Design Patterns

MSc in Communications Software

Produced by

Eamonn de Leastar (edeleastar@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

http://www.wit.ie

http://elearning.wit.ie

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit

Design Patterns Background

The Origins of Design Patterns

On Vocabulary

Janousek single scull

55-60kg

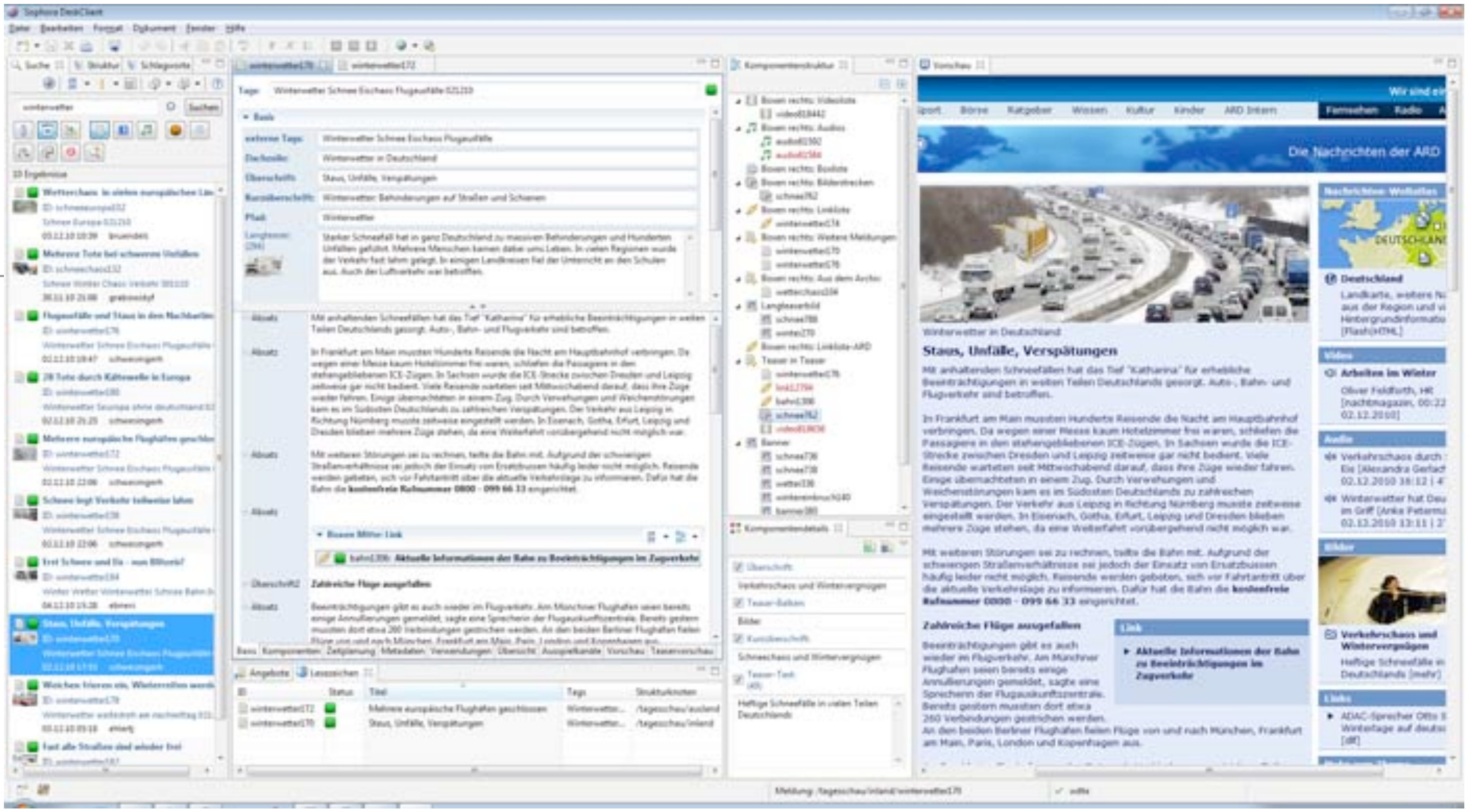
Croker blades

Carbon Honeycomb

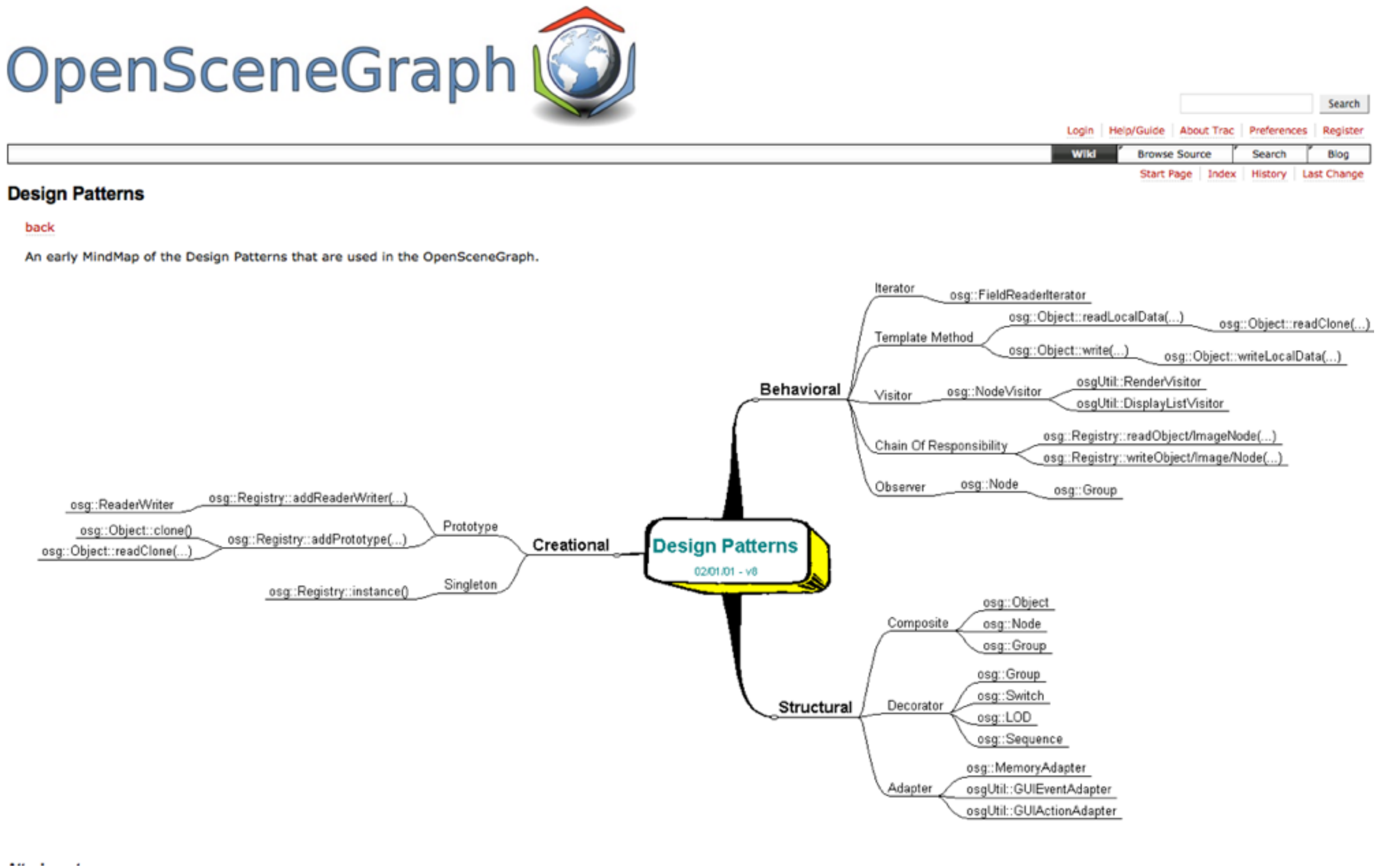
J13 Heavyweight Mould

Rowfit wing rigger

Patterns Vocabulary



- Hierarchical MVC UI Design
- Command pattern supporting multi-level undo/redo, incorporating Prototype based command creation.
- Composite contact list organisation
- Visitor contact search facility
- Strategy report generators


OpenSceneGraph
Search
Login | Help/Guide | About Trac | Preferences | Register
Wiki | Browse Source | Search | Blog
Start Page | Index | History | Last Change

Design Patterns

back

An early MindMap of the Design Patterns that are used in the OpenSceneGraph.

Design Patterns
02/01/01 - v8
Behavioral
Iterator
osg::FieldReaderIterator
Template Method
osg::Object::readLocalData(...)
osg::Object::readClone(...)
osg::Object::write(...)
osg::Object::writeLocalData(...)
Visitor
osg::NodeVisitor
osgUtil::RenderVisitor
osgUtil::DisplayListVisitor
Chain Of Responsibility
osg::Registry::readObject/ImageNode(...)
osg::Registry::writeObject/Image/Node(...)
Observer
osg::Node
osg::Group
Creational
Prototype
osg::Registry::addReaderWriter(...)
osg::ReaderWriter
osg::Registry::addPrototype(...)
osg::Object::clone()
osg::Object::readClone(...)
Singleton
osg::Registry::instance()
Structural
Composite
osg::Object
osg::Node
osg::Group
Decorator
osg::Group
osg::Switch
osg::LOD
osg::Sequence
Adapter
osg::MemoryAdapter
osgUtil::GUIEventAdapter
osgUtil::GUIActionAdapter

Purpose of Design Pattern

1.Reuse

- Reuse elegant, proven and high quality designs across multiple contexts

2.Flexibility

- Introduce greater flexibility into the design

3.Documentation

- Improving the documentation and maintenance of existing system by furnishing an explicit specification of class and object interactions and their intent

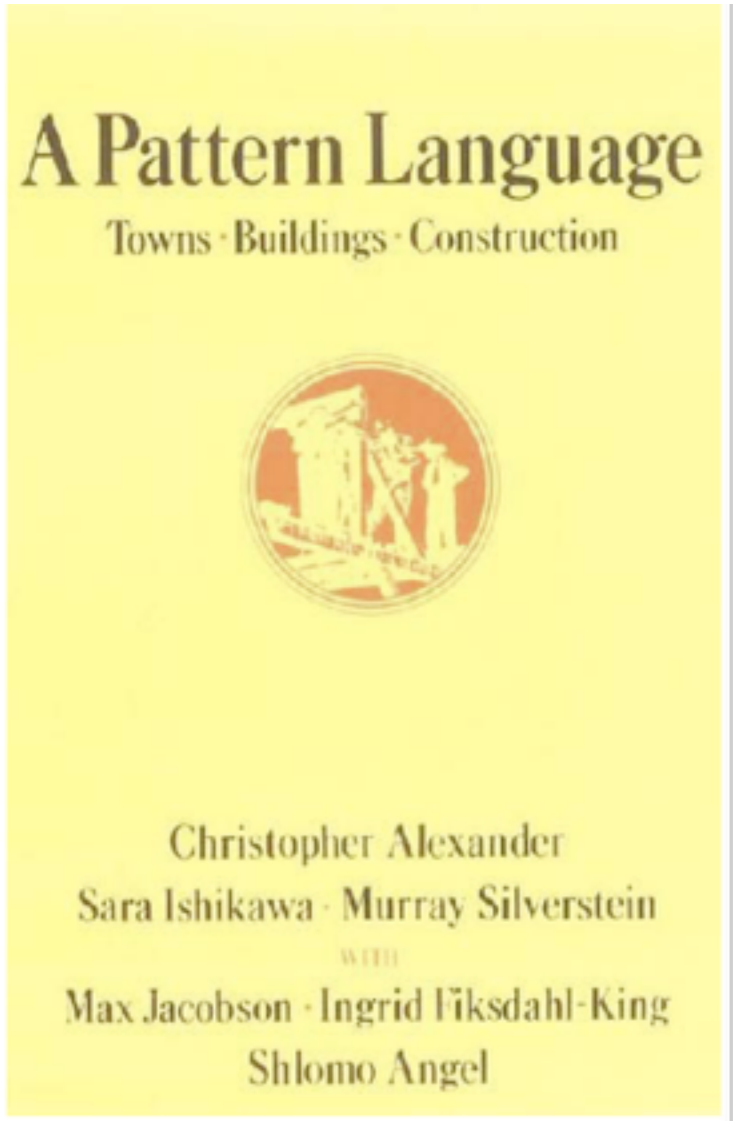
Origins

- "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice."

  Christopher Alexander, A Pattern Language: Towns/Buildings/Construction, 1977

- A object-oriented design pattern systematically names, explains and evaluates an important and recurring design in object-oriented systems

Origins



A Pattern Language

Towns · Buildings · Construction

Christopher Alexander

Sara Ishikawa · Murray Silverstein

WITH

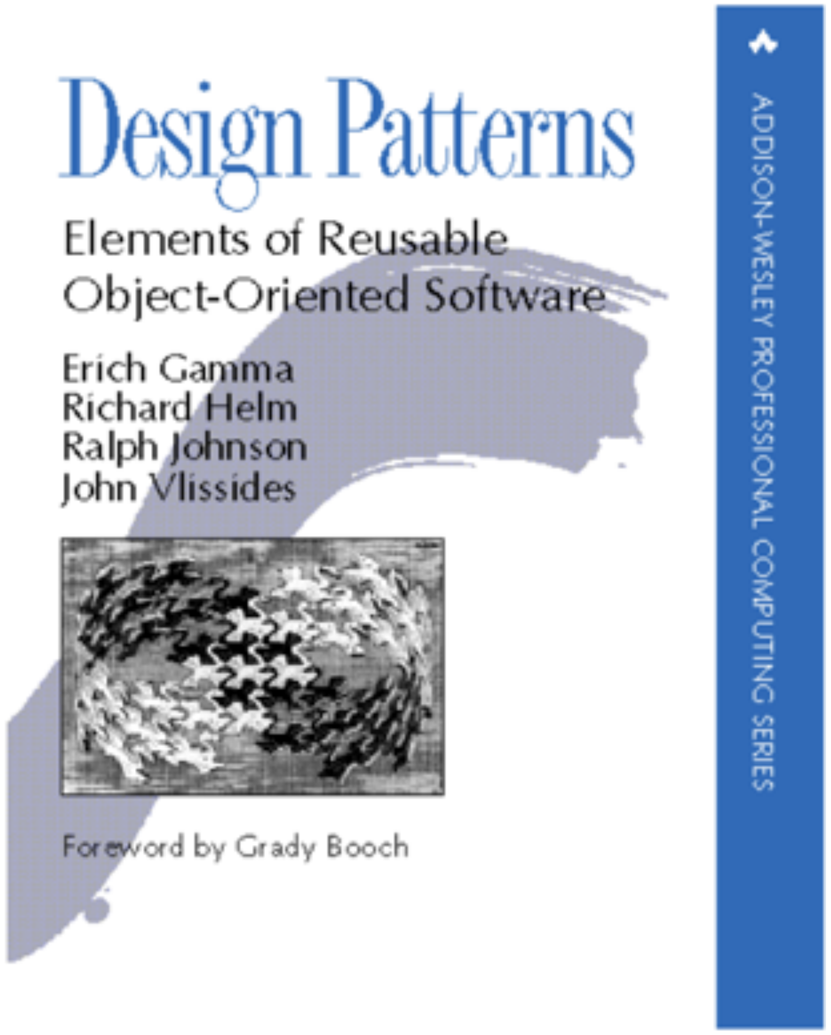
Max Jacobson · Ingrid Fiksdahl-King

Shlomo Angel

Alexander & Patterns

- Alexander studied the problem of objective quality by making observations of buildings, towns, streets, gardens, any spaces that human beings have built
  - He reasoned that high quality constructs had things in common
  - Architectural structures differed from each other even if they were of the same type solving the same problem. Yet different solutions were of high quality.
  - He understood that structures could not be separated from the problem they are solving
- He proposed that different structures yielded a high quality solution to similar problems and extracted the similarity of the structures, the core of the solution, which he calls a pattern:
  - solutions to a problem in a context
  - 253 patterns covering regions, towns, transportations, homes offices, rooms, lighthing, gardens, ...
  - each pattern defines subproblems solved by other smaller patterns

Patterns in Software Design



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

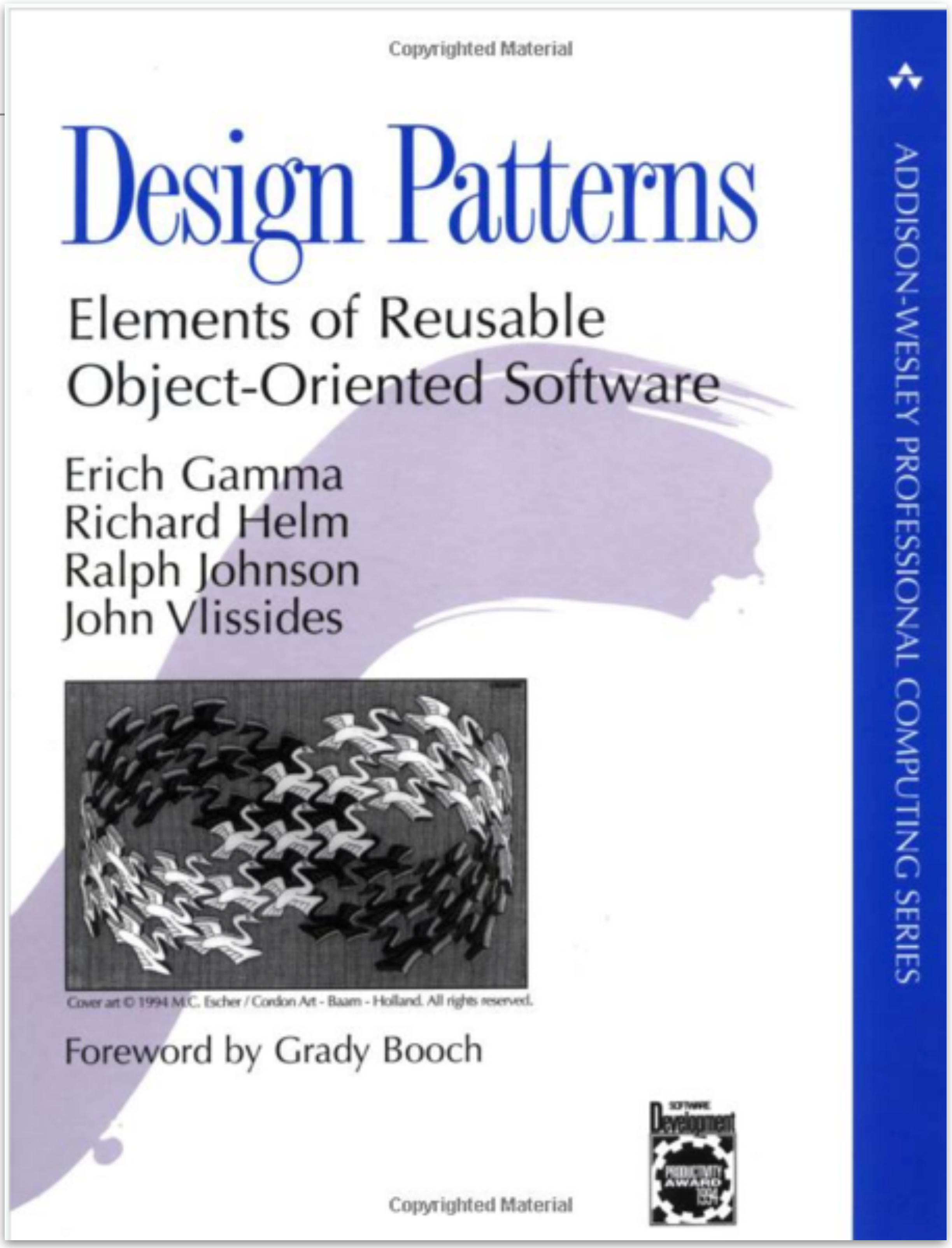
GoF

- The landmark book on software design patterns is:

  Design Patterns: Elements of Reusable Object-Oriented Software
  Erich Gamma, Richard Helm,
  Ralph Johnson, John Vlissides
  Addison-Wesley, 1995

- This is also known as the GOF ("Gang-of-Four") book.

- Design Patterns help you break out of first-generation (naive) OO thought patterns



Design Patterns

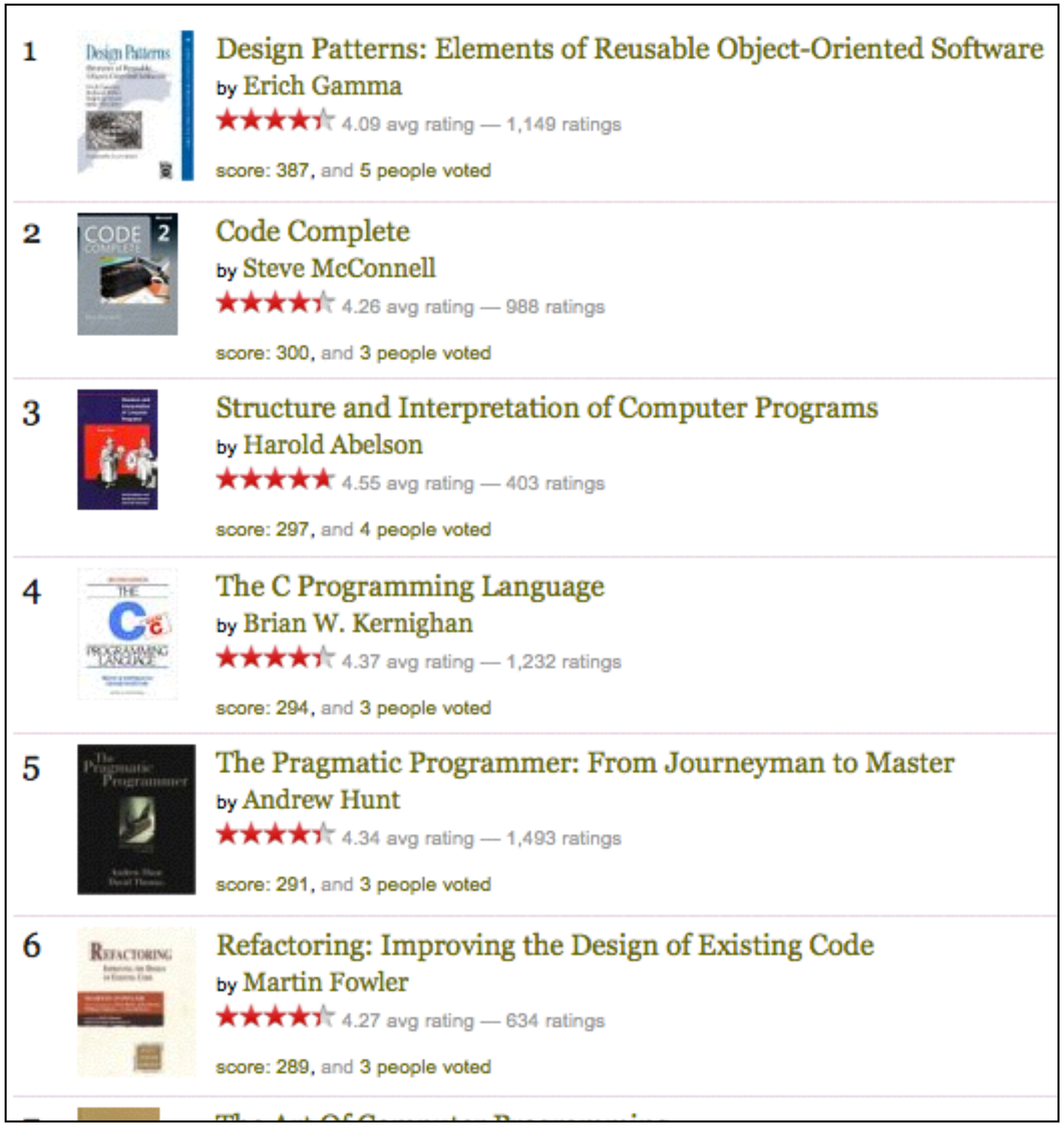
Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides

Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

Regularly Appears in top-10 programming lists



1 Design Patterns: Elements of Reusable Object-Oriented Software
by Erich Gamma
4.09 avg rating — 1,149 ratings
score: 387, and 5 people voted

2 Code Complete
by Steve McConnell
4.26 avg rating — 988 ratings
score: 300, and 3 people voted

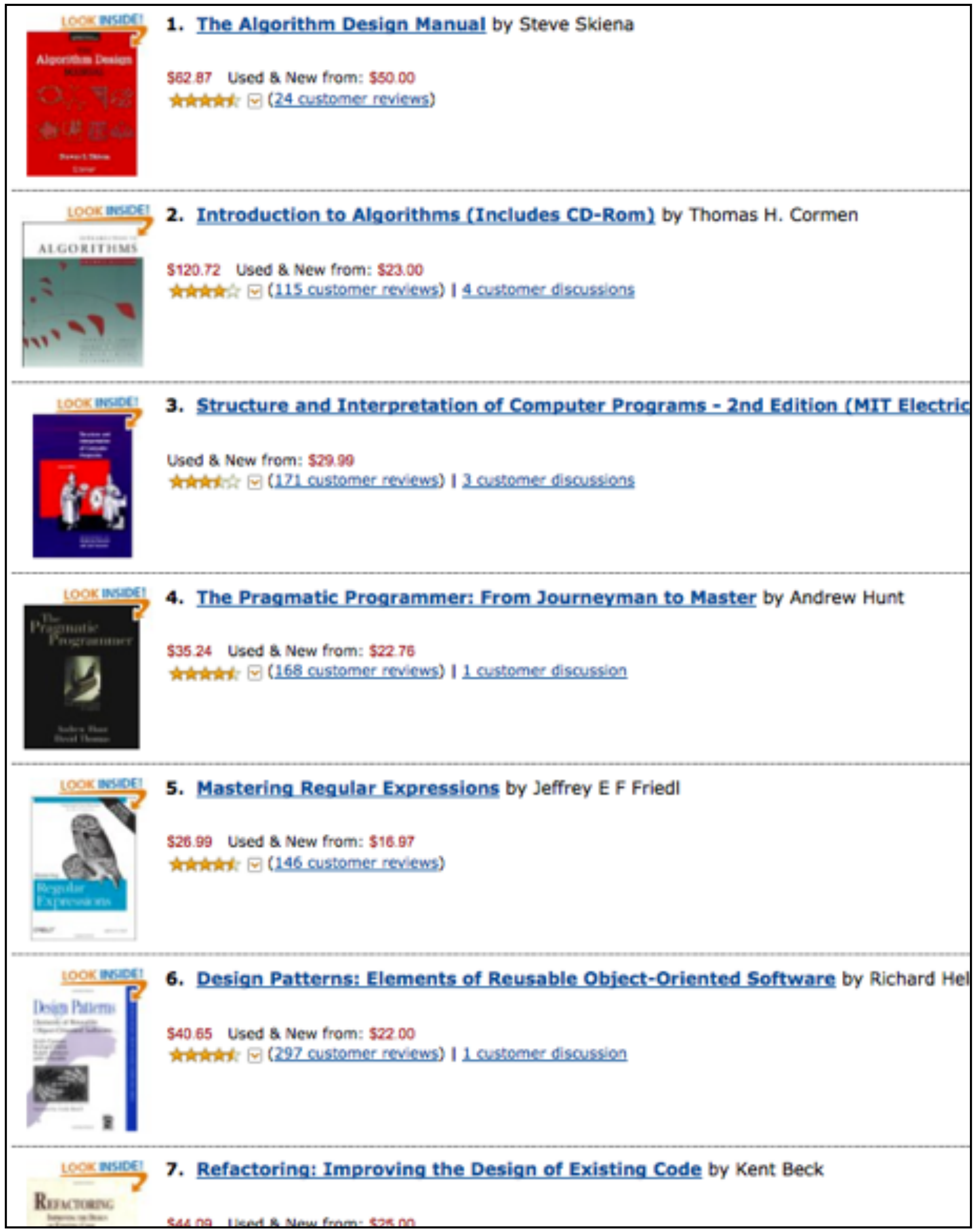
3 Structure and Interpretation of Computer Programs
by Harold Abelson
4.55 avg rating — 403 ratings
score: 297, and 4 people voted

4 The C Programming Language
by Brian W. Kernighan
4.37 avg rating — 1,232 ratings
score: 294, and 3 people voted

5 The Pragmatic Programmer: From Journeyman to Master
by Andrew Hunt
4.34 avg rating — 1,493 ratings
score: 291, and 3 people voted

6 Refactoring: Improving the Design of Existing Code
by Martin Fowler
4.27 avg rating — 634 ratings
score: 289, and 3 people voted

7 The Art Of Computer Programming



1. The Algorithm Design Manual by Steve Skiena
$62.87 Used & New from: $50.00
(24 customer reviews)

2. Introduction to Algorithms (Includes CD-Rom) by Thomas H. Cormen
$120.72 Used & New from: $23.00
(115 customer reviews) | 4 customer discussions

3. Structure and Interpretation of Computer Programs - 2nd Edition (MIT Electric
Used & New from: $29.99
(171 customer reviews) | 3 customer discussions

4. The Pragmatic Programmer: From Journeyman to Master by Andrew Hunt
$35.24 Used & New from: $22.76
(168 customer reviews) | 1 customer discussion

5. Mastering Regular Expressions by Jeffrey E F Friedl
$26.99 Used & New from: $16.97
(146 customer reviews)

6. Design Patterns: Elements of Reusable Object-Oriented Software by Richard Hel
$40.65 Used & New from: $22.00
(297 customer reviews) | 1 customer discussion

7. Refactoring: Improving the Design of Existing Code by Kent Beck
$44.09 Used & New from: $25.00

Pattern Definition

- Each design pattern systematically:
  - Names
  - Explains
  - Evaluates

  an important and recurring design in object-oriented systems.
- Patterns capture this design experience in a form that can be effectively communicated.
- Often presented as a catalogue.

To What End?

- Make it easier to reuse successful designs and architectures.
- Express proven techniques making them accessible to developers of new systems.
- Help developers choose appropriate design alternatives that make a system reusable and avoid options that compromise reusability.
- Improve the documentation and maintenance of existing systems by furnishing an explicit specification of class and object interactions and their underlying intent.

Design Pattern Characteristics

- Design patterns represent solutions to problems that arise when developing software within a particular context
  - Patterns = Problem/Solution pair in Context
- Capture static and dynamic structure and collaboration among key participants in software designs
  - key participants – an abstraction that occurs in a design problem
  - useful for articulating the how and why to solve non-functional forces.
- Facilitate reuse of successful software architectures and design

Documenting Patterns

- GoF used a standard procedure to describe and document design patterns.
    - Increases understandability.
    - Many books have adopted the similar approach.
- By documenting the design pattern, knowledge becomes explicit, instead of in the designer’s head.
    - Patterns are often presented as pattern catalogues
    - Important they are presented in a systematic form as a semi- formal document.

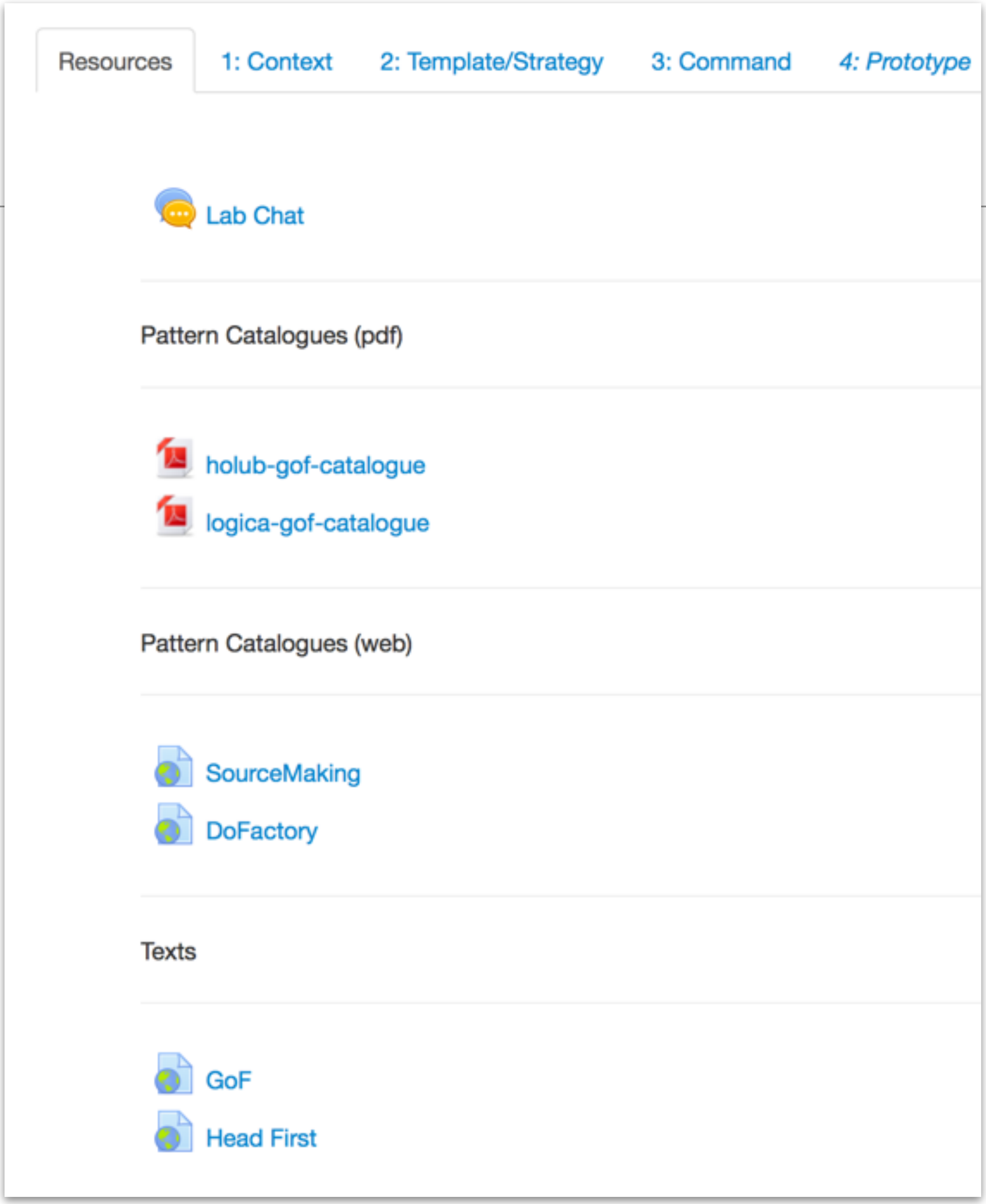
GoF Pattern Format

- **Pattern Name and Classification**
  - The pattern's name conveys the essence of the pattern succinctly. A good name is vital, because it will become part of your design vocabulary. The pattern's classification reflects a specific scheme (Creational, Behavioural, Structural).
- **Intent**
  - A short statement that answers the following questions: What does the design pattern do? What is its rationale and intent? What particular design issue or problem does it address?
- **Also Known As**
  - Other well-known names for the pattern, if any.
- **Motivation**
  - A scenario that illustrates a design problem and how the class and object structures in the pattern solve the problem. The scenario will help you understand the more abstract description of the pattern that follows.
- **Applicability**
  - What are the situations in which the design pattern can be applied? What are examples of poor designs that the pattern can address? How can you recognize these situations?
- **Structure**
  - A graphical representation of the classes in the pattern using UML. Usually class diagrams and interaction diagrams.

GoF Pattern Format

- **Participants**
  - The classes and/or objects participating in the design pattern and their responsibilities.
- **Collaborations**
  - How the participants collaborate to carry out their responsibilities.
- **Consequences**
  - How does the pattern support its objectives? What are the trade-offs and results of using the pattern? What aspect of system structure does it let you vary independently?
- **Implementation**
  - What pitfalls, hints, or techniques should you be aware of when implementing the pattern? Are there language-specific issues?
- **Sample Code**
  - Code fragments
- **Known Uses**
  - Examples of the pattern found in real systems. We include at least two examples from different domains.
- **Related Patterns**
  - What design patterns are closely related to this one? What are the important differences? With which other patterns should this one be used?

Catalogues


Resources | 1: Context | 2: Template/Strategy | 3: Command | *4: Prototype*

Lab Chat

Pattern Catalogues (pdf)

holub-gof-catalogue

logica-gof-catalogue

Pattern Catalogues (web)

SourceMaking

DoFactory

Texts

GoF

Head First

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

eLearning support unit