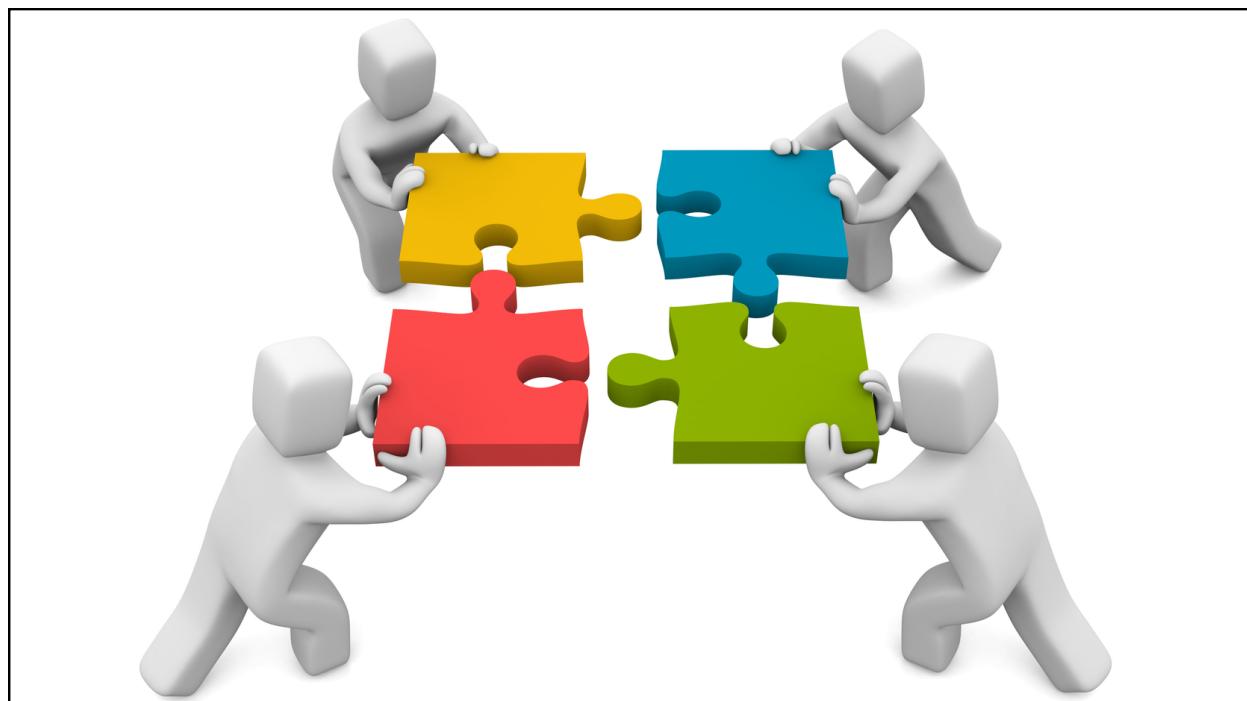


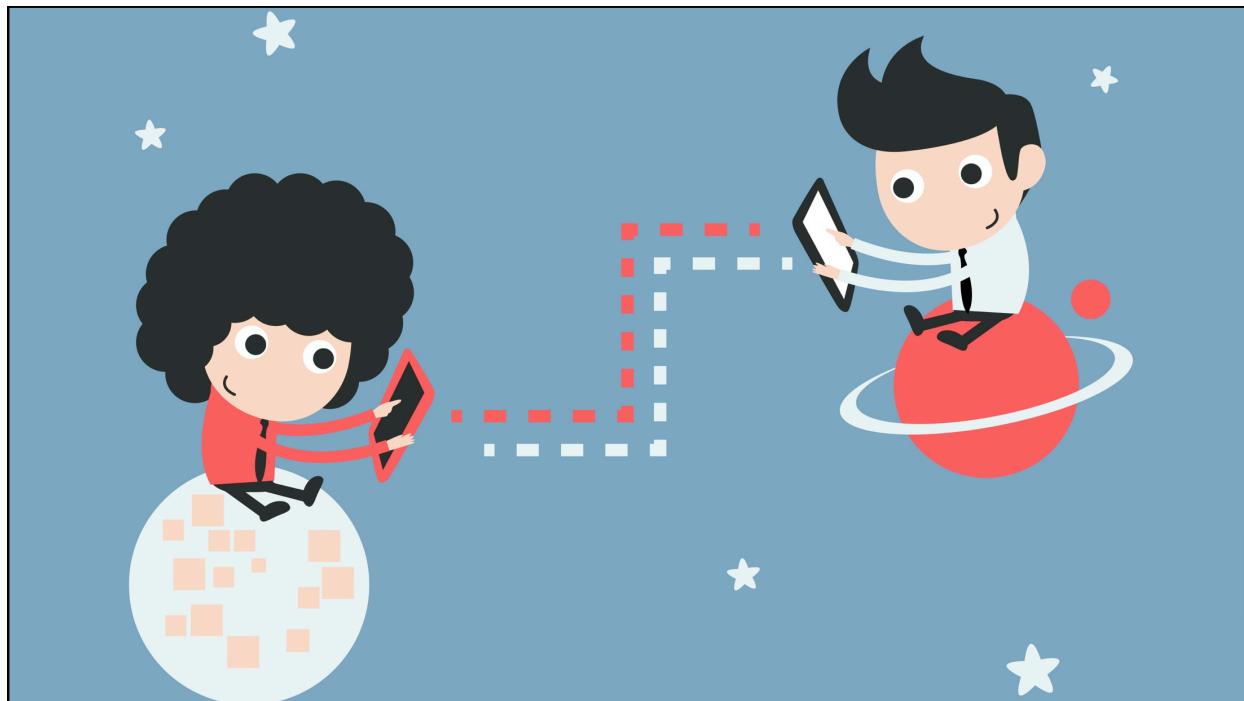


Behavioral Patterns

Part of the Design Patterns

www.elgoo.com







Content (1)
Behavioral Design patterns

PAGE 4

- 1 Command**
- 2 Strategy**
- 3 Visitor**
- 4 Observer**
- 5 Iterator**

www.elqoo.com



Content (2)

Behavioral Design patterns

PAGE 5

- 6 Memento
- 7 Mediator
- 8 Chain of responsibility
- 9 State
- X Template Method - Interpreter

www.elqoo.com



.....

Problem Statement


SD
Brad

- Hello Brad
- Hi Suzy
- The end users would like to have an undo button in the application. Could you take care of it?
- Sure, no problem


PM
Suzy

Elqoo

www.elqoo.com

PAGE 7

.....

Problem Statement Overview (1)

Analysis of the undo button



Assumption: Text Editor

User can:

- Copy/Past
- Format Text
- Type Text
- Open Documents

Elqoo

www.elqoo.com

PAGE 8

Problem Statement Overview (2)

Analysis of the undo button

PAGE 9



- Where will we problem the undo button?
- We need to know
 - Which actions were last executed
 - How to undo those actions
- Problem
 - Each action is fundamentally different

Elqoo

www.elqoo.com

Problem Statement Detail (1)

Motivation

PAGE 10

Generic GUI Toolkit

Pluggable Actions

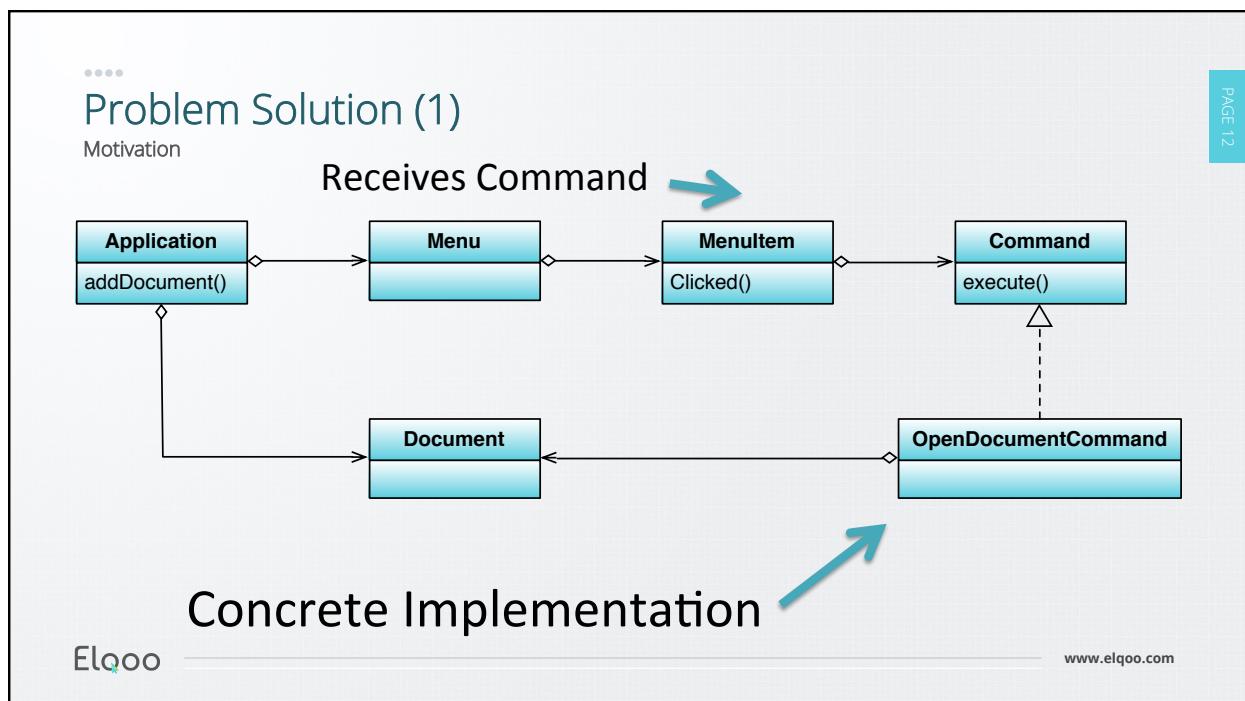
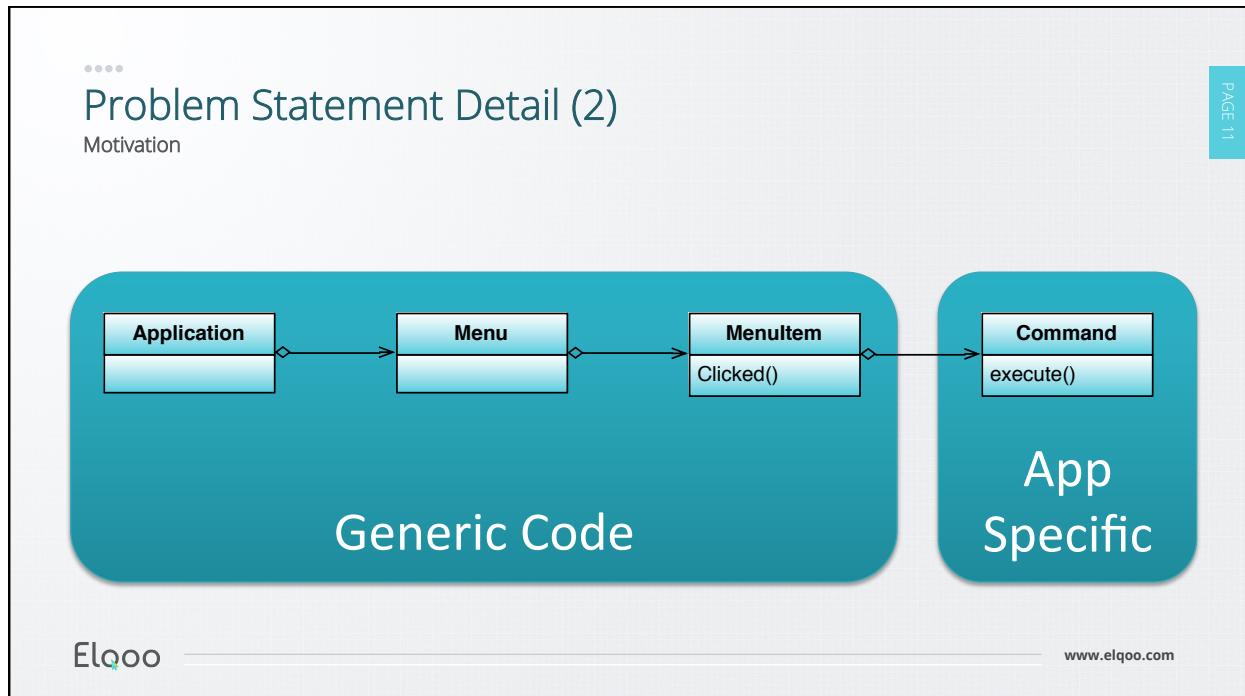
Open Document

Undo

Type Text

Elqoo

www.elqoo.com

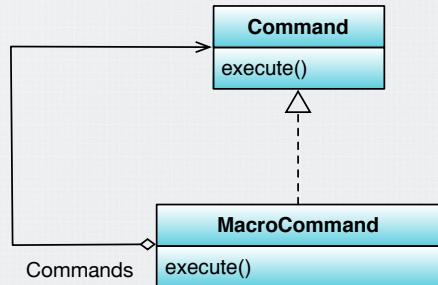


Problem Solution (2)

MacroCommand to execute multiple commands

PAGE 13

MacroCommand
`execute()` → Call
`execute()` for each
 command



Elqoo

www.elqoo.com

Design Pattern Details

PAGE 14

Structure

Participants

Collaborations

Consequences

Design
Pattern
Details

Intent

Known As

Motivation

Applicability

Elqoo

www.elqoo.com

Command Pattern

Intent and Known As

PAGE 15



Intent

Encapsulate a request as an object, thereby letting you **parameterize clients** with different **requests, queue or log requests**, and support **undoable operations**

Known As

Action, Transaction

Elqoo

www.elqoo.com

Apply Command Pattern

Applicability

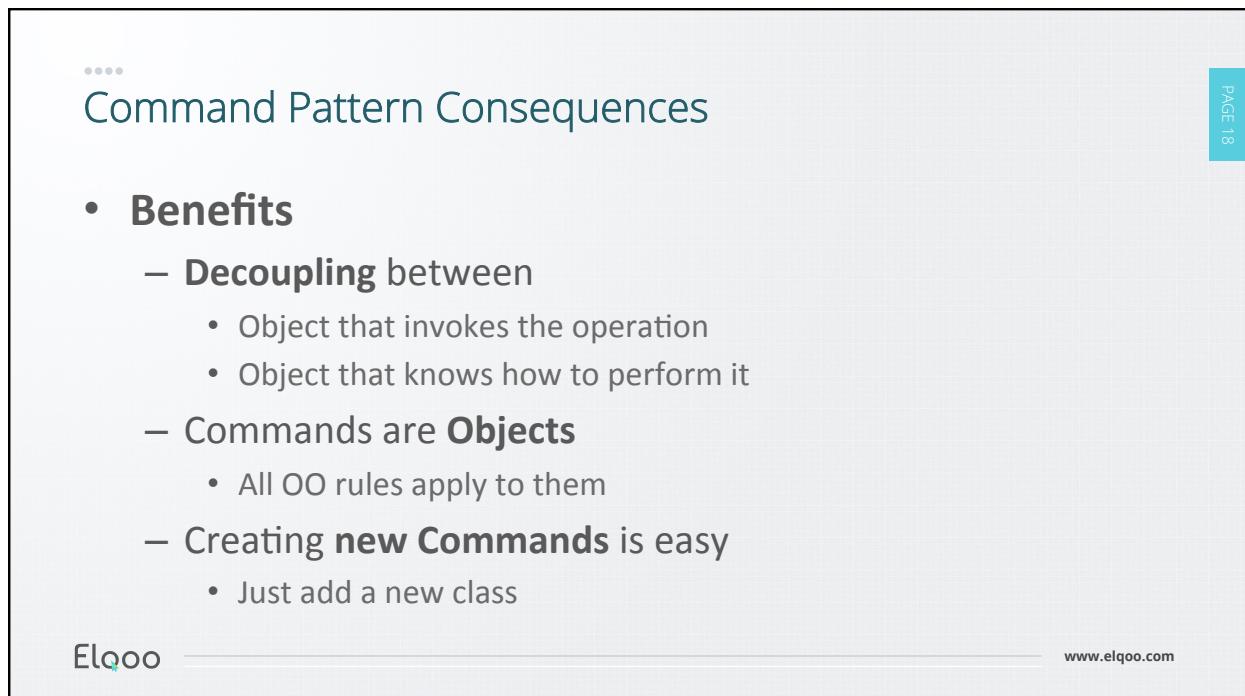
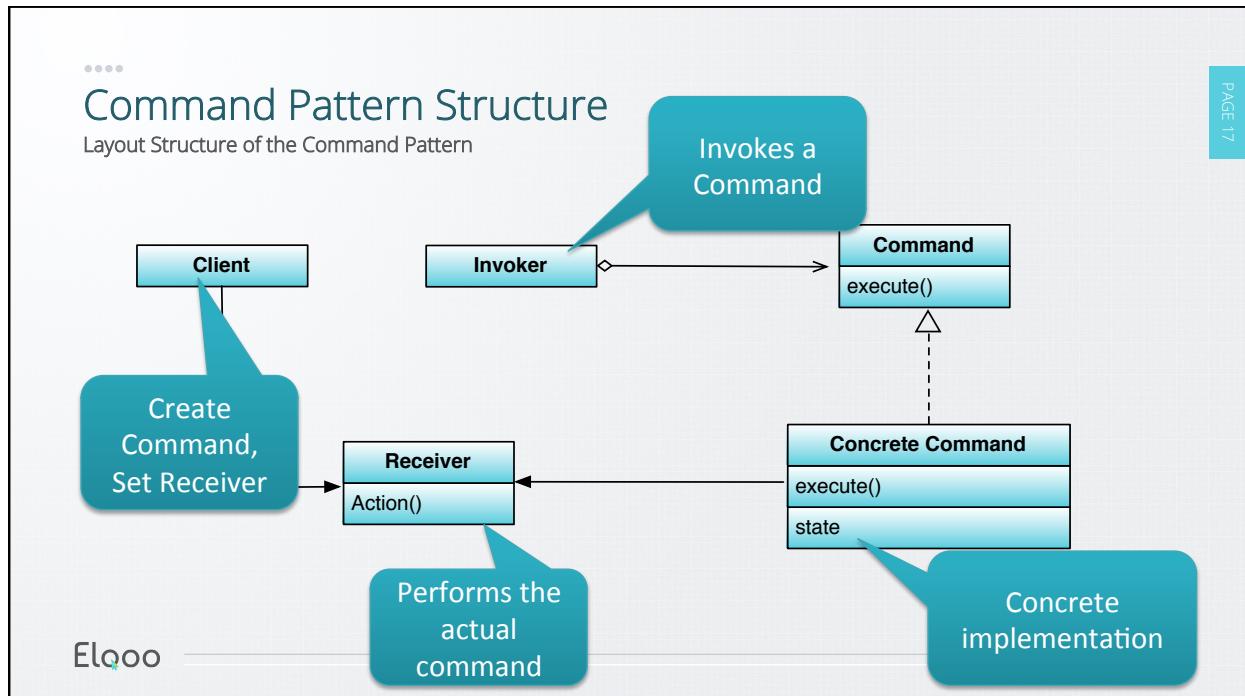
PAGE 16

- **Use**

- Command as **parameter**
- Pass command like general object
- **Queue Request**
- **Save request state**
- Undo functionality
- Provide an execute and undo method
- Support Logging
- **Re-execute** code in case of failure

Elqoo

www.elqoo.com



Conclusion

PAGE 19

- **Command pattern is great**
 - Capture a request
 - Centralize action functionality

Elqoo

www.elqoo.com



.....

Problem Statement

PAGE 21



SD

Brad

- Hello Brad
- Hi Suzy
- For our chess application:
we need to be able to set
the difficulty
- Sure not problem



PM

Suzy

Elqoo

www.elqoo.com

.....

Problem Statement Overview

PAGE 22

- **Multiple algorithms** to define to create an AI
- It needs to be **easily changed**
- No other code should be adapted in changing the algorithm

Elqoo

www.elqoo.com

Problem Statement Detail

Motivation

PAGE 23

- Programming algorithms into general application classes
 - More complex if algorithm grows
 - Difficult to maintain
 - Cannot switch between algorithms when needed

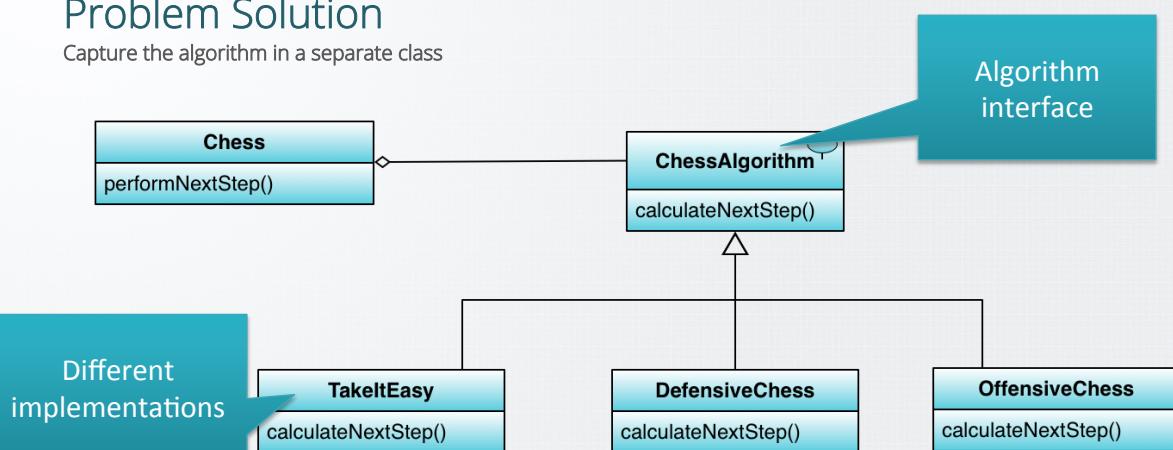
Elqoo

www.elqoo.com

Problem Solution

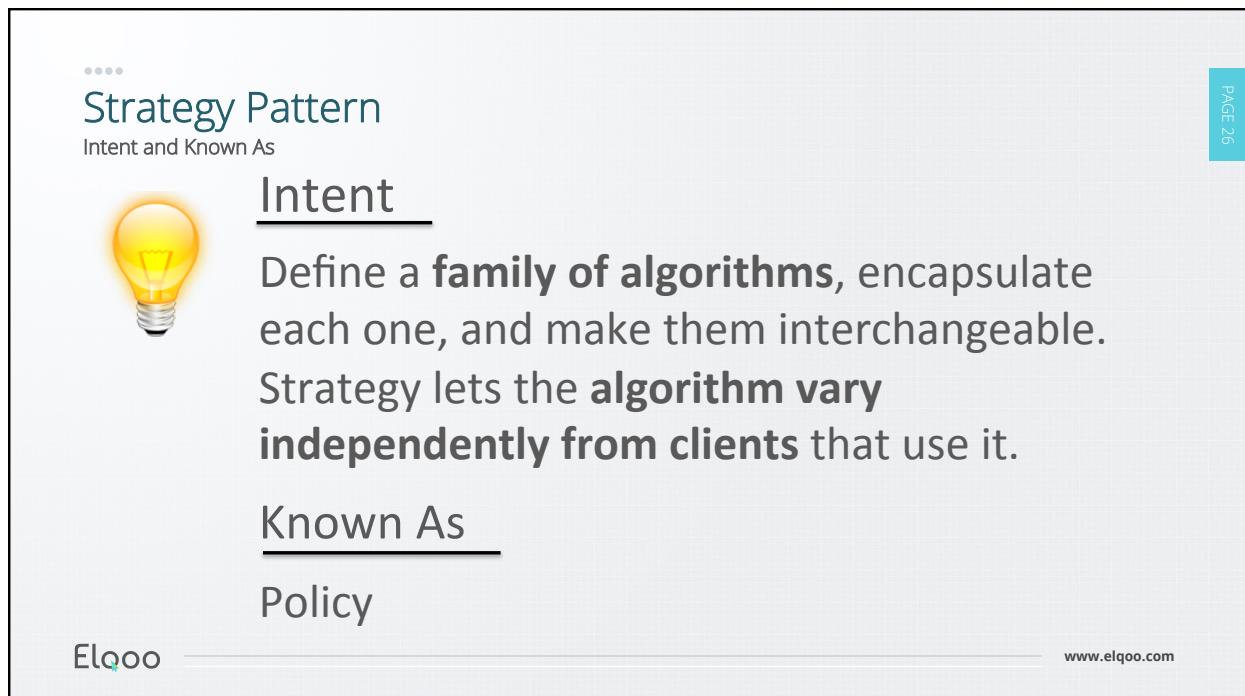
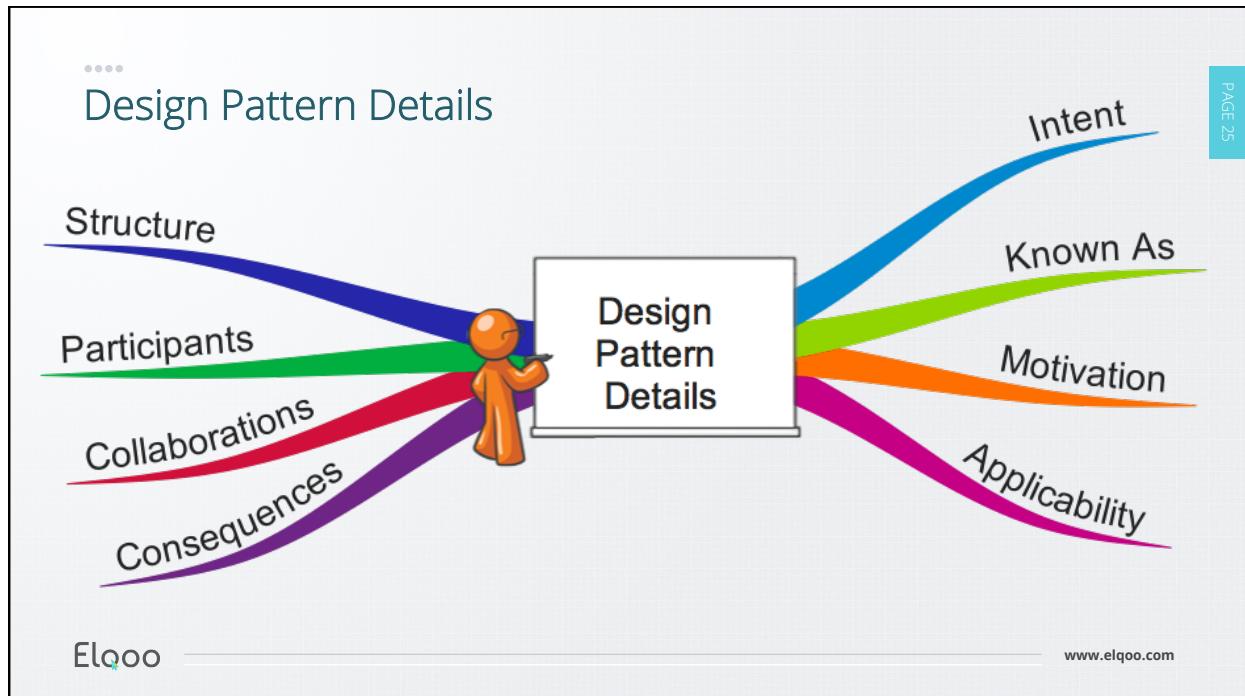
Capture the algorithm in a separate class

PAGE 24



Elqoo

www.elqoo.com



Apply Strategy Pattern

Applicability

PAGE 27

- **Use**

- Classes only change in behavior
- Different variants of an **algorithm**
- Algorithms that **use complex data** that clients shouldn't be aware of

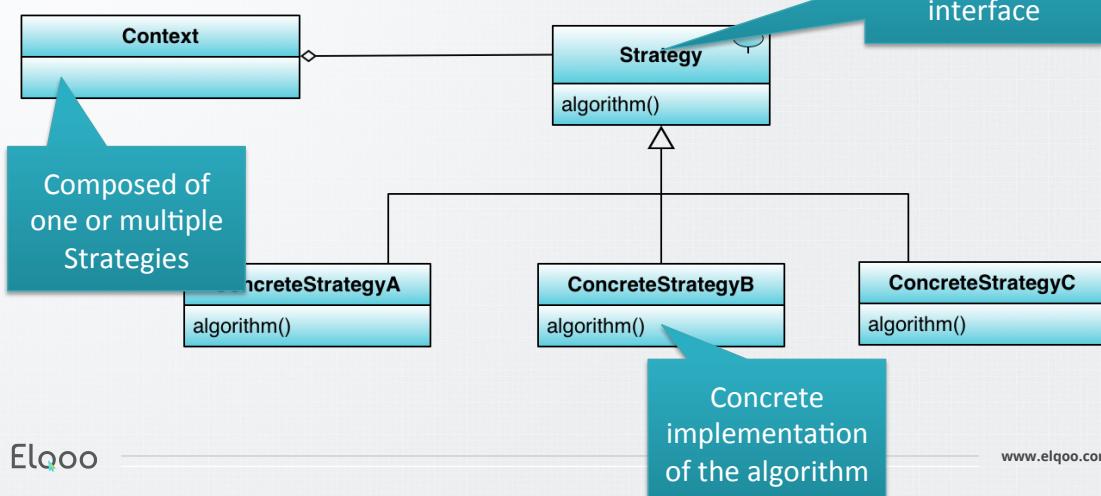
Elqoo

www.elqoo.com

Strategy Pattern Structure

Layout Structure of the strategy pattern

PAGE 28



Elqoo

www.elqoo.com

.....

Strategy Pattern Collaboration

Collaboration between objects

PAGE 29

- Context can **pass necessary data to strategy**
- Clients of the context can **pass a strategy**
- Context **forwards clients requests to the strategy**

.....

Builder Pattern Consequences

Benefits of the strategy pattern

PAGE 30

- **Benefits**

- Algorithm families use inheritance for common parts
- Avoid conditional statements using this pattern
- **Clients can choose the required behavior**

- **Drawbacks**

- Pattern increases nr of objects in application
 - Share strategies between objects.

Conclusion

PAGE 31

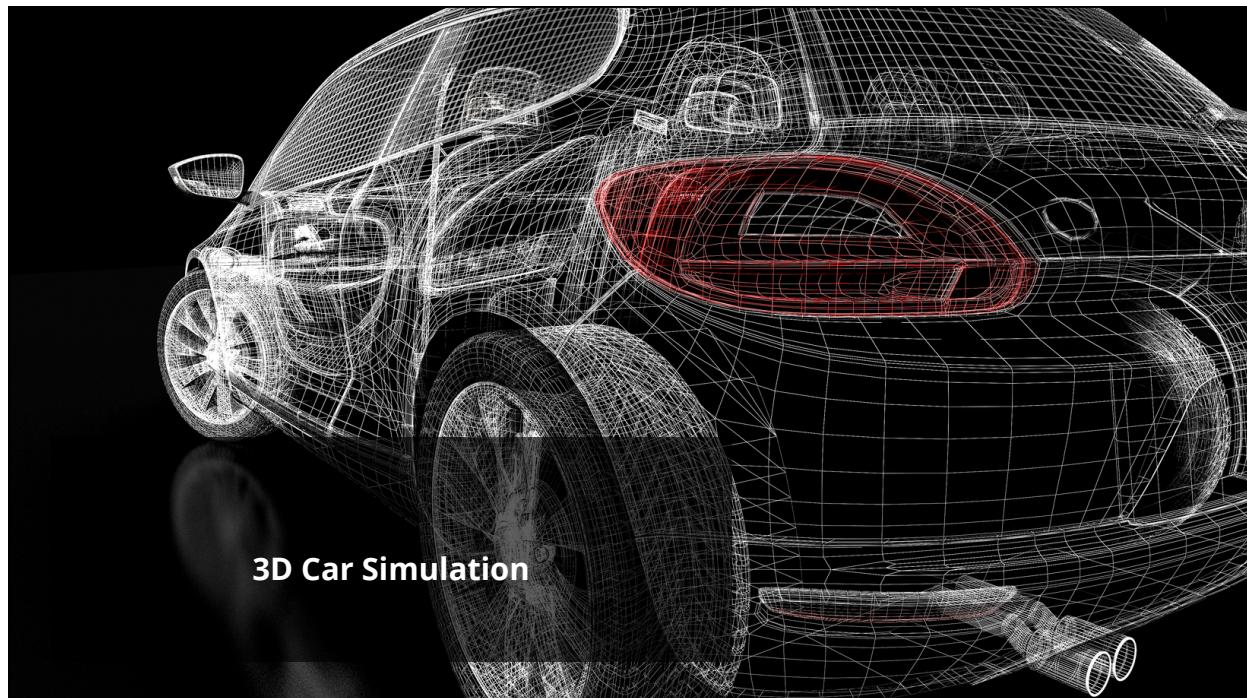
- **Strategy pattern is great**
 - Encapsulate an algorithm
 - Change algorithm at run-time

Elqoo

www.elqoo.com



•
- Visitor-
Visit Tree Nodes



.....

Problem Statement

PDGE 34



SD

Brad

- Hello Brad
- Hi Suzy
- We want to print and render cars in our 3D simulation program
- Ok

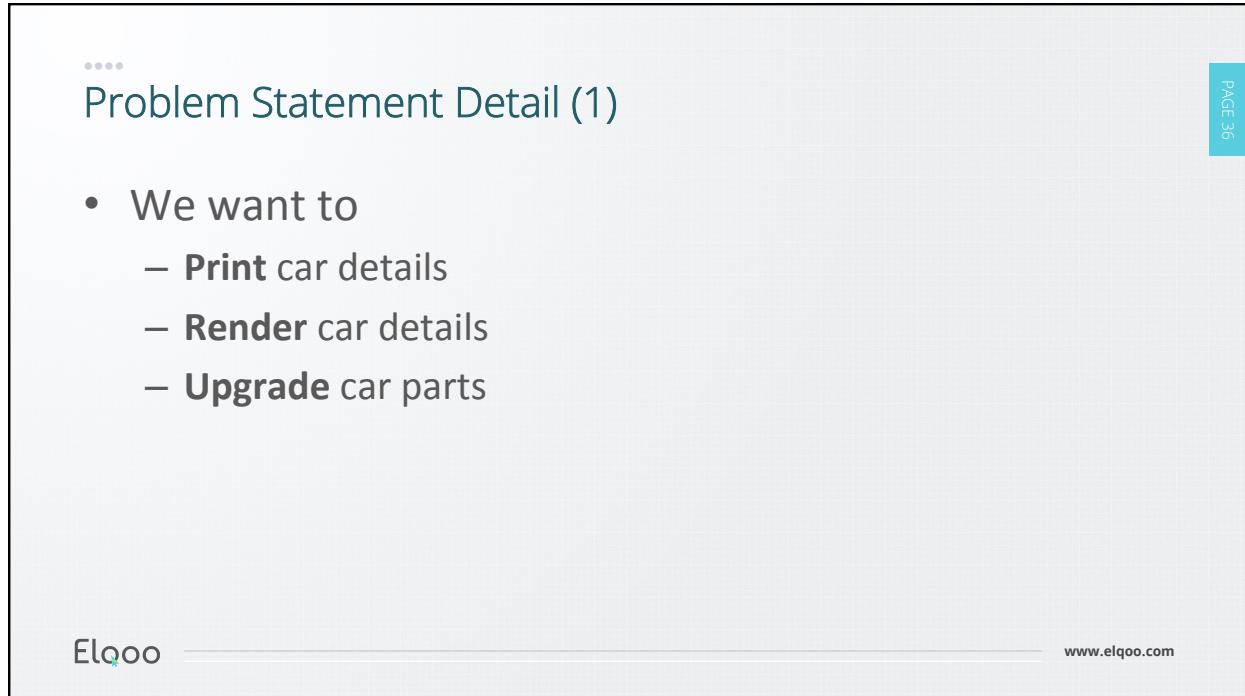
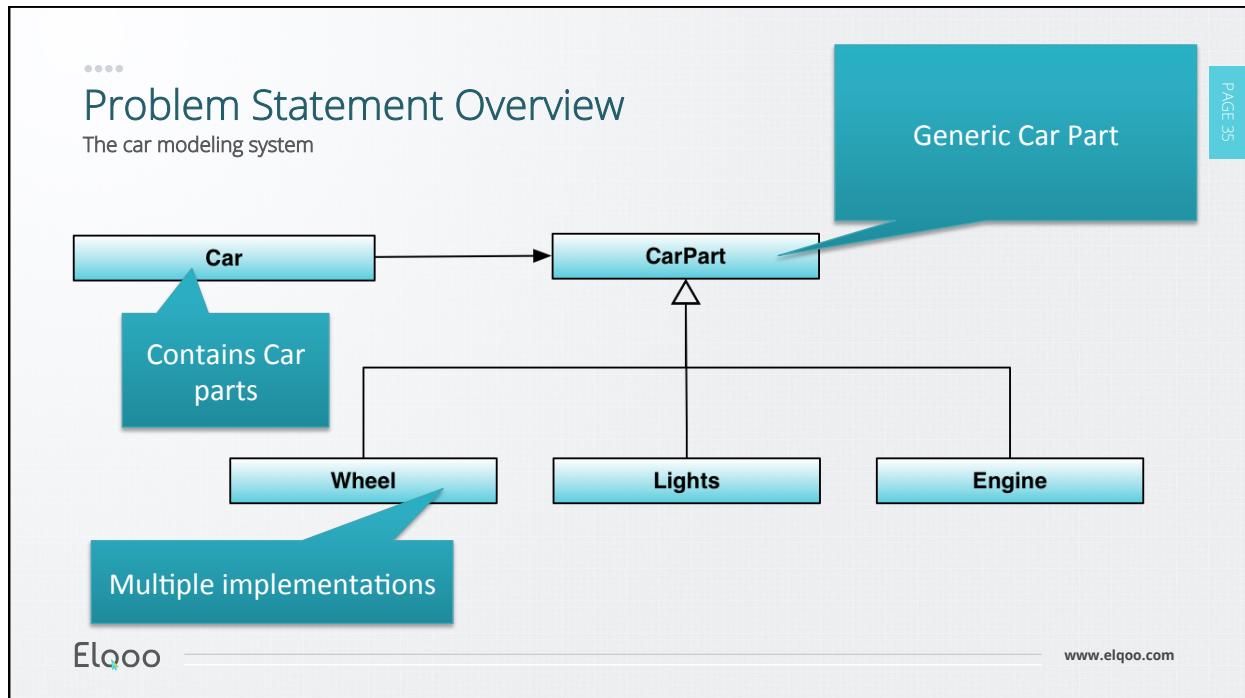


PM

Suzy

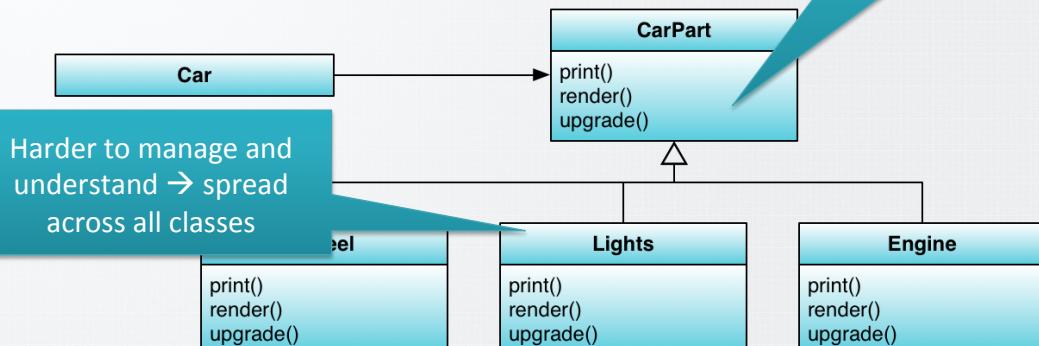
Elqoo

www.elqoo.com



Problem Statement Detail (2)

PAGE 37



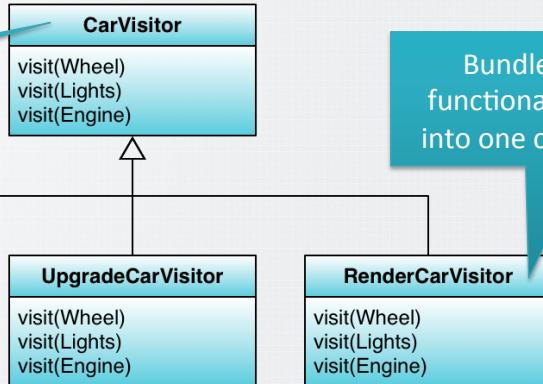
Elqoo

www.elqoo.com

Problem Solution (1)

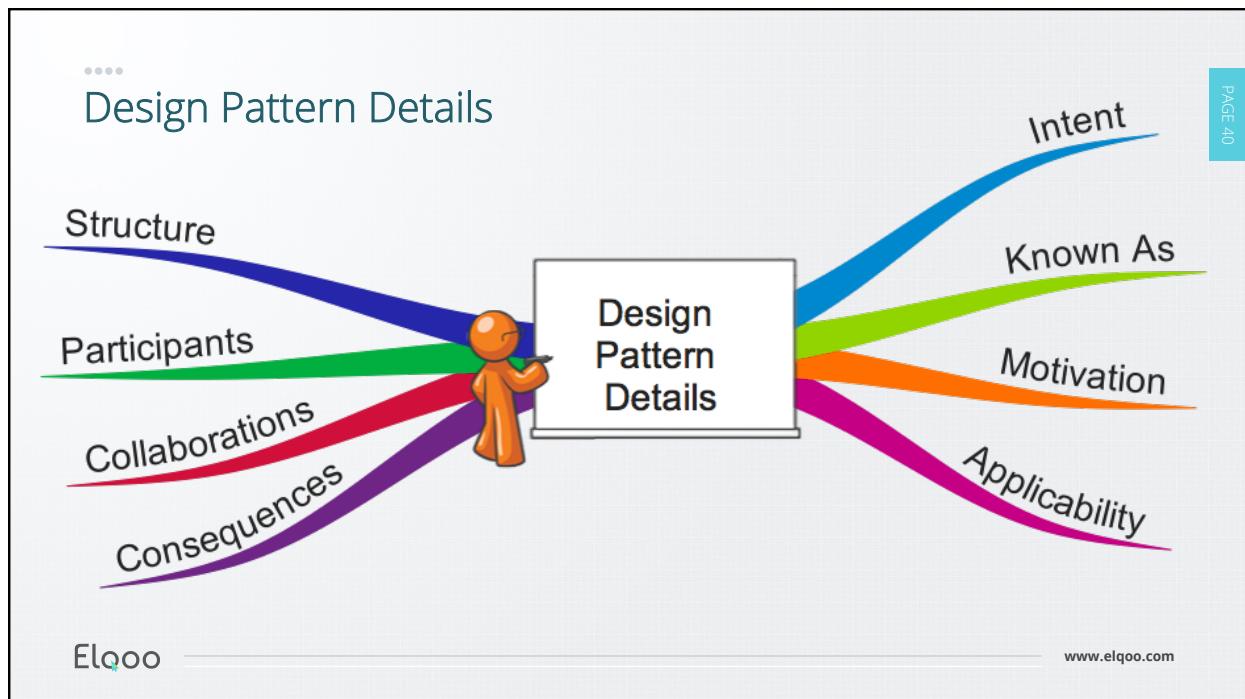
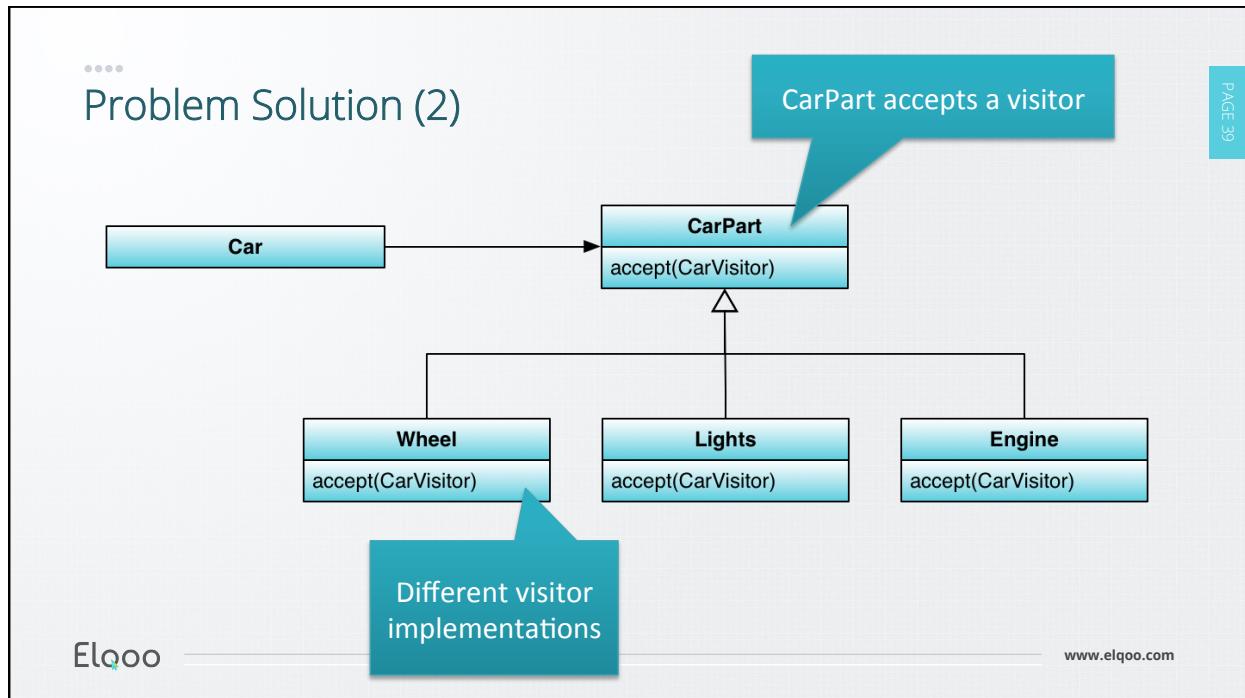
PAGE 38

Introduce a Car Visitor



Elqoo

www.elqoo.com



..... Visitor Pattern

Intent and Known As

PAGE 41



Intent

Represent an **operation** to be **performed** on the elements of an object structure. Visitor lets you **define a new operation** without changing the classes of the elements on which it operates.

Elqoo

www.elqoo.com

..... Apply Visitor Pattern

Applicability

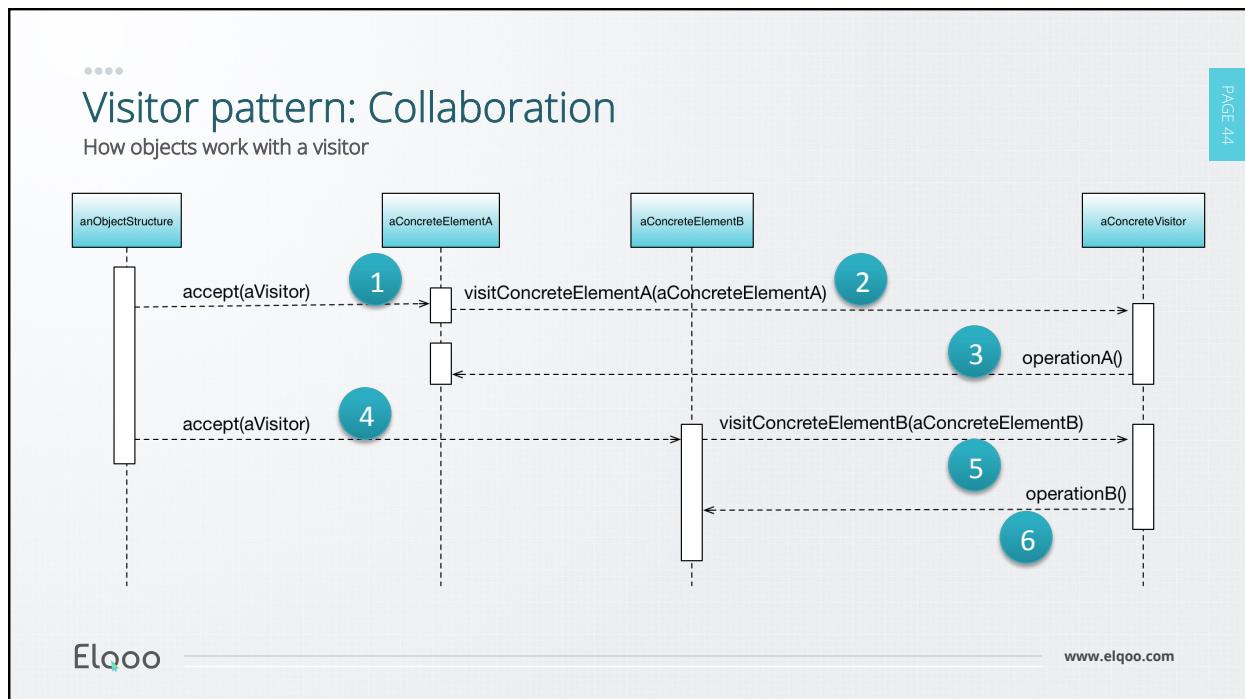
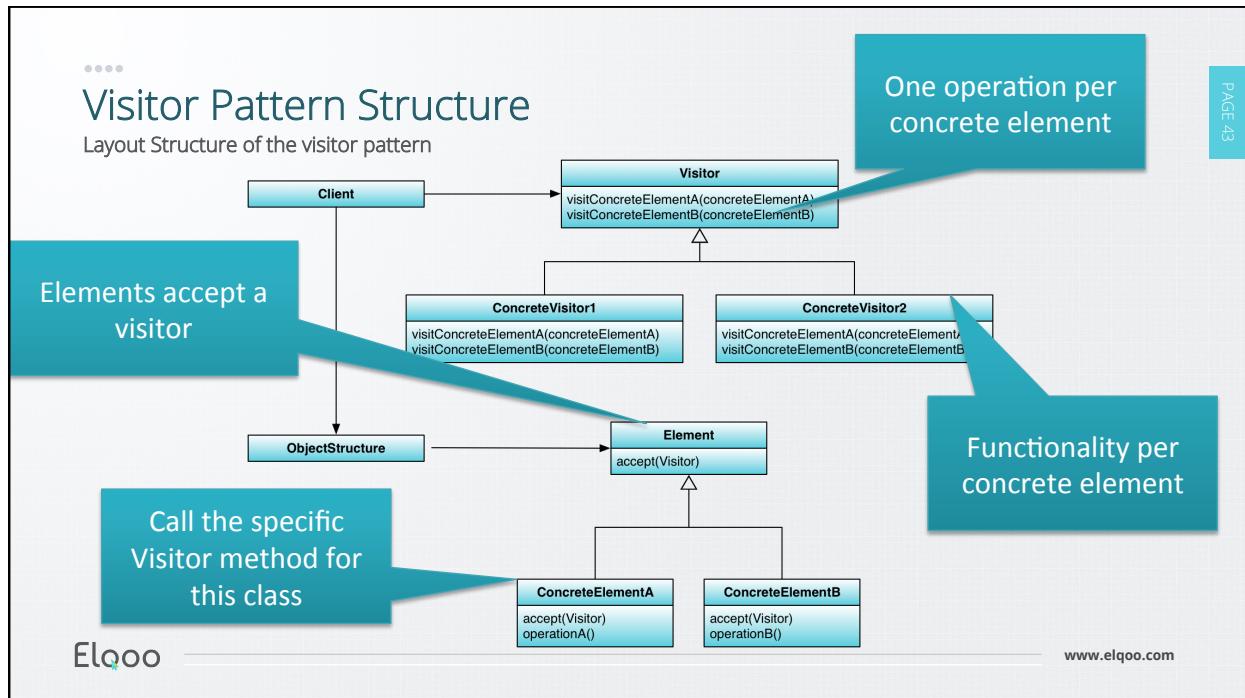
PAGE 42

- **Use**

- Visit complex object structure (inheritance)
 - Perform operations based upon concrete classes
- Avoid pollution of concrete classes with many different operations
 - Visitor groups this functionality
- Ability to easily define new operations without changing concrete classes.

Elqoo

www.elqoo.com



.... Visitor Pattern Consequences

Considerations for the Visitor pattern

- **Benefits**

- Adding **new operations** is **easy**
- Visitor separates operations that don't belong together
- **Accumulate state**
 - Visitors can maintain state across the hierarchy

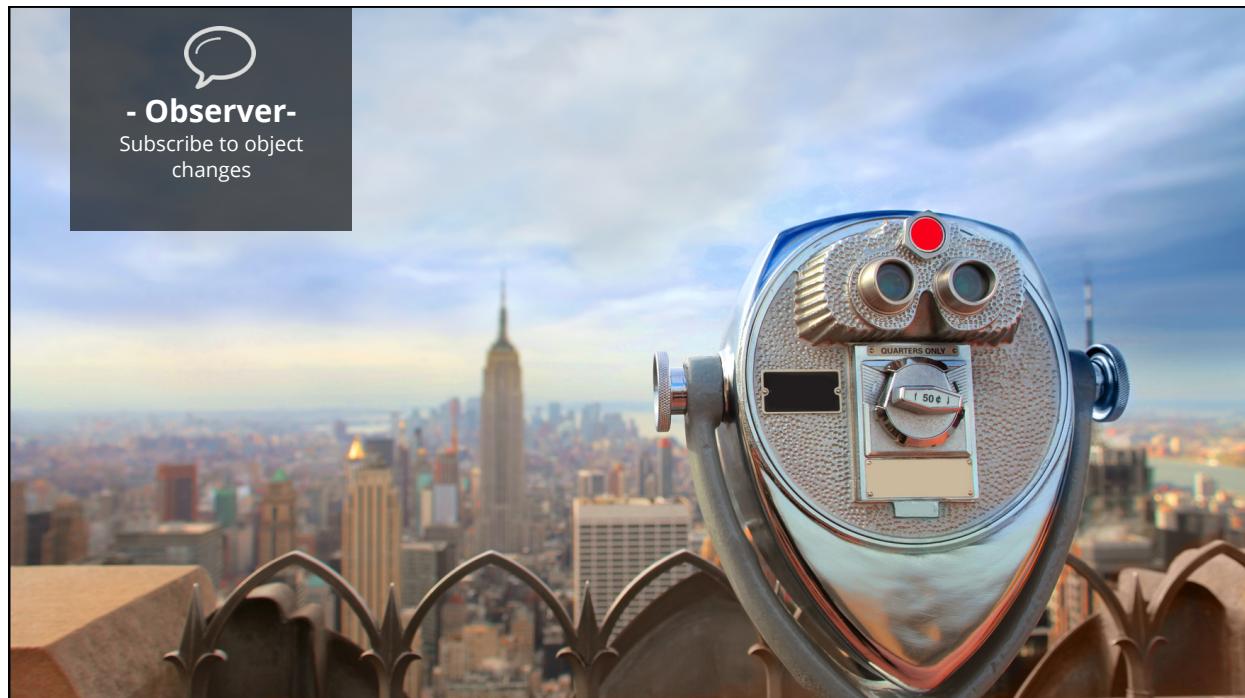
- **Drawbacks**

- **Adding new concrete elements** is **hard**
 - Requires a new method for all concrete visitors
- Visitors **break encapsulation**
 - They rely on the interface of the concrete element

.... Conclusion

- **Visitor pattern is great**

- **Visit complex object structure**
- Centralize logic




- Observer-
Subscribe to object
changes

.....

Problem Statement

PAGE 48

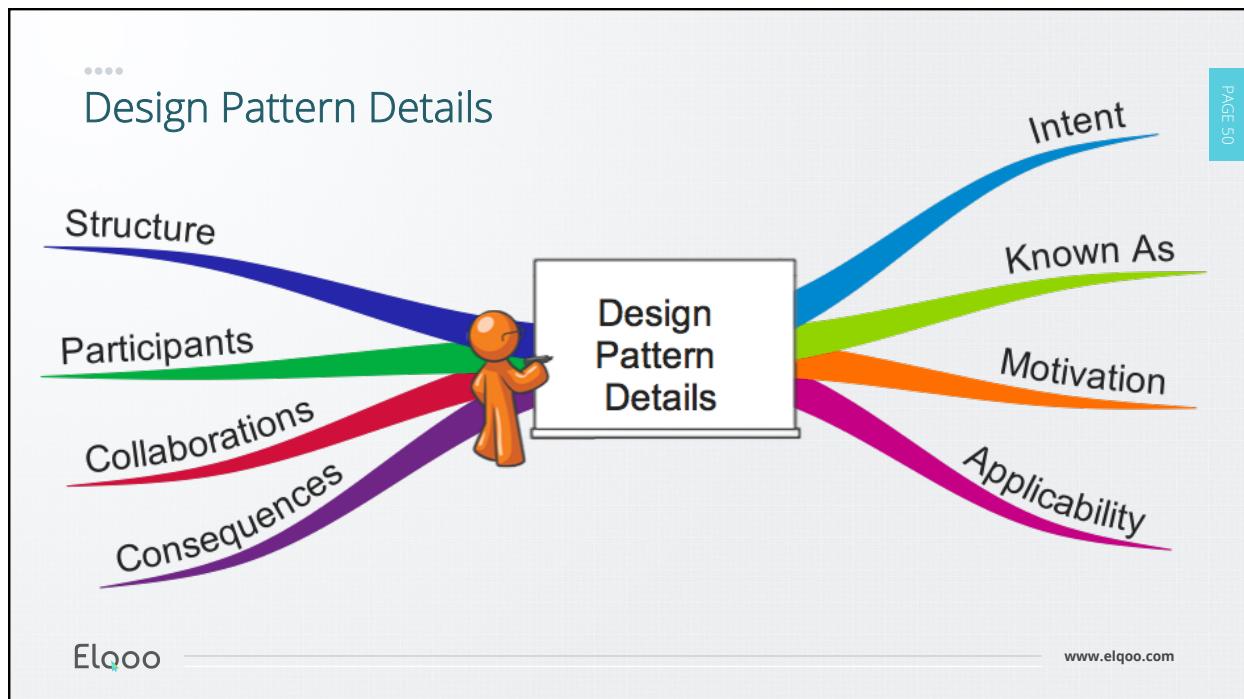
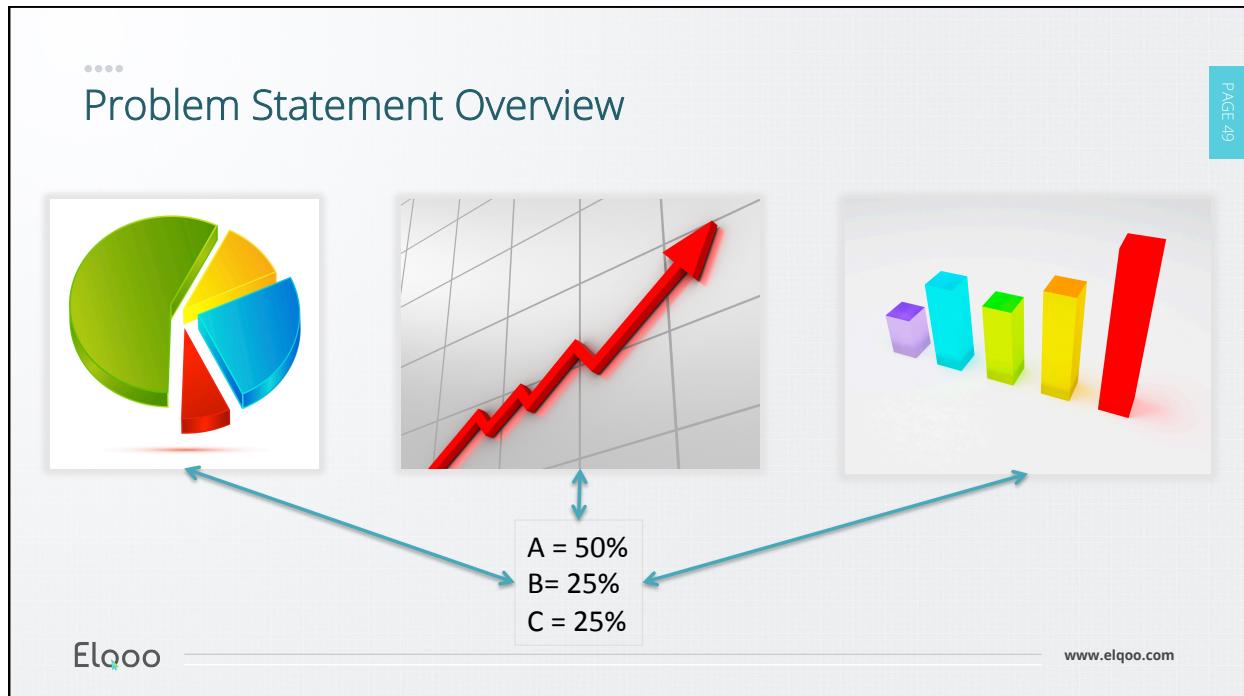

SD
Brad

- Hello Brad
- Hi Suzy
- When our data changes,
we want to automatically
change the graphical
representations in our
business application
- No problem


PM
Suzy

Elqoo

www.elqoo.com



Observer Pattern

Intent and Known As

PAGE 51



Intent

Define a **one-to-many dependency** between objects so that when one object changes state, **all its dependents are notified** and updated automatically

Known As

Dependents, Publish-Subscribe

Elqoo

www.elqoo.com

Apply Observer Pattern

Applicability

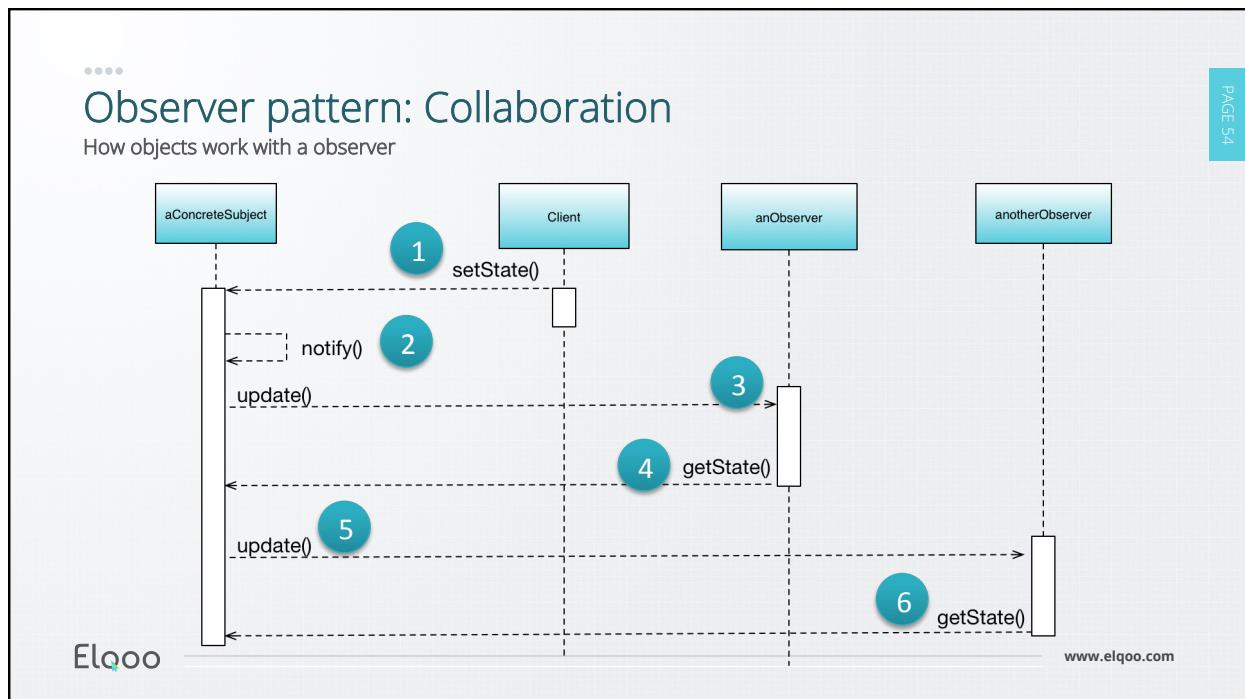
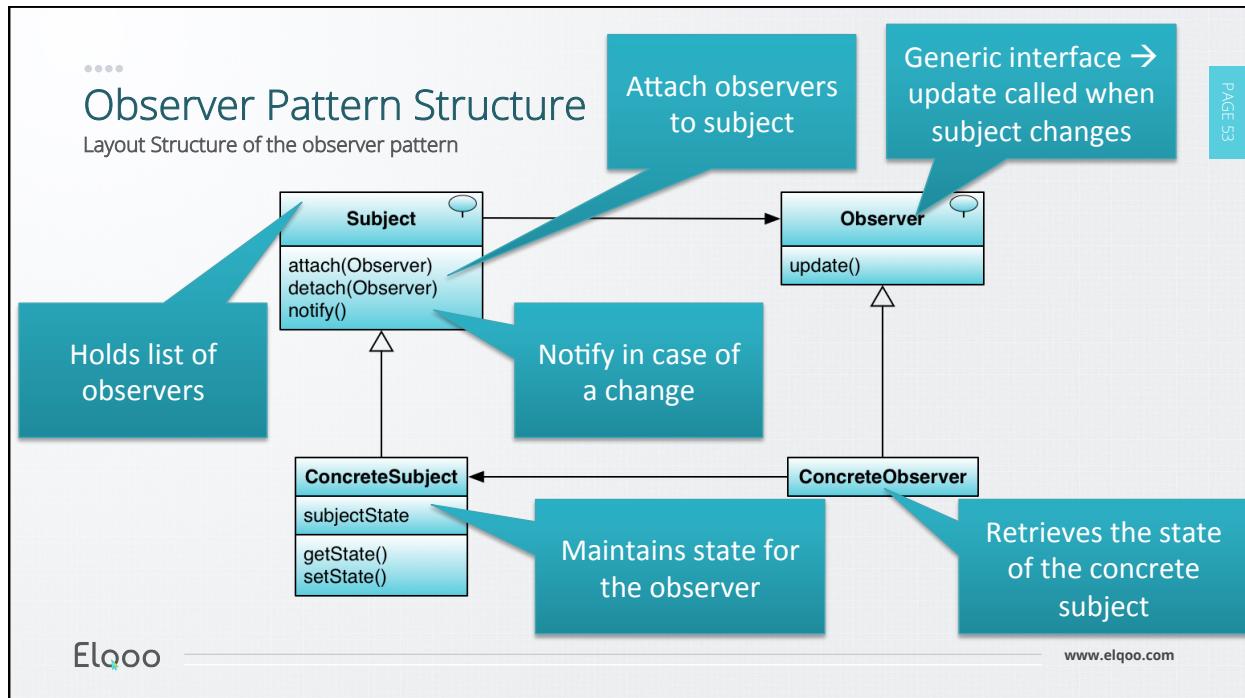
PAGE 52

- **Use**

- Change one object → changes others
- No idea how many objects need to be changed
- Object change notification
 - With preserving loose coupling
- One object may notify another without knowing them directly

Elqoo

www.elqoo.com



.... Observer Pattern Consequences

Considerations for the observer pattern

- **Benefits**

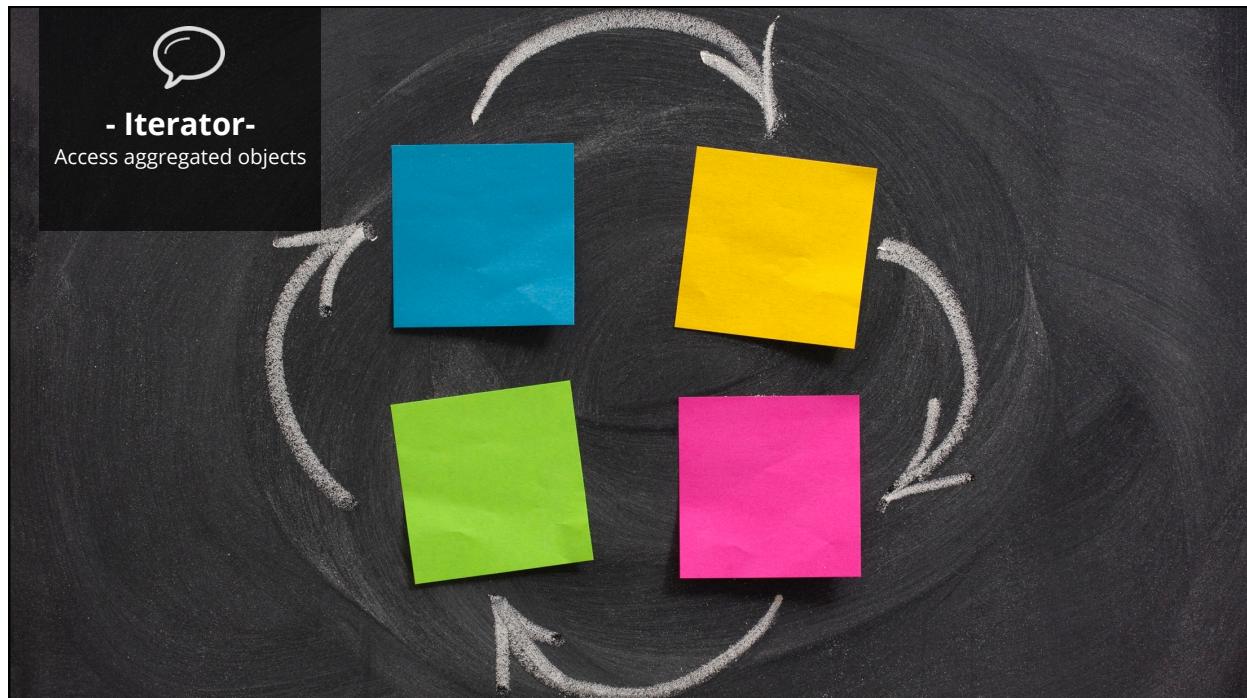
- Loose coupling between observers and subjects
- Supporting a broadcast model

- **Drawbacks**

- One change can result in multiple unnecessary notifications
- Clients don't know the ripple effects

.... Conclusion

- **Observer pattern is great**
 - Observe state in other objects
 - Preserve loose coupling



....

Problem Statement

PAGE 58



SD

Brad

- Hello Brad
- Hi Suzy
- Some of the new developers need help with traversing through some data.
- Ok



PM

Suzy

Elqoo

www.elqoo.com

.....

Problem Statement Overview

How to iterate through a collection without knowing how it's implemented

PAGE 59

Interface Collection<E>

Type Parameters:
E - the type of elements in this collection

All Superinterfaces:
Iterable<E>

All Known Subinterfaces:
BeanContext, BeanContextServices, BlockingDeque<E>, BlockingQueue<E>, Deque<E>, List<E>, NavigableSet<E>, Queue<E>, Set<E>, SortedSet<E>, TransferQueue<E>

All Known Implementing Classes:
AbstractCollection, AbstractList, AbstractQueue, AbstractSequentialList, AbstractSet, ArrayBlockingQueue, ArrayDeque, ArrayList, AttributeList, BeanContextServicesSupport, BeanContextSupport, ConcurrentLinkedDeque, ConcurrentLinkedQueue, ConcurrentSkipListSet, CopyOnWriteArrayList, CopyOnWriteArraySet, DelayQueue, EnumSet, HashSet, JobStateReasons, LinkedBlockingDeque, LinkedBlockingQueue, LinkedHashSet, LinkedList, LinkedTransferQueue, PriorityBlockingQueue, PriorityQueue, RoleList, RoleUnresolvedList, Stack, SynchronousQueue, TreeSet, Vector

Expand the Collection interface with traversal methods?

What if we need a new iteration method e.g. back to forth

Different implementations of a Collection

Elqoo

www.elqoo.com

.....

Design Pattern Details

PAGE 60

Structure

Participants

Collaborations

Consequences

Intent

Known As

Motivation

Applicability

Elqoo

www.elqoo.com

.... Iterator Pattern

Intent and Known As

PAGE 61



Intent

Provide a way to **access the elements of an aggregate object sequentially without exposing its underlying representation.**

Known As

Cursor

Elqoo

www.elqoo.com

.... Apply Iterator Pattern

Applicability

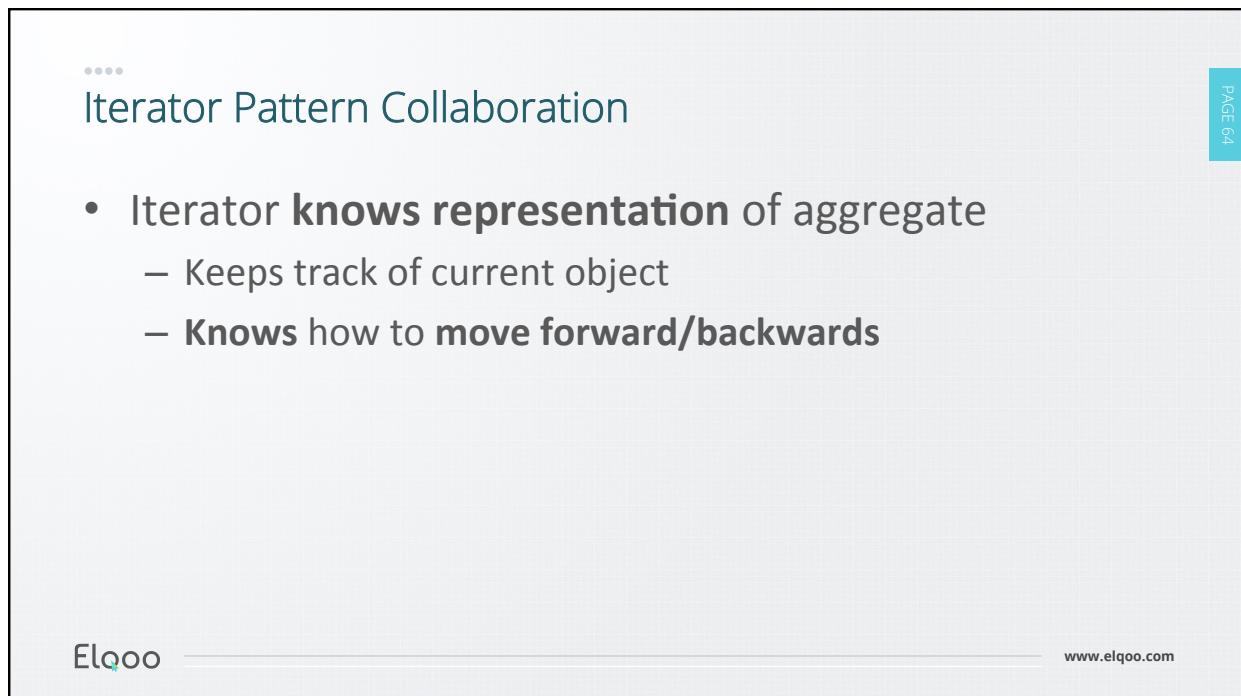
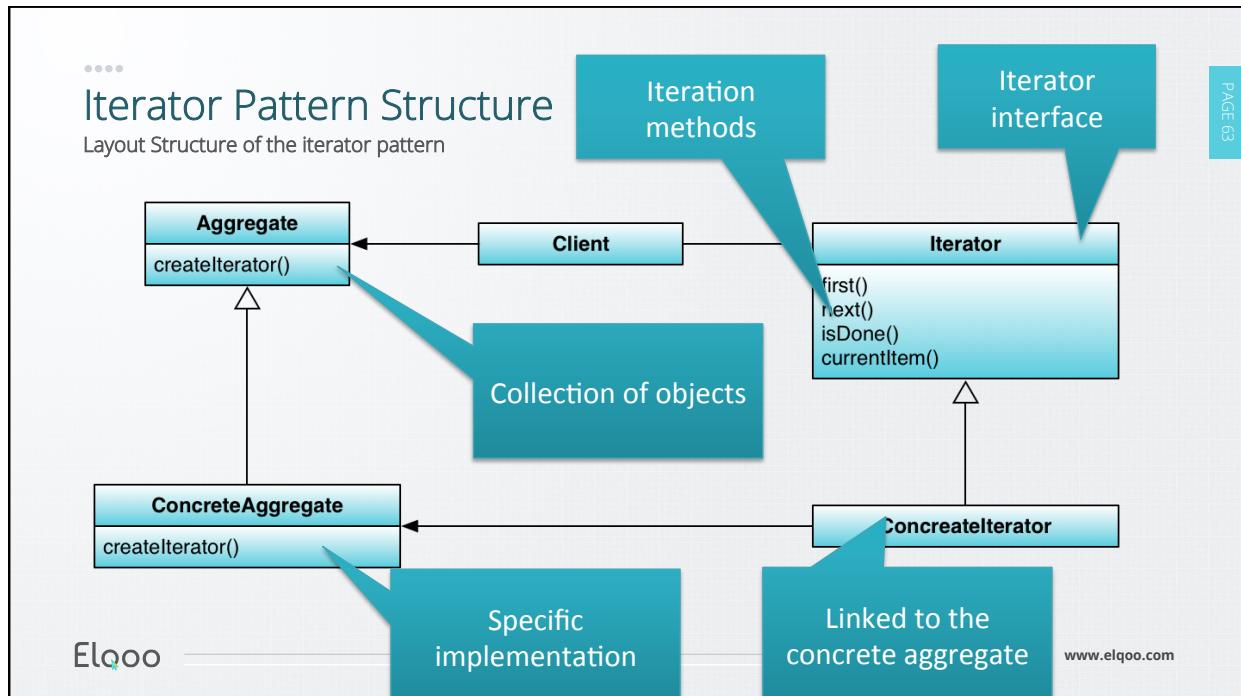
PAGE 62

- **Use**

- Access aggregated object's contents **without exposing its representation**
- Support multiple traversal of aggregated objects
- Provide **uniform interface** to
 - Traverse aggregated objects
 - Might be of different classes

Elqoo

www.elqoo.com



.... Iterator Pattern Consequences

Considerations for the Iterator pattern

PAGE 65

- **Benefits**

- **Multiple iteration** variations are possible
- Aggregate or Collection **interface is simplified**
- Each iterator keeps track where he is
 - Use **different iterators** to traverse an object **at same time**

.... Conclusion

PAGE 66

- **Iterator pattern is great**

- Traverse aggregated objects
- Provide unified interface



.....

Problem Statement

PAGE 68



SD

Brad

- Hello Brad
- Hi Suzy
- We want to support undo operations in our graphical drawing tool
- Sounds easy



PM

Suzy

Elqoo

www.elqoo.com

.....

Problem Statement Overview

Example of a graphical drawing tool

PAGE 69

The screenshot shows a graphical drawing application window. On the left is a toolbar with icons for selection, text, shape creation, and other functions. The main area is a grid-based workspace where three blue rectangular shapes are placed. A vertical line connects the top and bottom shapes. On the right is a 'Properties' panel with tabs for 'Connections' and 'Note'. Under 'Connections', there are checkboxes for 'Allow connections from lines' (checked), 'Connect to the group' (radio button selected), and 'Connect to the components'. A dropdown menu shows '8 magnets'. Under 'Note', there is a text input field and a table for data keys and values. At the bottom of the panel, it says '3 of 3 objects selected' and '100%'. A teal callout box labeled 'Ability to draw shapes' points to the workspace. Another teal callout box labeled 'Supports undo operation' points to the 'Connections' tab in the Properties panel.

Ability to draw shapes

Supports undo operation

Elqoo

www.elqoo.com

.....

Problem Statement Detail (1)

Implementing line connectivity

PAGE 70

The diagram illustrates the UML representation of line connectivity. It shows two horizontal rectangles representing objects. A vertical line connects them, with a double-headed arrow indicating bidirectional connectivity. To the right is a class diagram for 'ConnectionSolver' with three compartments: 'state' at the top, 'ConnectionSolver' in the middle, and 'calculateLine()' at the bottom. Arrows point from the objects to the vertical line and from the line to the 'ConnectionSolver' class.

UML representation of the line.

ConnectionSolver

state

calculateLine()

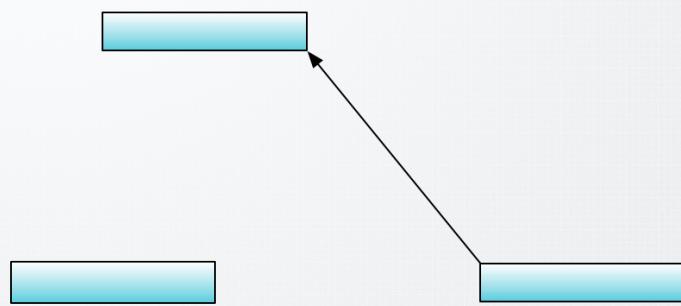
Elqoo

www.elqoo.com

Problem Statement Detail (2)

How would the undo functionality work?

PAGE 71



Move Rectangle

Undo Move?

Elqoo

www.elqoo.com

Problem Statement Detail (3)

PAGE 72



Undo Move FAIL →
We lost original
information

Elqoo

www.elqoo.com

Problem Solution

Solution to the undo graphical drawing tool problem

- We need to **restore the original state** of the ConnectionSolver

– Without breaking encapsulation

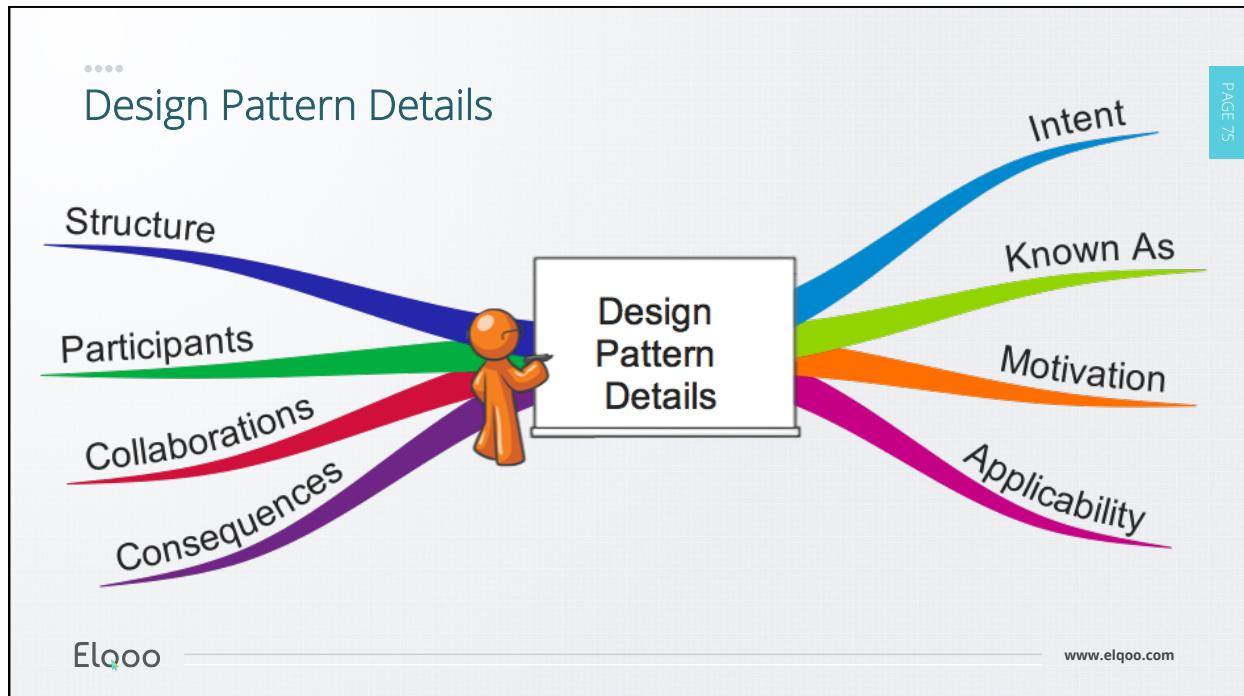
1. Memento pattern: **Save state** ConnectionSolver
2. Store memento state for later use
3. **Restore ConnectionSolver state** on undo

Problem Solution

Extract state from ConnectionSolver



Place state in a separate class that can be restored



Apply Memento Pattern

Applicability

PAGE 77

- **Use**

- Save a **snapshot** of an objects state
- Direct interface to object state would violate encapsulation

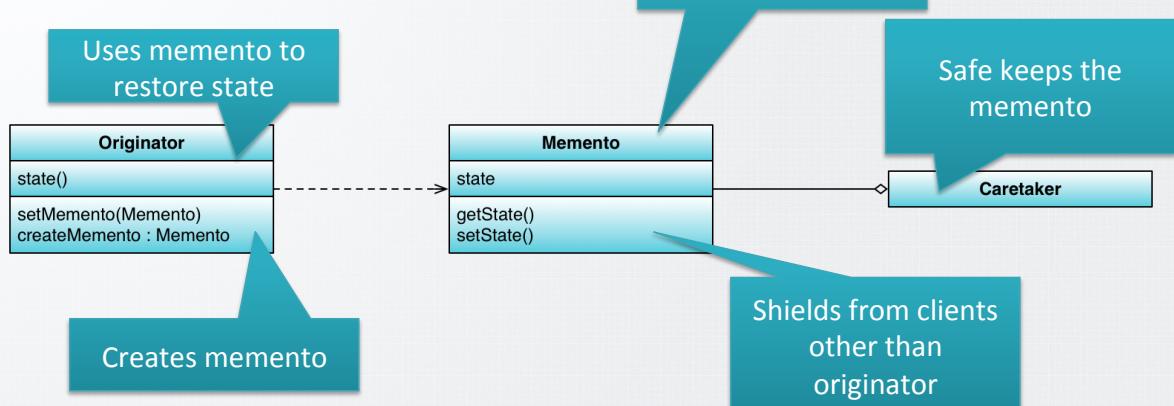
Elqoo

www.elqoo.com

Memento Pattern Structure

Layout Structure of the memento pattern

PAGE 78



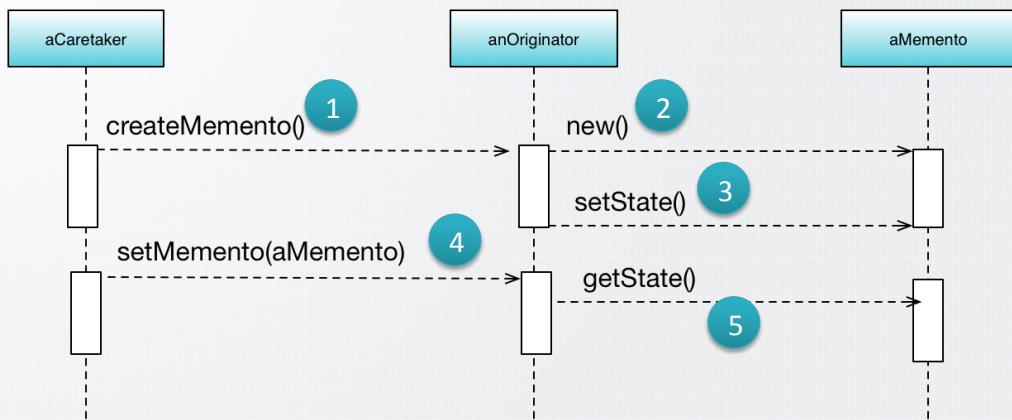
Elqoo

www.elqoo.com

Memento pattern Collaboration

How objects work with a memento

PAGE 79



Elqoo

www.elqoo.com

Memento Pattern Consequences

Considerations for the memento pattern

PAGE 80

- **Benefits**
 - Preserve encapsulation
 - Simplifies originator
- **Drawbacks**
 - Might be **expensive**
 - Object creation
 - Language must facilitate that only the originator can access the memento's state
 - **Extra management** for caretaking the mementos

Elqoo

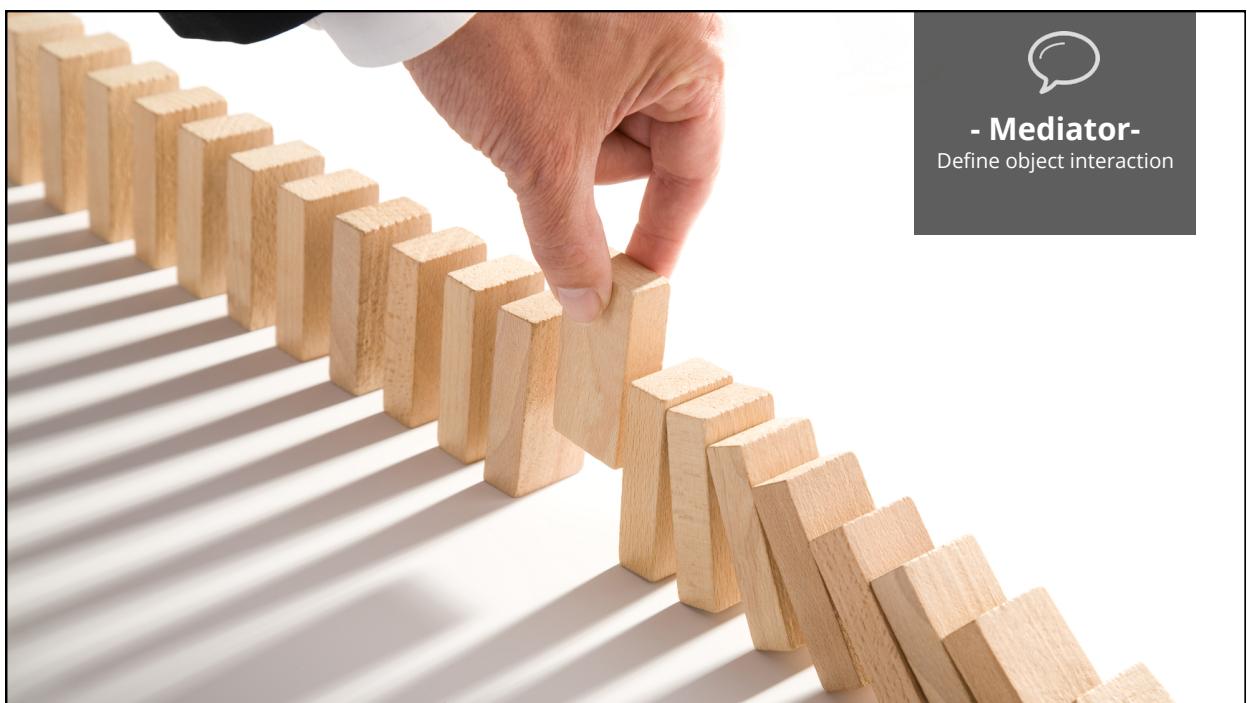
www.elqoo.com

Conclusion

PAGE 81

- **Memento pattern is great**
 - Access state but preserve **encapsulation**
 - Save **snapshot** of object data

Elqoo

www.elqoo.com

.....

Problem Statement

PAGE 83



SD

Brad

- Hello Brad
- Hi Suzy
- One of the developers is stuck with the implementation of a screen, could you have a look?
- Sure



PM

Suzy

Elqoo

www.elqoo.com

.....

Problem Statement Overview

PAGE 84

Selected: ListItem1

ListItem1
ListItem2
ListItem3



Changes upon selection of one item in the list

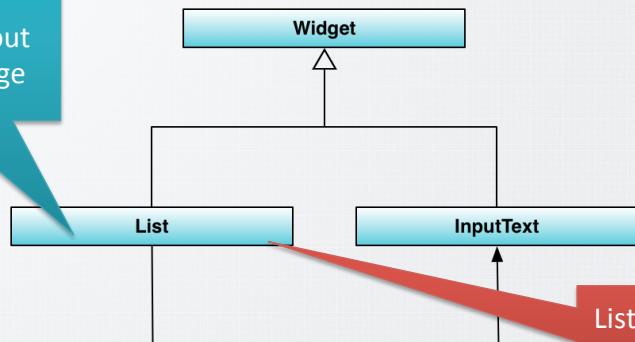
Elqoo

www.elqoo.com

Problem Statement Overview

The screen implementation

Sets state of input text upon change



List widget needs an InputText → No reuse possible

www.elqoo.com

Elqoo

Problem Solution

Loose coupling between List and InputText

ScreenDirector

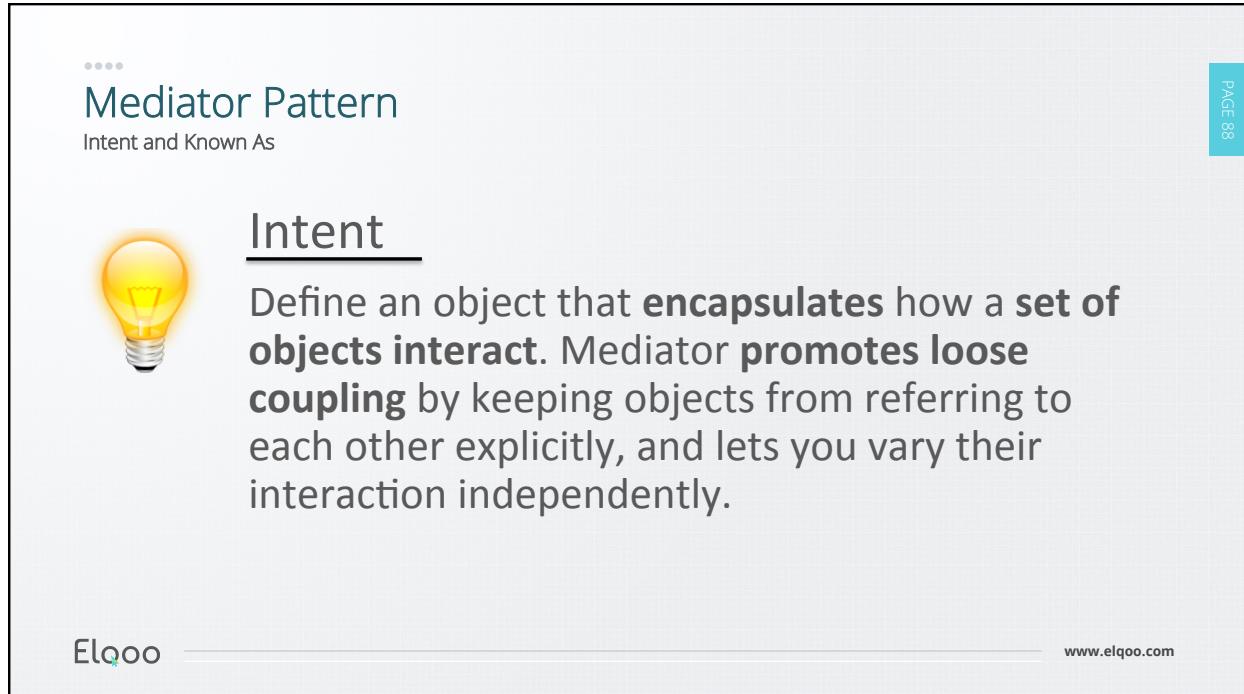
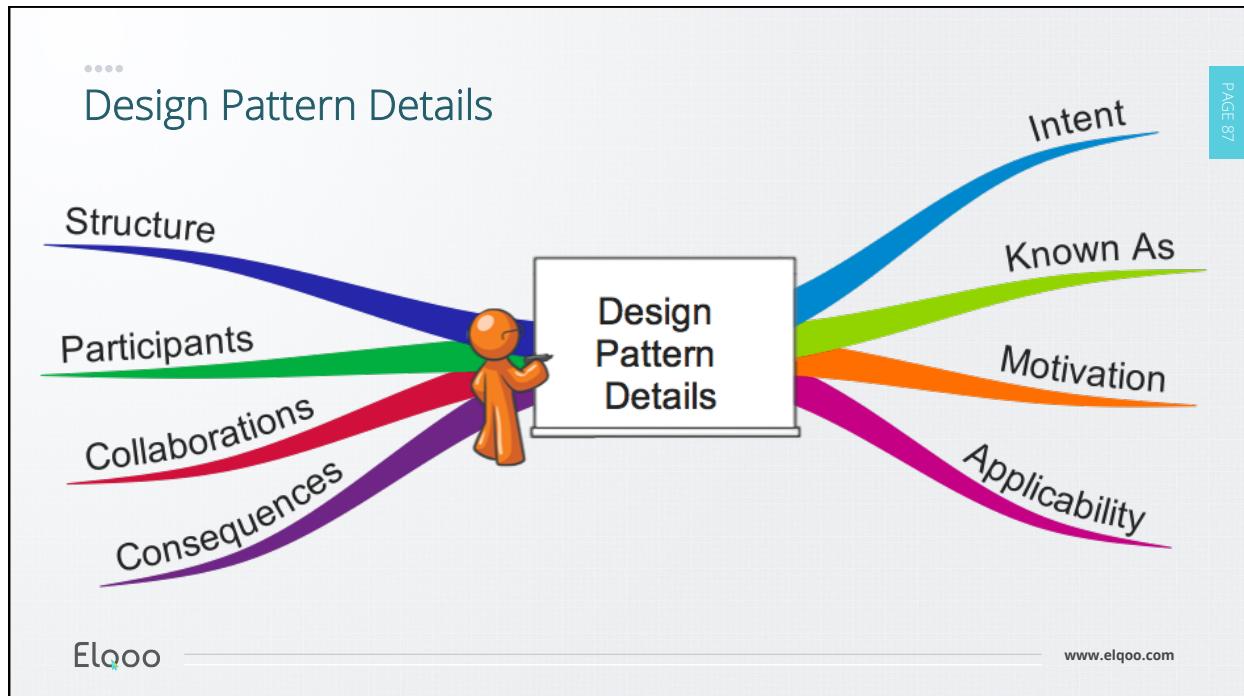
Screen director manages widget interaction

ScreenDirectorImpl

Implementation manages list and inputtext

Elqoo

www.elqoo.com



Apply Mediator Pattern

Applicability

PAGE 89

- **Use**

- Objects have **complex communication**, but it's well defined
- Hard to identify how the communication actually works
- Object **re-use** is **difficult**
 - These objects require a set of different objects (waterfall of dependencies)
- **Centralize behavior** between classes.

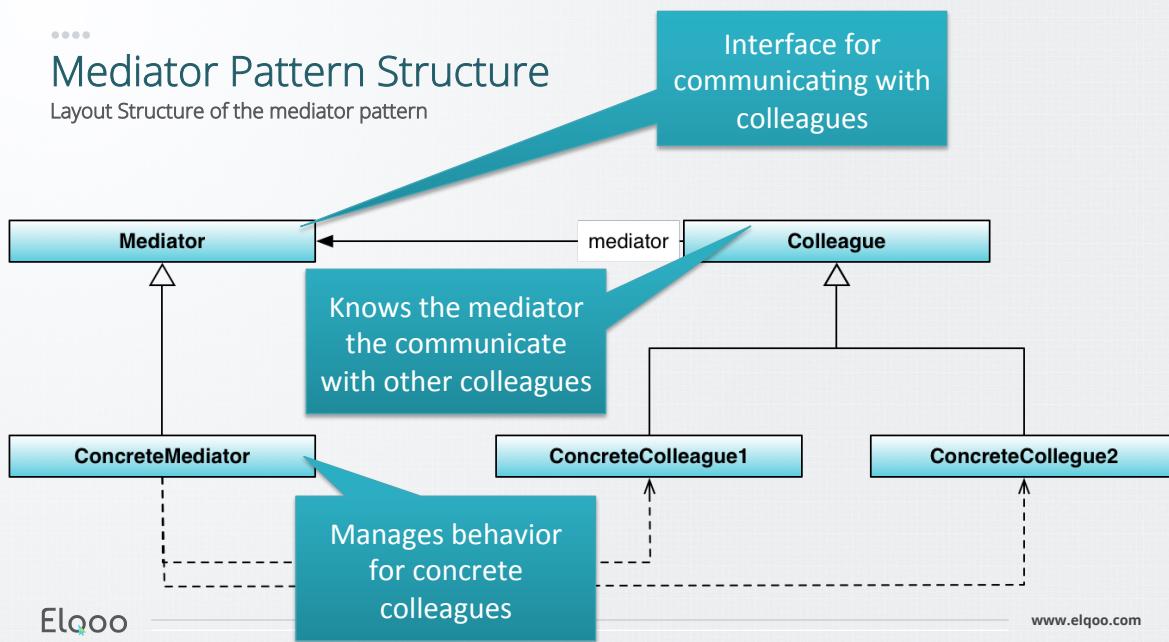
Elqoo

www.elqoo.com

Mediator Pattern Structure

Layout Structure of the mediator pattern

PAGE 90



.... Mediator Pattern Consequences

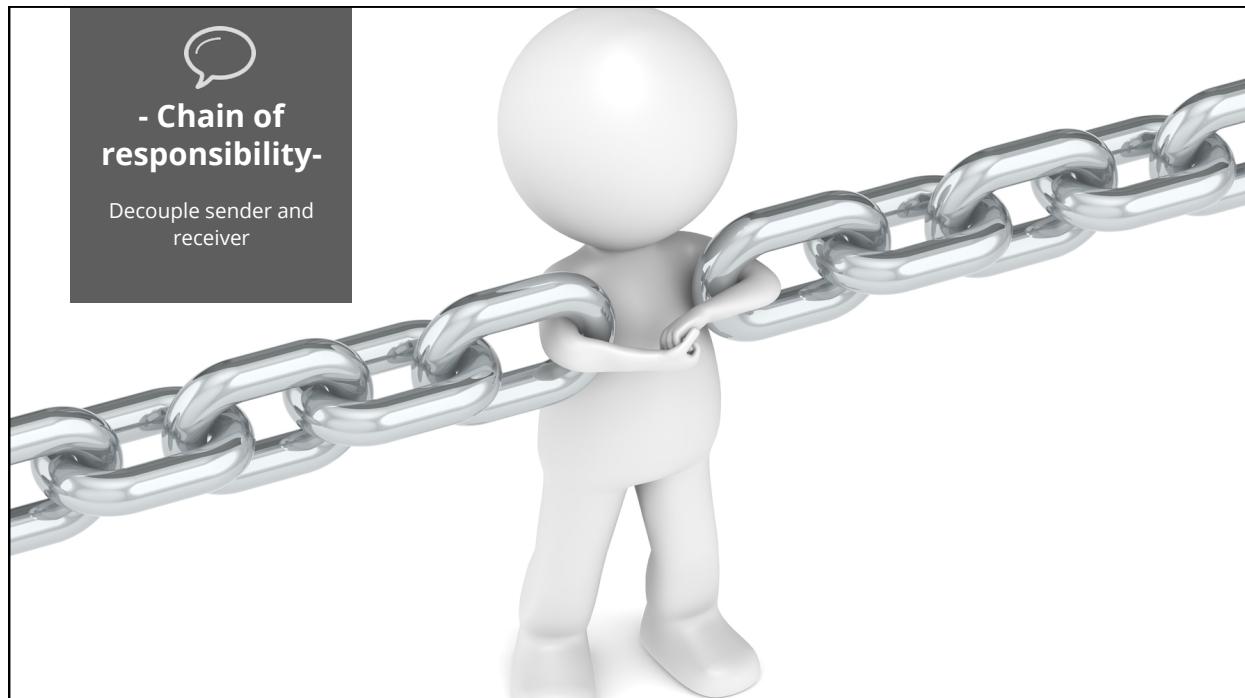
Considerations for the Mediator pattern

- **Benefits**

- **Centralize behavior**
 - Would otherwise be distributed among other objects
- **Decoupling of colleagues**
- Changes **many-to-many** interaction to **one-to-many** interaction
- Object collaboration is abstracted
- **Centralized control**

.... Conclusion

- **Mediator pattern is great**
 - Centralize behavior and control
 - Loose coupling between colleagues



.....

Problem Statement

PAGE 94



SD

Brad

- Hello Brad
- Hi Suzy
- The users of our e-commerce system want to add plugins to the checkout page.
- No problem



PM

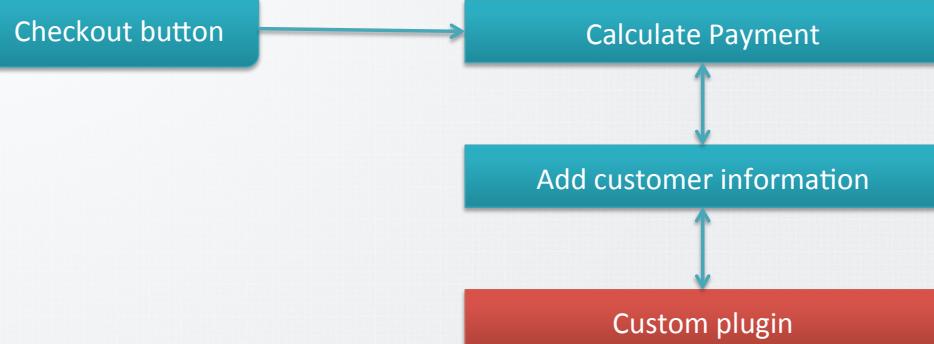
Suzy

Elqoo

www.elqoo.com

Problem Statement Overview

PAGE 95



Elqoo

www.elqoo.com

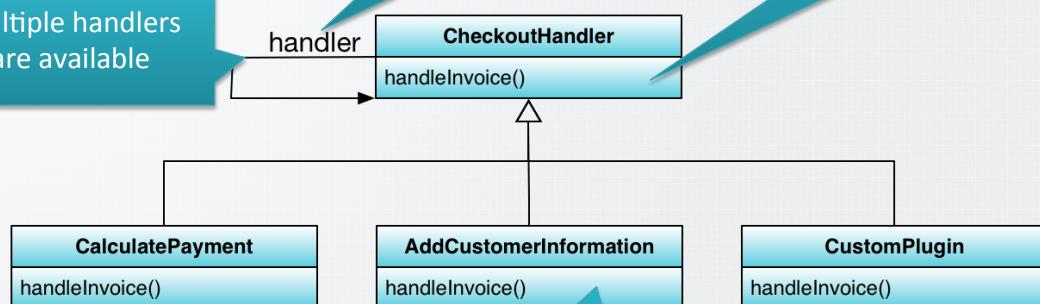
Problem Solution

PAGE 96

Multiple handlers are available

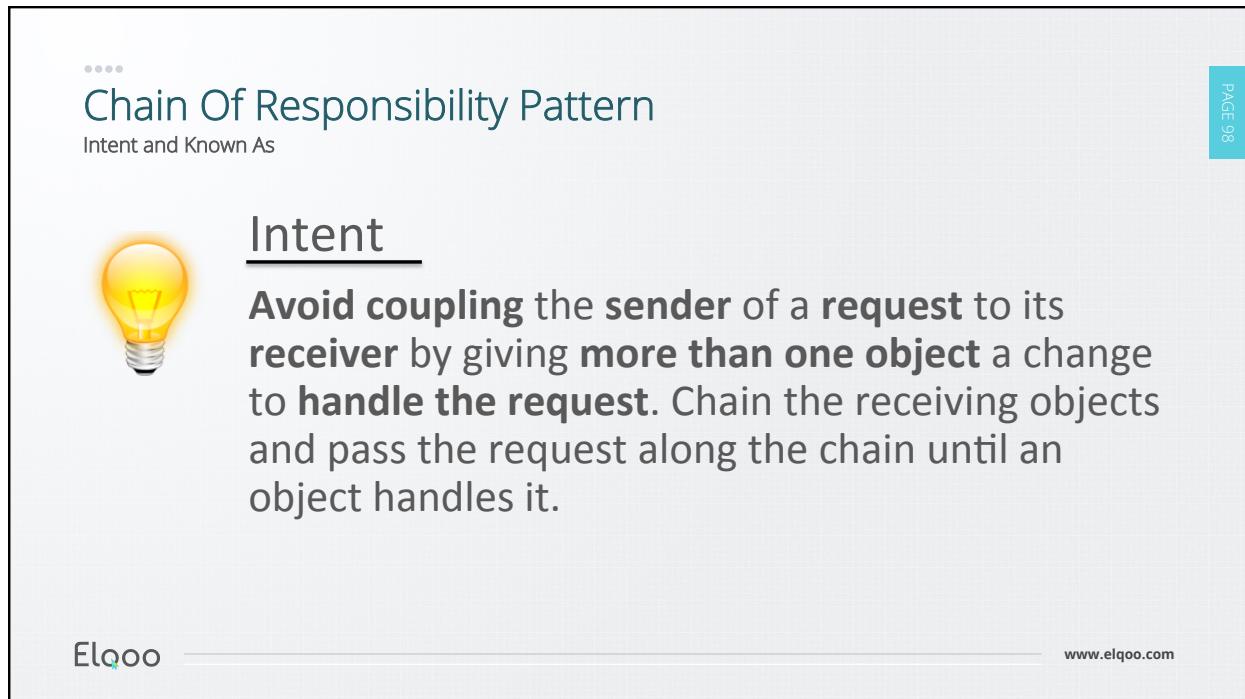
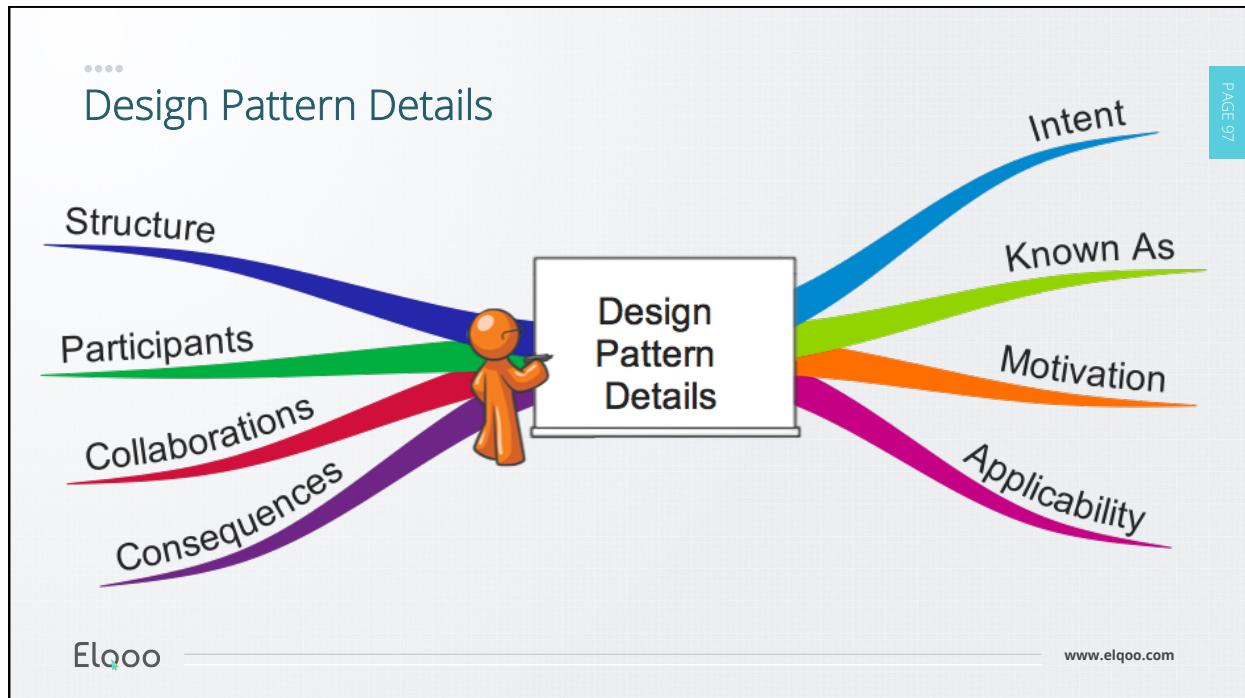
Handlers can be chained together

Generic interface to handle the checkout



Elqoo

www.elqoo.com



Apply Chain Of Responsibility Pattern

Applicability

PAGE 99

- **Use**

- Multiple objects need to handle a request
- Isn't clear upfront who will handle it
- Who can handle the request should be dynamic

Elqoo

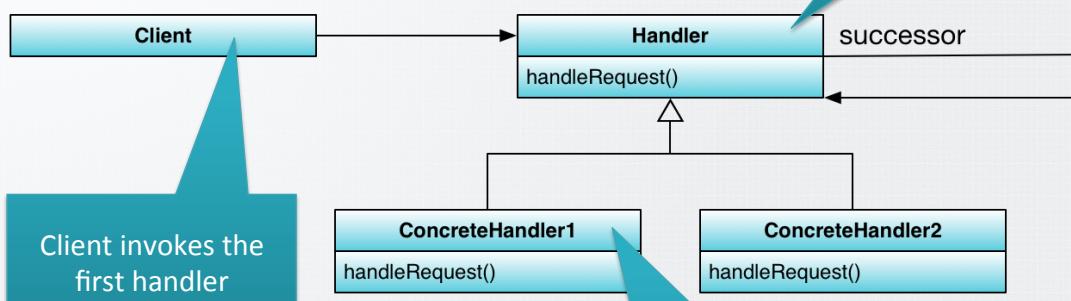
www.elqoo.com

Chain of Responsibility Pattern Structure

Layout Structure of the chain of resp. pattern

Interface for handling requests, keeps successor list

PAGE 100



Client invokes the first handler

Different handler implementations

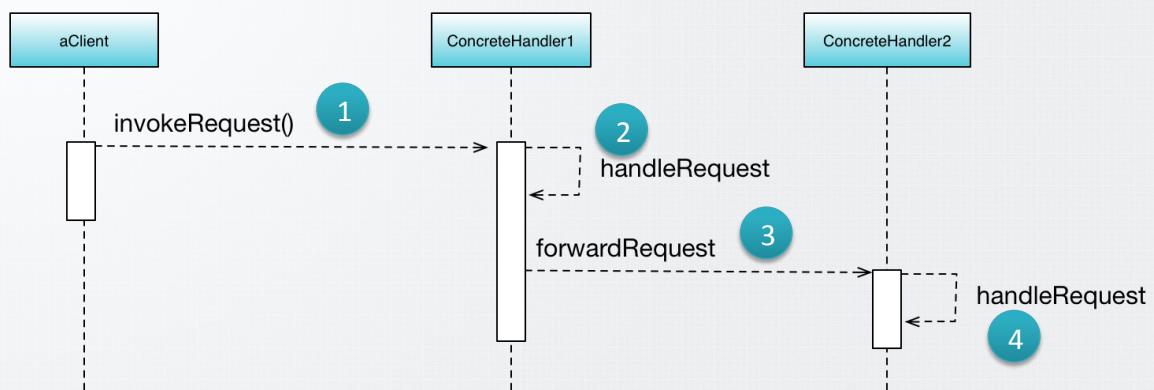
Elqoo

www.elqoo.com

Chain Of Responsibility Pattern Collaboration

Collaboration of objects using the chain of responsibility pattern

PAGE 101



Elqoo

www.elqoo.com

Chain Of Responsibility Pattern Consequences

Considerations for the chain of responsibility pattern

PAGE 102

- **Benefits**

- **Loose coupling** between **requester** and **receiver**
- Objects can **spread responsibility** in handling requests.

- **Drawbacks**

- No guarantee the **request** will be **handled**

Elqoo

www.elqoo.com

Conclusion

PAGE 103

- **Chain of responsibility pattern is great**
 - Handle requests **dynamically**
 - Enhanced loose coupling between sender and receiver

Elqoo

www.elqoo.com



- State -

Change behavior based
on state

.....

Problem Statement

PAGE 105


SD
Brad

- Hello Brad
- Hi Suzy
- We would like you to build us a networking framework.
- No problem


PM
Suzy

Elqoo

www.elqoo.com

.....

Problem Statement Overview

TCP Connection

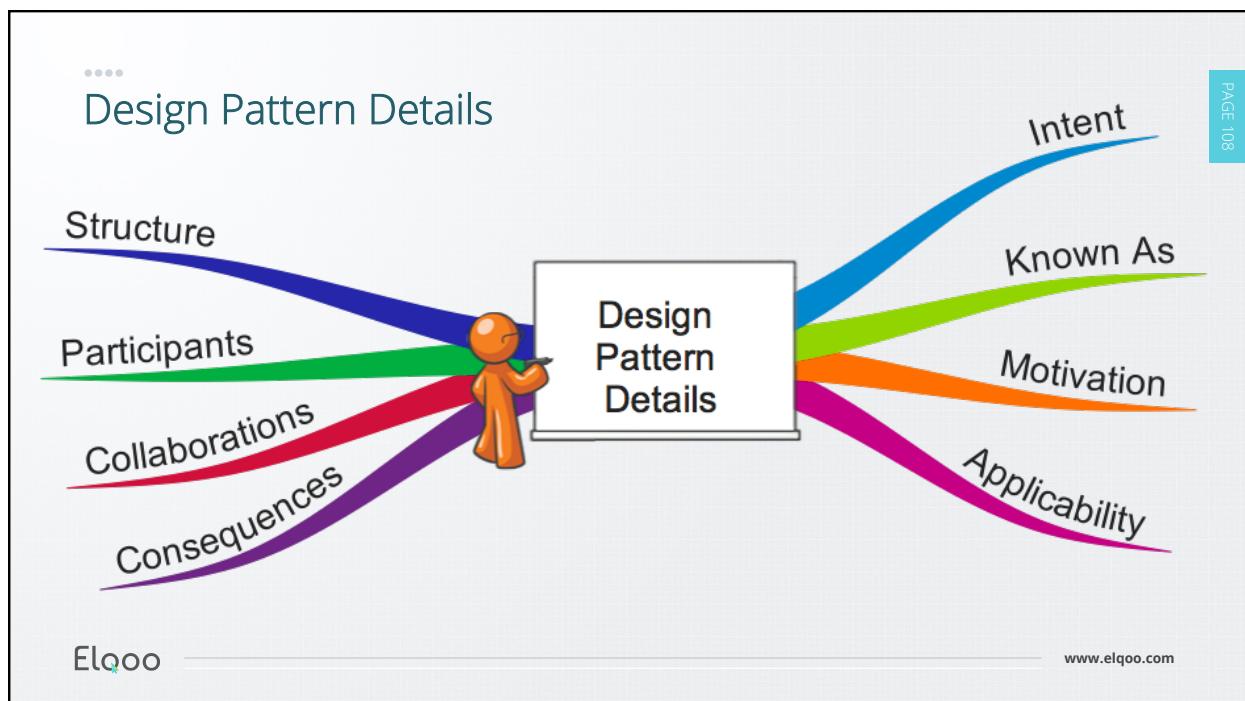
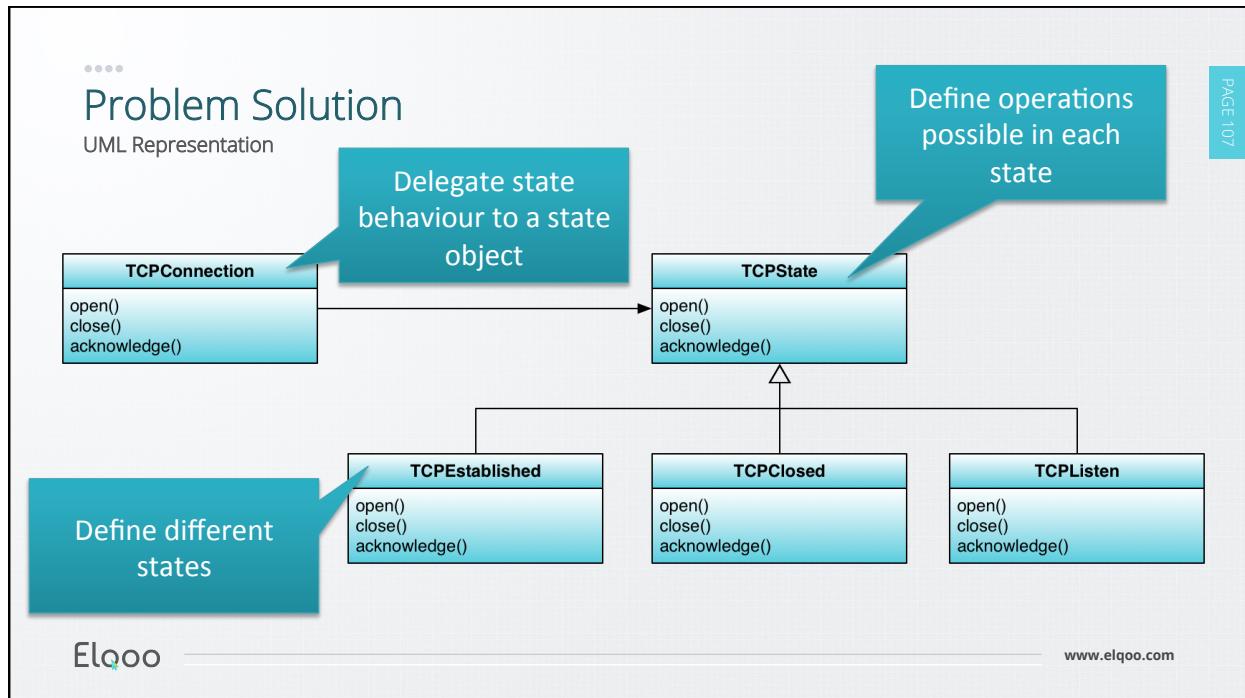
PAGE 106

- Has different **states**
 - Open, closed, acknowledge
- TCP Connection

- The **behavior** for the connection is **different in each state**
 - e.g. only an opened connection can be closed
- Should we keep the state and its implementation in the TCP Connection class itself?

Elqoo

www.elqoo.com



State Pattern

Intent and Known As

PAGE 109



Intent

Allow an object **to alter its behavior** when its internal **state changes**. The object will appear to change its class.

Known As

Object for states

Elqoo

www.elqoo.com

Apply State Pattern

Applicability

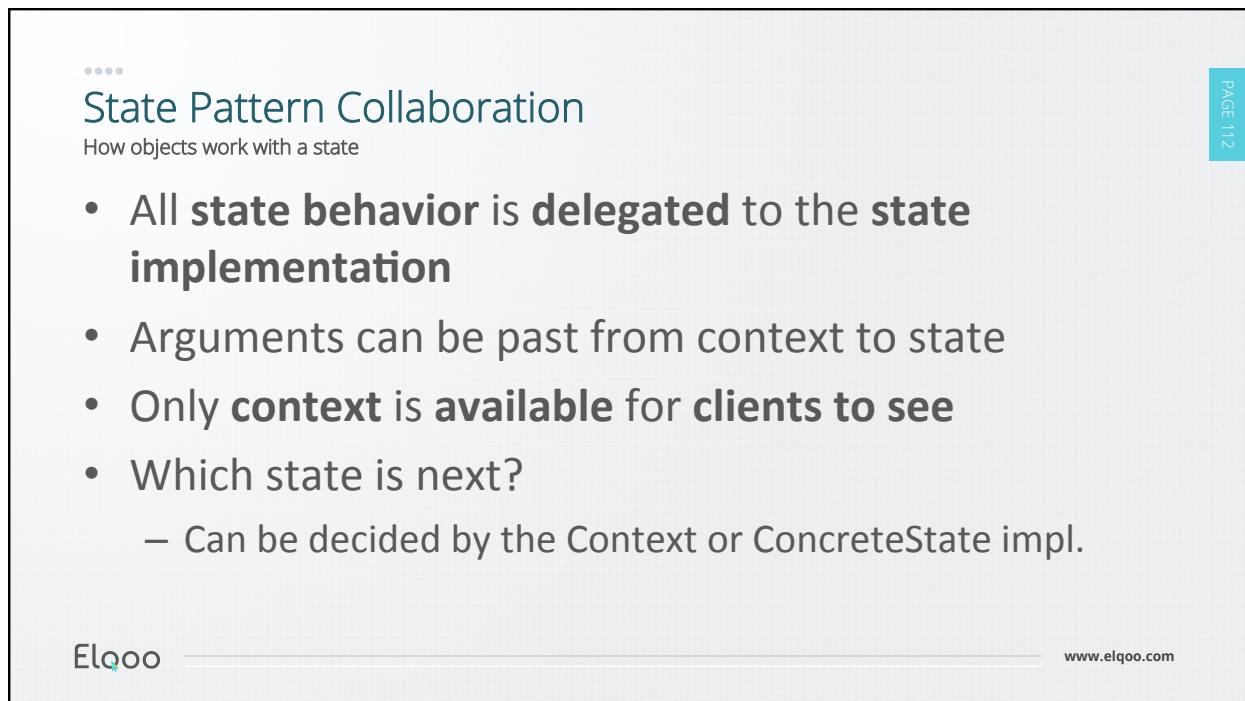
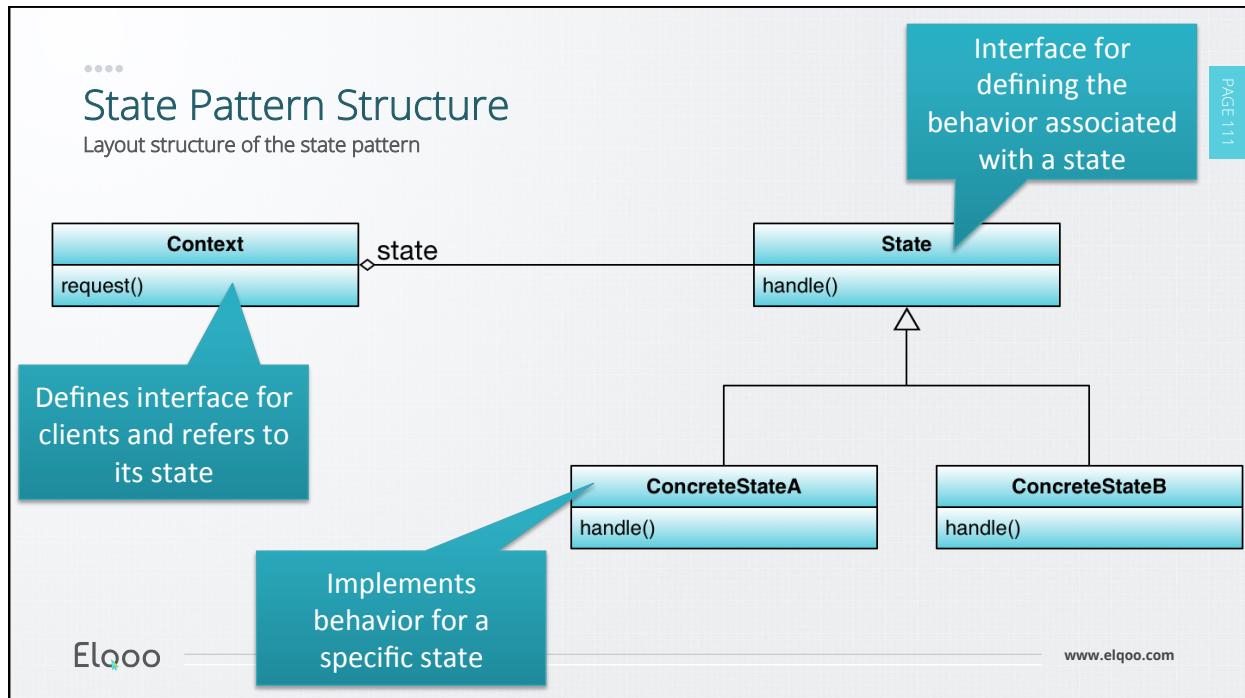
PAGE 110

- **Use**

- Object **behavior depends on object state**.
- **Avoid complex if-else-structures**
 - When state changes, simply change the state object
 - Implementation is done in the state object

Elqoo

www.elqoo.com



State Pattern Consequences

Considerations for the state pattern

PAGE 113

- **Benefits**

- State related **behavior** is **centralized**
- State **transitions** are **explicit** → state object must be changed
- State **changes** can be **atomic** → only one variable in the Context

- **Drawbacks**

- State objects must be **shareable**

Elqoo

www.elqoo.com

Conclusion

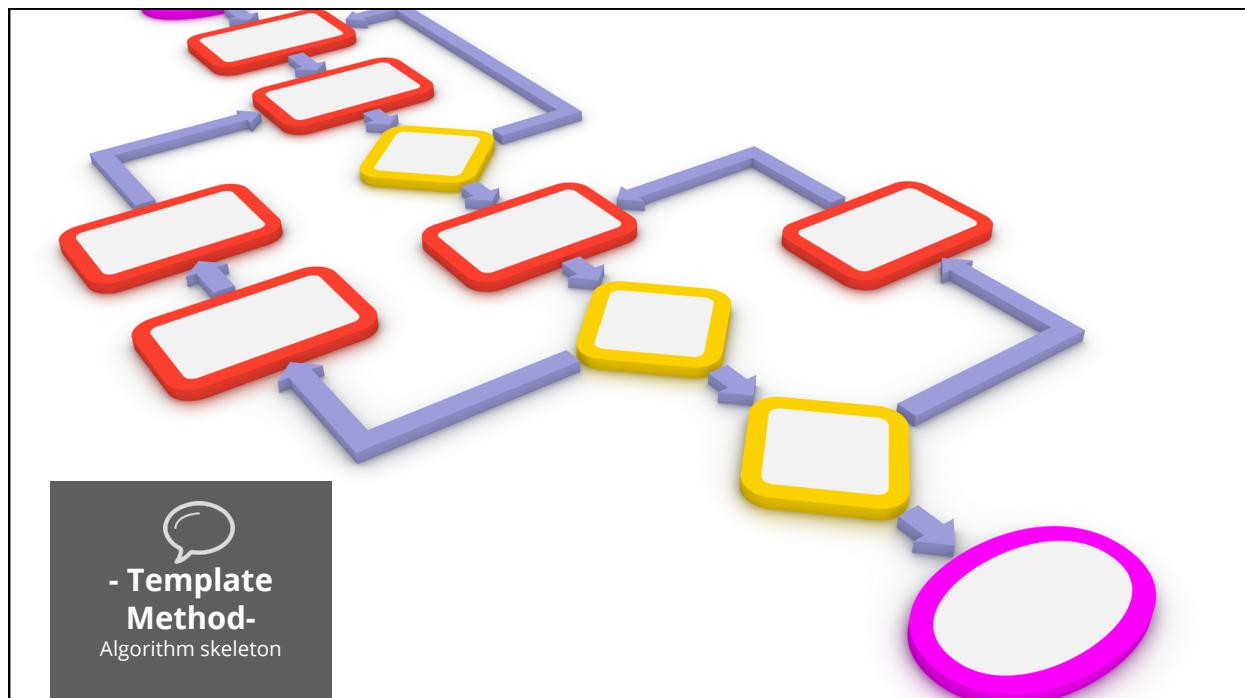
PAGE 114

- **State pattern is great**

- State behavior is **centralized**
- Reduce complexity due to state

Elqoo

www.elqoo.com



.....

Problem Statement

PAGE 116



SD

Brad

- Hello Brad
- Hi Suzy
- Can you help our developers to only focus on the new functionality to implement for our document framework?
- No problem



PM

Suzy

Elqoo

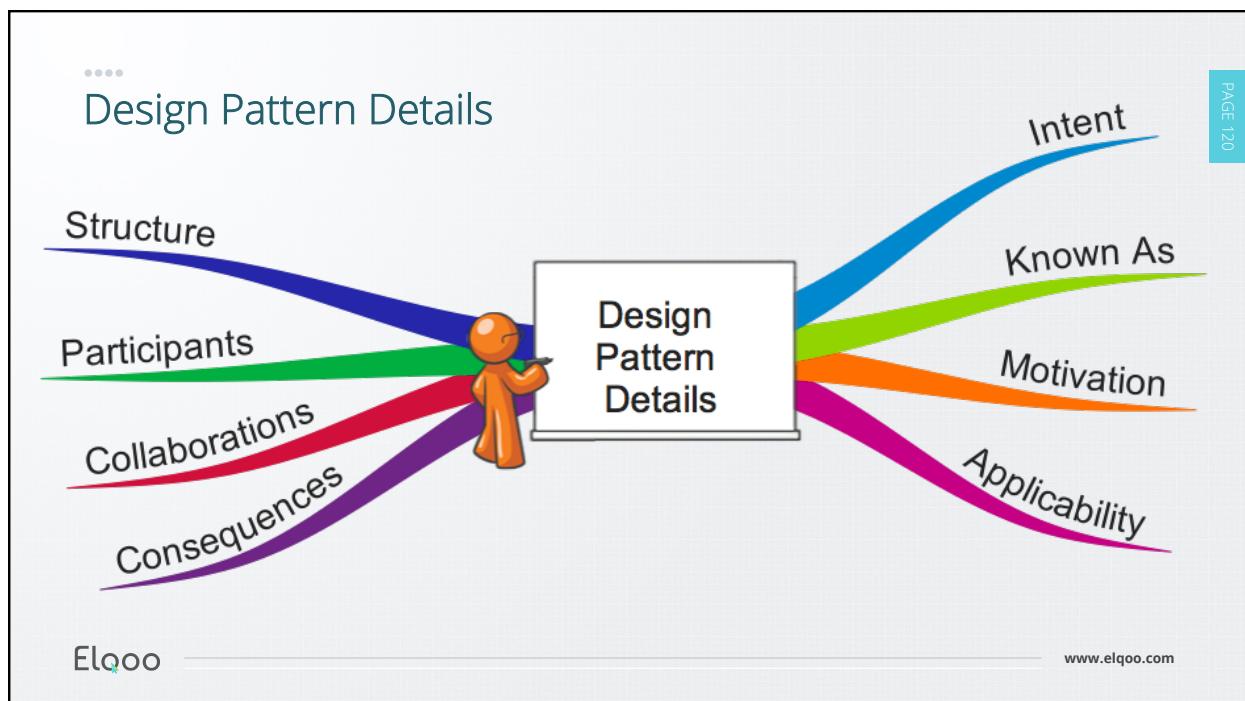
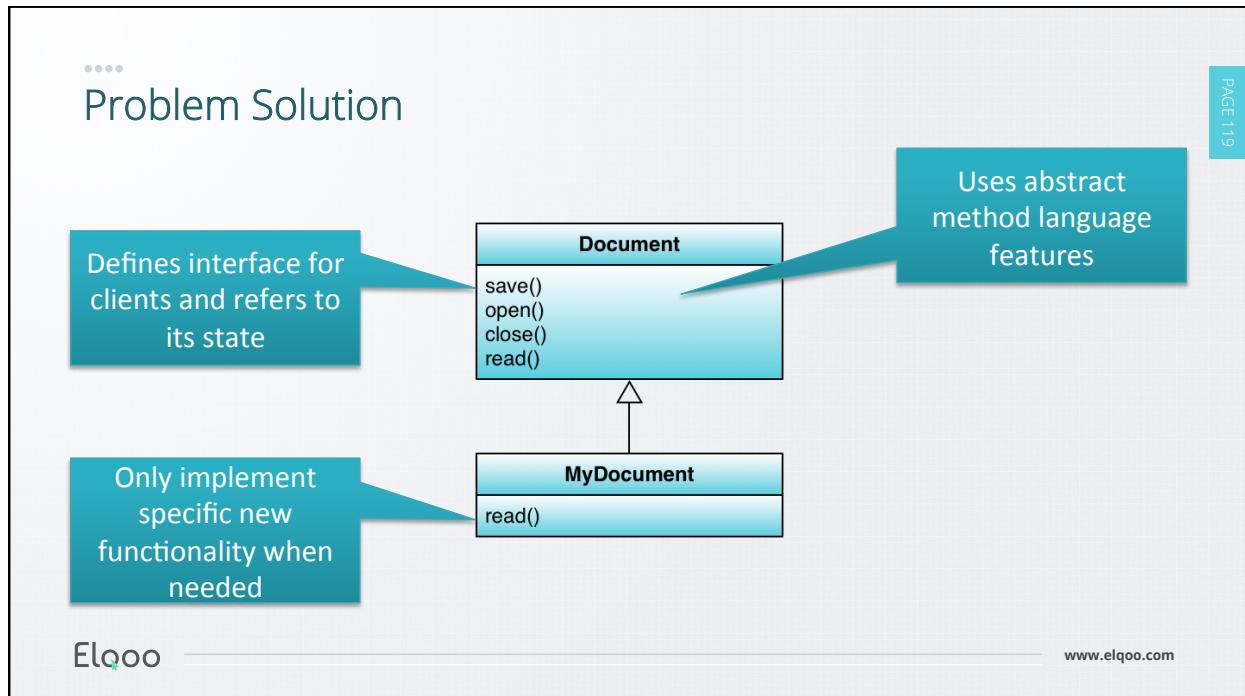
www.elqoo.com



..... Problem Statement Overview

PAGE 118

- Each **document** has the **same lifecycle**
 - Save
 - Open
 - Close
 - Read
- This lifecycle returns for each document
 - Do we need to implement it again?



Template Method Pattern

Intent and Known As

PAGE 121



Intent

Define the **skeleton of an algorithm** in an operation, deferring some steps to subclasses. Template Method lets subclasses redefine certain steps of an algorithm without changing the algorithm's structure.

Elqoo

www.elqoo.com

Apply Template Method Pattern

Applicability

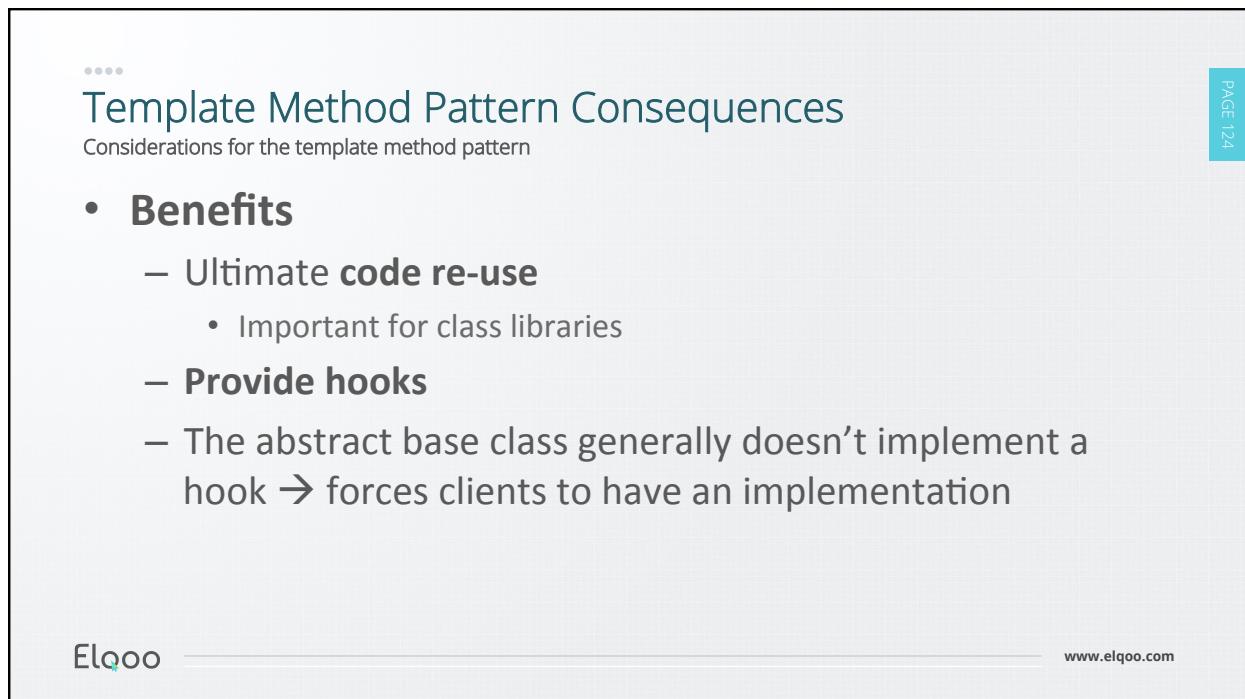
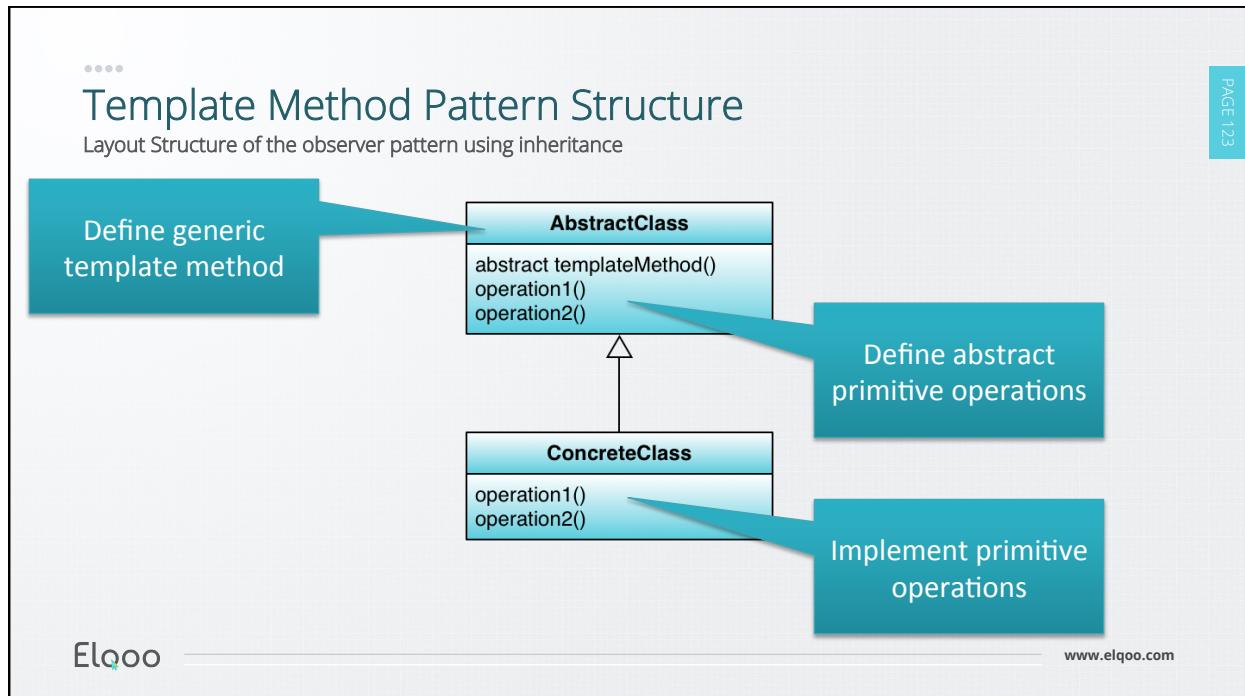
PAGE 122

- **Use**

- **Implement an algorithm once**
 - Subclasses can provide different implementation
- **Avoid code duplication**
- Define how a class should be extended
 - Implementing hooks

Elqoo

www.elqoo.com

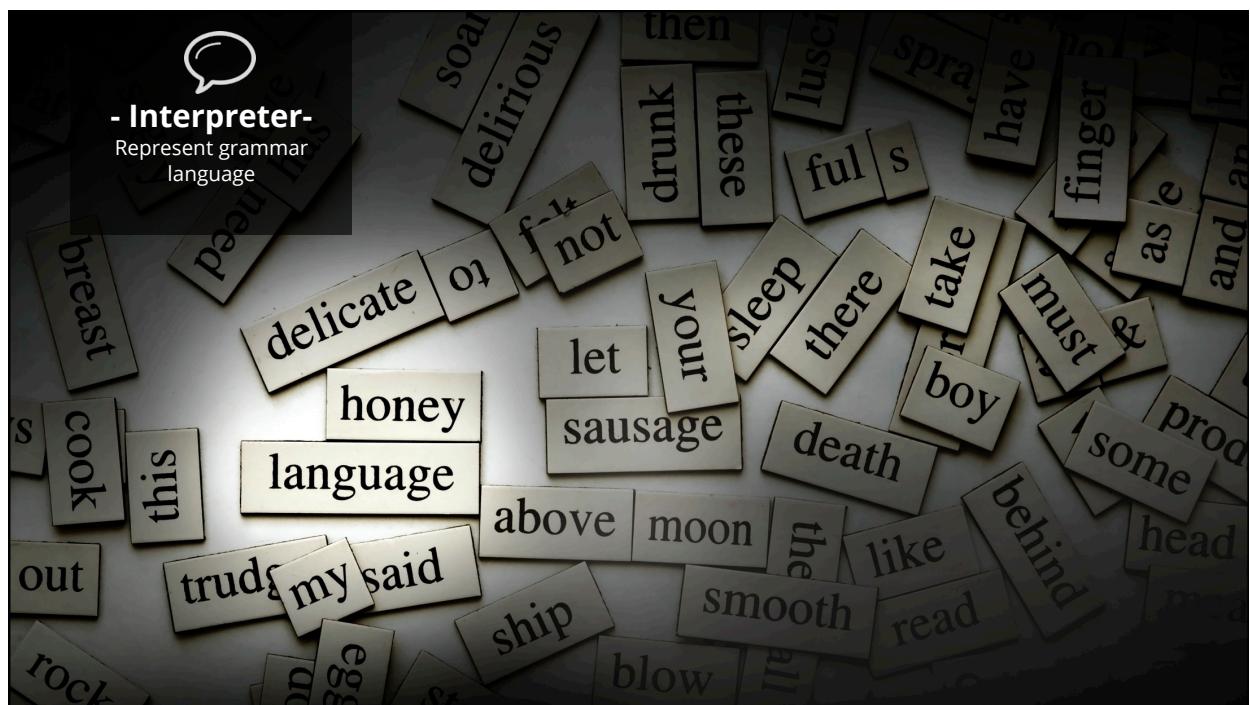


Conclusion

- **Template method pattern is great**
 - Heavily re-use code
 - Provide **hooks**

Elooo

www.elqoo.com



.....

Problem Statement

PAGE 127


SD
Brad

- Hello Brad
- Hi Suzy
- We have created a new calculation language and need a problem to help us.
- No problem


PM
Suzy

Elqoo

www.elqoo.com

.....

Problem Statement Overview (1)

The language definition

PAGE 128

```

expression ::= plus | minus | variable | number
plus ::= expression expression '+'
minus ::= expression expression '-'
variable ::= 'a' | 'b' | 'c' | ... | 'z'
digit = '0' | '1' | ... | '9'
number ::= digit | digit number

```

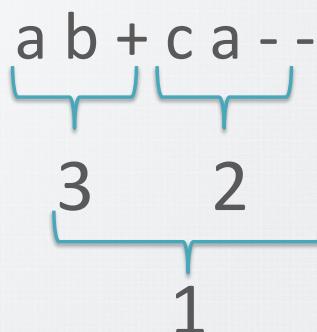
Elqoo

www.elqoo.com

Problem Statement Overview (2)

PAGE 129

Example:



$a = 1$
 $b = 2$
 $c = 3$

Elqoo

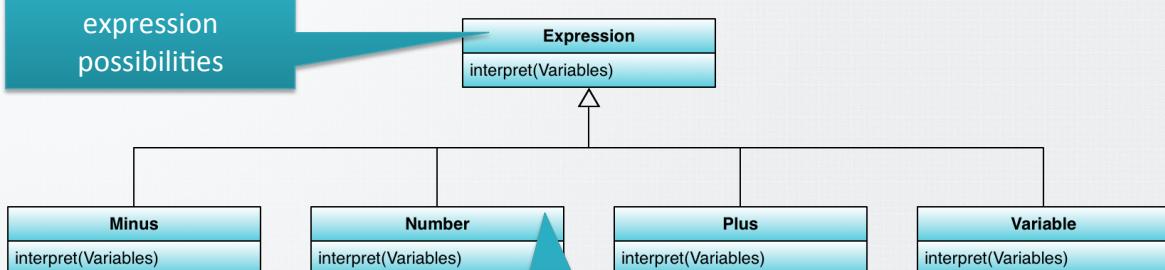
www.elqoo.com

Problem Solution

PAGE 130

UML language representation

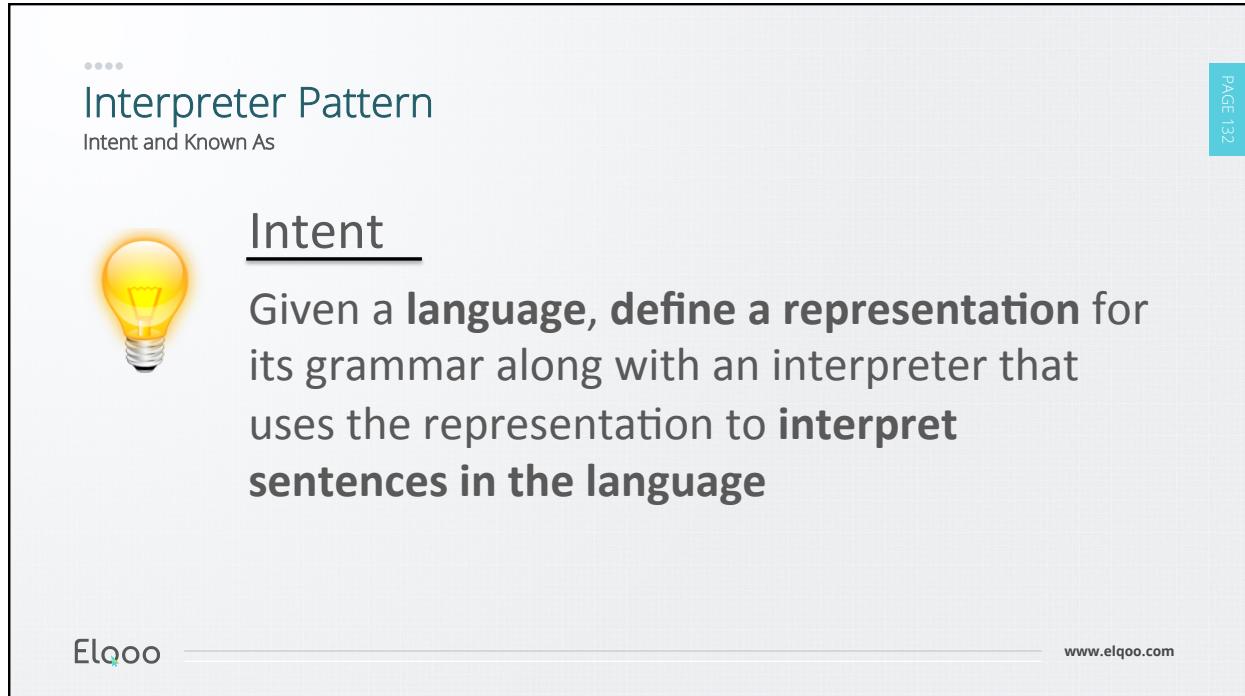
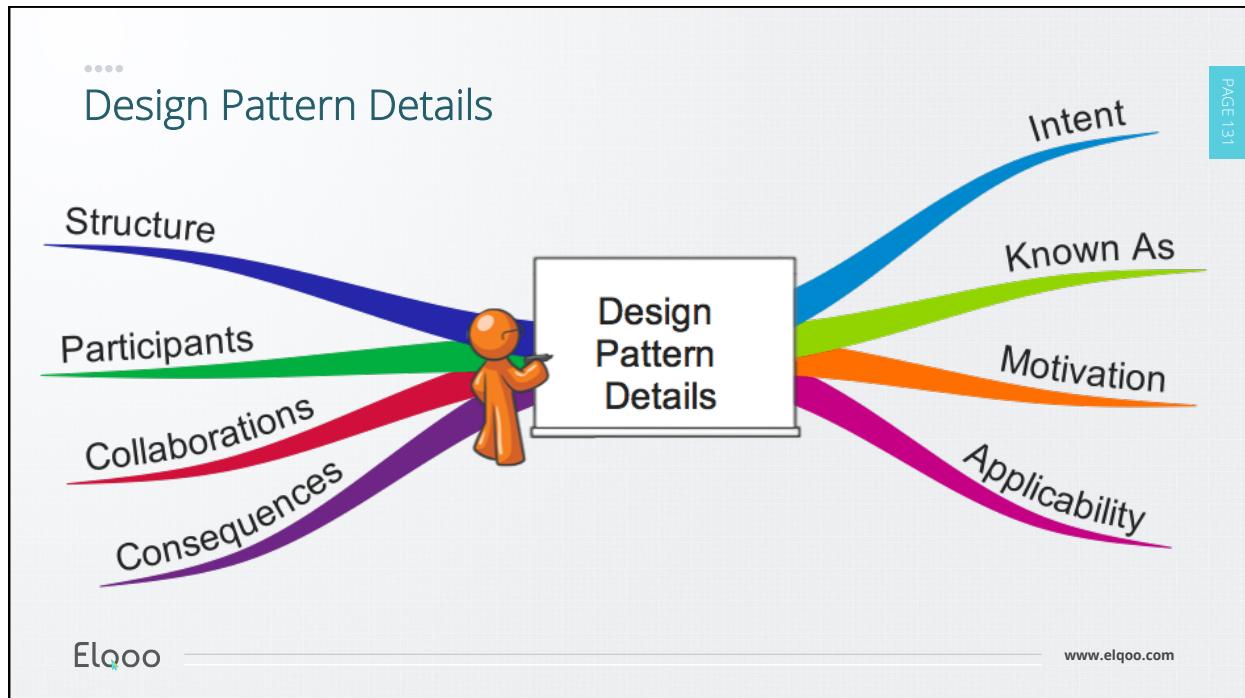
Base class for all
expression
possibilities



Each Base class is a
separate expression

Elqoo

www.elqoo.com



Apply Interpreter Pattern

Applicability

PAGE 133

- **Use**

- A language needs to be interpreted

- The language is simple

- Efficiency is not critical

- Note: languages are normally translated into a state machine

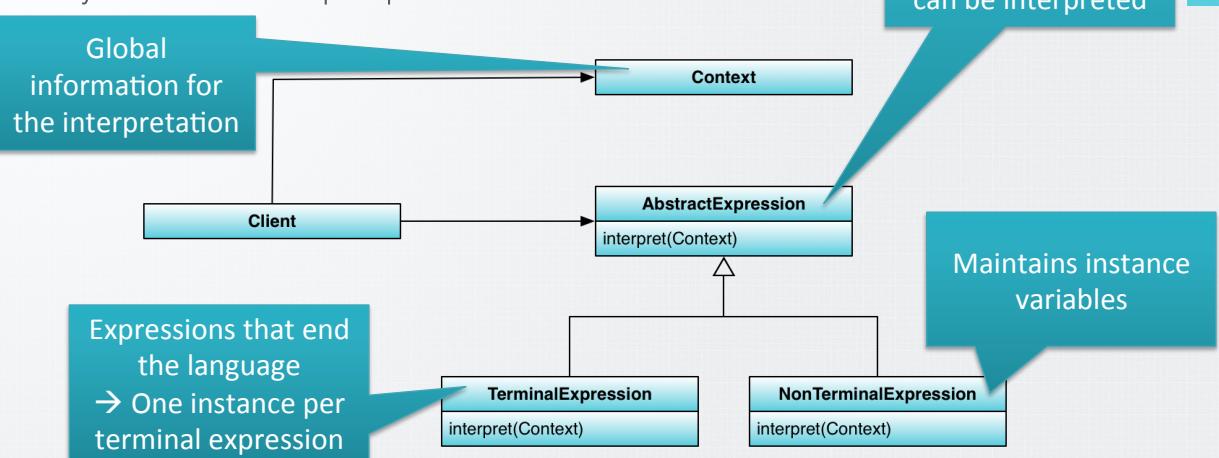
Elqoo

www.elqoo.com

Interpreter Pattern Structure

Layout Structure of the interpreter pattern

PAGE 134



Elqoo

www.elqoo.com

Interpreter Pattern: Collaborations

- **Client** is responsible for **building the syntax tree**
 - Representations of terminal & nonterminal expressions

Interpreter Pattern Consequences

Considerations for the interpreter pattern

- **Benefits**
 - **Easy to change or extend** the grammar
 - New interpretation expressions can be added easily
 - E.g. a pretty print
- **Drawbacks**
 - Hard for complex grammar

Conclusion

- Interpreter pattern is great
 - Simple syntax trees
 - Transform language into code objects