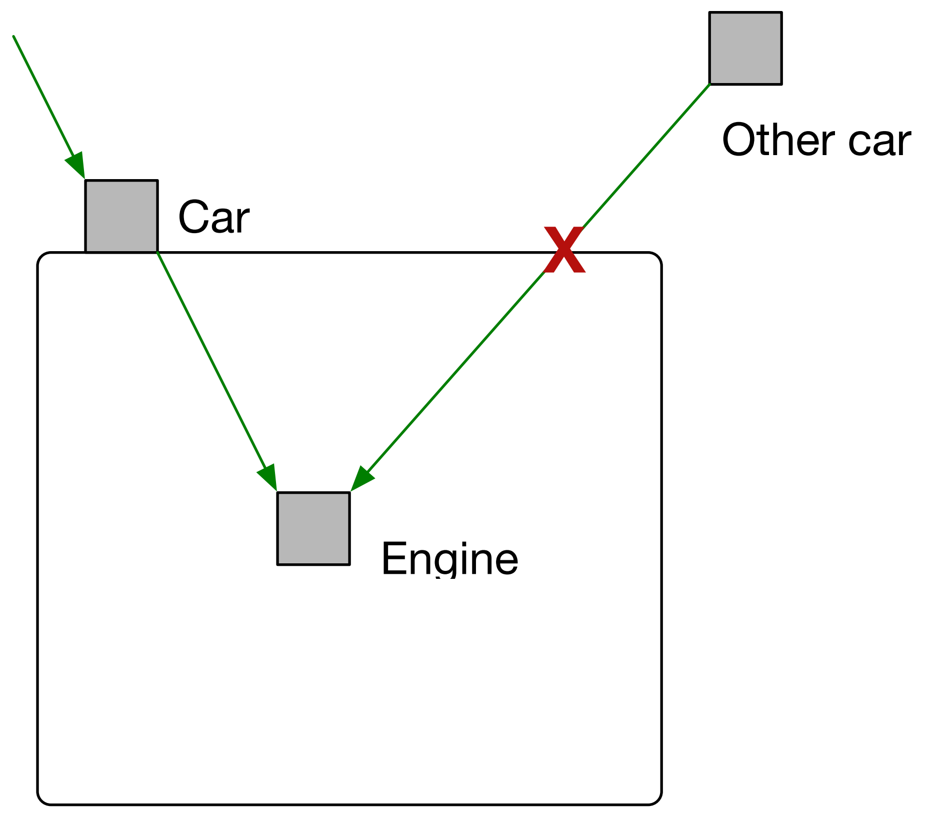
Ownership Types



- Strong notion of aggregate

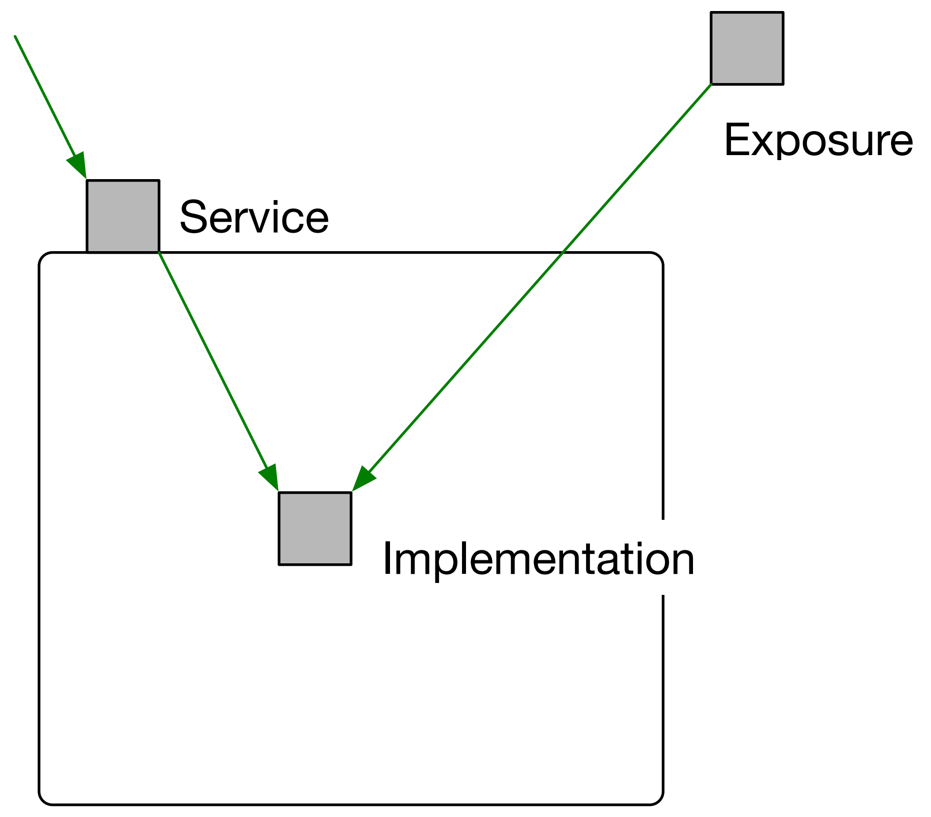
Ownership Types Models the Physical World





- Strong notion of aggregate

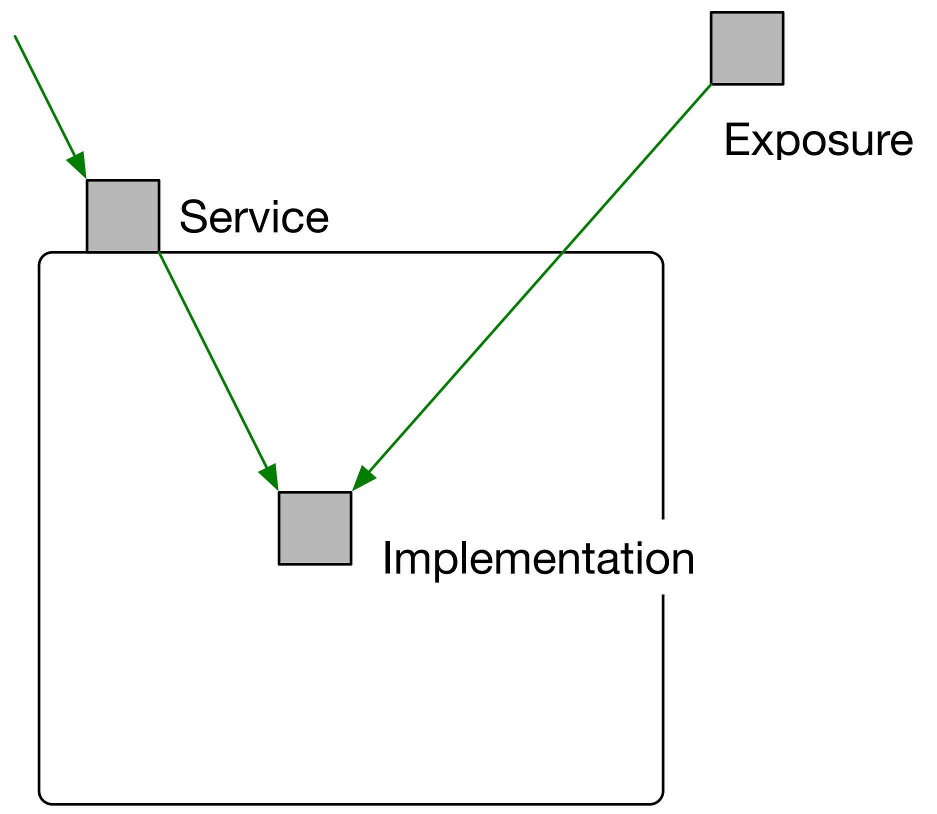
Limitations

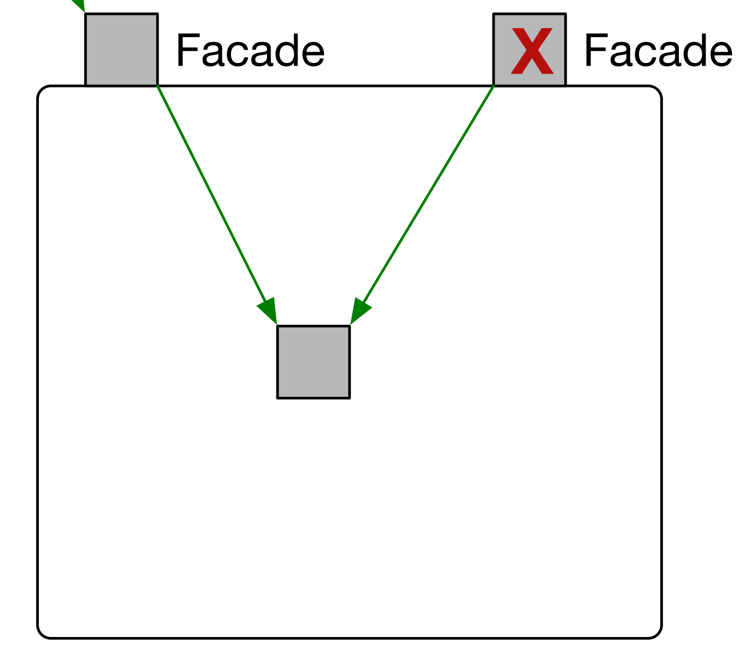


- Strong notion of aggregate

  — OAD forces single entry point

Limitations

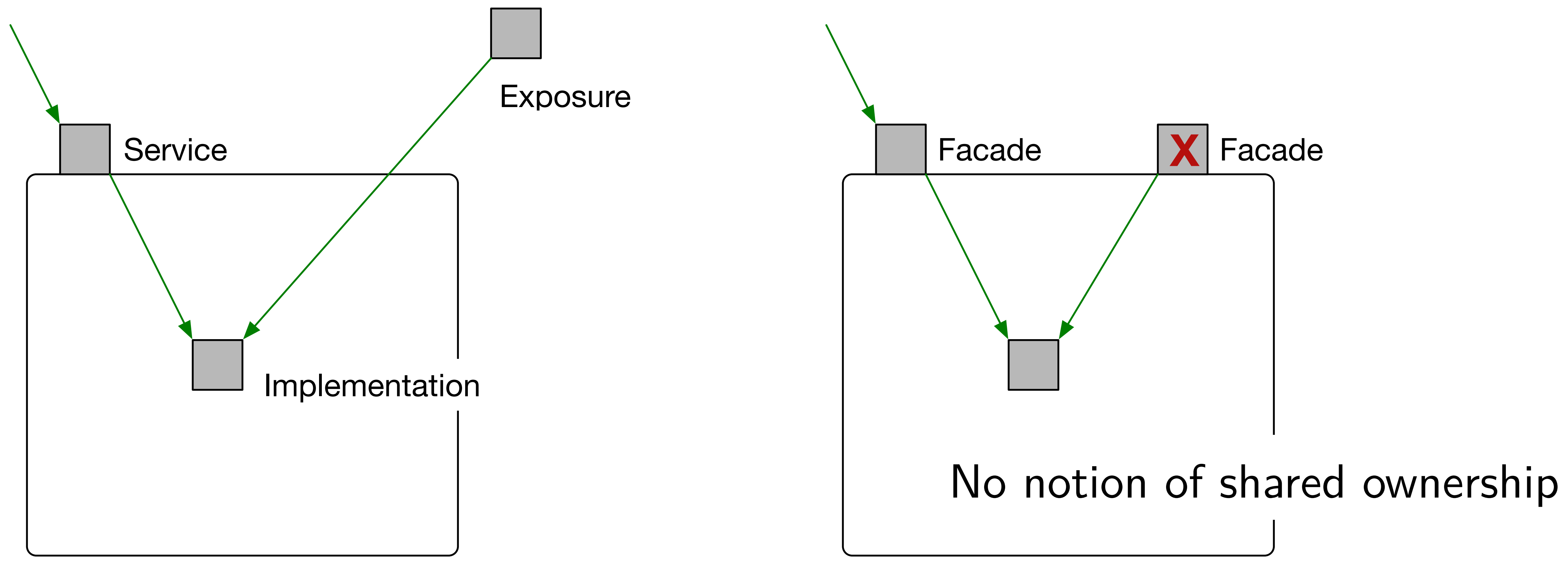




- Strong notion of aggregate

  — OAD forces single entry point

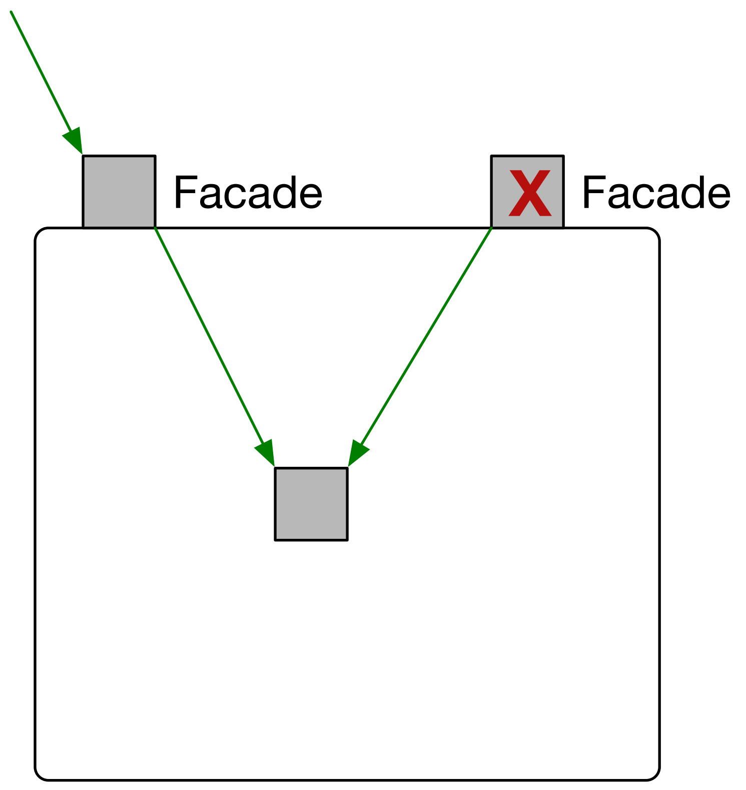
Limitations



- Strong notion of aggregate
  — OAD forces single entry point

No notion of shared ownership

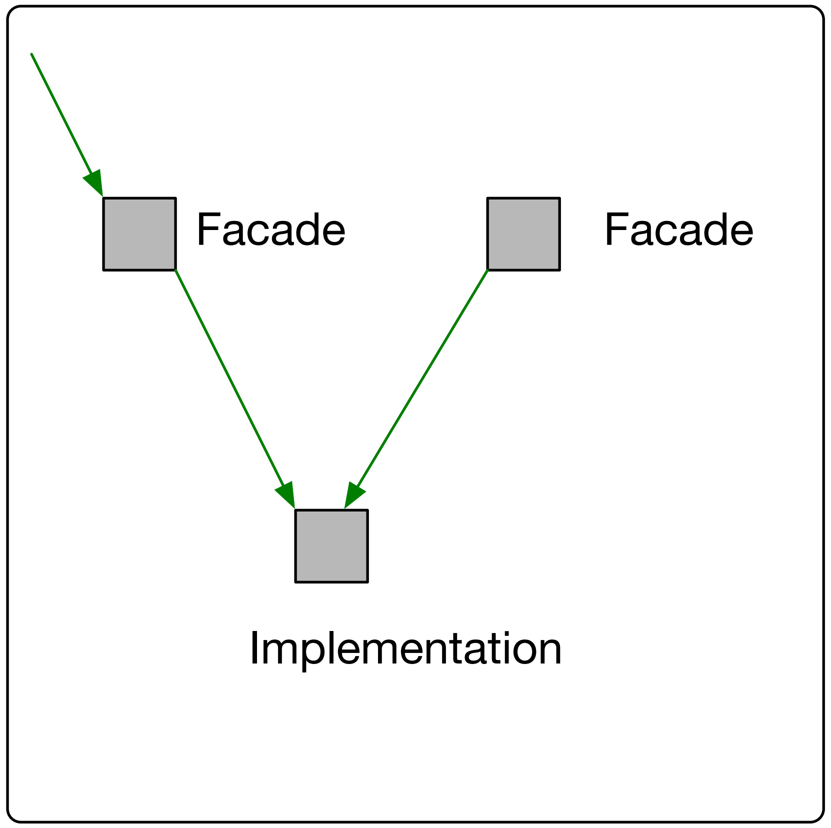
- Bad fit for components
- Many patterns temporarily break OAD (e.g., iterators)
- Solution: flatten ownership hierarchy



— *No multiple entry points*

— *No "friends"*

Flatten Ownership Hierarchy

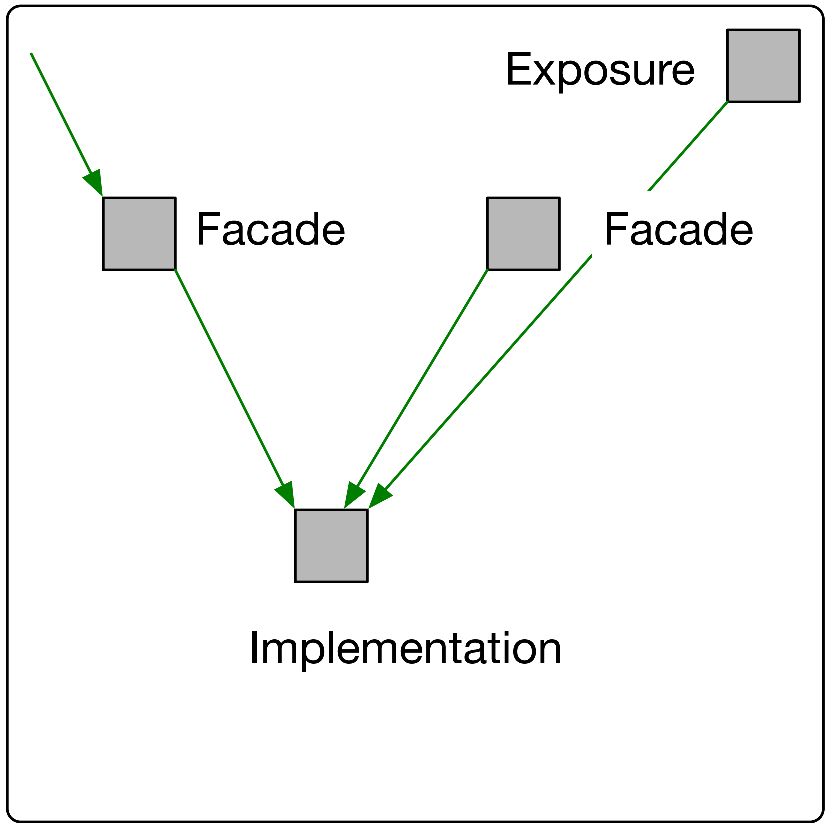


- Lift the ownership of the implementation to the level of the facade
- All objects become siblings (or peers)

— *No multiple entry points*

— *No "friends"*

Flatten Ownership Hierarchy



- Lift the ownership of the implementation to the level of the facade
- All objects become siblings (or peers)
- ...but, sadly, enables exposure

— *No multiple entry points*

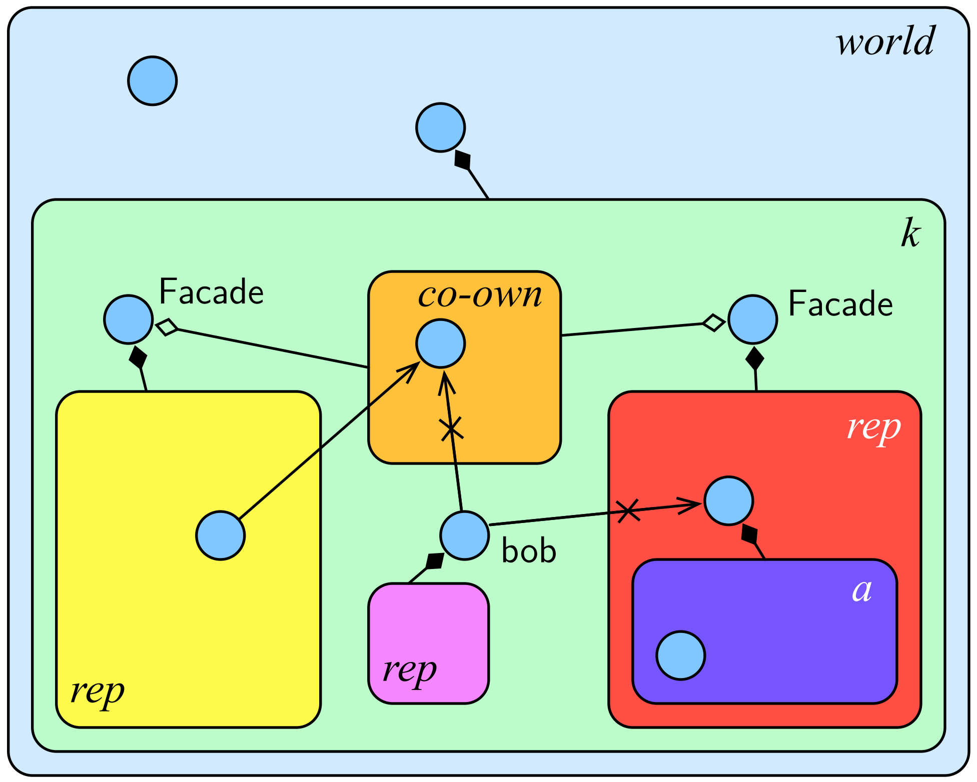
— *No "friends"*

This Talk: **Co-Ownership**

” *Allowing several objects to* ***collaboratively*** *and with* ***equal rights*** *define a* ***single, shared context***

This Talk: **Co-Ownership**

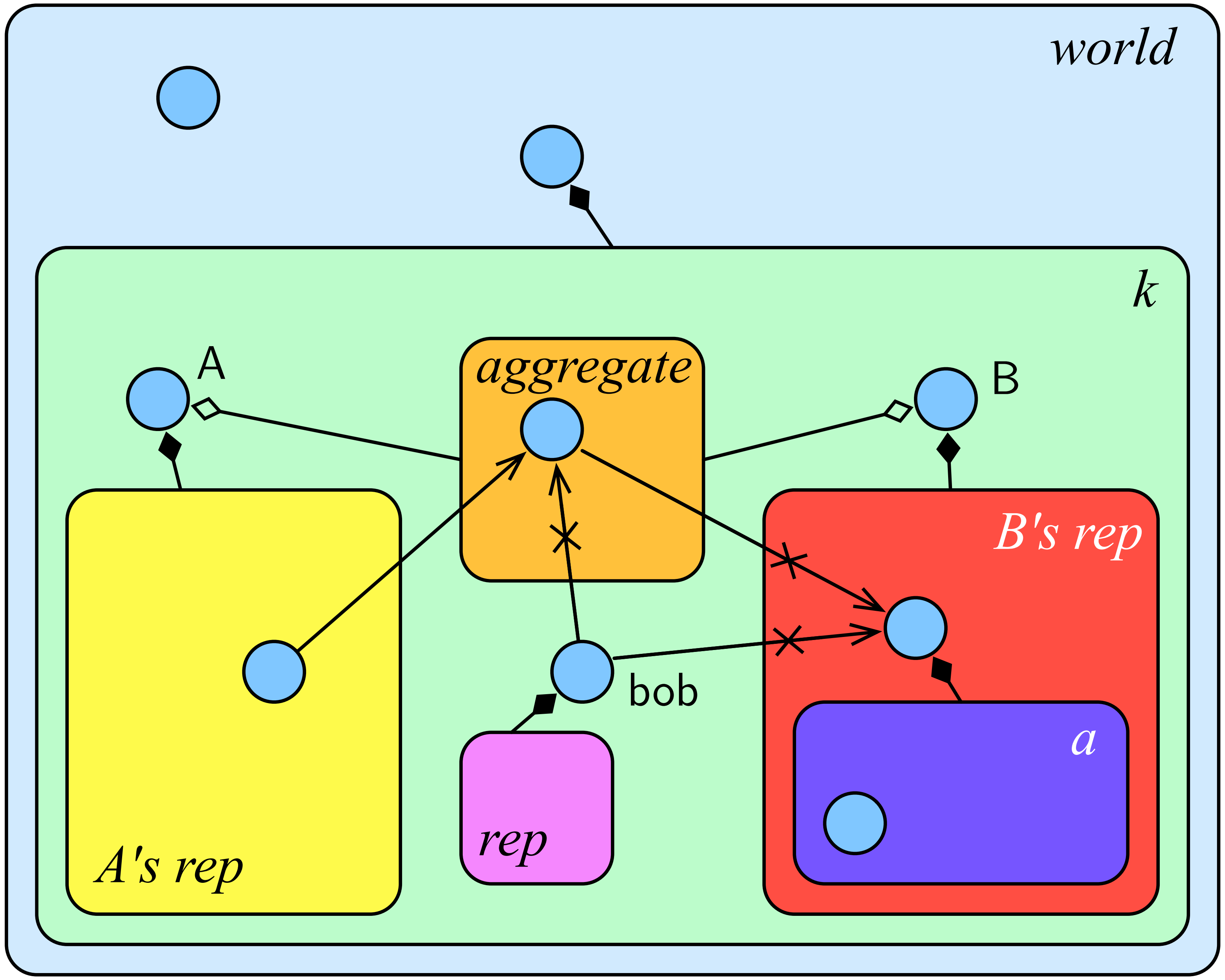
> *Allowing several objects to* ***collaboratively*** *and with* ***equal rights*** *define a* ***single, shared context***



- Multiple entry points to aggregates
- "Friendship"
- Essentially a simplification of multiple ownership [MOJO]
- A "disciplined relaxation of OAD"

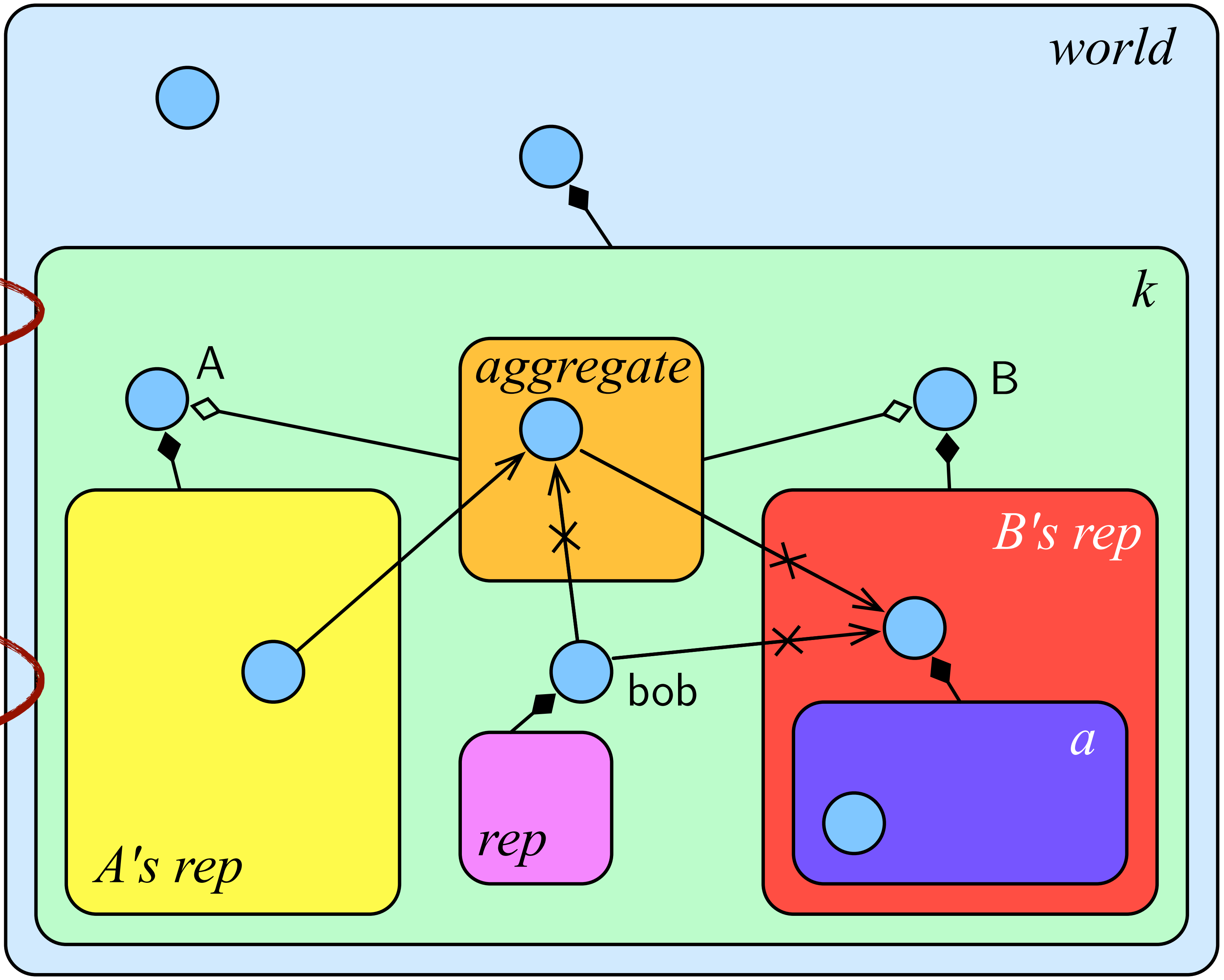
Co-Ownership

- An **aggregate** context is co-owned by a number of **bridge objects**
- **Bridge objects** are siblings
- Different siblings may have different aggregates
- An object cannot be a **bridge** for more than one **aggregate**
- Objects in the aggregate enjoy **strong** encapsulation



Co-Ownership

- An **aggregate** context is co-owned by a number of **bridge objects**
- **Bridge objects** are siblings
- Different siblings may have different aggregates
- An object cannot be a **bridge** for more than one **aggregate**
- Objects in the aggregate enjoy **strong** encapsulation



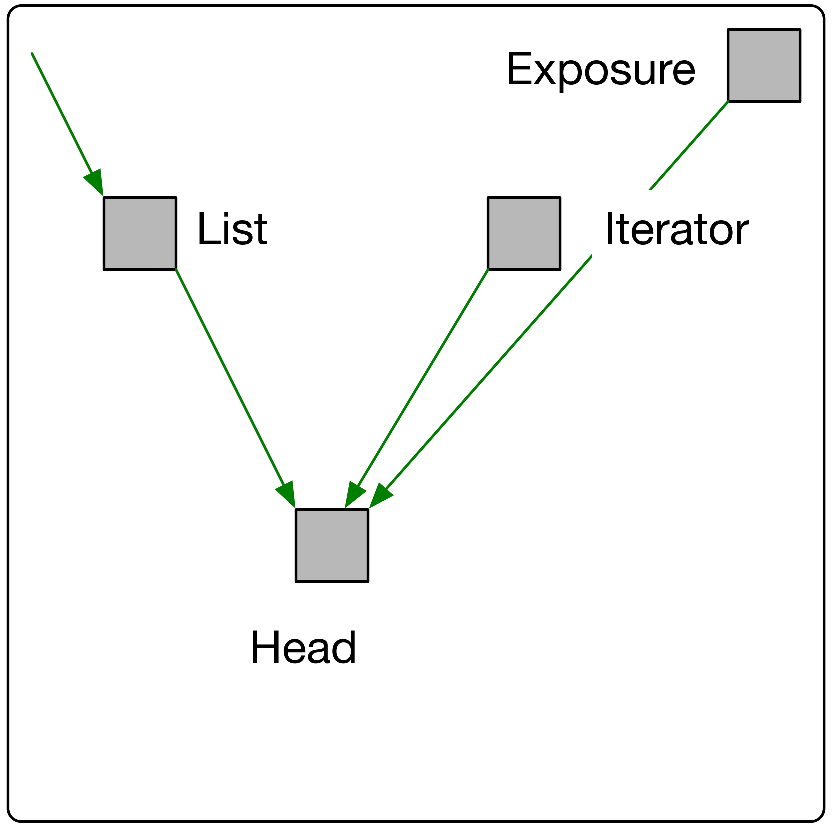
Example: Iterators

```
class List<data> { Link<rep, data> head; ... }
```

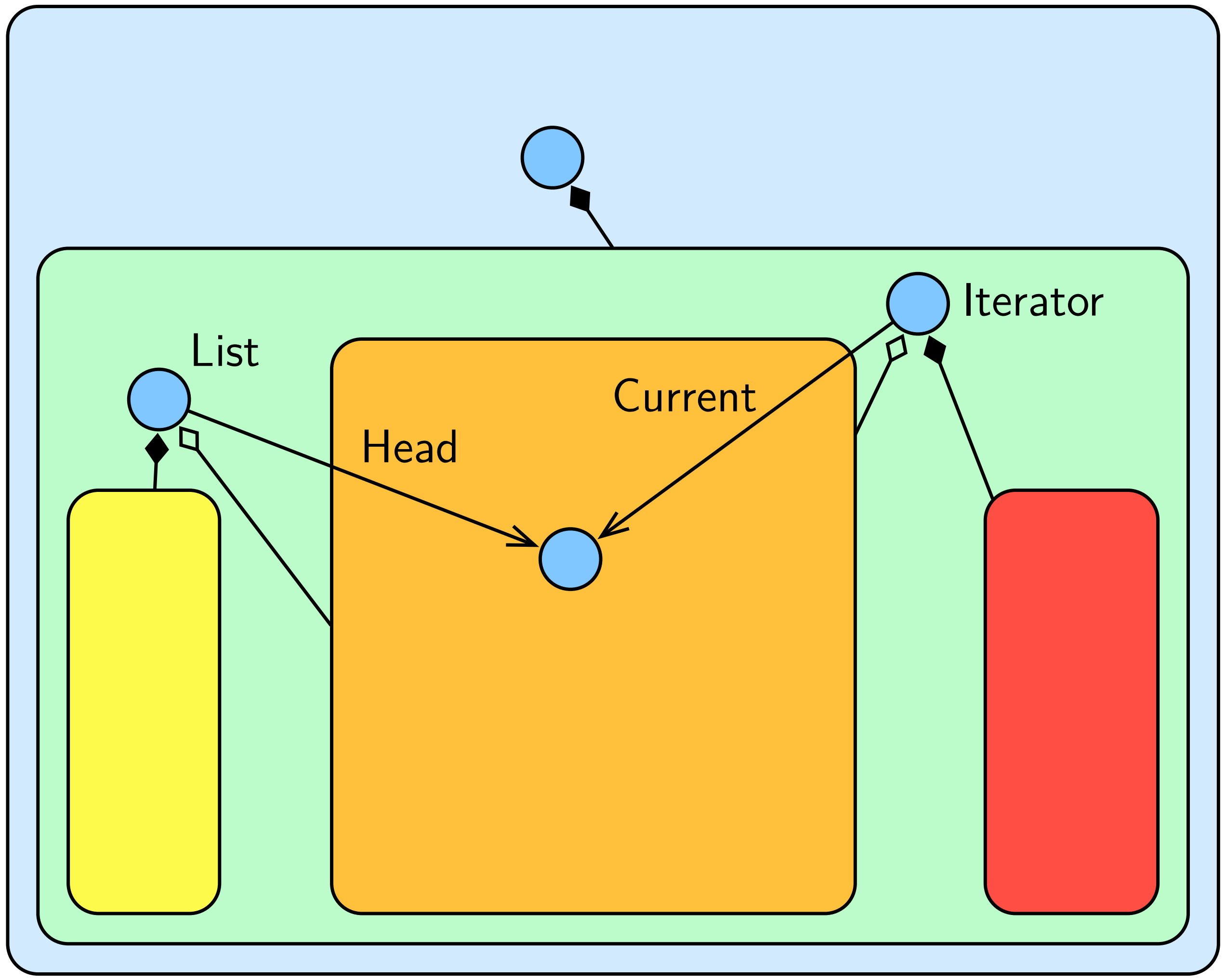
```
// Iterator through flattening
class List<data> {
  Link<owner, data> head; ...

  Iterator<owner, data> iter;

  Iterator<owner, data> iterator() {
    return iter;
  }
}
```



Example: Iterators



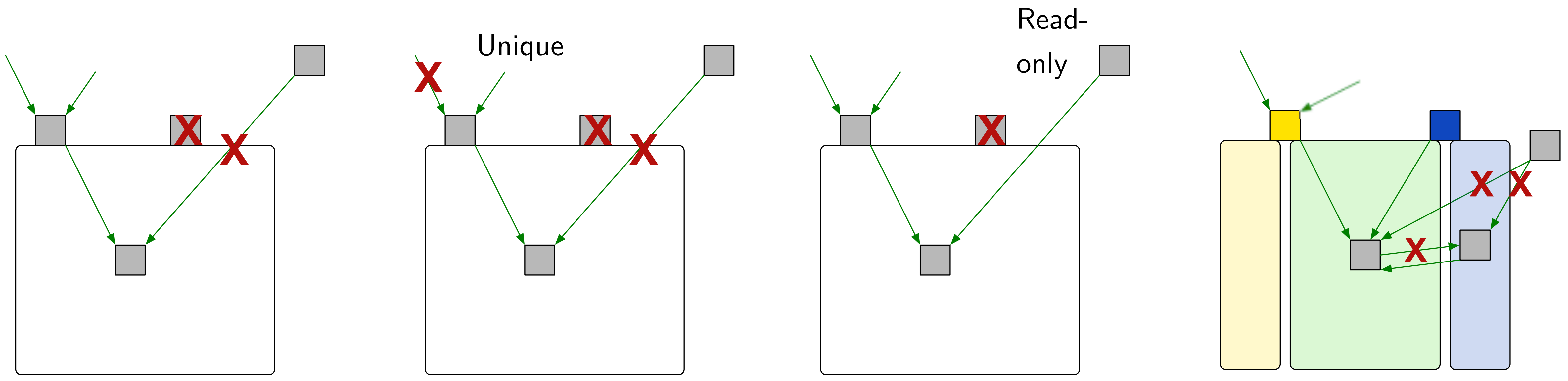
```
class List<data> {
  Link<aggregate, data> head; ...

  Iterator<bridge, data> iter;

  Iterator<bridge, data> iterator() {
    return iter;
  }
}
```

Comparison With Existing Systems

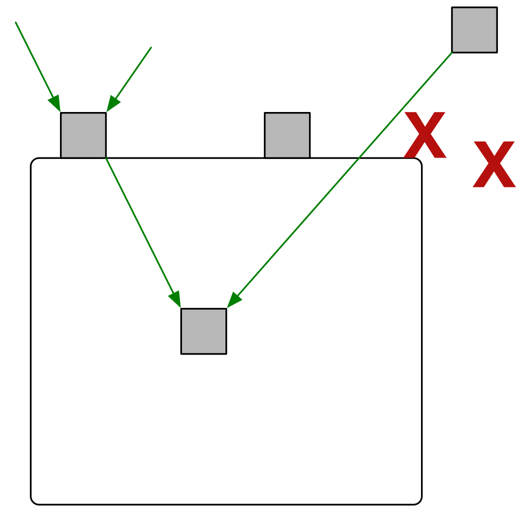
Owners-as-ombudsmen: *every path from a root in the system to an object in an aggregate context contains one of the context's bridge objects.*



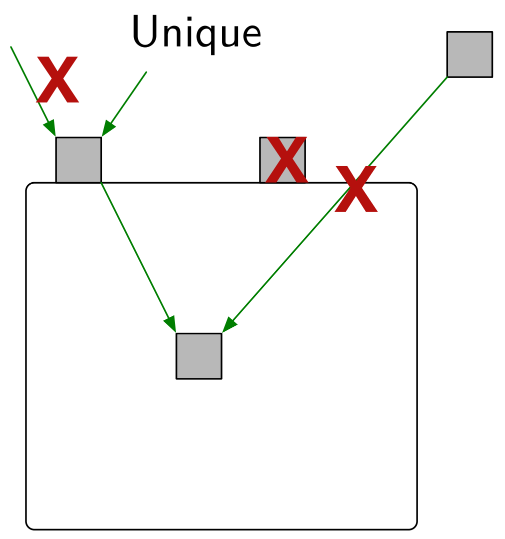
OAD EUADE OAM OAO

Comparison With Existing Systems

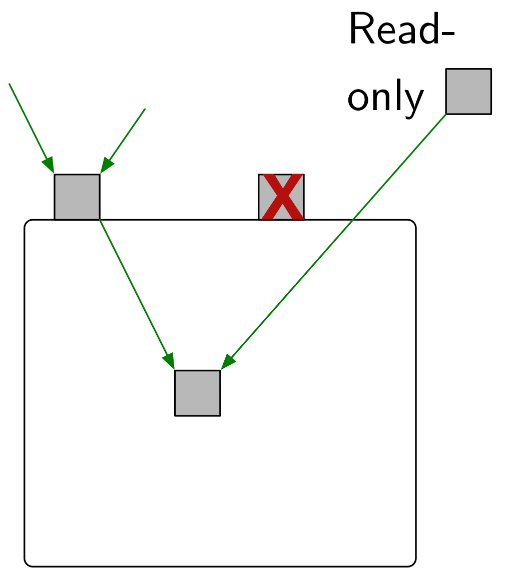
Owners-as-ombudsmen: *every path from a root in the system to an object in an aggregate context contains one of the context's bridge objects.*



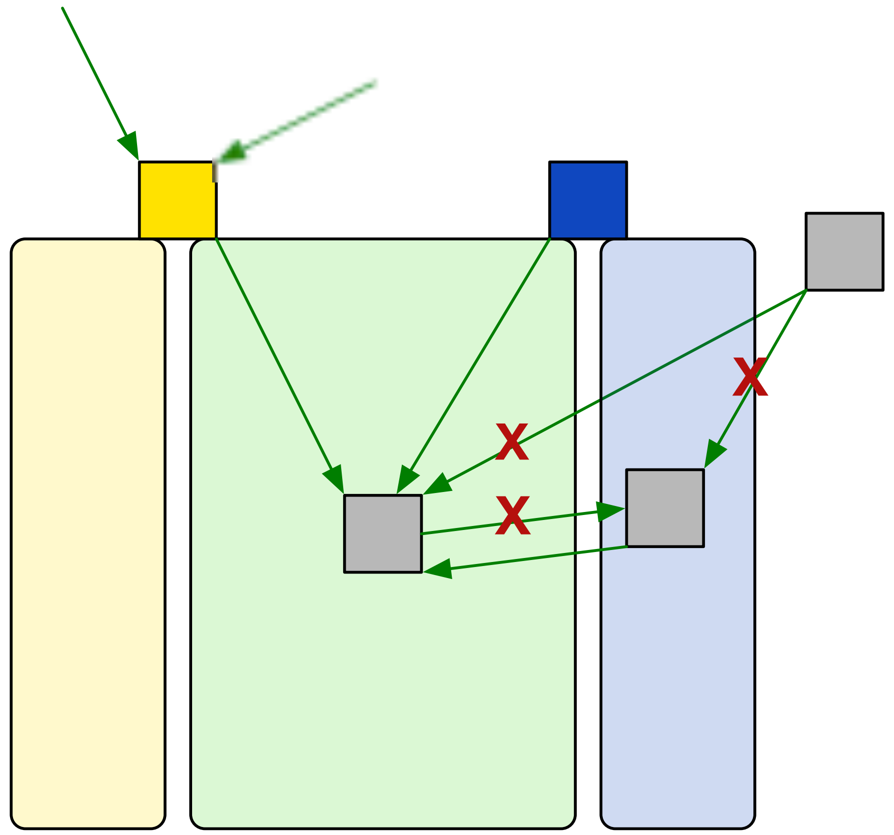
OAD



EUADE



OAM



OAO

Typing Co-Ownership

In addition to its explicitly given permissions, an object has access to

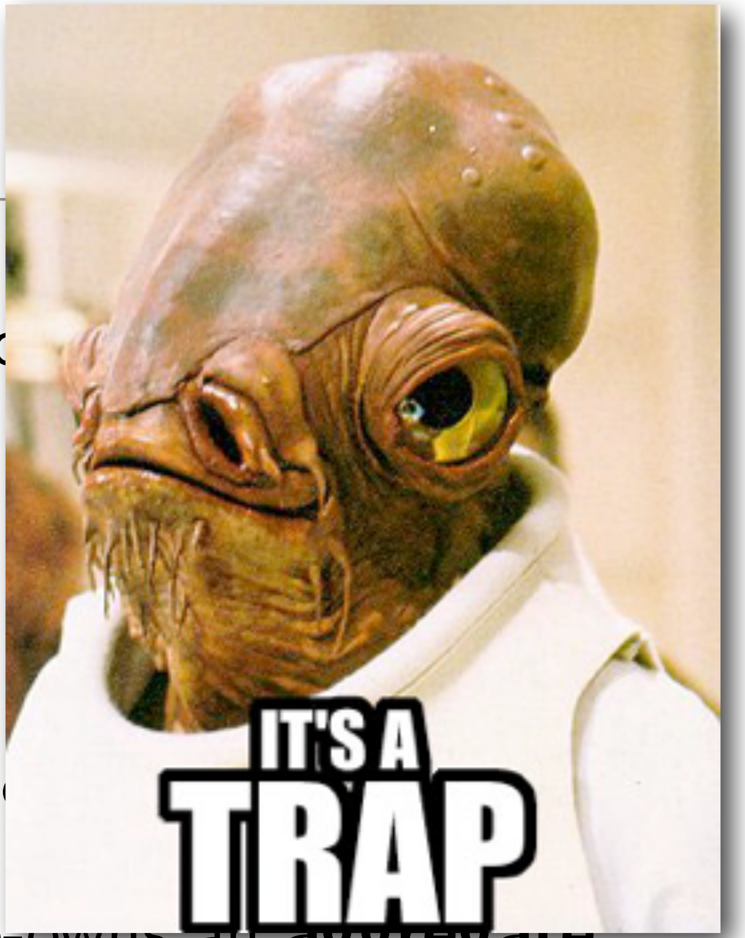
- **owner** — the objects in its owning context
- **rep** — the objects it owns
- **aggregate** — the objects in the aggregate it co-owns with others
- **bridge** — the objects in its **owner** context with which it co-owns an **aggregate**

$$\frac{\begin{array}{c}\text{(WF-CLASS)}\\ E = \mathbf{owner} \prec^* \mathbf{world}, \mathbf{rep} \prec^* \mathbf{owner}, \mathbf{bridge} \prec^* \mathbf{owner}, \backslash \\ \mathbf{aggregate} \prec^* \mathbf{owner}, \overline{p} \succ^* \mathbf{owner}, \mathbf{this} : \mathtt{C}\langle \mathbf{bridge}, \overline{p}\rangle \\ \{\overline{q}\} \subseteq \{\overline{p}\} \qquad \mathbf{owner} \notin \{\overline{p}\} \\ \tau_{\mathsf{s}} = \mathtt{D}\langle \mathbf{owner}, \overline{q}\rangle \qquad E; \tau_{\mathsf{s}} \vdash \overline{F} \qquad E; \tau_{\mathsf{s}} \vdash \overline{M}\end{array}}{\vdash \mathbf{class}\ \mathtt{C}\langle \mathbf{owner}, \overline{p}\rangle\ \mathbf{extends}\ \mathtt{D}\langle \overline{q}\rangle\ \{\ \overline{F}\ \overline{M}\ \}}$$

Typing Co-Ownership

In addition to its explicitly given permissions, an object has ac

- **owner** — the objects in its owning context
- **rep** — the objects it owns
- **aggregate** — the objects in the aggregate it co-owns with
- **bridge** — the objects in its **owner** context with which it co-owns an **aggregate**

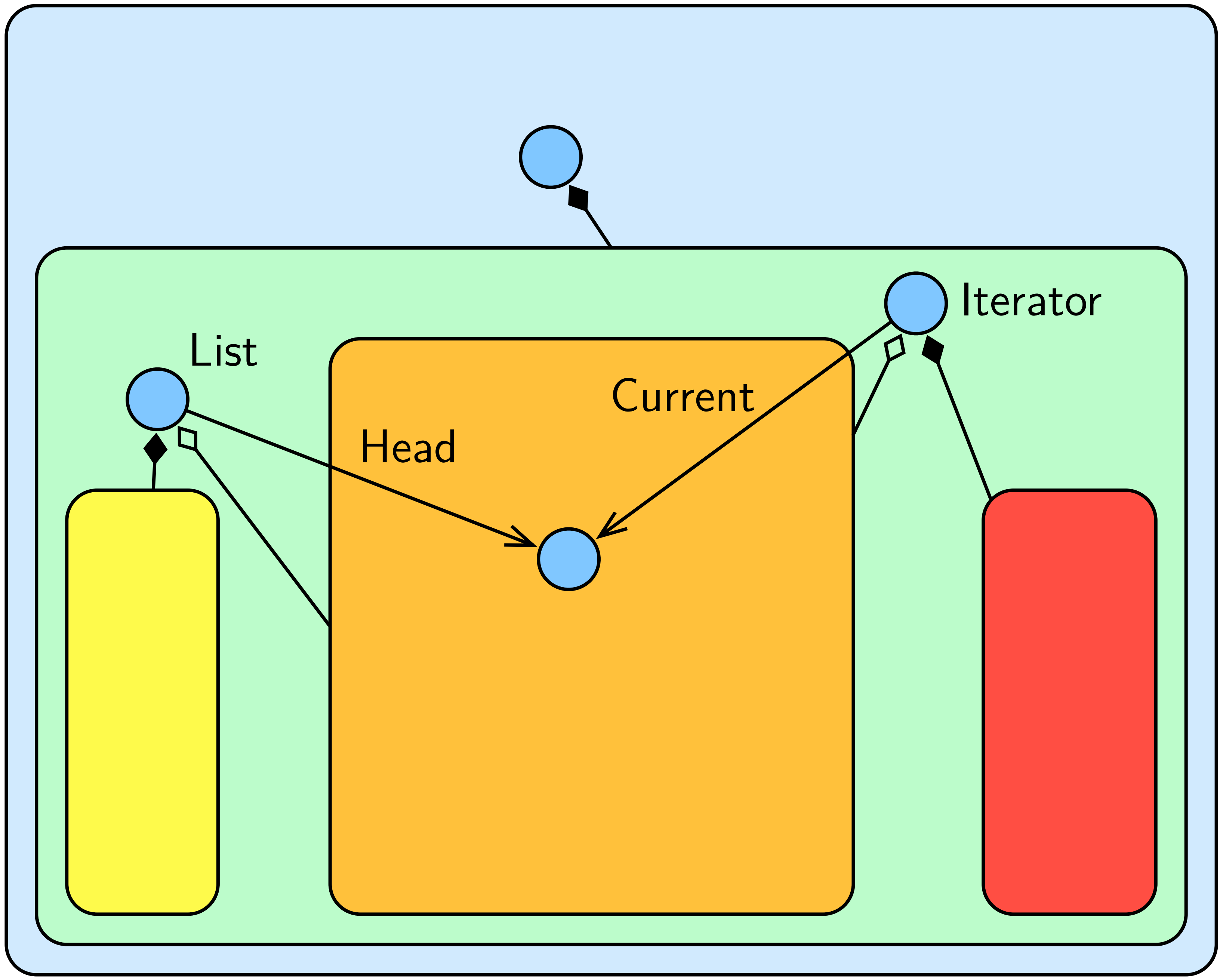


(WF-CLASS)

$$\frac{\begin{array}{c} E = \mathtt{owner} \prec^* \mathtt{world}, \mathtt{rep} \prec^* \mathtt{owner}, \mathtt{bridge} \prec^* \mathtt{owner}, \backslash \\ \mathtt{aggregate} \prec^* \mathtt{owner}, \overline{p} \succ^* \mathtt{owner}, \mathtt{this} : \mathtt{C}\langle \mathtt{bridge}, \overline{p} \rangle \\ \{\overline{q}\} \subseteq \{\overline{p}\} \qquad \mathtt{owner} \notin \{\overline{p}\} \\ \tau_s = \mathtt{D}\langle \mathtt{owner}, \overline{q} \rangle \qquad E; \tau_s \vdash \overline{F} \qquad E; \tau_s \vdash \overline{M} \end{array}}{\vdash \mathtt{class}\ \mathtt{C}\langle \mathtt{owner}, \overline{p} \rangle\ \mathtt{extends}\ \mathtt{D}\langle \overline{q} \rangle\ \{\ \overline{F}\ \overline{M}\ \}}$$

A disciplined (safe) flattening

An Aggregate is a "Hidden Subset" of Owner



Key Typing Issues 1/2

(EXPR-SELECT)

(a)

$$\frac{\begin{array}{c} E \vdash x : \mathsf{C}\langle \sigma^p \rangle \\ \mathsf{FieldType}(\mathsf{C}, f) = \tau \\ \mathbf{rep} \in \mathsf{Owners}(\tau) \Rightarrow x = \mathbf{this} \\ \mathbf{aggregate} \in \mathsf{Owners}(\tau) \Rightarrow p = \mathbf{bridge} \\ \mathsf{OmbudsmanAdaptation}(p, \tau) = \tau' \end{array}}{E \vdash x.f : \sigma^p(\tau')}$$

Only access **aggregate** objects from **bridge** objects

Lose "bridge status" if x is not a **bridge**

(EXPR-UPDATE)

(b)

$$\frac{\begin{array}{c} E \vdash x : \mathsf{C}\langle \sigma^p \rangle \\ \mathsf{FieldType}(\mathsf{C}, f) = \tau \\ E \vdash y : \sigma^p(\tau) \\ \mathbf{rep} \in \mathsf{Owners}(\tau) \Rightarrow x = \mathbf{this} \\ \mathbf{bridge}, \mathbf{aggregate} \in \mathsf{Owners}(\tau) \Rightarrow p = \mathbf{bridge} \end{array}}{E \vdash x.f = y : \sigma^p(\tau)}$$

Only assign **aggregate**/**bridge** objects from **bridge** objects

(EXPR-METHOD-CALL)

(a+b)

$$\frac{\begin{array}{c} E \vdash x : \mathsf{C}\langle \sigma^p \rangle \\ \mathsf{Signature}(\mathsf{C}, m) = \tau_1 \to \tau_2 \\ E \vdash y : \sigma^p(\tau_1) \\ \mathbf{rep} \in \mathsf{Owners}(\tau_1) \cup \mathsf{Owners}(\tau_2) \Rightarrow x = \mathbf{this} \\ \mathbf{bridge}, \mathbf{aggregate} \in \mathsf{Owners}(\tau_1) \Rightarrow p = \mathbf{bridge} \\ \mathbf{aggregate} \in \mathsf{Owners}(\tau_2) \Rightarrow p = \mathbf{bridge} \\ \mathsf{OmbudsmanAdaptation}(p, \tau_2) = \tau \end{array}}{E \vdash x.m(y) : \sigma^p(\tau)}$$

Lose "bridge status" if x is not a **bridge**

$$\begin{array}{rcl} \mathsf{OmbudsmanAdaptation}(\mathbf{bridge}, \tau) & = & \tau \\ \mathsf{OmbudsmanAdaptation}(p, \tau) & = & \tau\{^{\mathbf{owner}}/_{\mathbf{bridge}}\} \end{array}$$

Key Typing Issues 2/2

$$\frac{E \vdash p \prec^* q}{E \vdash p \rightarrow^{\mathsf{ok}} q} \;\text{(P-INSIDE)}$$

Objects owned by *q* are accessible to objects owned by *p*

$$\frac{\vdash E \qquad p \in \{\mathbf{bridge}, \mathbf{aggregate}\}}{E \vdash \mathbf{rep} \rightarrow^{\mathsf{ok}} p} \;\text{(P-REP)}$$

Bridge objects and aggregate objects are accessible by representation objects

$$\frac{E \vdash p \qquad E \vdash p \rightarrow^{\mathsf{ok}} \overline{p} \qquad \mathsf{Arity}(\mathsf{c}) = |p, \overline{p}|}{E \vdash \mathsf{c}\langle p, \overline{p} \rangle} \;\text{(GOOD-TYPE)}$$

Essentially the standard (GOOD-TYPE) rule of deep ownership types, extended with support for **bridge** and **aggregate**

Comparison with Previous Work

* = enforced at run-time

	Local reasoning power	Modularity	Flexibility	Co-ownership	Simplicity
Boyapati's inner classes	none	no	bad	no equal rights	yes
Lu et al.'s downgrading	none	yes	high	none	yes
Ownership Domains	~~none~~	yes	high	none	yes
CoBoxes	strong*	yes	high	yes	yes
Mojo & Mojojojo	strong	ham-pered	high	yes	no
This work	strong	yes	discip-lined	yes	yes

Notes and Future Work

Co-ownership is encoded as a flattening which is only visible to the collaborating bridge objects (not to the outside)

Aggregates can be built up in two ways:

- From within (similar to Boyapati's proposal)
- Through attachment (requires unique references)

Abstraction: It is not possible to tell whether two siblings belong to the same aggregate or not (once they lose bridge status they're owned by **owner**)

Current limitation #1: an object can only participate in one co-ownership

Current limitation #2: classes need to explicitly use **aggregate** and **bridge**

- *More details and examples in the paper!*

Thank you. Questions?