Separation of Concerns for Dependable Software Design

Daniel Jackson and Eunsuk Kang
MIT

Nov 7 · FoSER Workshop 2010

Achieving Dependability

Traditional approach

- process + testing: necessary, but not sufficient*
- reliance on *ex post facto* analysis: too late?

Static analysis & verification

- stronger guarantees, but need guidance
- correctness proof ⇏ dependability

*D. Jackson, M. Thomas, and L. I. Millet. *Software for Dependable Systems: Sufficient Evidence?* The National Academies Press, Washington, DC, 2007.

A Different Approach

Dependability case

- explicit, end-to-end argument
- ENV $\wedge$ SPEC $\Rightarrow$ REQ

Design for dependability

- most critical requirements first
- smaller trusted base $\Rightarrow$ simpler case, lower cost

Mixed-Criticality System

Many critical properties are *partial*

- factor out from full functional requirements
- ex. "perform good op X" vs. "prevent bad op Y"

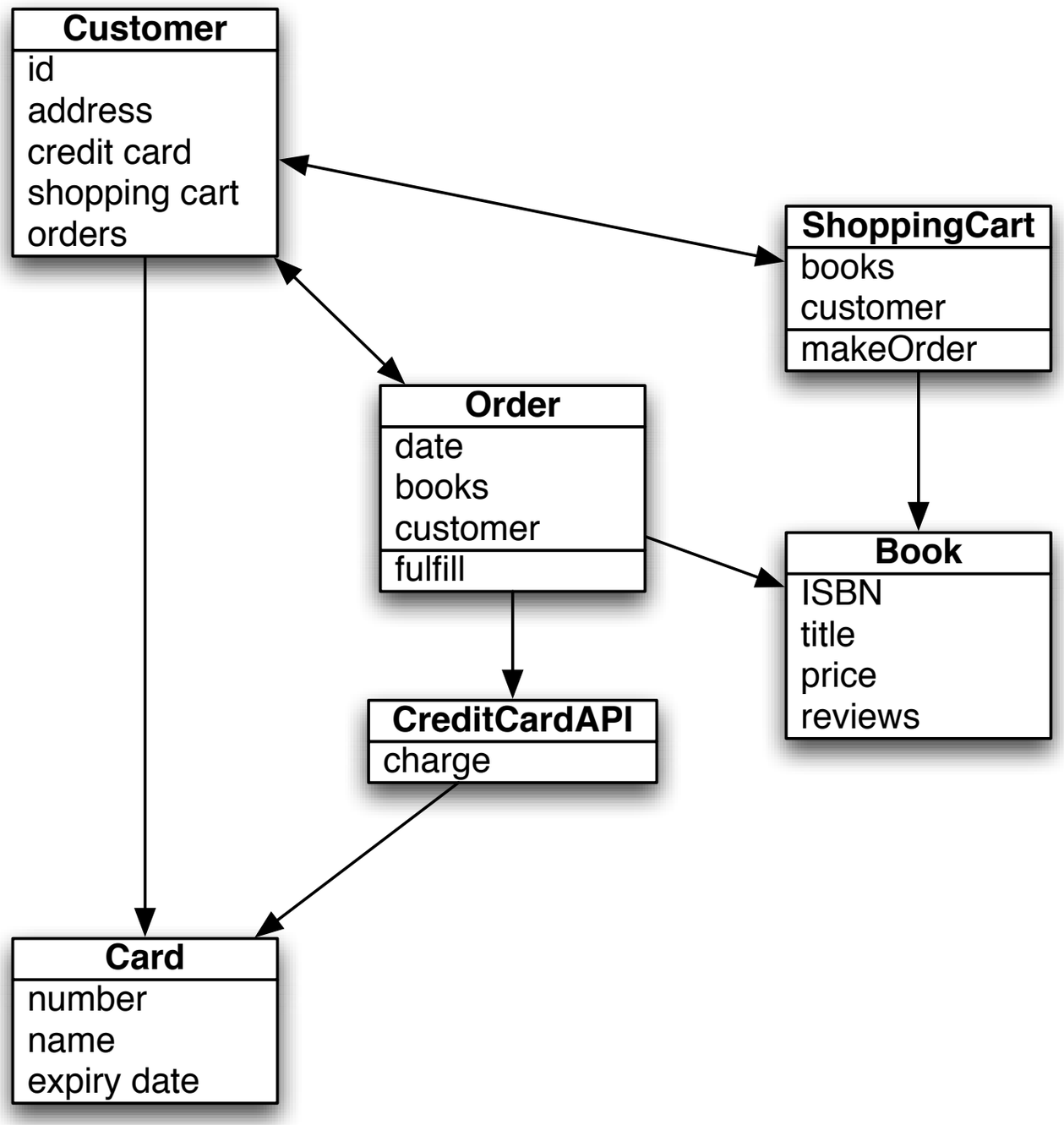
Non-uniform allocation of resources

Example: Online Bookstore

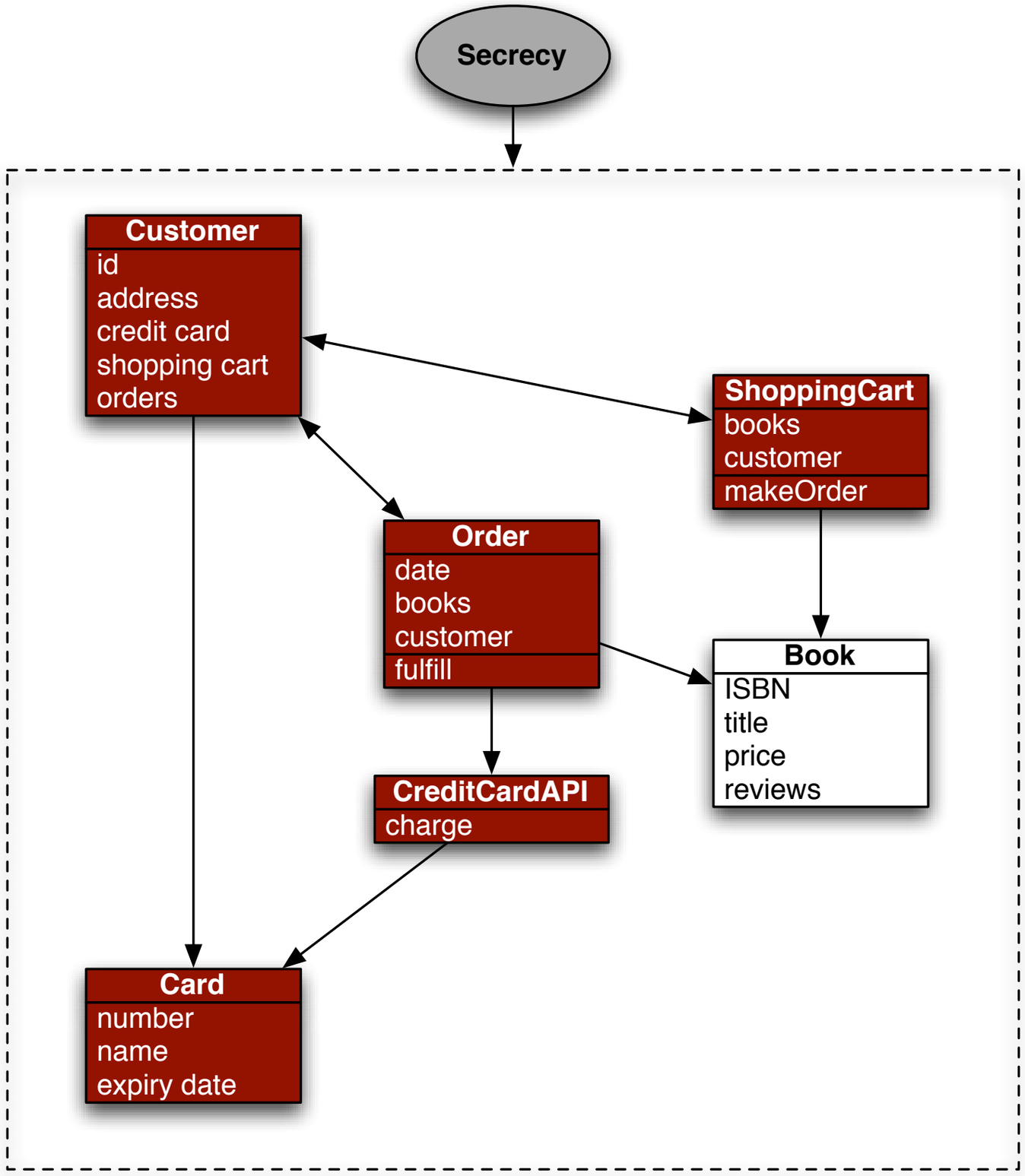
Two requirements

- ordering: "fulfill a customer order"
- secrecy: "don't leak a customer's credit card"

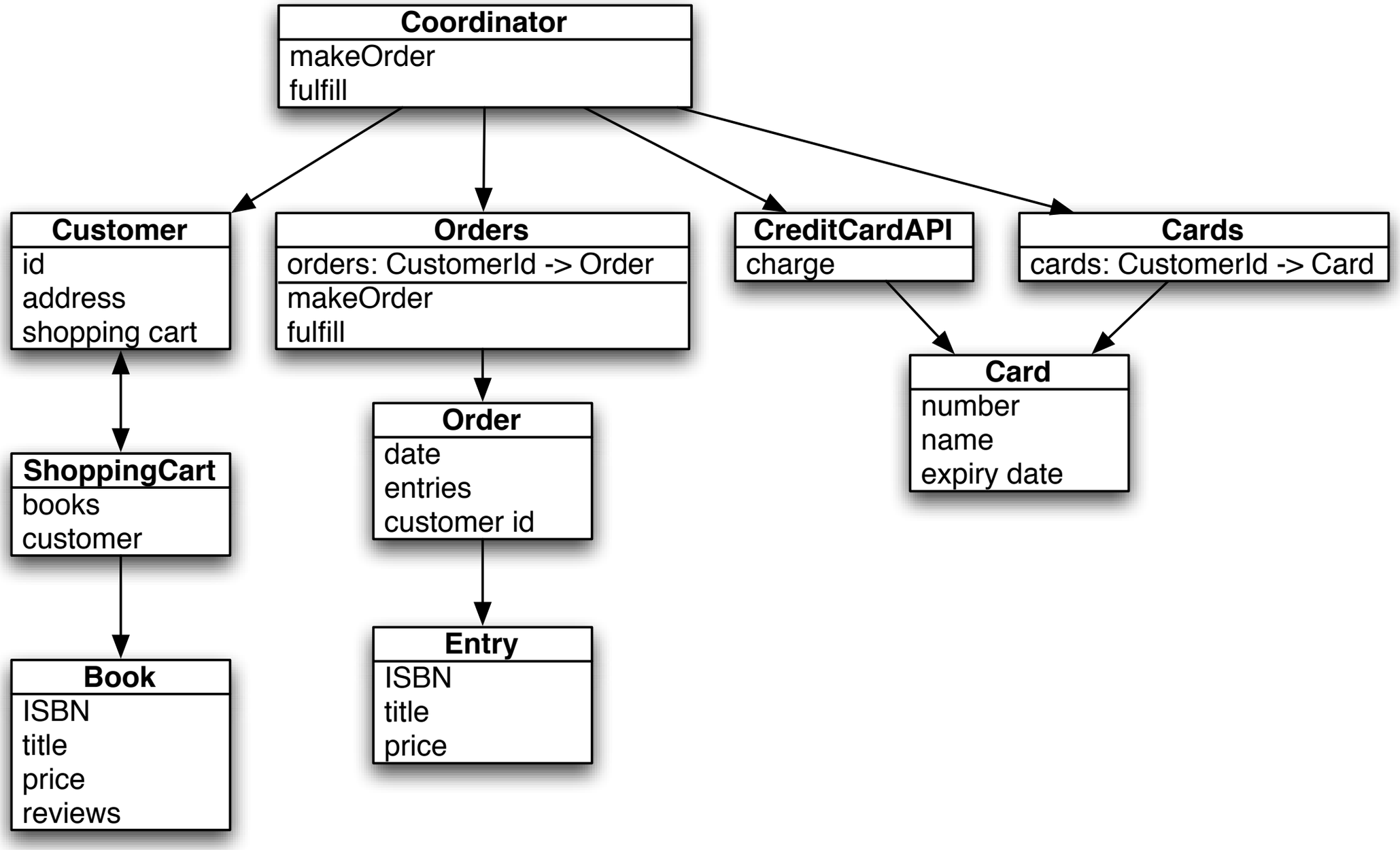
Design Candidate



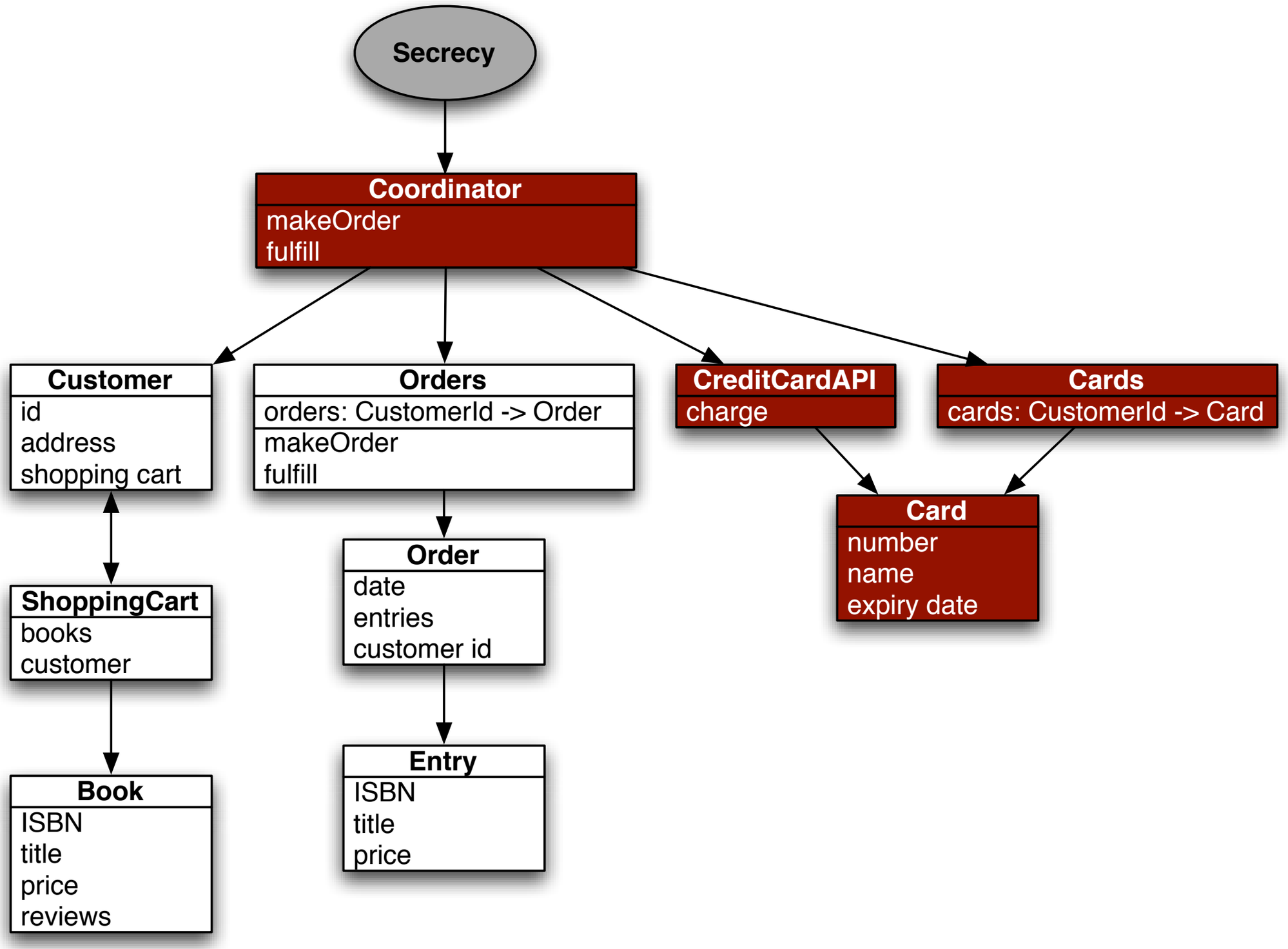
Trusted Base for Secrecy



Alternative Design



Reduced Trusted Base



Discussions

Dependability case

- if you can't say why it works, it probably doesn't

Design for dependability

- untapped potential; shift in research focus?

Our on-going research

- design method for small trusted bases
- case studies: Tokeneer, radiation therapy, e-voting