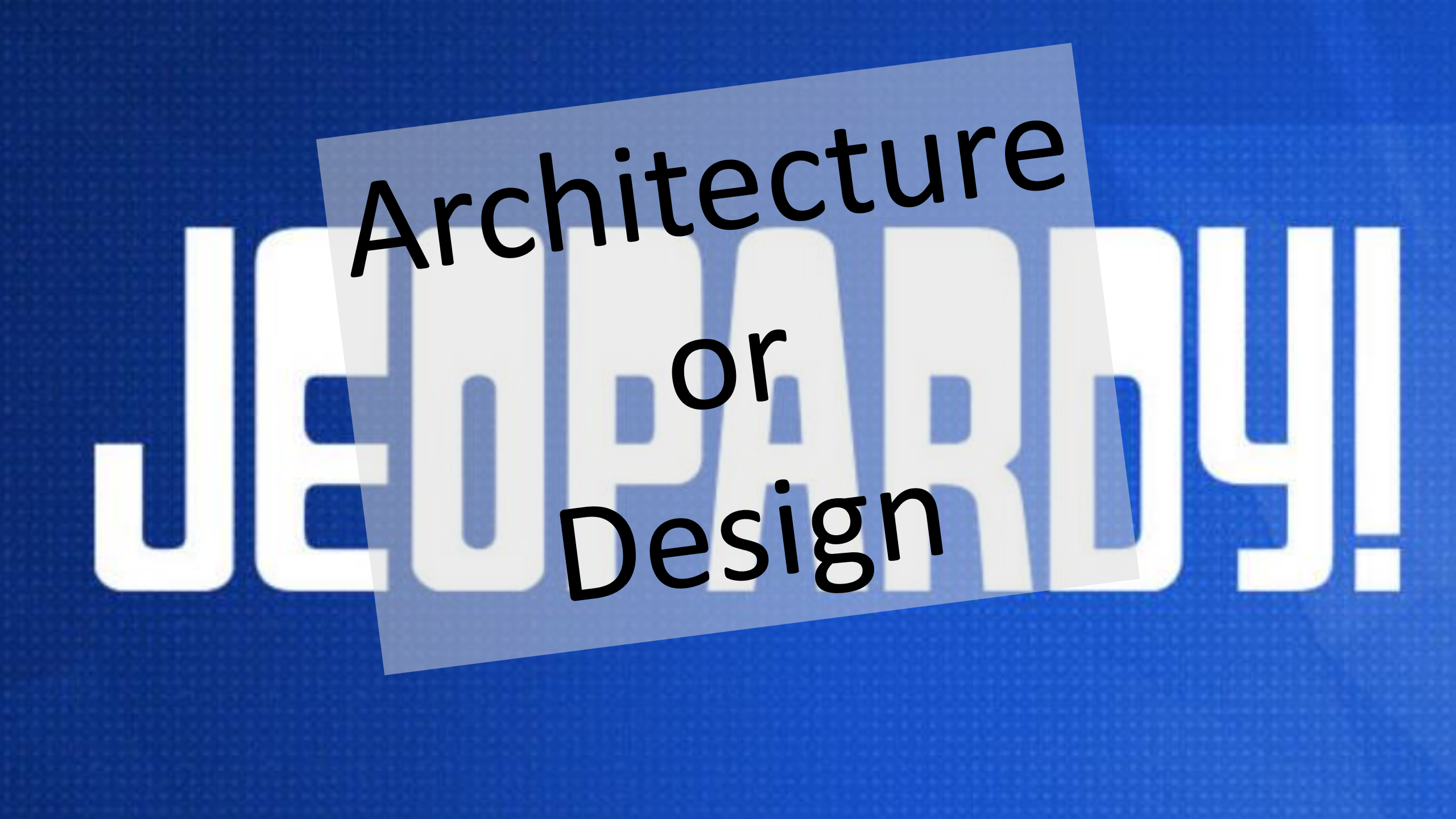

Architecture
or
Design
JEOPARDY!

“We want a GUI layer, an analytics layer, and a data storage layer.”

Architecture Design

“We want a GUI layer, an analytics layer, and a data storage layer.”

Architecture

Design

“We want a GUI layer, an analytics layer, and a data storage layer.”

Architecture

The statement refers to high level modules of the system, namely layers.

Design

“All new application need to extend the *Application* interface.”

Architecture Design

"All new application need to extend the *Application* interface."

Architecture

Design

“All new application need to extend the *Application* interface.”

Architecture

Design

The statement refers to a specific implementation detail.

It requires modifications at a class level.

“The application is available as a service deployed on cloud.”

Architecture Design

“The application is available as a service deployed on cloud.”

Architecture

Design

“The application is available as a service deployed on cloud.”

Architecture

Design

The statement refers to the deployment/execution environment of the application.

“We need a NoSQL database with a high availability rate.”

Architecture Design

“We need a NoSQL database with a high availability rate.”

Architecture

Design

“We need a NoSQL database with a high availability rate.”

The statement refers to a specific layer/tier of the system.
It also references a non-functional requirement. NoSQL may be a specific technology, but not a particular implementation.

Architecture

Design

“A method takes the type of an object as parameter and returns an instance of this type by calling the private constructor of the corresponding class.”

Architecture Design

“A method takes the type of an object as parameter and returns an instance of this type by calling the private constructor of the corresponding class.”

The statement refers to a specific implementation detail. It requires modifications at a class level. It also references a particular design pattern (Singleton)

Architecture

Design

“A method takes the type of an object as parameter and returns an instance of this type by calling the private constructor of the corresponding class.”

Architecture

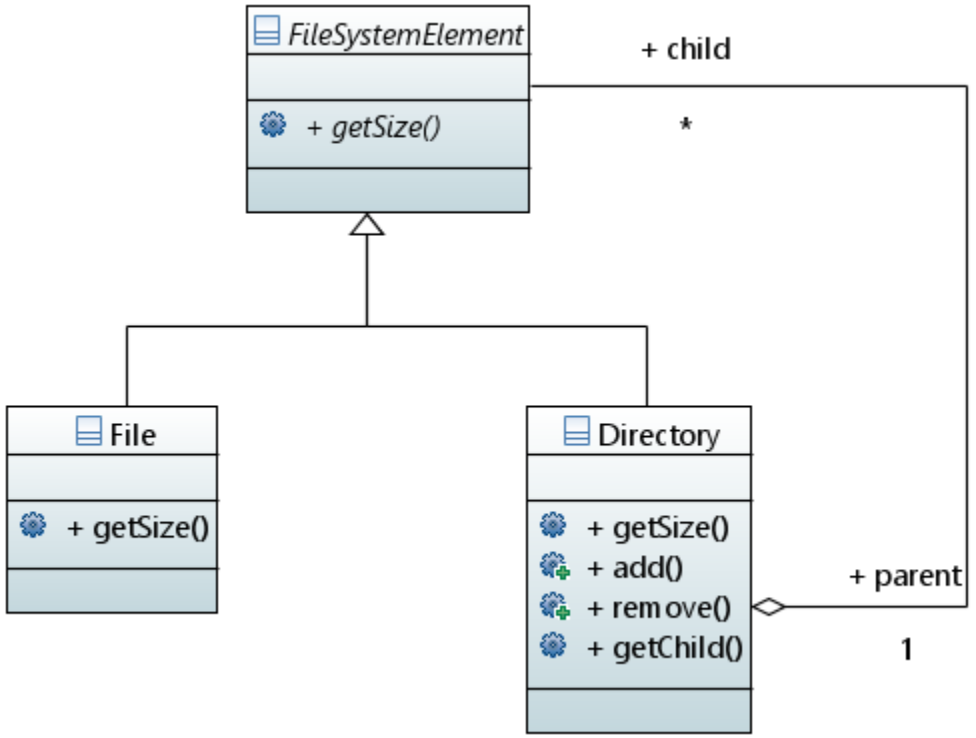
Design

Design Patterns

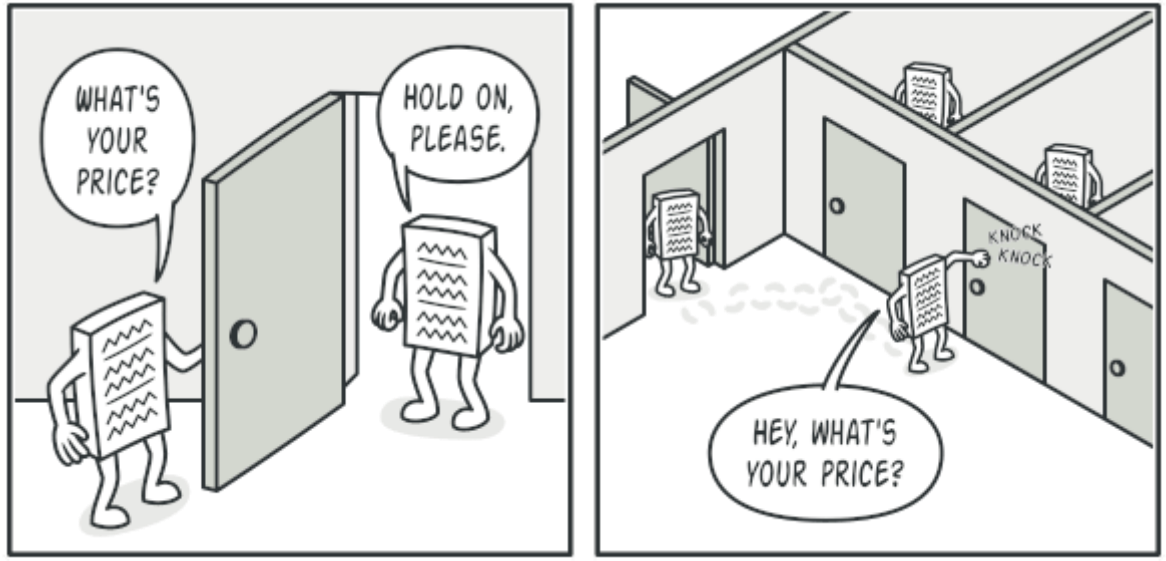
Design Patterns for $200

- You work for an accounting office and you develop a software to calculate the size of the entire file system. The `size` of a `Directory` is the sum of sizes of all `Files` in the directory.

Solution #1 (composite)



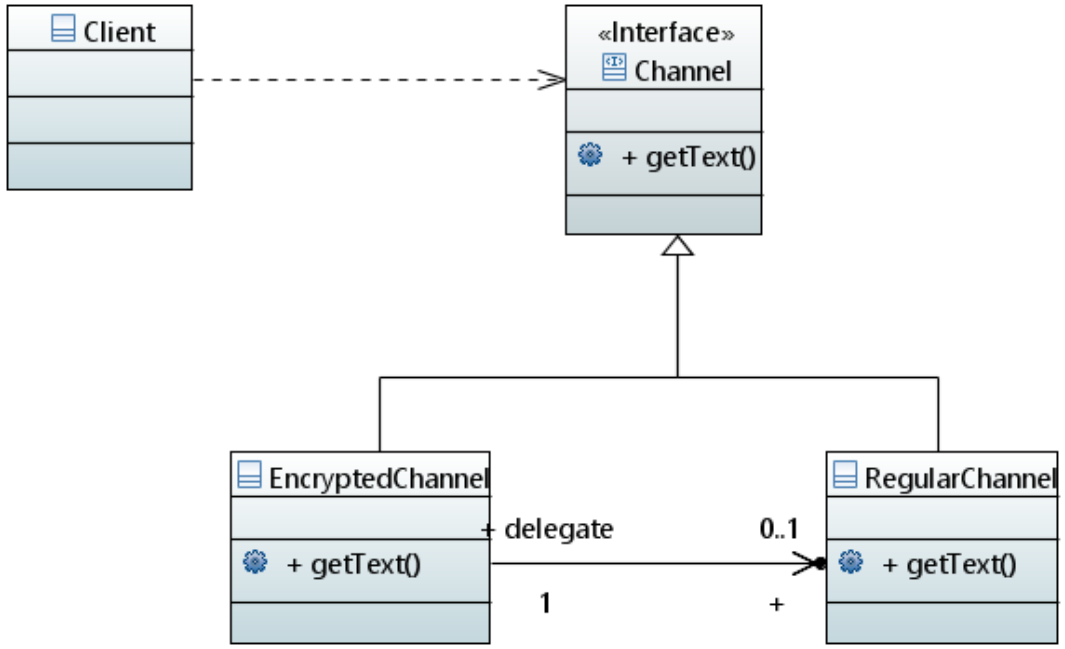
- You have objects that need to be treated similarly (i.e., add size).
- But some of them are *composite* containing other simpler ones. (Directories contain Files).
- State/Strategy also define hierarchies, but you do not have the composite relationship.



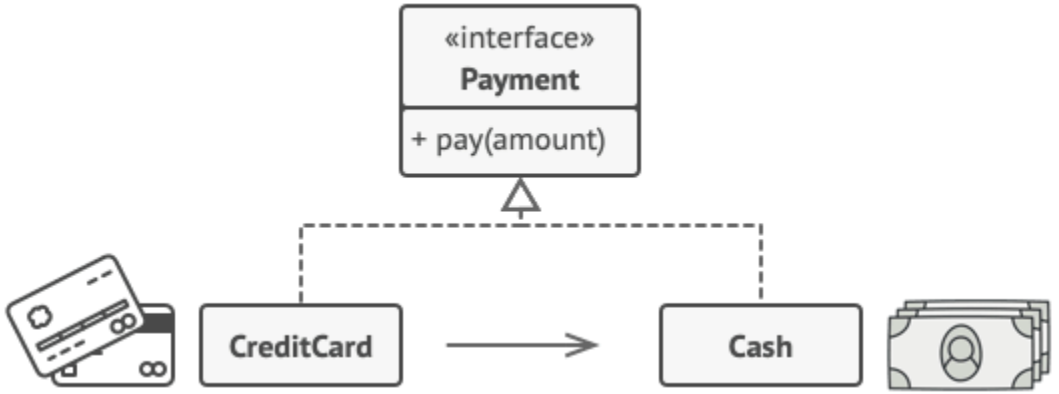
Design Patterns for $400

- We have a configuration, where we send and receive data over a network. At a given moment, the flow of input data changes to encrypted. How do we need to change the client code?

Solution #2 (proxy)



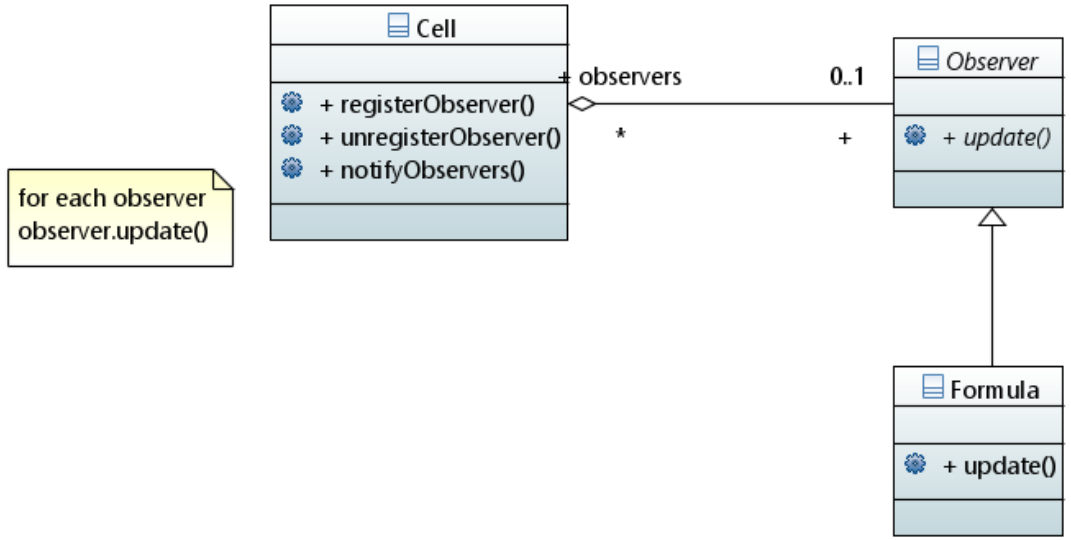
- You need to control the access to a given object (RegularChannel).
- You provide a placeholder (Channel) that dynamically determines how the object will be accessed.



Design Patterns for $600

- You develop a spreadsheet application that allows to automatically calculate the contents of cells using functions, which depend on other cells.

Solution #3 (observer)

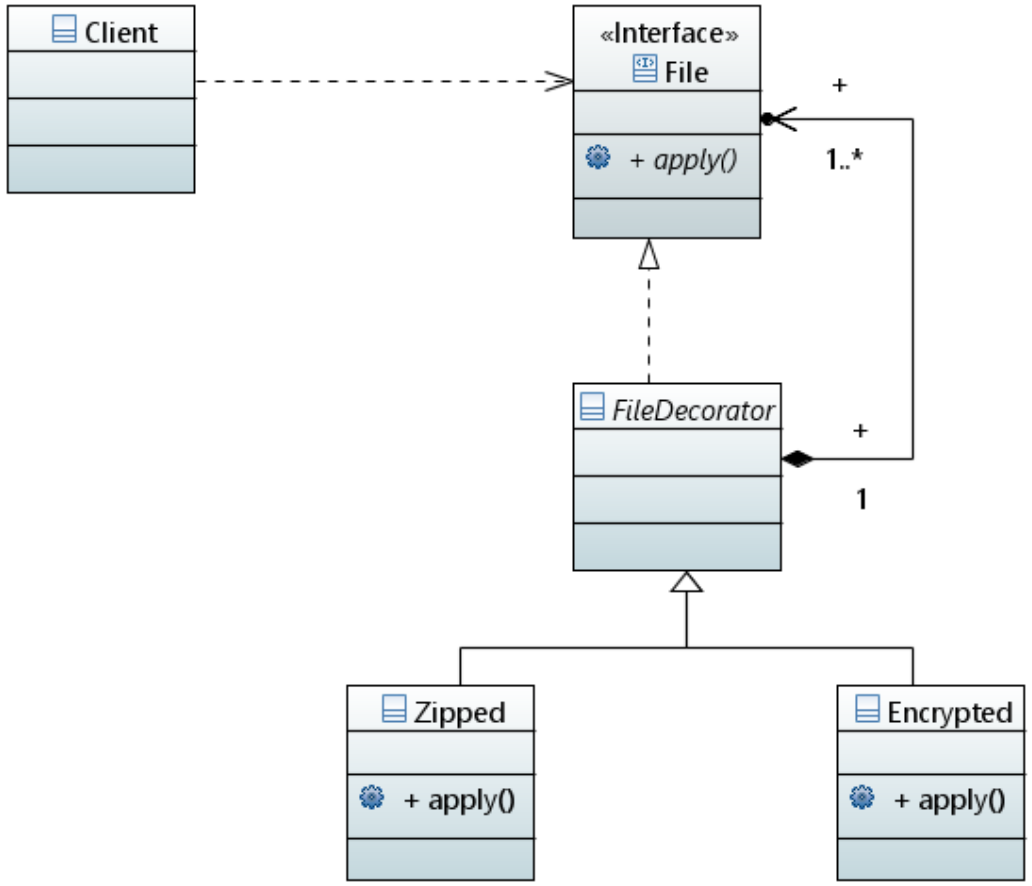


- An object (Cell) needs to change its state based on what happens to another object (Formula).
- The object that changes is the subject and the object that needs to change is an observer that subscribes to the subject.

Design Patterns for $800

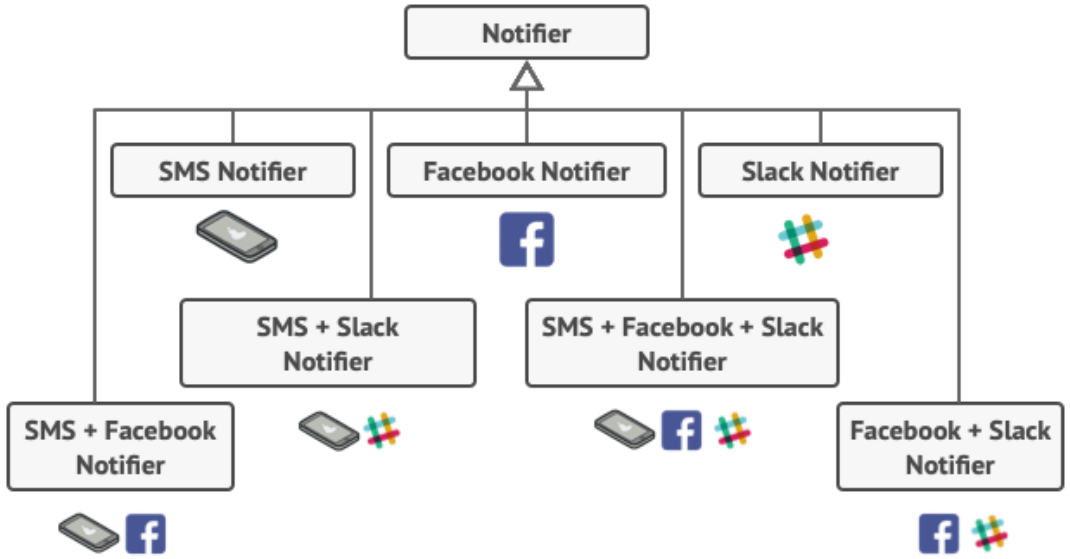
- You wish to develop a file reader that is capable of reading a file, which may be (a) compressed, (b) encrypted, (c) compressed an encrypted, or (d) encrypted, then compressed, and then encrypted again.

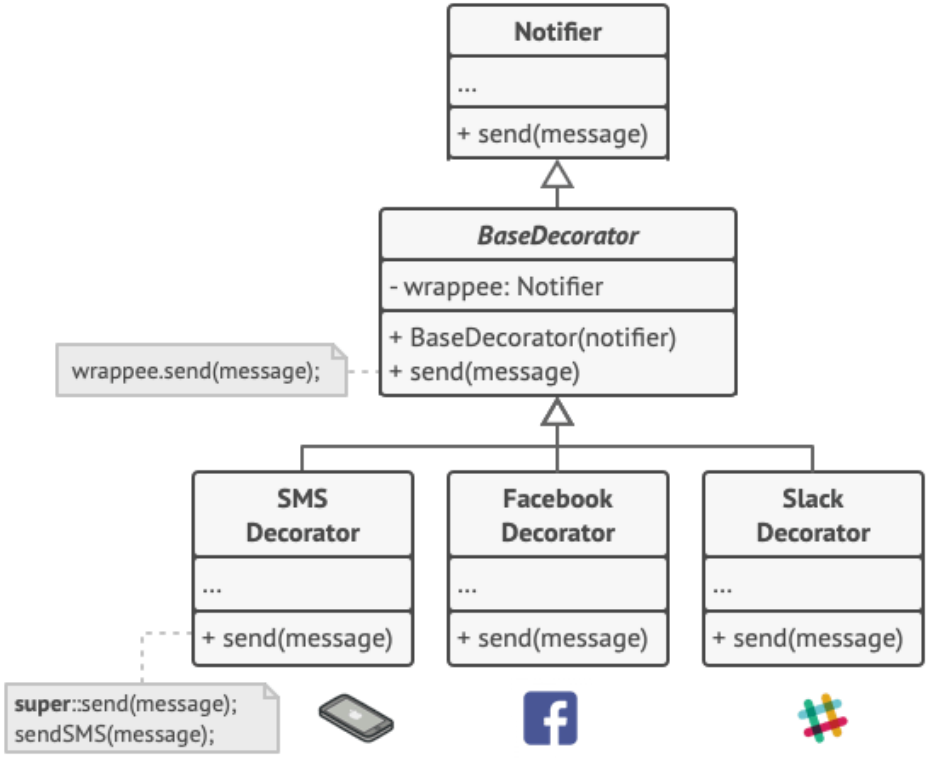
Solution #4 (decorator)



- You have objects that may have different behaviors or states (such as Zipped or Encrypted).
- On top of that, you want these states to be combined in any possible way.
- State/Strategy would require you to create subclasses for all possible combinations and do not account for future extensions.

Solution #4 (decorator)

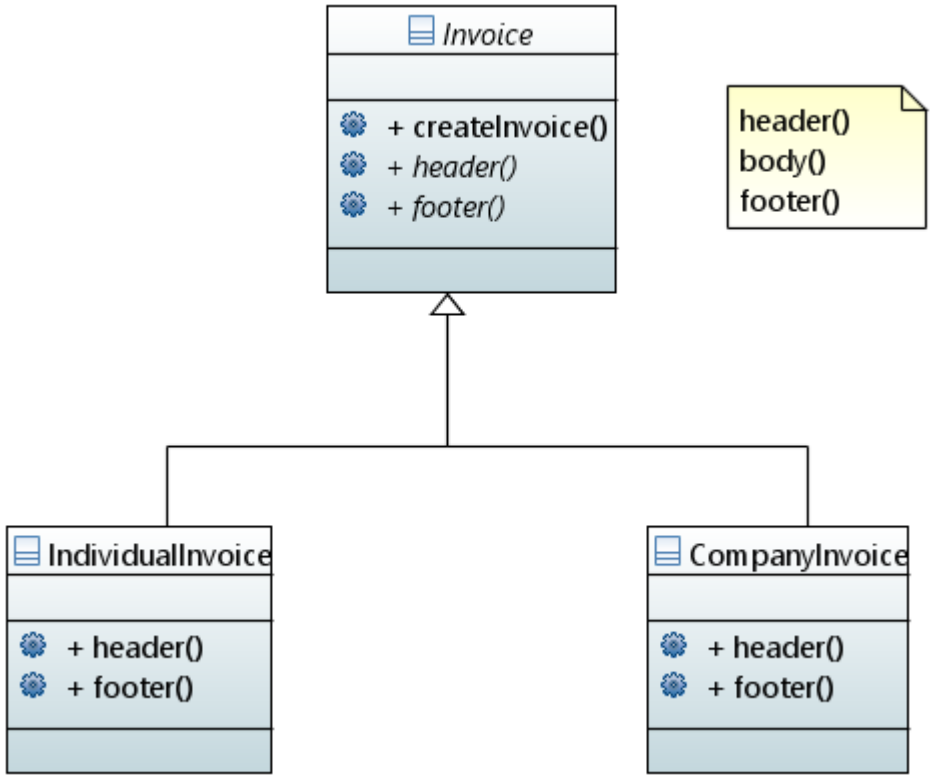




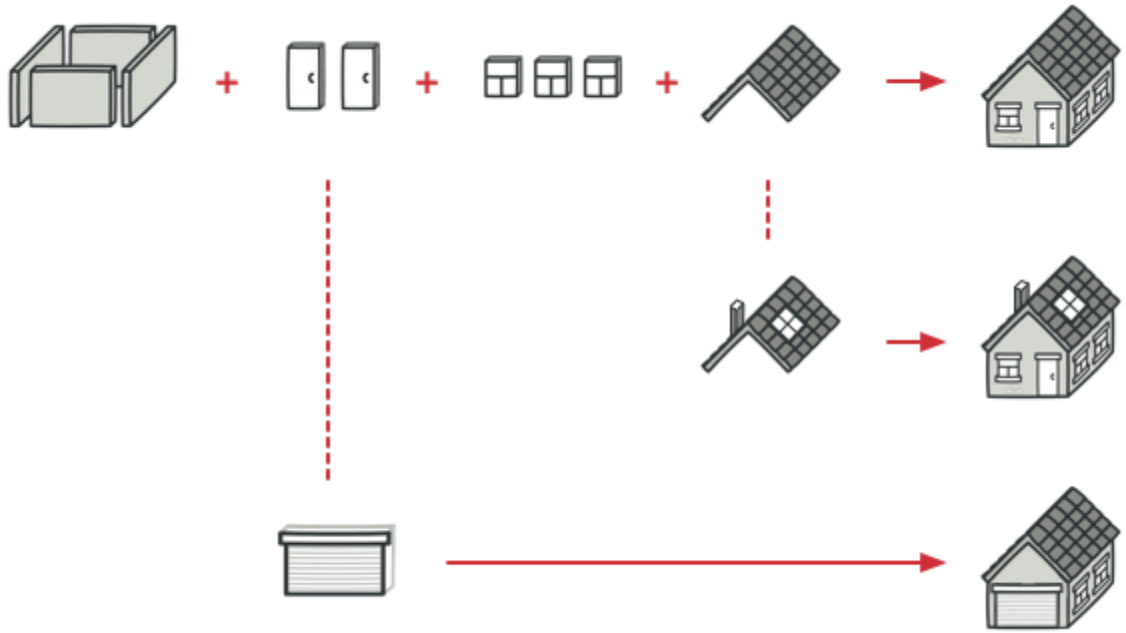
Design Patterns for $1000

- You have a system that prints invoices, one for individuals and an other for companies, which differ between each other on the header and the footer of the page. The content in the body of the invoice is a list of all elements, their prices and the total.

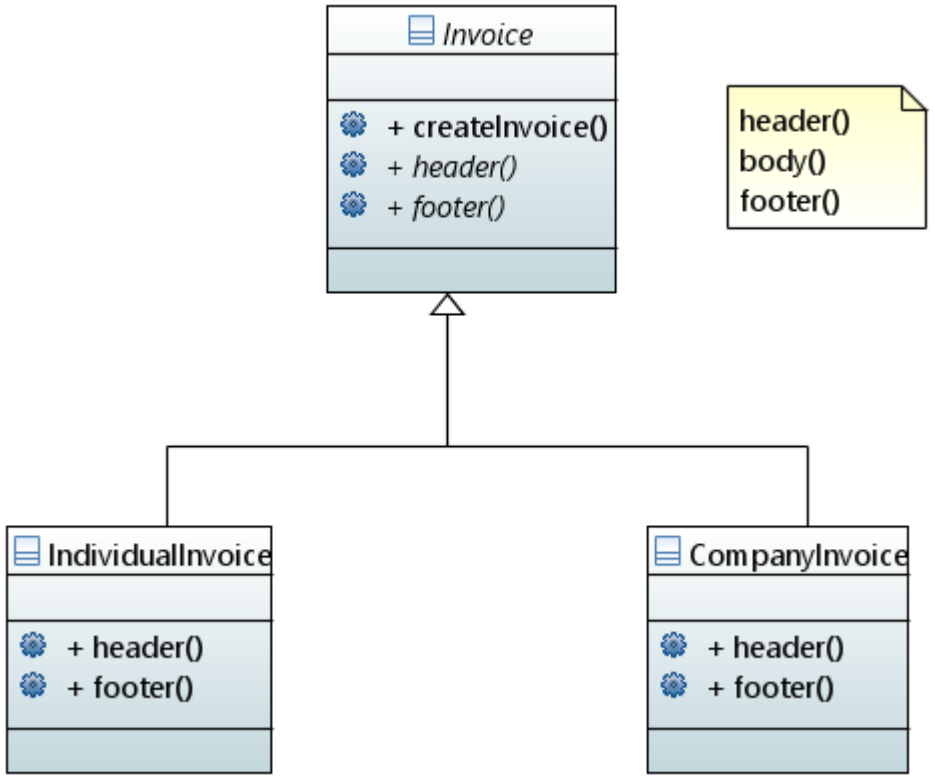
Solution #5 (template method)



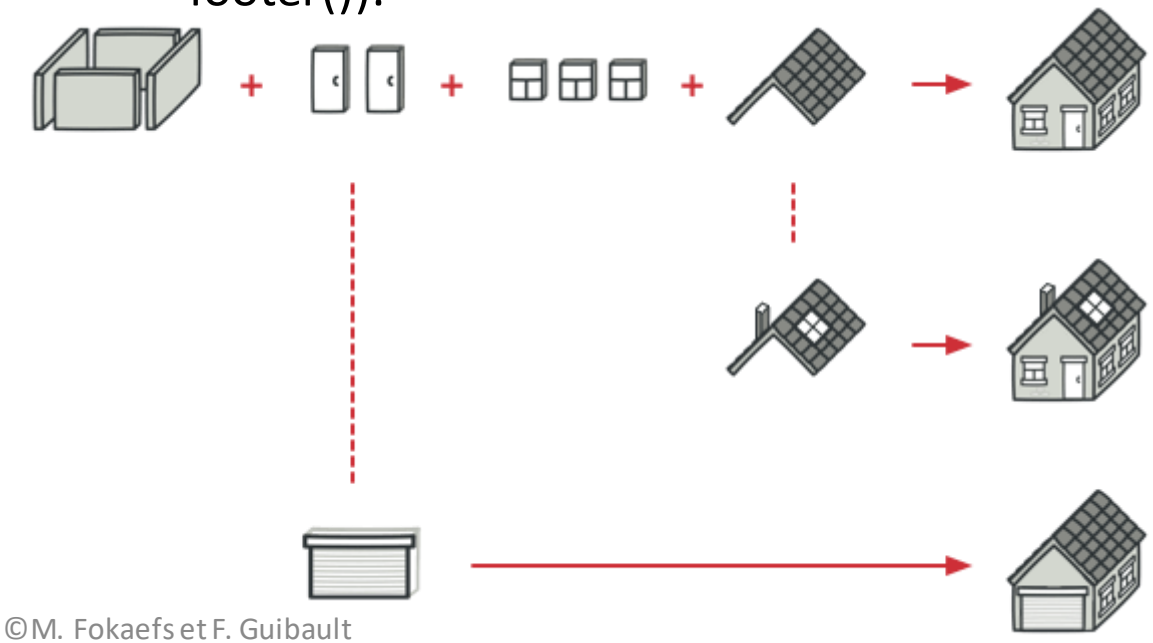
- You have a complex behavior that changes in different scenarios.
- However, certain parts of the behavior remain the same regardless of the scenario.



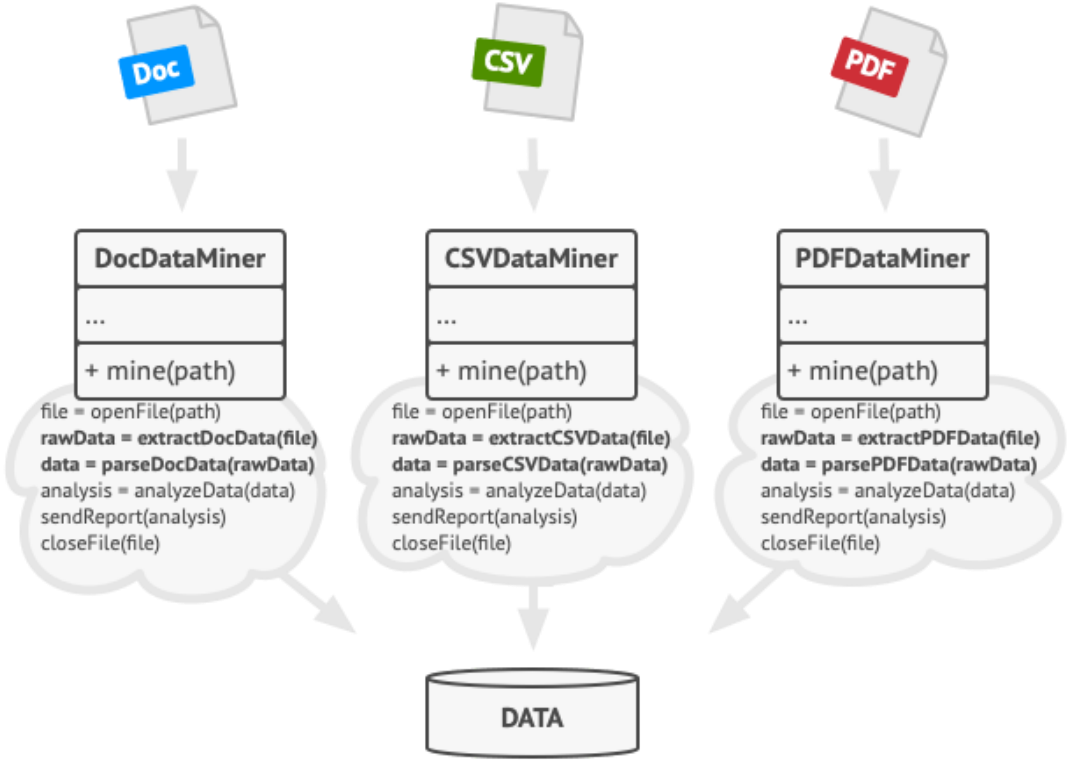
Solution #5 (template method)

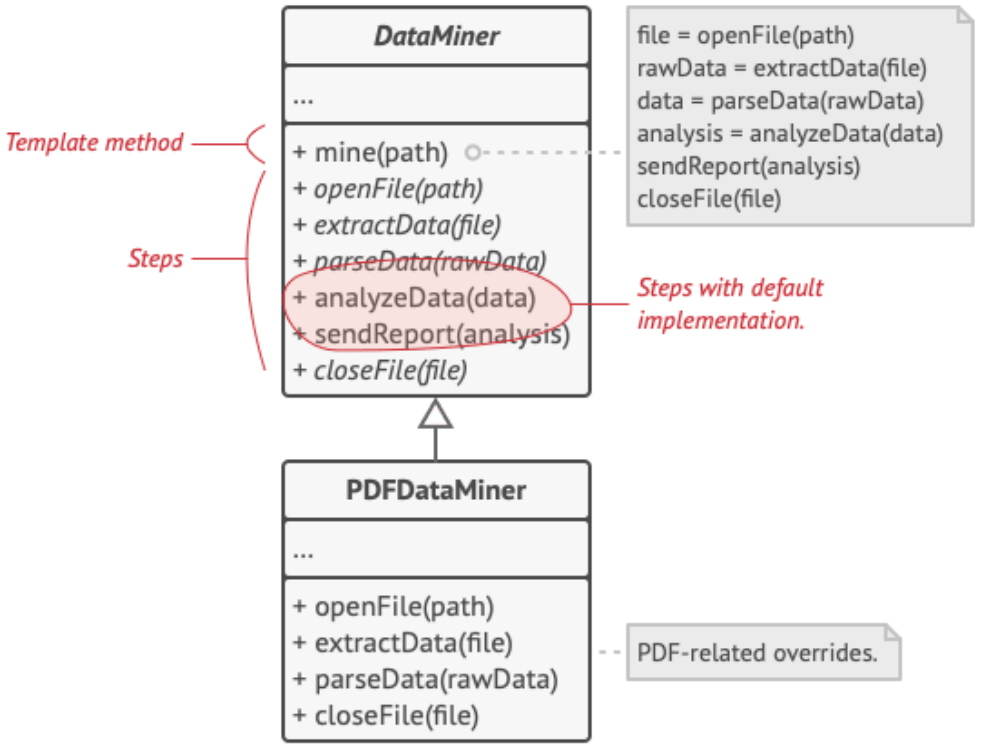


- You have a complex behavior (createInvoice()) that changes in different scenarios (IndividualInvoice, CompanyInvoice).
- However, certain parts of the behavior remain the same regardless of the scenario (header(), footer()).



Solution #5 (template method)

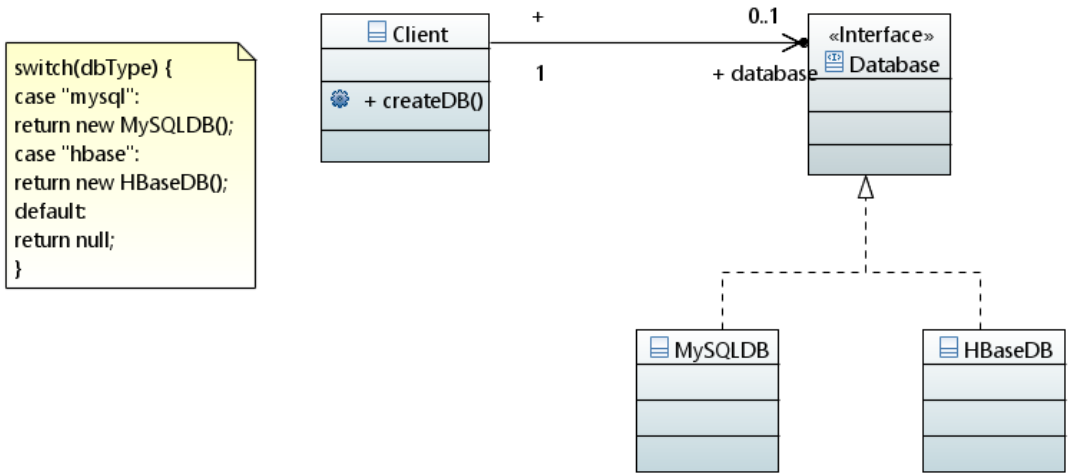




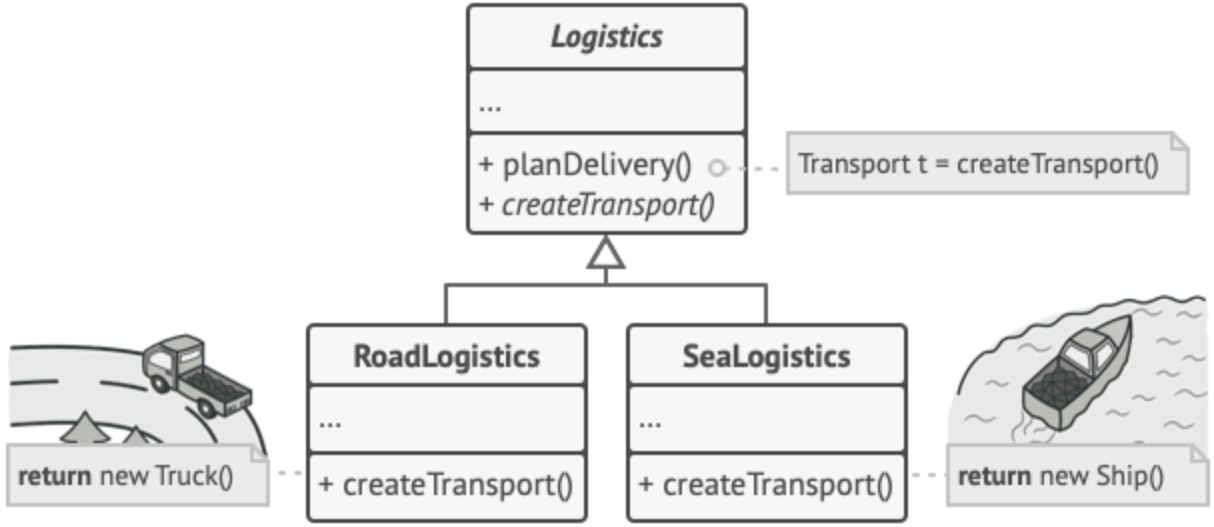
Design Patterns for $2000

- You have multiple databases, each of which having their own initialization, and you wish that your own application will be able to construct and use each of them in an interchangeable way (db1, db2, …) at the user's choice.

Solution #6 (factory method)



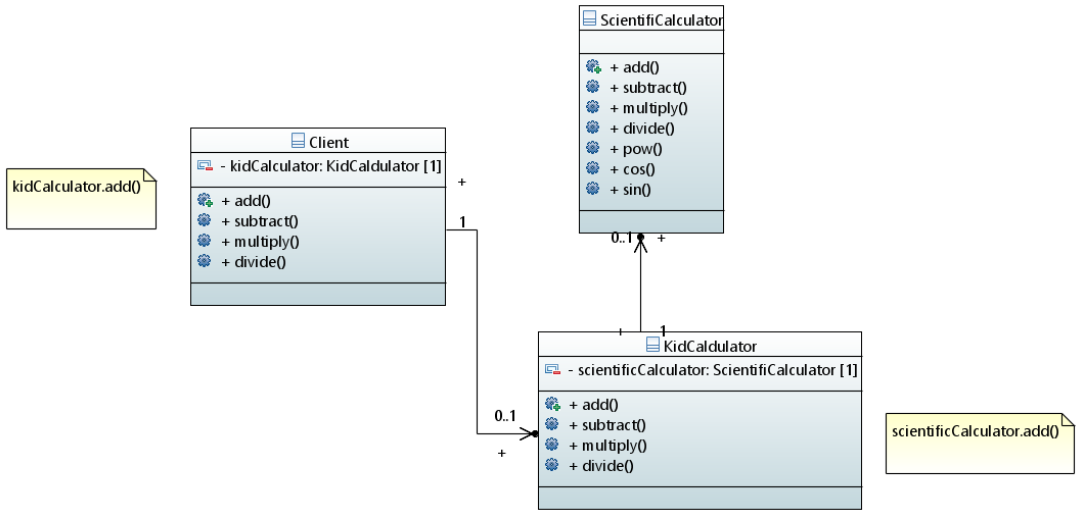
- You have an object (Database) with different implementations (MySQLDB, HBaseDB).
- An implementation is created only upon the choice of a client.



DAILY DOUBLE! ($4000)

- You are hired by MathWorks (the developers of Matlab). The company would like to provide software to elementary schools, byt they want to reuse existing code. Your are given code for a scientific calculator and your task is to develop a calculator for children with just the basic operations.

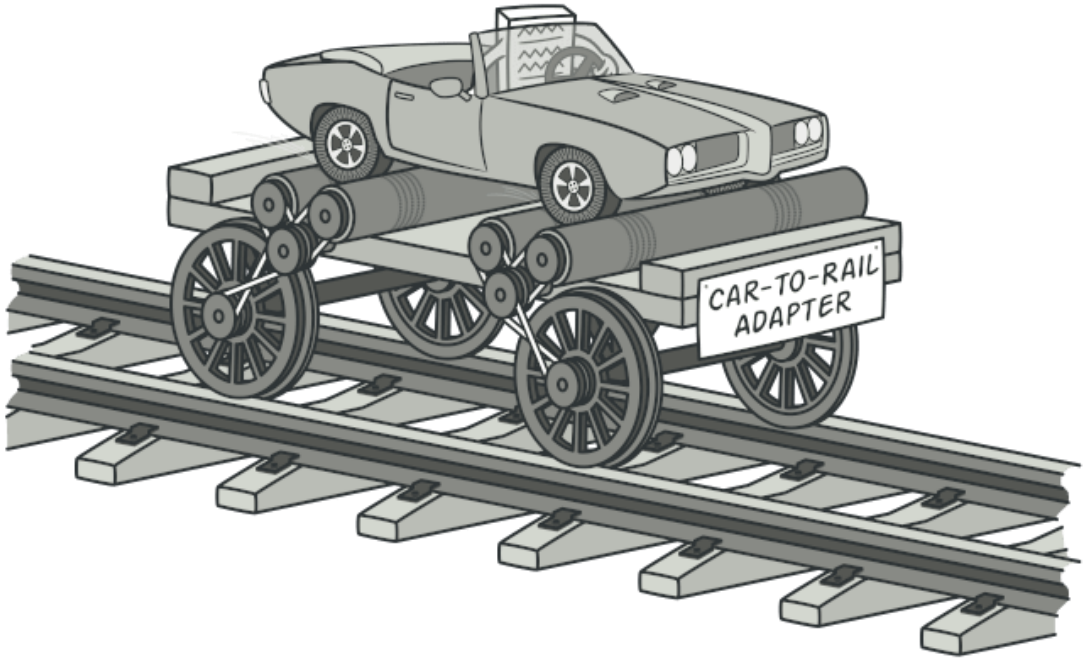
Solution #7 (adapter or façade)



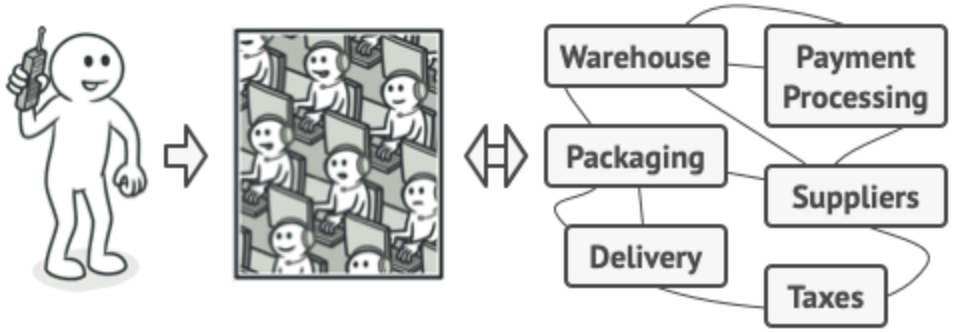
- You have a complex interface (ScientificCalculator) and an object (Client) that cannot understand it or most of it.
- Adapter translates one interface into another understood by the client.
- Façade simplifies a complex interface or merges multiple ones to publish only certain operations.

Solution #7 (adapter or façade)

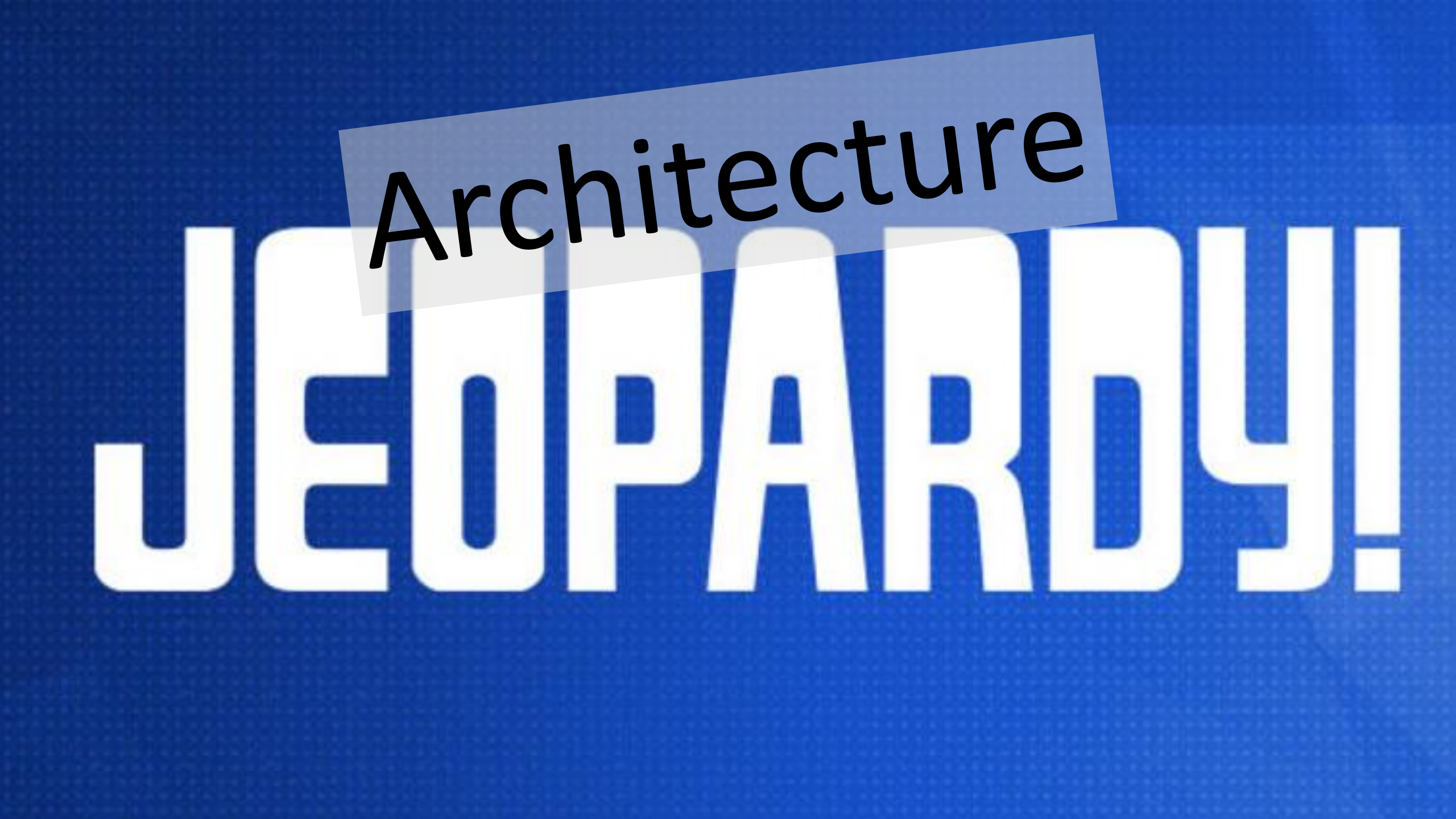
Adapter



Façade



Architecture



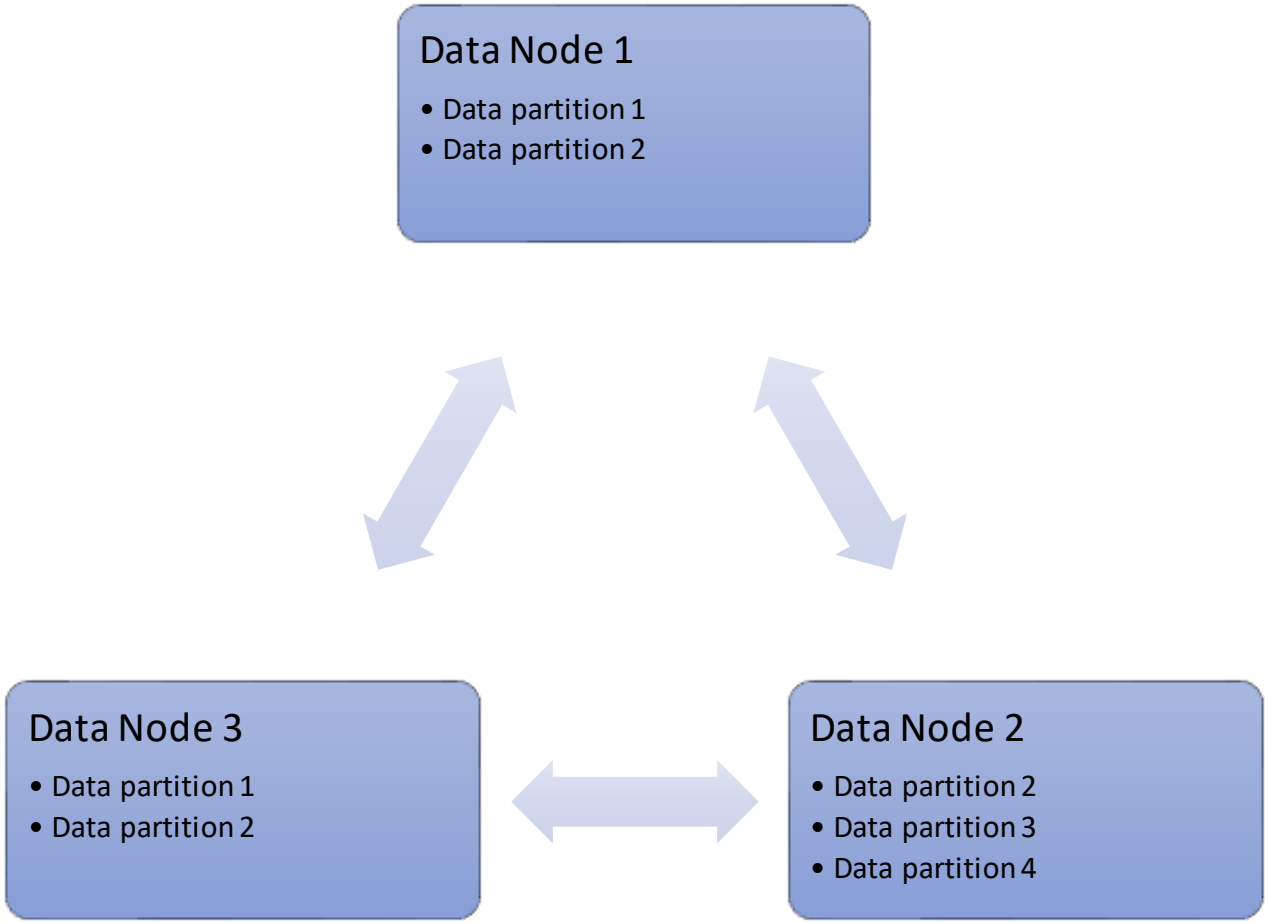
Software Architecture for $200

- We search for a distributed data system with high fault tolerance and with high availability. The database contains numerous replicas and the nodes communicate with each other to identify errors.

Software Architecture for $200

- We search for a distributed data system with high fault tolerance and with high availability. The database contains numerous replicas and the nodes communicate with each other to identify errors.
- What is… “peer-to-peer”
- Why? There is no central entity that controls the rest of the system. The nodes communicate with each other, among others to maintain their reliability and availability.
- The system is the NoSQL database Apache Cassandra.

Software Architecture for $200



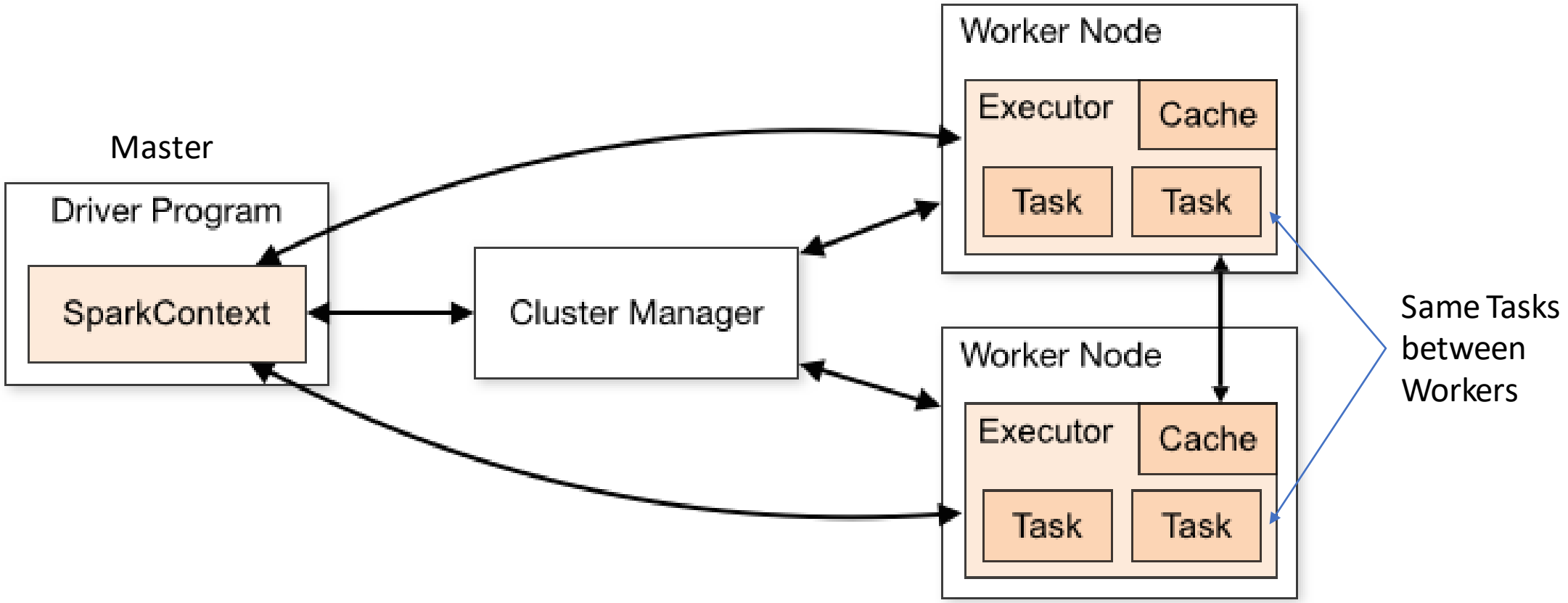
Software Architecture for $400

- We search for a data analytics system. An analytics task is submitted and it is replicated on multiple nodes. Each node works on a slice of the data. The results are assembled and aggregated before being returned.

Software Architecture for $400

- We search for a data analytics system. An analytics task is submitted and it is replicated on multiple nodes. Each node works on a slice of the data. The results are assembled and aggregated before being returned.
- What is… “master-slave”
- Why? There is a single task being replicated and executed in parallel by multiple workers. The fact that the task is being applied on different partitions of data is this that produces meaningful, and not repeated results. If the data partition was one and the tasks different we would have talked about Blackboard or pipe-filer.
- MapReduce: Hadoop, Spark…

Software Architecture for $400



Software Architecture for $600

- We search for a system to store and analyse tax returns. The computation and storage capacity is not a problem and the number of tax returns is roughly predicted.

Software Architecture for $600

- We search for a system to store and analyse tax returns. The computation and storage capacity is not a problem and the number of tax returns is roughly predicted.
- What is… “monolithic”
- Why? Security is of the highest importance. Scalability is not an issue, so we can directly deploy the mainframe on an infrastructure that will not change. Some interface may be necessary to submit the tax returns (nowadays, also the web), but the analysis happen on-site internally.

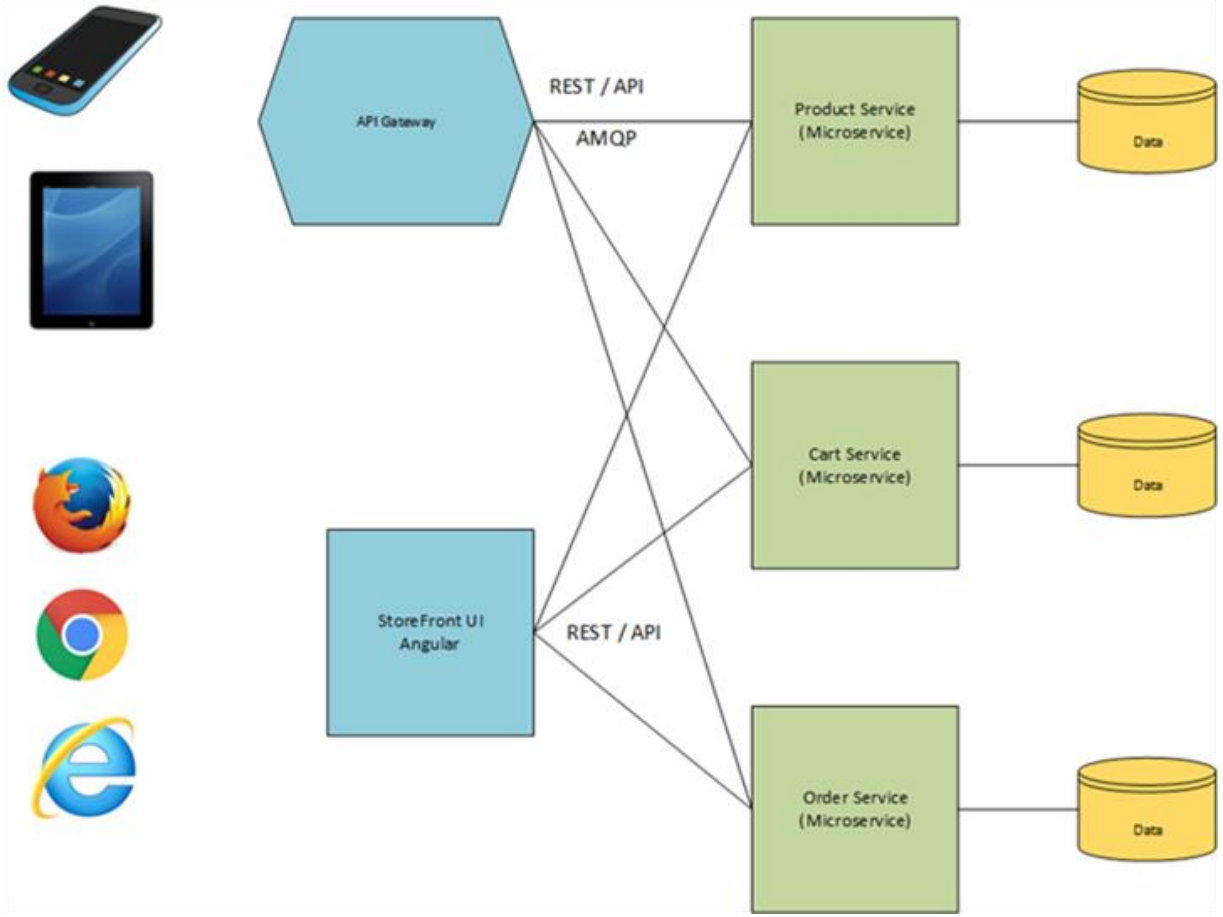
Software Architecture for $800

- We search for a data system to sell products of an e-store. Each product can be identified uniquely. The user can simply view a product, add a product, delete a product and update a product.

Software Architecture for $800

- We search for a data system to sell products of an e-store. Each product can be identified uniquely. The user can simply view a product, add a product, delete a product and update a product.
- What is… “REST” or “Microservices”
- Why? We definitely look for some kind of web architecture, so most probably client-server. To be more specific, this is a good case for SOA or multitier (GUI, service, DB). Based on the unique product identifiers, we are led to a specific SOA solution: REST that works with URIs. Based on the set of specific functionalities (CRUD), SOAP is too complicated for this application. The difference between Microservices and REST can be perceived as the one between Architecture and Implementation.

Software Architecture for $800



https://www.accenture.com/us-en/blogs/software-engineering-blog/bowler-microservice-architecture

Software Architecture for $1000

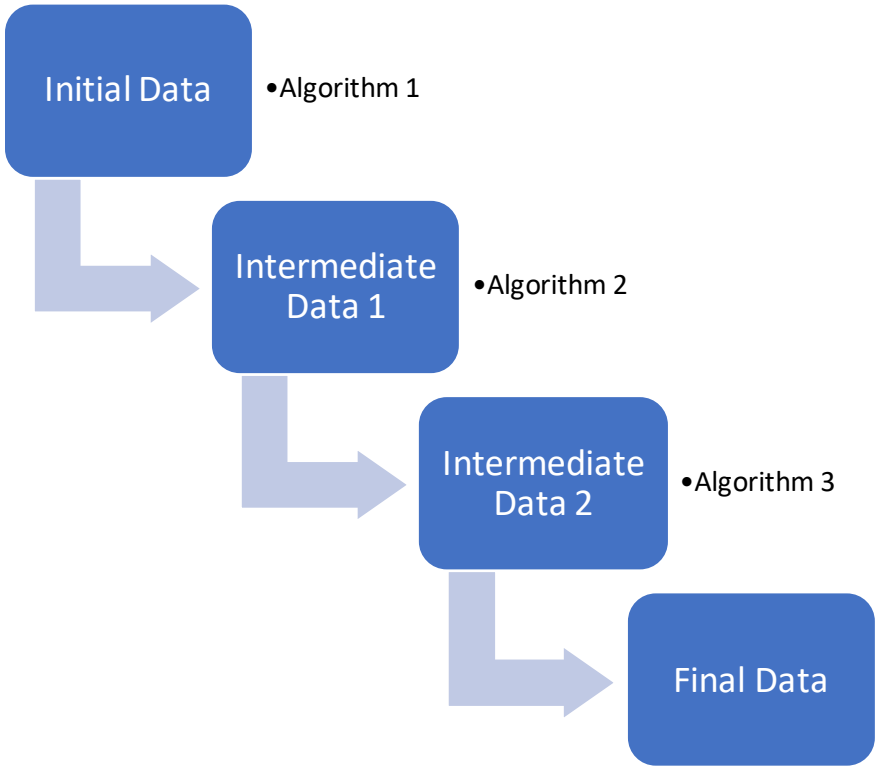
- We search for a data analytics system. There are many algorithms to analyse the data. Each algorithm can change the data and prepare it for the following algorithm

Software Architecture for $1000

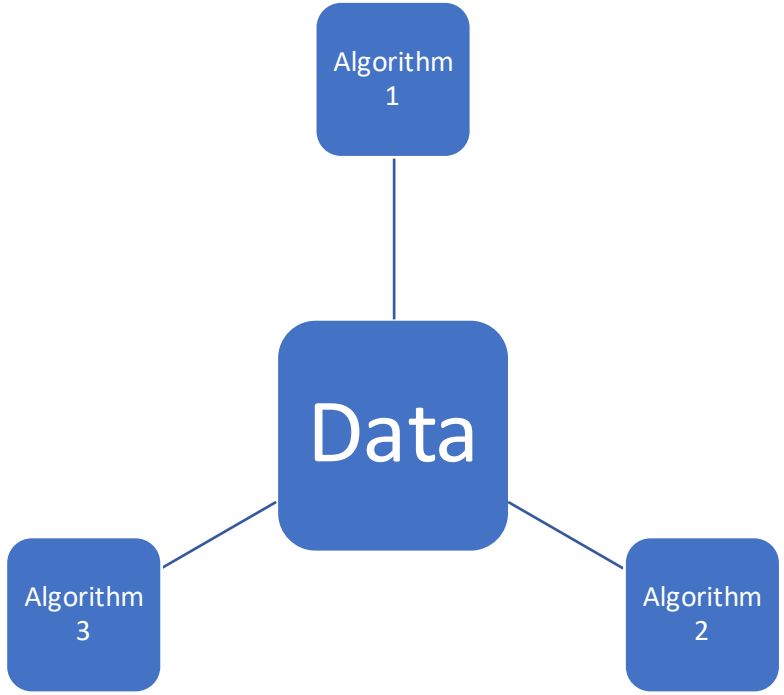
- We search for a data analytics system. There are many algorithms to analyse the data. Each algorithm can change the data and prepare it for the following algorithm
- What is… “pipe-filter”, if the order is predetermined.
- What is… “blackboard”, if the order is determined based on the most recent results.
- Why? We have many functionalities (so, not master-slave). There is a sequence of applied functionalities on a given dataset.

Software Architecture for $1000

Pipe-filter



Blackboard




Architecture
Quality Version
JEOPARDY!

Software architecture quality for $200

- We are looking for a financial system that monitors investments. The requirements are well defined. The system needs to be available 100% of the time. Our clients are our priority, their data need to remain private and we need to maintain their integrity.

Software architecture quality for $200

- We are looking for a financial system that monitors investments. The requirements are well defined. The system needs to be available 100% of the time. Our clients are our priority, their data need to remain private and we need to maintain their integrity.
- What is… CBA
  - …unless monolithic is an option.
  - Potentially client-server with a thin client (if not distributed)
  - Layered or MVC are the best options.
- The system is critical. The conformity to the requirements, the security and the reliability are the priorities.

Software architecture quality for $400

- We are looking for a smart assistance system with the ability to understand and produce natural language. We will use multiple tools that already exist. As an innovative project, we can expect a lot of changes.

Software architecture quality for $400

- We are looking for a smart assistance system with the ability to understand and produce natural language. We will use multiple tools that already exist. As an innovative project, we can expect a lot of changes.
- What is… SOA
  - Potentially SOAP or microservices.
- Functional Suitability, Maintainability, Compatibility

Software architecture quality for $600

- We are looking for a system to monitor the environmental conditions of buildings. The system will notify the building manager in real time if there are problems. The system should function for any building and it should be integrated with existing equipment.

Software architecture quality for $600

- We are looking for a system to monitor the environmental conditions of buildings. The system will notify the building manager in real time if there are problems. The system should function for any building and it should be integrated with existing equipment.
- What is… EDA
- Efficiency, Portability, Compatibility

Software architecture quality for $800

- We are looking for a system for an autonomic vehicle. The system will monitor the vehicle at all times and it cannot make errors. The security of the passengers is of utmost importance. The system should function for any vehicle.

Software architecture quality for $800

- We are looking for a system for an autonomic vehicle. The system will monitor the vehicle at all times and it cannot make errors. The security of the passengers is of utmost importance. The system should function for any vehicle.
- What is… AOA
- Efficiency, Reliability, Security, Portability

Software architecture quality for $1000

- We are looking for a flight simulator. The system should have a simple interface, clear and concise to train pilots. It should be adaptive and easy to change in order to accommodate multiple scenarios in the future. It should avoid errors to ensure the proper training of the pilots.

Software architecture quality for $1000

- We are looking for a flight simulator. The system should have a simple interface, clear and concise to train pilots. It should be adaptive and easy to change in order to accommodate multiple scenarios in the future. It should avoid errors to ensure the proper training of the pilots.
- What is… MDA
  - Pipe-filter or Blackboard. Something were modules can be replaced and buggy modules can be fixed in an uninterrupted manner.
- Usability, Maintainability, Reliability


Bad design
JEOPARDY!

Bad design for $200

ActivityModel.java

```
27  * <em>Framework</em><br>
28  * The interfaces and classes listed below define a framework for progress
29  * management.<br>
30  * Contract: {@link ActivityManager}, {@link ActivityModel},
31  * {@link JActivityWindow}, {@link JActivityIndicator}.
32  *
33  * @author Werner Randelshofer
34  * @version $Id$
35  */
36 public interface ActivityModel extends BoundedRangeModel {
37
38     public static final String INDETERMINATE_PROPERTY = "indeterminate";
39     public static final String NOTE_PROPERTY = "note";
40     public static final String WARNING_PROPERTY = "warning";
41     public static final String ERROR_PROPERTY = "error";
42     public static final String CANCELABLE_PROPERTY = "cancelable";
43     public static final String CANCELED_PROPERTY = "canceled";
44     public static final String CLOSED_PROPERTY = "closed";
45
46     /**
47      * Gets the owner of the progress model. This is typically a {@link org.
48      * or a {@link org.jhotdraw.api.app.Application}.
49      */
50     public Object getOwner();
51
52     /**
53      * Set cancelable to false if the operation can not be canceled.
54      */
55     public void setCancelable(boolean b);
56
57     /**
58      * Returns true if the operation can be canceled.
59      */
```

JActivityView.java

```
private void updateProperties(PropertyChangeEvent evt) {
    if (evt == null || evt.getPropertyName() == null) {
        updateNote();
        updateWarning();
        updateError();
        updateCancelable();
        updateCanceled();
        updateClosed();
        updateIndeterminate();
        return;
    }
    String name = evt.getPropertyName();
    if ((name == null && ActivityModel.NOTE_PROPERTY == null) || (name != null && name.equals(ActivityModel.NOTE_PROPERTY))) {
        updateNote();
    } else if ((name == null && ActivityModel.WARNING_PROPERTY == null) || (name != null && name.equals(ActivityModel.WARNING_PROPERTY))) {
        updateWarning();
    } else if ((name == null && ActivityModel.ERROR_PROPERTY == null) || (name != null && name.equals(ActivityModel.ERROR_PROPERTY))) {
        updateError();
    } else if ((name == null && ActivityModel.CANCELABLE_PROPERTY == null) || (name != null && name.equals(ActivityModel.CANCELABLE_PROPERTY))) {
        updateCancelable();
    } else if ((name == null && ActivityModel.CANCELED_PROPERTY == null) || (name != null && name.equals(ActivityModel.CANCELED_PROPERTY))) {
        updateCanceled();
        updateCancelable();
    } else if ((name == null && ActivityModel.INDETERMINATE_PROPERTY == null) || (name != null && name.equals(ActivityModel.INDETERMINATE_PROPERTY))) {
        updateIndeterminate();
    } else if ((name == null && ActivityModel.CLOSED_PROPERTY == null) || (name != null && name.equals(ActivityModel.CLOSED_PROPERTY))) {
        updateCancelable();
        updateCanceled();
        updateClosed();
    }
}
```

Bad design for $200

- The method updateProperties() in class JActivityView obviously suffers of a case of Type Checking.
- The problem in this case is a lack of flexibility when adding or removing types of properties in class ActivityModel.
  - This is a violation of OCP (see slides 14-18 in SOLID course).
- Type checking is a known antipattern, causing the same problems. (see slides 42-44 in Bad Design course).
  - The example in slide 44 is very similar to the one found in JHotDraw.
- If a type is added or removed in ActivityModel this would cause necessary changes to other places in code, including for example JActivityView.
  - This ripple effect is another known antipattern called Shotgun Surgery (slide 40 in Bad Design course).
- Type code can be solved by introducing a hierarchy (through a State or Strategy design pattern) if one doesn't exist or by exploiting polymorphism if a hierarchy does exist.
  - In our example we can extract a PropertyType class, with a concrete class for every specific type that can contain the special update method for every type.

Bad design for $400

MDIApplication.java

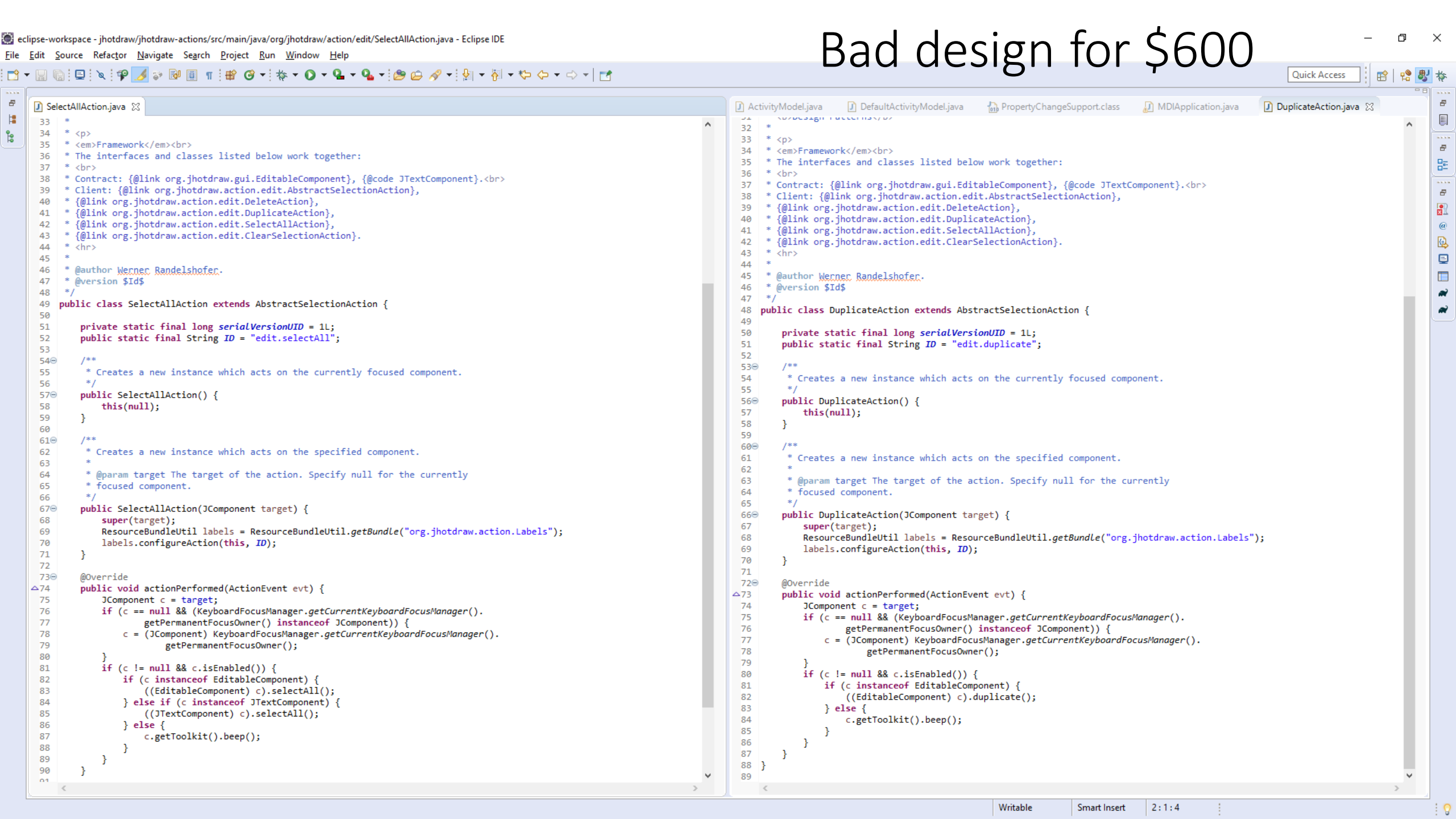
```
@Override
public void show(final View v) {
    if (!v.isShowing()) {
        v.setShowing(true);
        final JInternalFrame f = new JInternalFrame();
        f.setDefaultCloseOperation(JInternalFrame.DO_NOTHING_ON_CLOSE);
        f.setClosable(getAction(v, CloseFileAction.ID) != null);
        f.setMaximizable(true);
        f.setResizable(true);
        f.setIconifiable(false);
        f.setSize(new Dimension(400, 400));
        updateViewTitle(v, f);
        PreferencesUtil.installInternalFramePrefsHandler(prefs, "view", f, desktopPane);
        Point loc = new Point(desktopPane.getInsets().left, desktopPane.getInsets().top);
        boolean moved;
        do {
            moved = false;
            for (View aView : views()) {
                if (aView != v && aView.isShowing()
                        && SwingUtilities.getRootPane(aView.getComponent()).getParent().
                                getLocation().equals(loc)) {
                    Point offset = SwingUtilities.convertPoint(SwingUtilities.getRootPane(aView.getC
                    loc.x += Math.max(offset.x, offset.y);
                    loc.y += Math.max(offset.x, offset.y);
                    moved = true;
                    break;
                }
            }
        } while (moved);
        f.setLocation(loc);
        //paletteHandler.add(f, v);
        f.addInternalFrameListener(new InternalFrameAdapter() {
            @Override
            public void internalFrameClosing(final InternalFrameEvent evt) {
                getAction(v, CloseFileAction.ID).actionPerformed(
                        new ActionEvent(f, ActionEvent.ACTION_PERFORMED,
                                "windowClosing"));
            }

            @Override
            public void internalFrameClosed(final InternalFrameEvent evt) {
                v.stop();
            }
        });
        v.addPropertyChangeListener(new PropertyChangeListener() {
            @Override
            public void propertyChange(PropertyChangeEvent evt) {
                String name = evt.getPropertyName();
                if (((name == null && View.HAS_UNSAVED_CHANGES_PROPERTY == null) || (name != null &&
                        || ((name == null && View.URI_PROPERTY == null) || (name != null && name.equ
```

Bad design for $400

- (If we had access to the entire code of method show(),) the method has a Cyclomatic Complexity of 17, maximum nesting depth of 6, lines of code of 82.
- This is a clear case of Long Method (see slide 35 in Bad Design course).
  - This can also be detected by metrics (see slide 51 in Bad Design course). Note: LOC may not be too high, but high complexity could be considered enough reason.
- The method also seems to use a lot from classes View and JInternalFrame.
  - This can be an indication of Feature Envy (see slide 41 in Bad Design course).
- Long method, also in this case, can be solved by extracting parts in smaller less complex methods or using the Compose Method refactoring to pattern (slide 36 in Bad Design course).
  - We can also extract and move the parts that correspond to View and JInternalFrame classes to solve the Feature Envy problem (see slide 41 in Bad Design course).

Bad design for $600




```java
/**
 * <p>
 * <em>Framework</em><br>
 * The interfaces and classes listed below work together:
 * <br>
 * Contract: {@link org.jhotdraw.gui.EditableComponent}, {@code JTextComponent}.<br>
 * Client: {@link org.jhotdraw.action.edit.AbstractSelectionAction},
 * {@link org.jhotdraw.action.edit.DeleteAction},
 * {@link org.jhotdraw.action.edit.DuplicateAction},
 * {@link org.jhotdraw.action.edit.SelectAllAction},
 * {@link org.jhotdraw.action.edit.ClearSelectionAction}.
 * <hr>
 *
 * @author Werner Randelshofer.
 * @version $Id$
 */
public class SelectAllAction extends AbstractSelectionAction {

    private static final long serialVersionUID = 1L;
    public static final String ID = "edit.selectAll";

    /**
     * Creates a new instance which acts on the currently focused component.
     */
    public SelectAllAction() {
        this(null);
    }

    /**
     * Creates a new instance which acts on the specified component.
     *
     * @param target The target of the action. Specify null for the currently
     * focused component.
     */
    public SelectAllAction(JComponent target) {
        super(target);
        ResourceBundleUtil labels = ResourceBundleUtil.getBundle("org.jhotdraw.action.Labels");
        labels.configureAction(this, ID);
    }

    @Override
    public void actionPerformed(ActionEvent evt) {
        JComponent c = target;
        if (c == null && (KeyboardFocusManager.getCurrentKeyboardFocusManager().
                getPermanentFocusOwner() instanceof JComponent)) {
            c = (JComponent) KeyboardFocusManager.getCurrentKeyboardFocusManager().
                    getPermanentFocusOwner();
        }
        if (c != null && c.isEnabled()) {
            if (c instanceof EditableComponent) {
                ((EditableComponent) c).selectAll();
            } else if (c instanceof JTextComponent) {
                ((JTextComponent) c).selectAll();
            } else {
                c.getToolkit().beep();
            }
        }
    }
```

```java
/**
 * <p>
 * <em>Framework</em><br>
 * The interfaces and classes listed below work together:
 * <br>
 * Contract: {@link org.jhotdraw.gui.EditableComponent}, {@code JTextComponent}.<br>
 * Client: {@link org.jhotdraw.action.edit.AbstractSelectionAction},
 * {@link org.jhotdraw.action.edit.DeleteAction},
 * {@link org.jhotdraw.action.edit.DuplicateAction},
 * {@link org.jhotdraw.action.edit.SelectAllAction},
 * {@link org.jhotdraw.action.edit.ClearSelectionAction}.
 * <hr>
 *
 * @author Werner Randelshofer.
 * @version $Id$
 */
public class DuplicateAction extends AbstractSelectionAction {

    private static final long serialVersionUID = 1L;
    public static final String ID = "edit.duplicate";

    /**
     * Creates a new instance which acts on the currently focused component.
     */
    public DuplicateAction() {
        this(null);
    }

    /**
     * Creates a new instance which acts on the specified component.
     *
     * @param target The target of the action. Specify null for the currently
     * focused component.
     */
    public DuplicateAction(JComponent target) {
        super(target);
        ResourceBundleUtil labels = ResourceBundleUtil.getBundle("org.jhotdraw.action.Labels");
        labels.configureAction(this, ID);
    }

    @Override
    public void actionPerformed(ActionEvent evt) {
        JComponent c = target;
        if (c == null && (KeyboardFocusManager.getCurrentKeyboardFocusManager().
                getPermanentFocusOwner() instanceof JComponent)) {
            c = (JComponent) KeyboardFocusManager.getCurrentKeyboardFocusManager().
                    getPermanentFocusOwner();
        }
        if (c != null && c.isEnabled()) {
            if (c instanceof EditableComponent) {
                ((EditableComponent) c).duplicate();
            } else {
                c.getToolkit().beep();
            }
        }
    }
}
```

Bad design for $600

- Duplicate code between classes SelectAllAction and DuplicateAction.
  - see slide 33 in Bad Design course
- Duplication can overcomplicate maintenance; if one clone changes, all other instances need to be updated. So, we have to minimize the points of change.
- Extracting the common elements so that they can be reused is the solution for duplicated code.
  - When the two classes are related, we can extract the common elements in a superclass and make the clone classes extend it.
- Given that in this case, the hierarchy exists, we can also talk about Unfactored Hierarchy (see slide 29 in Bad Design course).
  - An other acceptable, but possibly not as good a response, is Duplicate Abstraction (see slide 23 in Bad Design course).

Bad design for $800

SVGUtil.java/toPathData()

```
public static String toPathData(BezierPath path) {
    StringBuilder buf = new StringBuilder();

    if (path.size() == 0) {
        // nothing to do
    } else if (path.size() == 1) {
        BezierPath.Node current = path.get(0);
        buf.append("M ");
        buf.append(current.x[0]);
        buf.append(' ');
        buf.append(current.y[0]);
        buf.append(" L ");
        buf.append(current.x[0]);
        buf.append(' ');
        buf.append(current.y[0] + 1);
    } else {
        BezierPath.Node previous;
        BezierPath.Node current;

        previous = current = path.get(0);
        buf.append("M ");
        buf.append(current.x[0]);
        buf.append(' ');
        buf.append(current.y[0]);
        for (int i=1, n = path.size(); i < n; i++) {
            previous = current;
            current = path.get(i);

            if ((previous.mask & BezierPath.C2_MASK) =
                if ((current.mask & BezierPath.C1_MASK
                    buf.append(" L ");
                    buf.append(current.x[0]);
                    buf.append(' ');
                    buf.append(current.y[0]);
                } else {
                    buf.append(" Q ");
                    buf.append(current.x[1]);
                    buf.append(' ');
                    buf.append(current.y[1]);
                    buf.append(' ');
                    buf.append(current.x[0]);
                    buf.append(' ');
                    buf.append(current.y[0]);
                }
            } else {
                if ((current.mask & BezierPath.C1_MASK
                    buf.append(" Q ");
                    buf.append(current.x[2]);
                    buf.append(' ');
                    buf.append(current.y[2]);
                    buf.append(' ');
                    buf.append(current.x[1]);
                    buf.append(' ');
                    buf.append(current.y[1]);
                    buf.append(' ');
                    buf.append(current.x[0]);
                    buf.append(' ');
                    buf.append(current.y[0]);
                } else {
                    buf.append(" C ");
                    buf.append(current.x[2]);
                    buf.append(' ');
                    buf.append(current.y[2]);
                    buf.append(' ');
                    buf.append(current.x[1]);
                    buf.append(' ');
                    buf.append(current.y[1]);
                    buf.append(' ');
                    buf.append(current.x[0]);
                    buf.append(' ');
                    buf.append(current.y[0]);
                }
            }
        }
    }
    if (path.isClosed()) {
        if (path.size() > 1) {
            previous = path.get(path.size() - 1);
            current = path.get(0);

            if ((previous.mask & BezierPath.C2_MASK) == 0) {
                if ((current.mask & BezierPath.C1_MASK) == 0) {
                    buf.append(" L ");
                    buf.append(current.x[0]);
                    buf.append(' ');
                    buf.append(current.y[0]);
                } else {
                    buf.append(" Q ");
                    buf.append(current.x[1]);
                    buf.append(' ');
                    buf.append(current.y[1]);
                    buf.append(' ');
                    buf.append(current.x[0]);
                    buf.append(' ');
                    buf.append(current.y[0]);
                }
            } else {
                if ((current.mask & BezierPath.C1_MASK) == 0) {
                    buf.append(" Q ");
                    buf.append(previous.x[2]);
                    buf.append(' ');
                    buf.append(previous.y[2]);
                    buf.append(' ');
                    buf.append(current.x[0]);
                    buf.append(' ');
                    buf.append(current.y[0]);
                } else {
                    buf.append(" C ");
                    buf.append(previous.x[2]);
                    buf.append(' ');
                    buf.append(previous.y[2]);
                    buf.append(' ');
                    buf.append(current.x[1]);
                    buf.append(' ');
                    buf.append(current.y[1]);
                    buf.append(' ');
                    buf.append(current.x[0]);
                    buf.append(' ');
                    buf.append(current.y[0]);
                }
            }
            buf.append(" Z");
        }
```

Bad design for $800

- (If we had access to the entire code of method toPathData(),) the method has a Cyclomatic Complexity of 12, maximum nesting depth of 6, lines of code of 115.
- This is a clear case of Long Method (see slide 35 in Bad Design course).
  - This can also be detected by metrics (see slide 51 in Bad Design course). Note: LOC may not be too high, but high complexity could be considered enough reason.
- The method also seems to use a lot from classes StringBuilder and BezierPath.
  - This can be an indication of Feature Envy (see slide 41 in Bad Design course).
- Long method, also in this case, can be solved by extracting parts in smaller less complex methods or using the Compose Method refactoring to pattern (slide 36 in Bad Design course).
  - We can also extract and move the parts that correspond to StringBuilder and BezierPath classes to solve the Feature Envy problem (see slide 41 in Bad Design course).
  - We can further minimize the length of the method by collapsing all the calls to the append() method.

Bad design for $1000

```java
/**
 * Creates a new legend item.
 *
 * @param label  the label ({@code null} not permitted).
 * @param description  the description (not currently used,
 *         {@code null} permitted).
 * @param toolTipText  the tool tip text ({@code null} permitted).
 * @param urlText  the URL text ({@code null} permitted).
 * @param shapeVisible  a flag that controls whether or not the shape is
 *                      displayed.
 * @param shape  the shape ({@code null} permitted).
 * @param shapeFilled  a flag that controls whether or not the shape is
 *                     filled.
 * @param fillPaint  the fill paint ({@code null} not permitted).
 * @param shapeOutlineVisible  a flag that controls whether or not the
 *                             shape is outlined.
 * @param outlinePaint  the outline paint ({@code null} not permitted).
 * @param outlineStroke  the outline stroke ({@code null} not
 *                       permitted).
 * @param lineVisible  a flag that controls whether or not the line is
 *                     visible.
 * @param line  the line ({@code null} not permitted).
 * @param lineStroke  the stroke ({@code null} not permitted).
 * @param linePaint  the line paint ({@code null} not permitted).
 */
public LegendItem(AttributedString label, String description,
                  String toolTipText, String urlText,
                  boolean shapeVisible, Shape shape,
                  boolean shapeFilled, Paint fillPaint,
                  boolean shapeOutlineVisible, Paint outlinePaint,
                  Stroke outlineStroke,
                  boolean lineVisible, Shape line, Stroke lineStroke,
                  Paint linePaint) {

    Args.nullNotPermitted(label, "label");
    Args.nullNotPermitted(fillPaint, "fillPaint");
    Args.nullNotPermitted(lineStroke, "lineStroke");
    Args.nullNotPermitted(line, "line");
    Args.nullNotPermitted(linePaint, "linePaint");
    Args.nullNotPermitted(outlinePaint, "outlinePaint");
    Args.nullNotPermitted(outlineStroke, "outlineStroke");
    this.label = characterIteratorToString(label.getIterator());
    this.attributedLabel = label;
    this.description = description;
    this.shapeVisible = shapeVisible;
    this.shape = shape;
    this.shapeFilled = shapeFilled;
    this.fillPaint = fillPaint;
    this.fillPaintTransformer = new StandardGradientPaintTransformer();
    this.shapeOutlineVisible = shapeOutlineVisible;
    this.outlinePaint = outlinePaint;
    this.outlineStroke = outlineStroke;
    this.lineVisible = lineVisible;
    this.line = line;
    this.lineStroke = lineStroke;
    this.linePaint = linePaint;
    this.toolTipText = toolTipText;
    this.urlText = urlText;
}
```

```java
@Override
public boolean equals(Object obj) {
    if (obj == this) {
        return true;
    }
    if (!(obj instanceof LegendItem)) {
        return false;
    }
    LegendItem that = (LegendItem) obj;
    if (this.datasetIndex != that.datasetIndex) {
        return false;
    }
    if (this.series != that.series) {
        return false;
    }
    if (!this.label.equals(that.label)) {
        return false;
    }
    if (!AttributedStringUtils.equal(this.attributedLabel,
            that.attributedLabel)) {
        return false;
    }
    if (!ObjectUtils.equal(this.description, that.description)) {
        return false;
    }
    if (this.shapeVisible != that.shapeVisible) {
        return false;
    }
    if (!ShapeUtils.equal(this.shape, that.shape)) {
        return false;
    }
    if (this.shapeFilled != that.shapeFilled) {
        return false;
    }
    if (!PaintUtils.equal(this.fillPaint, that.fillPaint)) {
        return false;
    }
    if (!ObjectUtils.equal(this.fillPaintTransformer,
            that.fillPaintTransformer)) {
        return false;
    }
    if (this.shapeOutlineVisible != that.shapeOutlineVisible) {
        return false;
    }
    if (!this.outlineStroke.equals(that.outlineStroke)) {
        return false;
    }
    if (!PaintUtils.equal(this.outlinePaint, that.outlinePaint)) {
        return false;
    }
    if (!this.lineVisible == that.lineVisible) {
        return false;
    }
    if (!ShapeUtils.equal(this.line, that.line)) {
        return false;
    }
    if (!this.lineStroke.equals(that.lineStroke)) {
        return false;
    }
```

Bad design for $1000

- The constructor of the class LegentItem has 15 parameters.
  - This can be classified as a Long Parameter List antipattern (see slide 38 in Bad Design course).
- This can cause increased complexity as in the case of the equals() method.
  - This can result in Long Method or even Switch Statements (see slides 35 and 42 in Bad Design course).
- A solution here is to extract groups of parameters as Parameter Objects.
  - For example, parameters related to paint, shape, or lines.
- If Parameter Objects are extractable, we can also think of this as a case of Missing Abstraction (see slide 21 in Bad Design course).

Bad design for $2000

LogarithmicAxis.java/
refreshTicksVertical()

```java
/**
 * Calculates the positions of the tick labels for the axis, storing the
 * results in the tick label list (ready for drawing).
 *
 * @param g2  the graphics device.
 * @param dataArea  the area in which the plot should be drawn.
 * @param edge  the location of the axis.
 *
 * @return A list of ticks.
 */
@Override
protected List refreshTicksVertical(Graphics2D g2, Rectangle2D dataArea,
        RectangleEdge edge) {

    List ticks = new java.util.ArrayList();

    //get lower bound value:
    double lowerBoundVal = getRange().getLowerBound();
    //if small log values and lower bound value too small
    // then set to a small value (don't allow <= 0):
    if (this.smallLogFlag && lowerBoundVal < SMALL_LOG_VALUE) {
        lowerBoundVal = SMALL_LOG_VALUE;
    }
    //get upper bound value
    double upperBoundVal = getRange().getUpperBound();

    //get log10 version of lower bound and round to integer:
    int iBegCount = (int) Math.rint(switchedLog10(lowerBoundVal));
    //get log10 version of upper bound and round to integer:
    int iEndCount = (int) Math.rint(switchedLog10(upperBoundVal));

    if (iBegCount == iEndCount && iBegCount > 0
            && Math.pow(10, iBegCount) > lowerBoundVal) {
        //only 1 power of 10 value, it's > 0 and its resulting
        // tick value will be larger than lower bound of data
        --iBegCount;       //decrement to generate more ticks
    }

    double tickVal;
    String tickLabel;
    boolean zeroTickFlag = false;
    for (int i = iBegCount; i <= iEndCount; i++) {
        //for each tick with a label to be displayed
        int jEndCount = 10;
        if (i == iEndCount) {
            jEndCount = 1;
        }

        for (int j = 0; j < jEndCount; j++) {
            //for each tick to be displayed
            if (this.smallLogFlag) {
                //small log values in use
                tickVal = Math.pow(10, i) + (Math.pow(10, i) * j);
                if (j == 0) {
                    //first tick of group; create label text
                    if (this.log10TickLabelsFlag) {
                        //if flag then
                        tickLabel = "10^" + i;   //create "log10"-type label
                    }
                    else {    //not "log10"-type label
                        if (this.expTickLabelsFlag) {
                            //if flag then
                            tickLabel = "1e" + i;  //create "1e#"-type label
                        }
                        else {    //not "1e#"-type label
                            if (i >= 0) {   // if positive exponent then
                                            // make integer
                                NumberFormat format
                                    = getNumberFormatOverride();
                                if (format != null) {
                                    tickLabel = format.format(tickVal);
                                }
                                else {
                                    tickLabel = Long.toString((long)
                                            Math.rint(tickVal));
                                }
                            }
                            else {
                                //negative exponent; create fractional value
                                //set exact number of fractional digits to
                                // be shown:
                                this.numberFormatterObj
                                    .setMaximumFractionDigits(-i);
                                //create tick label:
                                tickLabel = this.numberFormatterObj.format(
                                        tickVal);
                            }
                        }
                    }
                }
                else {   //not first tick to be displayed
                    tickLabel = "";     //no tick label
                }
            }
            else { //not small log values in use; allow for values <= 0
                if (zeroTickFlag) {      //if did zero tick last iter then
                    --j;
                }               //decrement to do 1.0 tick now
                tickVal = (i >= 0) ? Math.pow(10, i) + (Math.pow(10, i) * j)
                        : -(Math.pow(10, -i) - (Math.pow(10, -i - 1) * j));
                if (j == 0) {  //first tick of group
                    if (!zeroTickFlag) {     // did not do zero tick last
                                             // iteration
                        if (i > iBegCount && i < iEndCount
                                && Math.abs(tickVal - 1.0) < 0.0001) {
                            // not first or last tick on graph and value
                            // is 1.0
                            tickVal = 0.0;        //change value to 0.0
                            zeroTickFlag = true;  //indicate zero tick
                            tickLabel = "0";      //create label for tick
                        }
                        else {
                            //first or last tick on graph or value is 1.0
                            //create label for tick:
                            if (this.log10TickLabelsFlag) {
                                //create "log10"-type label
                                tickLabel = (((i < 0) ? "-" : "")
                                        + "10^" + Math.abs(i));
```

Bad design for $2000

- (If we had access to the entire code of method refreshTicksVertical(),) the method has a Cyclomatic Complexity of 29, maximum nesting depth of 10, lines of code of 139.
- This is a clear case of Long Method (see slide 35 in Bad Design course).
  - This can also be detected by metrics (see slide 51 in Bad Design course). Note: LOC may not be too high, but high complexity could be considered enough reason.
- Long method, also in this case, can be solved by extracting parts in smaller less complex methods or using the Compose Method refactoring to pattern (slide 36 in Bad Design course).

MapReduce

MapReduce for $200

- We want to develop a program to count the words of thousands of documents organized in folders with the first letter of the file's name ("A", "B", "C"…). You have 27 workers. Describe the MapReduce implementation for this problem.

MapReduce for $200

- We want to develop a program to count the words of thousands of documents organized in folders with the first letter of the file's name ("A", "B", "C"…). You have 27 workers. Describe the MapReduce implementation for this problem.

1) Send a folder to each worker.
   a) Be careful! Not all folders contain the same amount of files! You can assign an appropriate size to the workers or use less workers and send some of the least popular letters to a single worker.
2) Map tasks count the words of files per file.
3) Reduce tasks add the results per folder and for the entire file system.
   a) You can have multiple levels of Reduce tasks and the input of a reduce task can be the output of a previous reduce task.

MapReduce for $400

- We have the CRA database for income taxes. The CRA has multiple data servers around the country. We want to find who paid the most taxes in 2018. Describe the MapReduce implementation for this problem.

MapReduce for $400

- We have the CRA database for income taxes. The CRA has multiple data servers around the country. We want to find who paid the most taxes in 2018 per province and in the entire country.. Describe the MapReduce implementation for this problem.

1. Every data server can run a replica of the maximum algorithm.
2. The Map tasks are to find the maximum in a single server.
3. Reduce tasks to find the maximum per province and another Reduce task to find the overall maximum of the country.

MapReduce for $600

- We want to find the player with the most average goals per game in the entire history of NHL. Each team keep record of their own statistics. (Do not consider teams that do not exist now). Describe the MapReduce implementation for this problem.

MapReduce for $600

- We want to find the player with the most average goals per game in the entire history of NHL. Each team keep record of their own statistics. (Do not consider teams that do not exist now). Describe the MapReduce implementation for this problem.

1. The data server of each team will run a replica of the algorithm.
2. Map tasks are to find the average per team.
3. Reduce tasks will find the maximum over the returned averages.
4. And the answer is… Mike Bossy (0.762) of the NY Islanders (1977-1987).

MapReduce for $800

- Each University holds all defended theses in their respective library. We want to search in the entire world for theses that contain the term “DevOps” in their title. Describe the MapReduce implementation for this problem.

MapReduce for $800

- Each University holds all defended theses in their respective library. We want to search in the entire world for theses that contain the term “DevOps” in their title. Describe the MapReduce implementation for this problem.

1. The data server of each university will run a replica of the grep/search algorithm.
2. The Map tasks are the searches with the “DevOps” string on the title.
3. The reduce tasks will simply aggregate the individual results from the map tasks in a list.

MapReduce for $1000

- We want a complete lists of all cities, villages and communities of the world order by population in decreasing order. Census servers in each country contain population data. Describe the MapReduce implementation for this problem.

MapReduce for $1000

- We want a complete lists of all cities, villages and communities of the world order by population in descending order. Census servers in each country contain population data. Describe the MapReduce implementation for this problem.

1. Each census server can run a replica of the sorting algorithm.
2. The map tasks will store the cities per country.
3. The reduce task will aggregate the total list ordered in descending order.

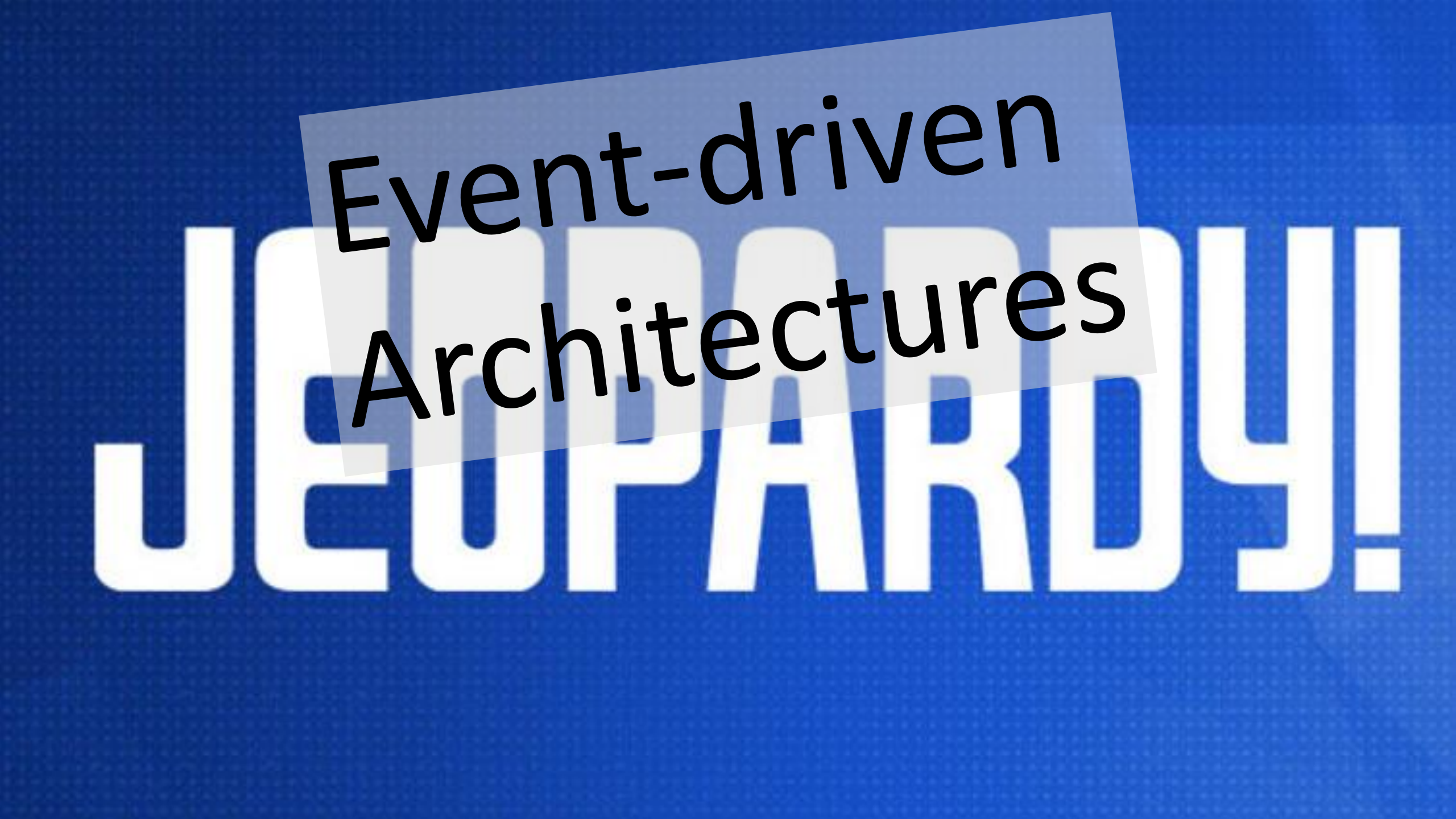
MapReduce for $2000

- We want a complete lists of all cities, villages and communities of the world order by population in decreasing order. Census servers in each country contain population data. Describe the MapReduce implementation for this problem.
- Bonus question: What is the most suitable sorting algorithm for this case?

MapReduce for $2000

- We want a complete lists of all cities, villages and communities of the world order by population in decreasing order. Census servers in each country contain population data. Describe the MapReduce implementation for this problem.
- Bonus question: What is the most suitable sorting algorithm for this case?
- “What is…” mergesort
- In fact, RDDs (or similar data formats) are key-value pairs, which means that sorting can be very efficient and simple over keys.
- If we want to sort by value, one solution is with “compound keys”, where we can combine the primary key with the column we want to sort and then sort by key automatically.

Event-driven Architectures



Event-driven Architectures for $200

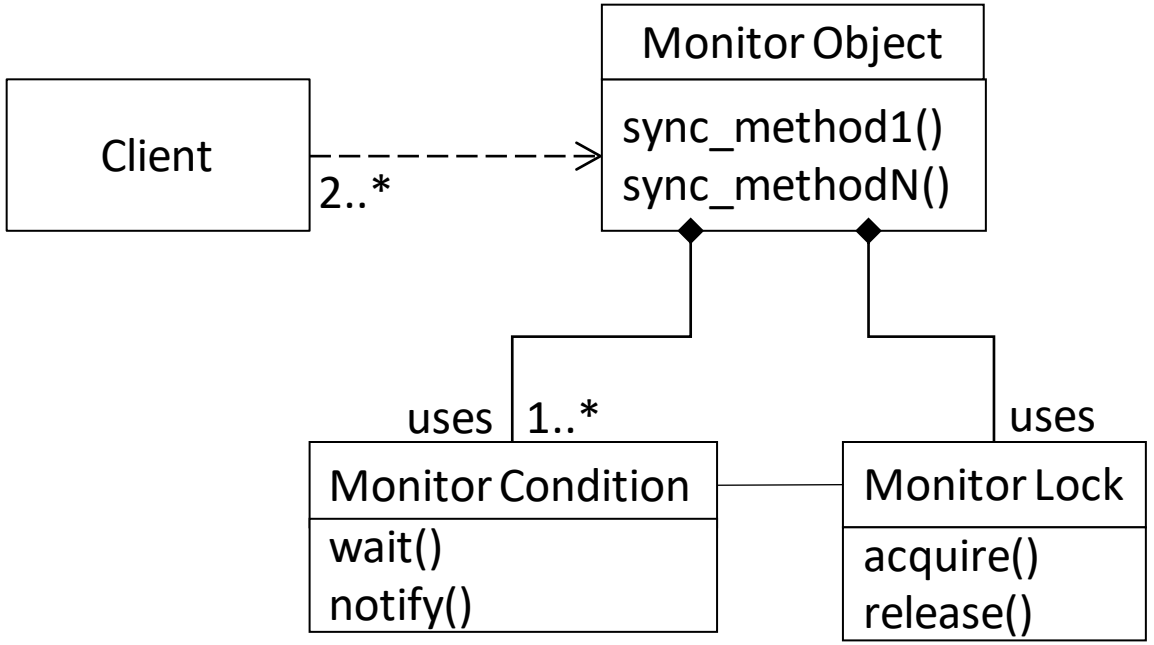
- We are in a fast food restaurant. Clients place their order in the cashier. If the order cannot be served immediately, the clients step aside and wait. When the order is ready, they step back at the front of the line to receive their meal from the cashier.

Event-driven Architectures for $200

- We are in a fast food restaurant. Clients place their order in the cashier. If the order cannot be served immediately, the clients step aside and wait. When the order is ready, they step back at the front line to receive their meal from the cashier.
- What is… Monitor Object
- The cashier is the Monitor Object. The clients are the synchronized requests. Only one client can access the cashier at a given time. Clients “wait” while the cashier prepares an order or serves other clients. The cashier notifies the next client when they are ready.

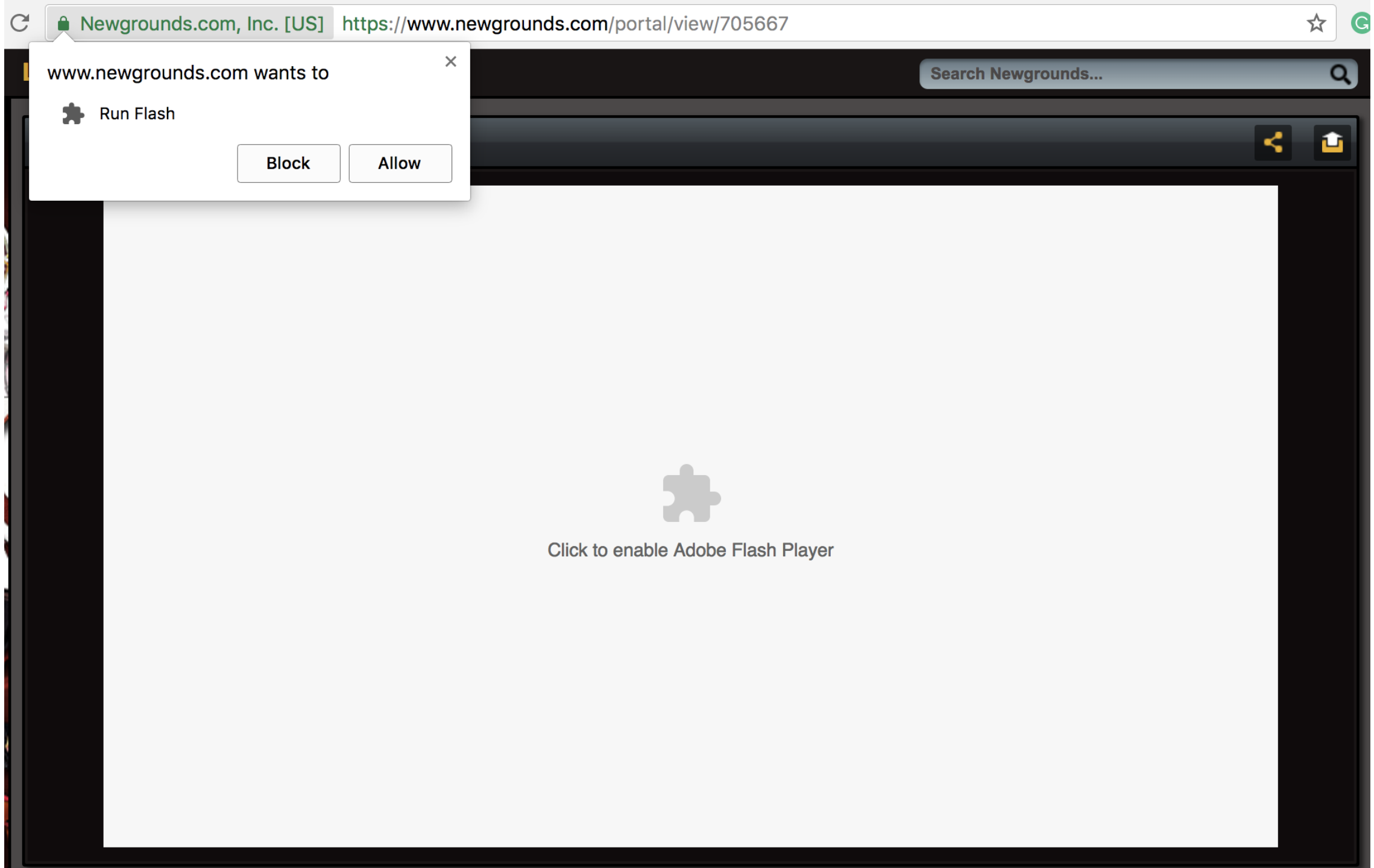
Monitor Object

- **Structure** :



Event-driven Architectures for $400

- Chrome uses plugins to allow the presentation of certain media types, like Flash. When a page that contains such media needs to be loaded the browser checks if the appropriate plugin is installed. If it is, the browser automatically invokes the plugin to handle the media.


Newgrounds.com, Inc. [US]
https://www.newgrounds.com/portal/view/705667
www.newgrounds.com wants to
Run Flash
Block
Allow
Search Newgrounds...
Click to enable Adobe Flash Player

Event-driven Architectures for $400

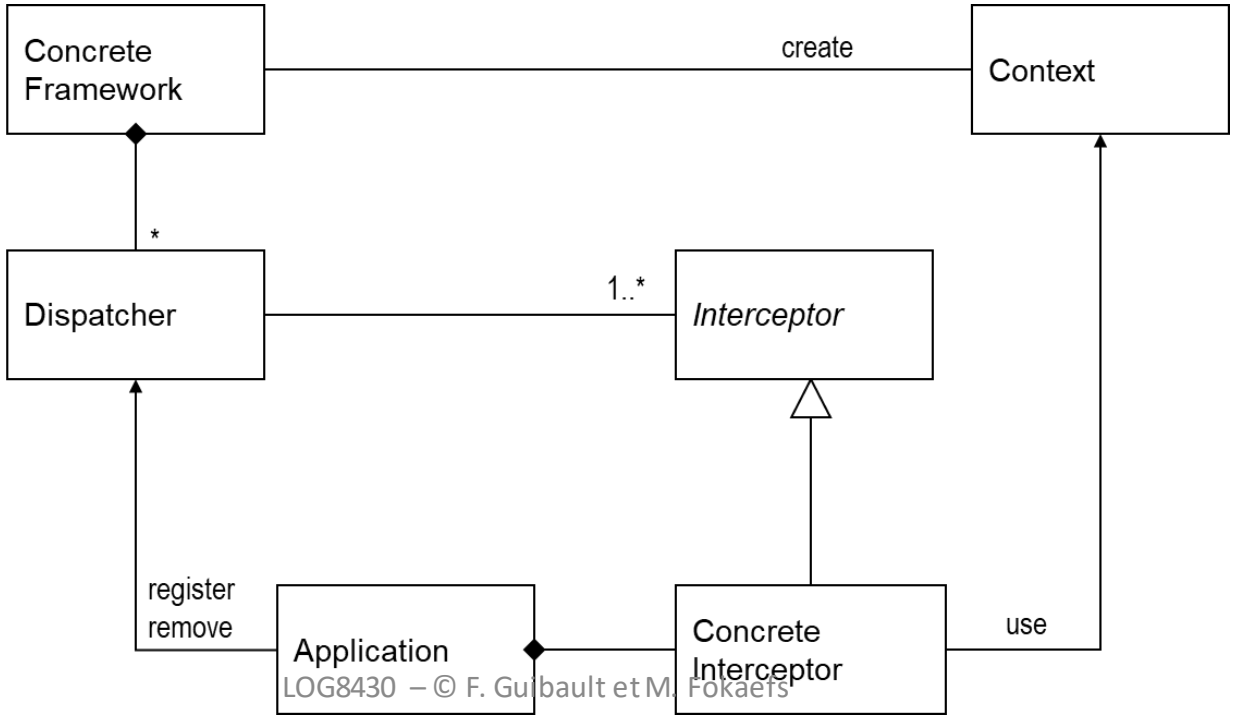
- Chrome uses plugins to allow the presentation of certain media types, like Flash. When a page that contains such media needs to be loaded the browser checks if the appropriate plugin is installed. If it is, the browser automatically invokes the plugin to handle the media.

Event-driven Architectures for $400

- Chrome uses plugins to allow the presentation of certain media types, like Flash. When a page that contains such media needs to be loaded the browser checks if the appropriate plugin is installed. If it is, the browser automatically invokes the plugin to handle the media.
- What is… Interceptor
- The browser is the framework whose functionality can be extended. The plugins are the interceptors. The dispatcher identifies dynamically what is the best plugin to be added.

Interceptor

- **Objective:** Allow services to be added in a framework in a transparent way and to be started automatically when certain events occur.
- **Application:** When a framework needs to be able to register and trigger new services that were not originally planned. Also, to allow applications to control the behavior and the functionality of the framework.
- **Structure** :



Event-driven Architectures for $600

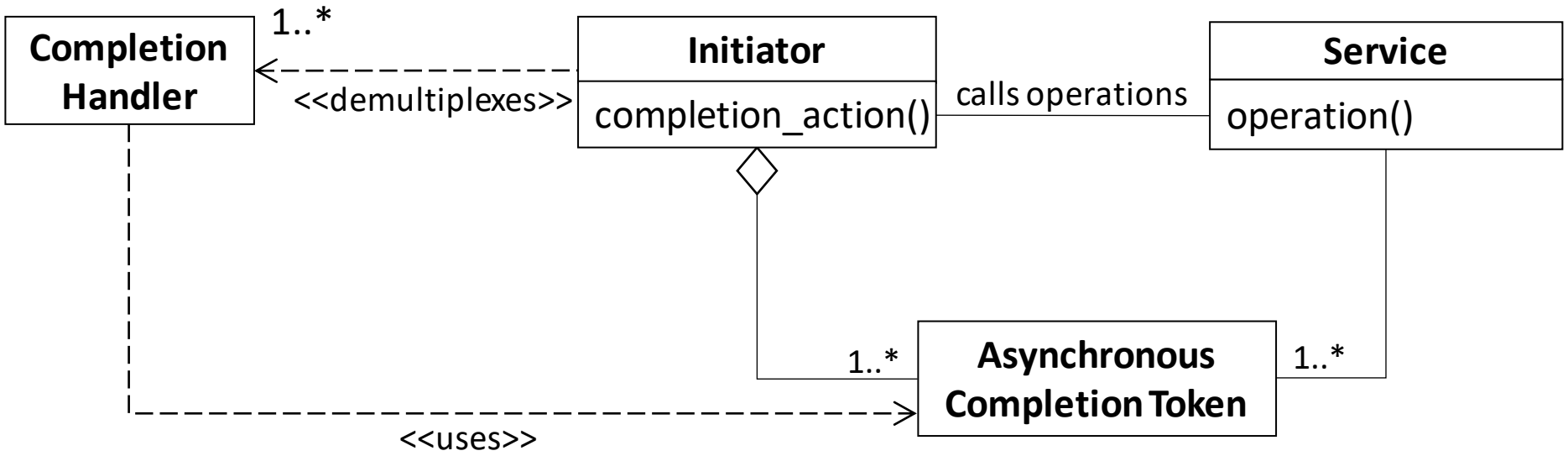
- A FedEx Airbill contains a section labeled: 'Your Internal Billing Reference Information (Optional: First 24 characters will appear on invoice).' The sender of a package uses this field as a confirmation. This confirmation is returned by FedEx to the sender with the invoice that notifies the sender that the transaction has completed. FedEx deliberately defines this field very loosely: it is a maximum of 24 characters, which are otherwise 'untyped.' Therefore, senders can use the field in a variety of ways. For example, a sender can populate this field with the index of a record for an internal database or with a name of a file containing a 'to-do list' to be performed after the acknowledgment of the FedEx package delivery has been received.

Event-driven Architectures for $600

- A FedEx Airbill contains a section labeled: 'Your Internal Billing Reference Information (Optional: First 24 characters will appear on invoice).’ The sender of a package uses this field as a confirmation. This confirmation is returned by FedEx to the sender with the invoice that notifies the sender that the transaction has completed. FedEx deliberately defines this field very loosely: it is a maximum of 24 characters, which are otherwise 'untyped.' Therefore, senders can use the field in a variety of ways. For example, a sender can populate this field with the index of a record for an internal database or with a name of a file containing a 'to-do list' to be performed after the acknowledgment of the FedEx package delivery has been received.
- What is… Asynchronous Completion Token (ACT)
- The “Internal Billing Reference Information” is the ACT. This is ignored by FedEx but it is returned to the sender so that they can handle the response. For example, it can be a product category so that all tracking information for this category are grouped together.

Asynchronous Completion Token (ACT)

- **Structure** :



Event-driven Architectures for $800

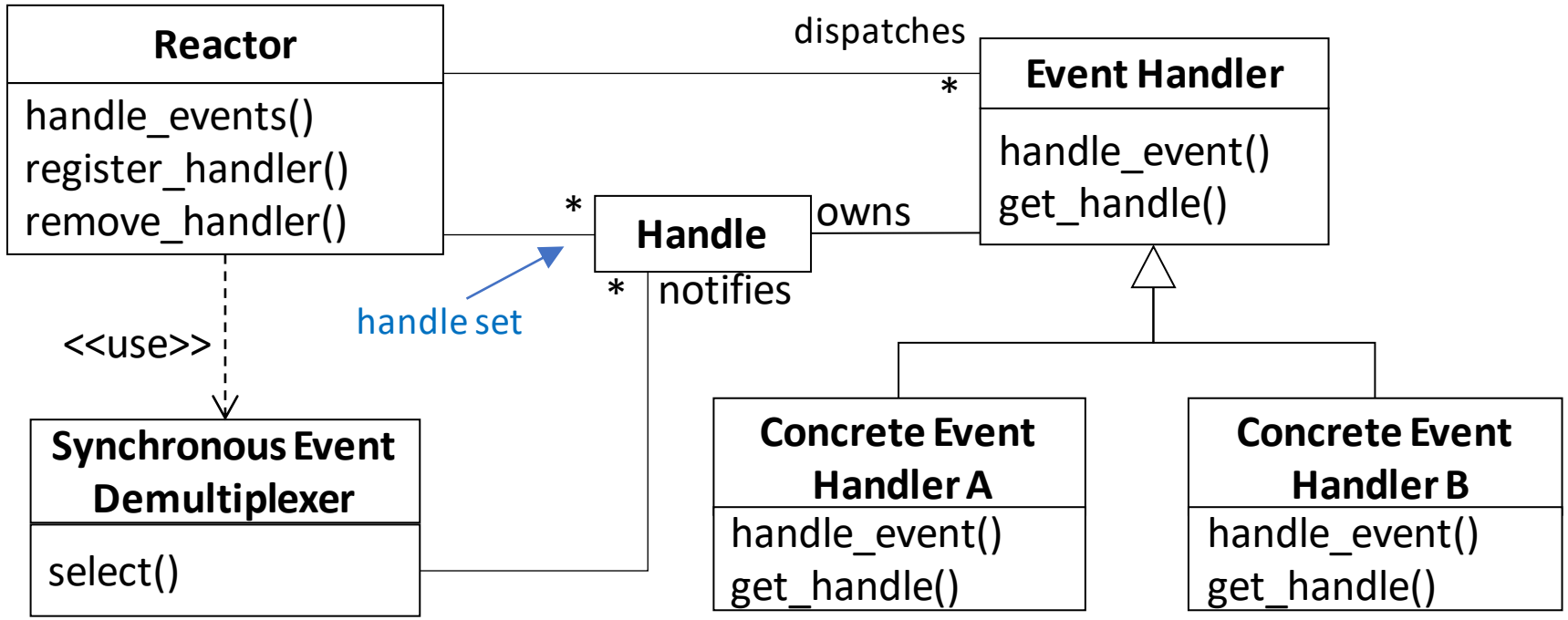
- You probably have a mobile phone associated with a specific phone number. People can call that number and reach you. When somebody does, your phone rings and you may choose to respond and carry out the conversation with the person on the other side of the line.

Event-driven Architectures for $800

- You probably have a mobile phone associated with a specific phone number. People can call that number and reach you. When somebody does, your phone rings and you may choose to respond and carry out the conversation with the person on the other side of the line.
- What is… Reactor
- The telecom provider is the Reactor. Your phone number is the Handle. You (with your mobile phone) are the EventHandler.
- Acceptor/Connector can be another response, but usually accepting a call comes after the connection part, so Reactor is a better answer.

Reactor

- **Structure** :



Event-driven Architectures for $1000

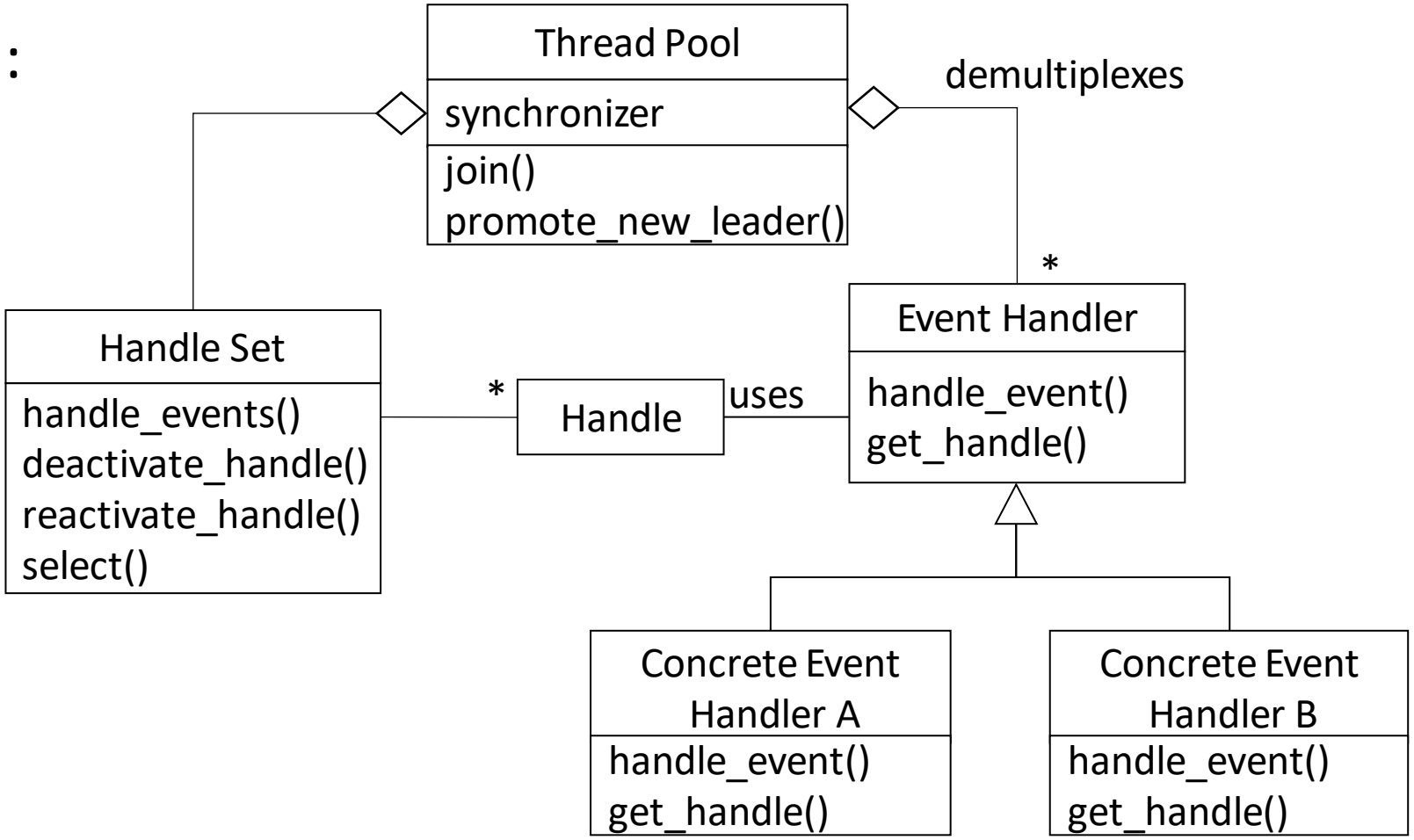
- Travelers arrive at an airport and some of them go to grab a taxi. The taxis are waiting in a line and so do the passengers, wait to be served one at a time. Normally, the first passenger will be served by the first taxi, but in certain cases a taxi driver may choose to serve a particular destination.

Event-driven Architectures for $1000

- Travelers arrive at an airport and some of them go to grab a taxi. The taxis are waiting in a line and so do the passengers, wait to be served one at a time. Normally, the first passenger will be served by the first taxi, but in certain cases a taxi driver may choose to serve a particular destination.
- What is… Leader/Followers
- The line of taxis is the thread pool. The taxis are the event handlers. The passengers are the event sources.

Leader/Followers

- **Structure** :



Microservices

Microservices for $200

- We have a system to manage the academic content of Polytechnique. We have courses, programs, concentrations. Students enter programs, they choose courses, and they receive their degrees based on a concentration (which courses they took). Each department manages its programs and courses independently, but students can also choose courses between departments. When creating new courses, we must submit a course analysis and plan, which will need to be approved by the department. When selecting a course, students are listed in the course catalog, and they are given access and materials. When a program is created, it must include courses outside the department's catalog according to a certain number of credits and be approved by the department and the University. How would you break the system down into microservices? Which would have their own database and which should be shared?

Microservices for $200

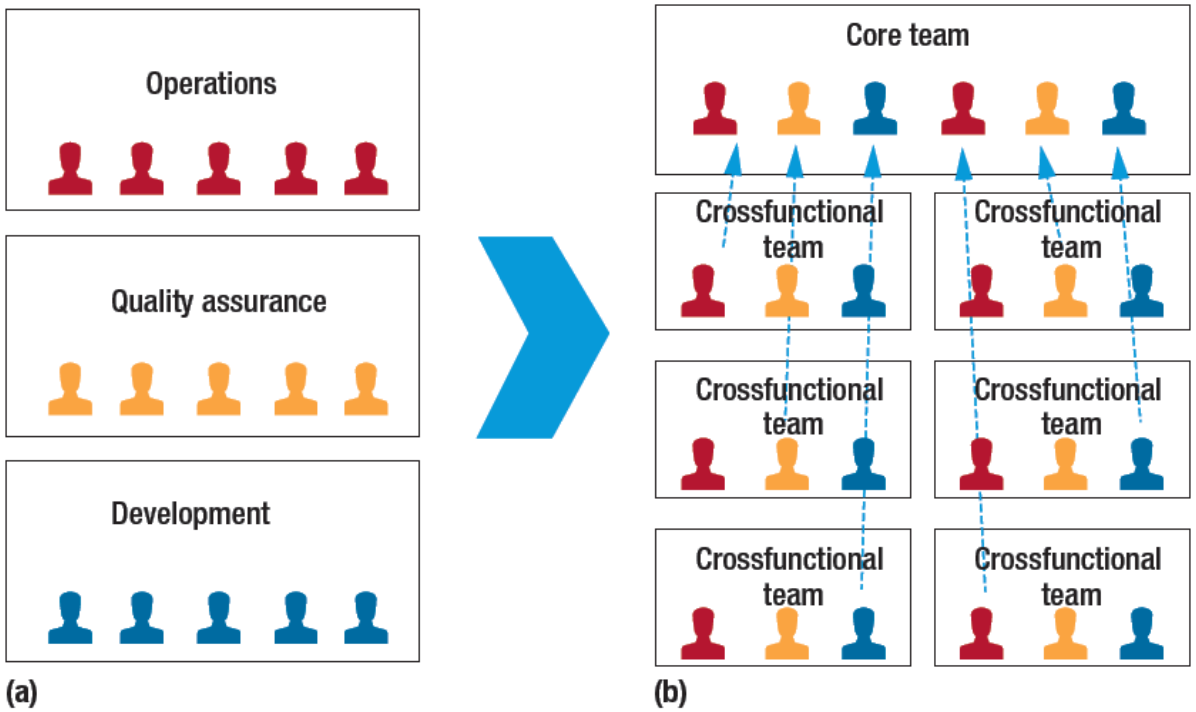
- We have a system to manage the academic content of Polytechnique. We have courses, programs, concentrations. Students enter programs, they choose courses, and they receive their degrees based on a concentration (which courses they took). Each department manages its programs and courses independently, but students can also choose courses between departments. When creating new courses, we must submit a course analysis and plan, which will need to be approved by the department. When selecting a course, students are listed in the course catalog, and they are given access and materials. When a program is created, it must include courses outside the department's catalog according to a certain number of credits and be approved by the department and the University. How would you break the system down into microservices? Which would have their own database and which should be shared?
- Course Creation → Courses DB
- Program Creation → Programs DB (access to Courses DB)
- Course Selection → Access to Courses DB, potential access to a Students DB, eventually access to the Programs BD.
- Each department has its own set of microservices (decompose by domain), the University has one microservice with access to all Programs databases.

Microservices for $400

- We want to design the administration of Polytechnique. The school is concerned with questions of research, teaching and education, infrastructure and spaces, human resources, outreach and others. For each of these subjects, the school has a committee made up of the chairs of similar committees from each department. Departmental committees are made up of professors, students and other employees with specific roles and specialties. How would you break down the organization of these committees? What are the domains and what are the “services”?

Microservices for $400

- We want to design the administration of Polytechnique. The school is concerned with questions of research, teaching and education, infrastructure and spaces, human resources, outreach and others. For each of these subjects, the school has a committee made up of the chairs of similar committees from each department. Departmental committees are made up of professors, students and other employees with specific roles and specialties. How would you break down the organization of these committees? What are the domains and what are the "services"?
- Domains: research, teaching and education, infrastructures and spaces, human resources, outreach and others.
- Services: The departmental committees.
- A central committee on a university level which will coordinate the work of the sub-committees and obtain their contributions.



Microservices for $600

- You are developing a mobile application to help Polytechnique monitor food services at Polytechnique and UdeM. The app contains information about what's available, when it's open, where it's located, what's on the menu and more. Students can log in, search for restaurants / cafes / snack counters, verify their information, leave a review or comment, and provide feedback for the app. You collect the data yourself from each food station on campus. How would you design this application?

Microservices for $600

- You are developing a mobile application to help Polytechnique monitor food services at Polytechnique and UdeM. The app contains information about what's available, when it's open, where it's located, what's on the menu and more. Students can log in, search for restaurants / cafes / snack counters, verify their information, leave a review or comment, and provide feedback for the app. You collect the data yourself from each food station on campus. How would you design this application?
- What is… monolith
- Small team (one person, maybe two), one clear functionality, modifications via the application (add/update/remove a food station).

Microservices for $800

- Your catering app is a huge success! Campus food stations are reaching out to you to send you their new menu items, to post student specials, to get feedback on seasonal menu items. Your friends think it would be a good idea to link the app to social media and allow cross-posting and chatting between app users. Additionally, due to COVID-19, students cannot be on campus all the time, and food services are having difficulty planning their production for people on campus. They ask you to set up a pre-order feature for people who will be on campus on any given day. You get in touch with all 33 of your LOG8430E classmates to extend your application. What type of architecture would you present to them? What components would you use? How will the development team be organized?

Microservices for $800

- Your catering app is a huge success! Campus food stations are reaching out to you to send you their new menu items, to post student specials, to get feedback on seasonal menu items. Your friends think it would be a good idea to link the app to social media and allow cross-posting and chatting between app users. Additionally, due to COVID-19, students cannot be on campus all the time, and food services are having difficulty planning their production for people on campus. They ask you to set up a pre-order feature for people who will be on campus on any given day. You get in touch with all 33 of your LOG8430E classmates to extend your application. What type of architecture would you present to them? What components would you use? How will the development team be organized?
- What is… a microservice architecture.
- One service per business case:
  - Food station service (add/update/remove) → Food station DB
  - Social Media service (connect to a platform, publish comments/questions) → Comments DB
  - Chat service (connect, chat) → Users DB
  - Preorder service → Order DB, potential access to the Food station DB et to the Users DB.
- Interfunctional teams per service (backend, UI, testers, QA, integrators of external systems) no more than 12 people per team.

Exam Sample

Structure

- 15-20 multiple choice questions (2-3 points each)
- 2 “75-point” quiz questions (given a description, specify the style/pattern to use). (10-15 points each)
- 2 “complex” questions (25-35 points each)
- You need at least 40% in the finals to pass the course and allow assignments and quizzes to count.

Multiple-choice

- According to this architectural style, a single task is replicated and executed by multiple nodes.
  - Client-Server
  - REST
  - Blackboard
  - Master-Slave

Complex Question

- [DESCRIPTION]

1. What architectural style or styles would you use to implement this system? Justify and explain your choices. Also, use arguments relevant to quality to support your decision.
    1. NOTE: Do not answer with CBA, MDA, AOA or EDA. Use specific architectural styles from the list.
2. Provide a diagram of your chosen architecture(s) to describe the system.
3. Mention at least two patterns (design patterns or event-driven patterns) that you would use in your system. Provide a UML diagram for each of the patterns. *Do not use the generic diagram of the pattern but make it specific to this system*. Briefly describe the role of each class in the context of the pattern.
4. How would you implement the data part of your system? Comment on the organization of data and whether you will use any special solutions for storing, ingesting, or analyzing data.