The Law of Demeter

For Operating System Course

Motivation

- Several programs were written in bad style
- Are difficult to maintain
- There is a simple remedy

The Law of Demeter (LoD) and how it should be used

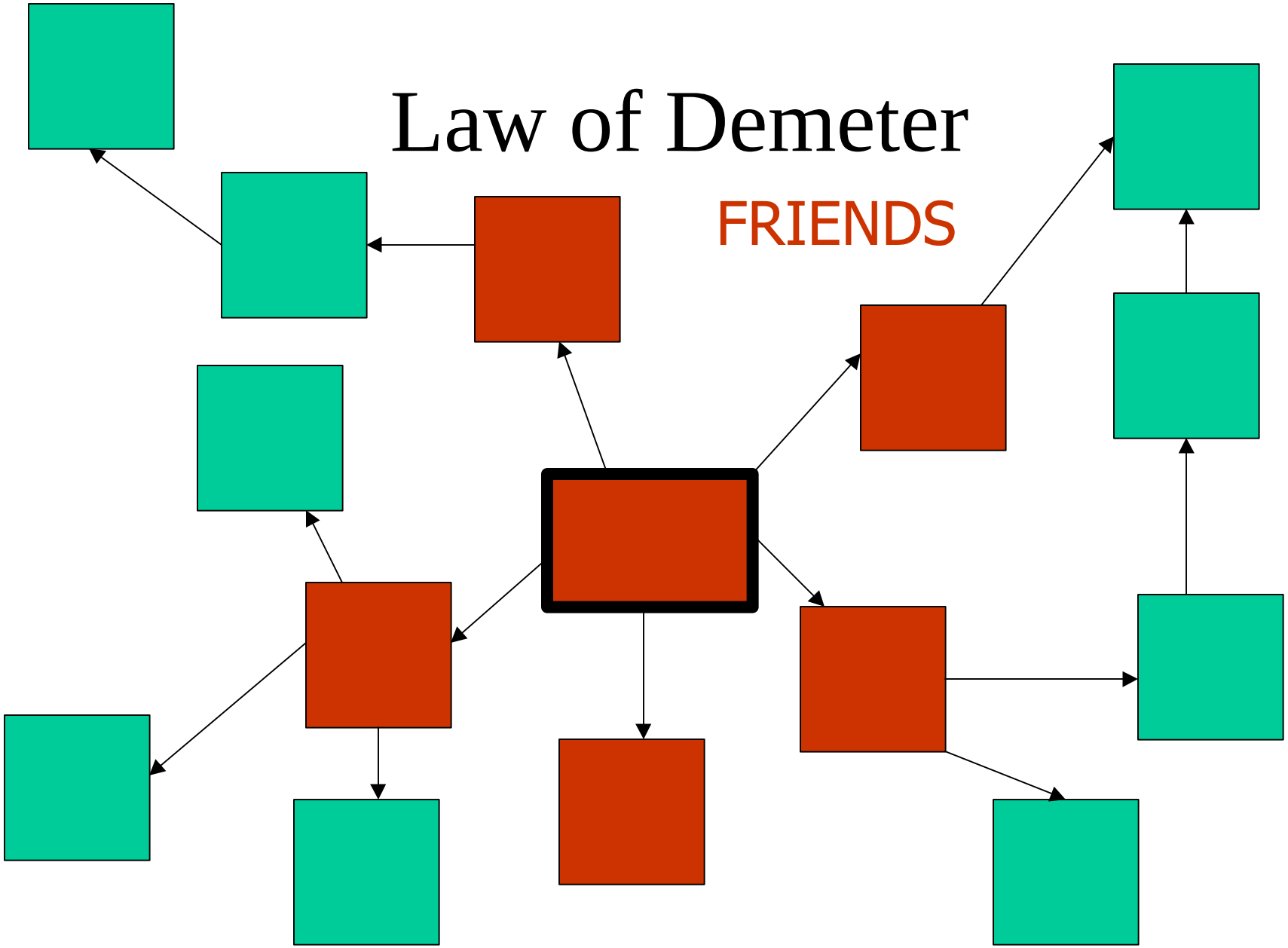
Law of Demeter

- What is it: Style Rule for building object-oriented systems.
- Proposed by my research group: The Demeter Research Group in 1987, published in 1988.
- Covered in many major books on OO design and programming.

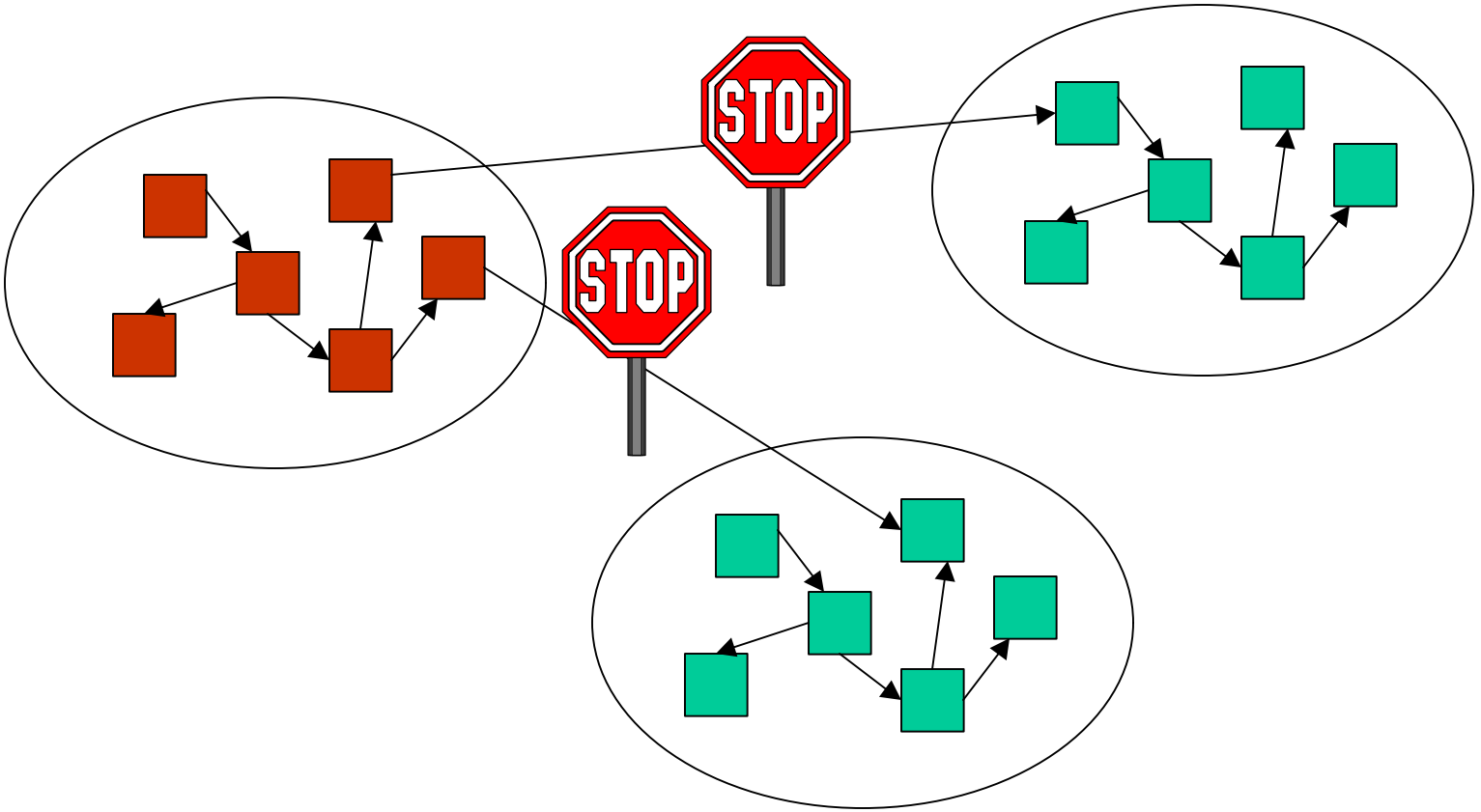
Law of Demeter Principle

- **Each unit should only use a limited set of other units: only units "closely" related to the current unit.**
- "Each unit should only talk to its friends." "Don't talk to strangers."
- Main Motivation: Control information overload. We can only keep a limited set of items in short-term memory.

Law of Demeter



“closely related”



Application to OO

- Unit = method
  - closely related =
    - methods of class of `this/self` and other argument classes
    - methods of immediate part classes (classes that are return types of methods of class of `this/self`)
- In the following we talk about this application of the Law of Demeter Principle to OO: example follows in a few slides.

Citibank Quote: Law of Demeter

- The Law of Demeter forms one of the **cornerstones** of the **design approach** of the **Global Finance Application Architecture** (quote from: Global Finance Application Architecture: Business Elements Analysis, Oct. 1991, Citibank confidential document)
- Widely used in big projects, for example, at JPL for the Mars exploration software.

Jet Propulsion Laboratory(JPL) Quote: Law of Demeter

- The Law of Demeter … has taken a firm hold in many areas of JPL. Major systems which have used LoD extensively include … Mars Pathfinder Software (begun in 1993). We are going to use LoD as a foundational software engineering principle for the X2000 Europa orbiter mission.

What others say about the Law of Demeter

- To get a better understanding
  - Booch
  - Rumbaugh

Booch and the Law of Demeter

Context

Chapter: Classes and Objects, Section: On Building Quality Classes and Objects, Subsection: Choosing Relationships

Booch and the Law of Demeter

Quote: The basic effect of applying this Law is the creation of **loosely coupled classes**, whose implementation secrets are encapsulated. Such classes are fairly unencumbered, meaning that to understand the meaning of one class, you **need not understand the details of many other classes**.

Rumbaugh and the Law of Demeter

Context

Chapter: Programming Style, Section: Extensibility

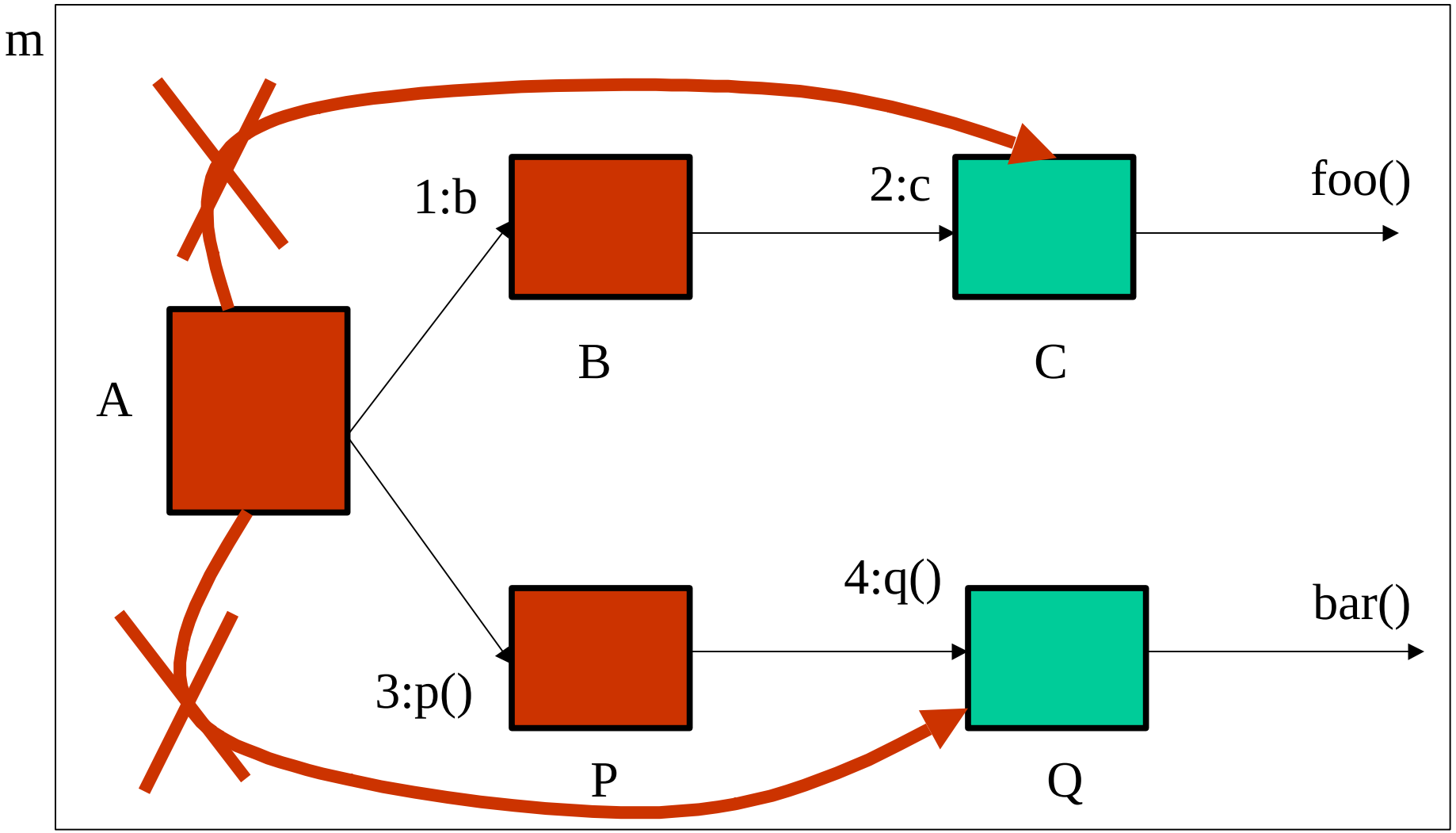
Rumbaugh and the Law of Demeter

Quote: Avoid traversing multiple links or methods. **A method should have limited knowledge of an object model.** A method must be able to traverse links to obtain its neighbors and must be able to call operations on them, but it should not traverse a second link from the neighbor to a third class.

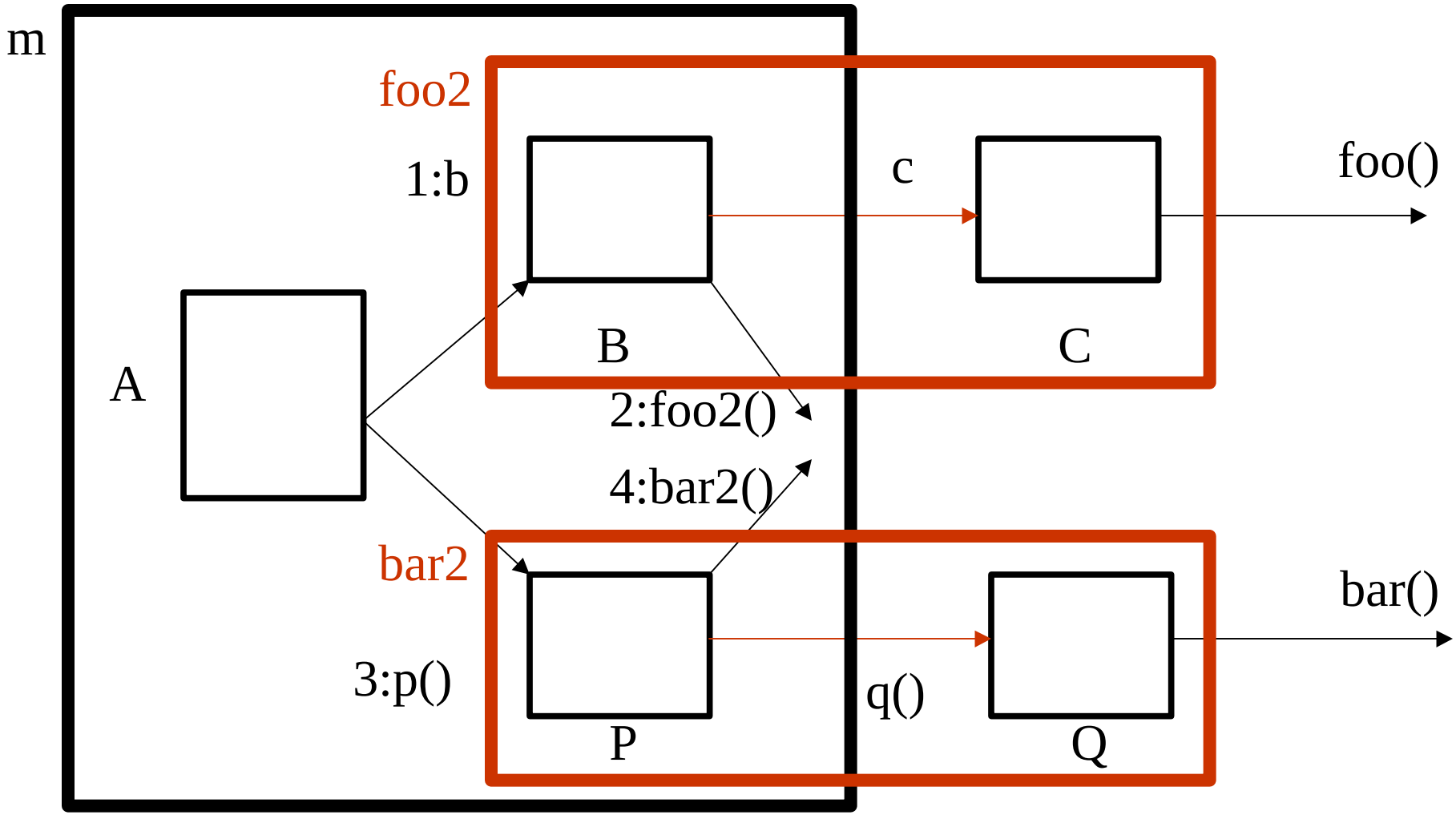
The Law of Demeter (cont.)
Violation of the Law

```
class A {public: void m(); P p(); B b; };
class B {public: C c; };
class C {public: void foo(); };
class P {public: Q q(); };
class Q {public: void bar(); };
void A::m() {
    this.b.c.foo(); this.p().q().bar();}
```

Violations: Dataflow Diagram



OO Following of LoD



Experience

- Following the Law of Demeter leads to code that is
  - easier to understand and maintain
  - structured in a more natural way (fewer dependencies between classes)

Recommendation

- Follow the Law of Demeter in all object-oriented programs you write, specifically in the ones you write in this class.
- Operating systems are complex and it is important that we keep them well structured.