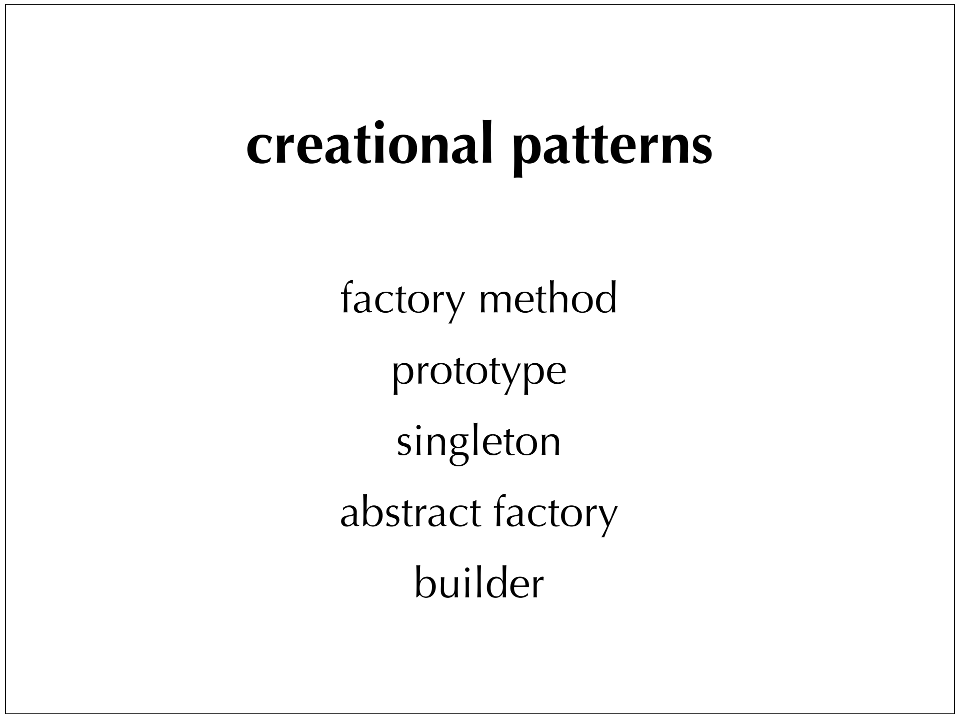
Based on your reading for today...

1. **What constraints affect Lexi's implementation of user operations?**
2. **What design pattern does Lexi use to implement user operations?**
3. **List three other design patterns used in Lexi.**



creational patterns

factory method

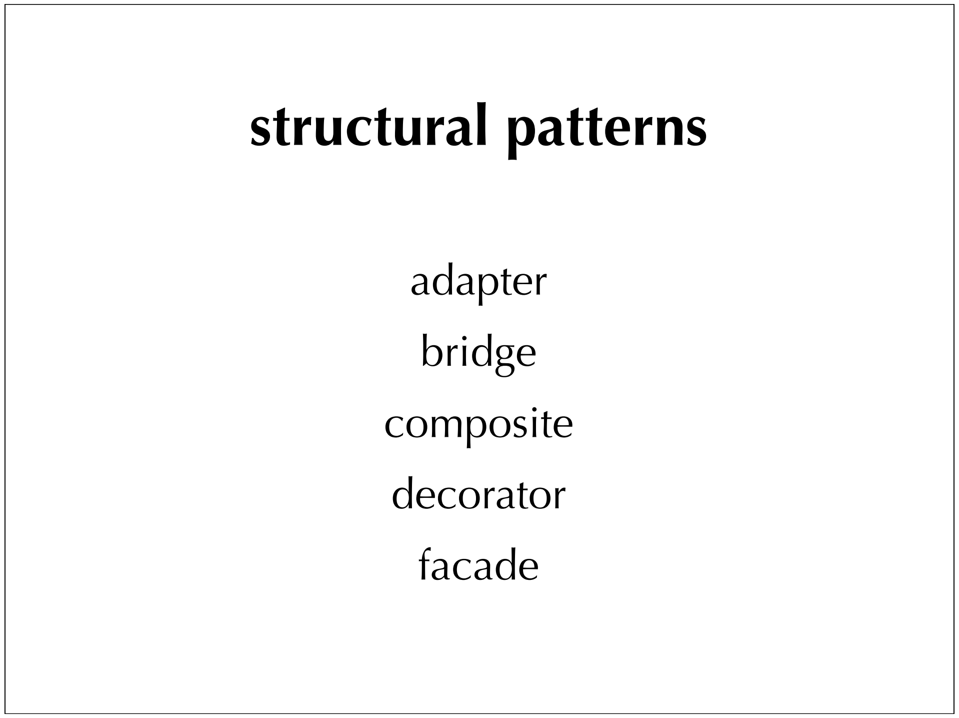
prototype

singleton

abstract factory

builder

You see factory method in 810:053.



structural patterns

adapter

bridge

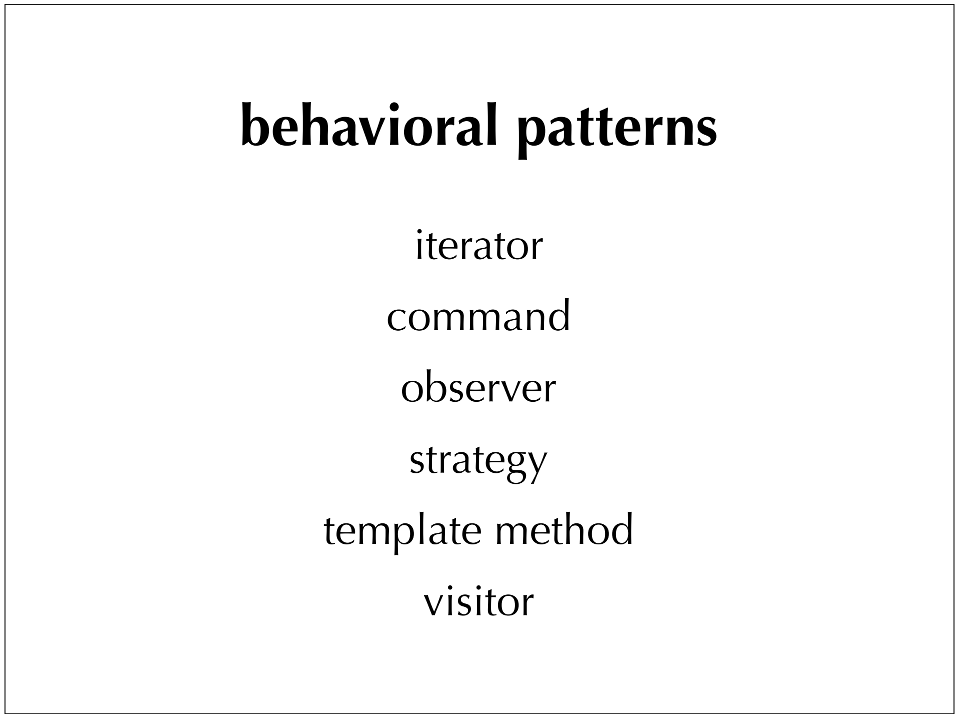
composite

decorator

facade

This is a subset — 5 of 7.

You see some of these in 810:053 (adapter, composite, decorator).



behavioral patterns

iterator

command

observer

strategy

template method

visitor

This is a subset — 6 of 11.

You see some of these in 810:053 (iterator, strategy, template method).

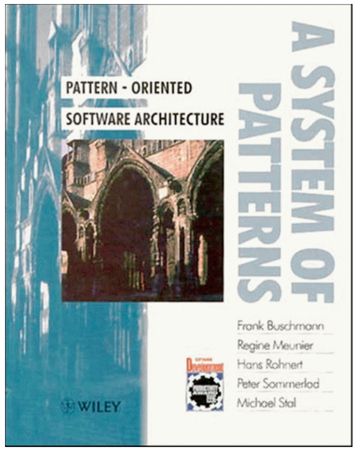
Consider the idea of a pattern as a way to make up for a language’s deficiencies...

Reactor embodies the idea of MVC

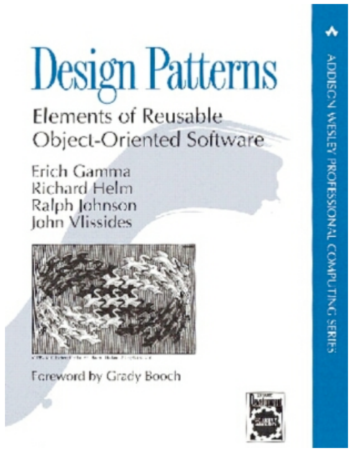
```
public class Reactor
{

   // THIS PART ACTS LIKE A MODEL

   public void addDependent( Reactor watcher )
   {
      if (numberOfDependents < MaximumNumberOfDependents)
      {
         myDependents.addElement( watcher );
         numberOfDependents++;
      }
      else
         System.out.println( "I already have too many dependents!" );
   }

   public void change()
   {
      System.out.println( myName + " is changing." );
      for (int i = 0; i < numberOfDependents; i++)
      {
          Reactor aDependent = (Reactor) myDependents.elementAt( i );
          aDependent.update( this );
      }
   }


   // THIS PART ACTS LIKE A VIEW

   public void update ( Reactor sender )
   {
      System.out.println( "Uh-oh...  "  + myName + " has to change because " +
                          sender.name() + " has changed." );
      change();
   }
}
```

<discuss Reactor>

<demo MVC>



Model-View-Controller

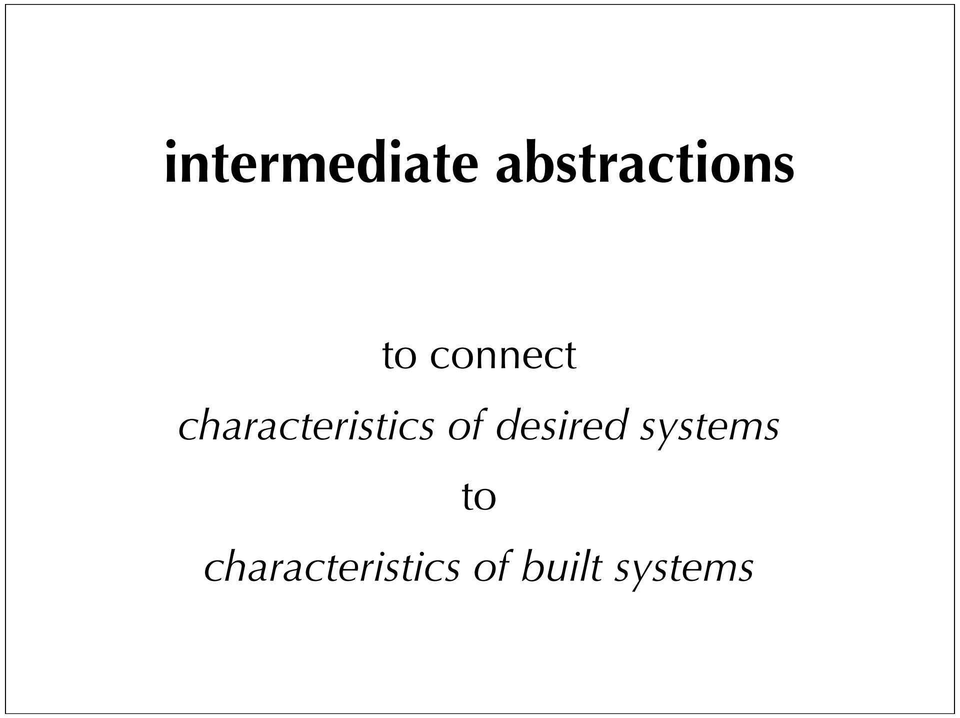

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON WESLEY PROFESSIONAL COMPUTING SERIES

Observer

Toward handbooks for software engineers.

Design Patterns, the “Gang of Four book” or GoF, is the **de facto** standard for OO design.

Pattern-Oriented Software Architecture, the PoSA book, is a part of a series aimed at large distributed systems.



intermediate abstractions

to connect

characteristics of desired systems

to

characteristics of built systems

We need a way to **reliably** and **repeatably** map problem types onto solution types.

Recur “a million times, but never the same way”.

All models are wrong;
some are useful.

— George Box