The Exploding While Loop Part III

Object-Oriented Software Architecture and Design Patterns

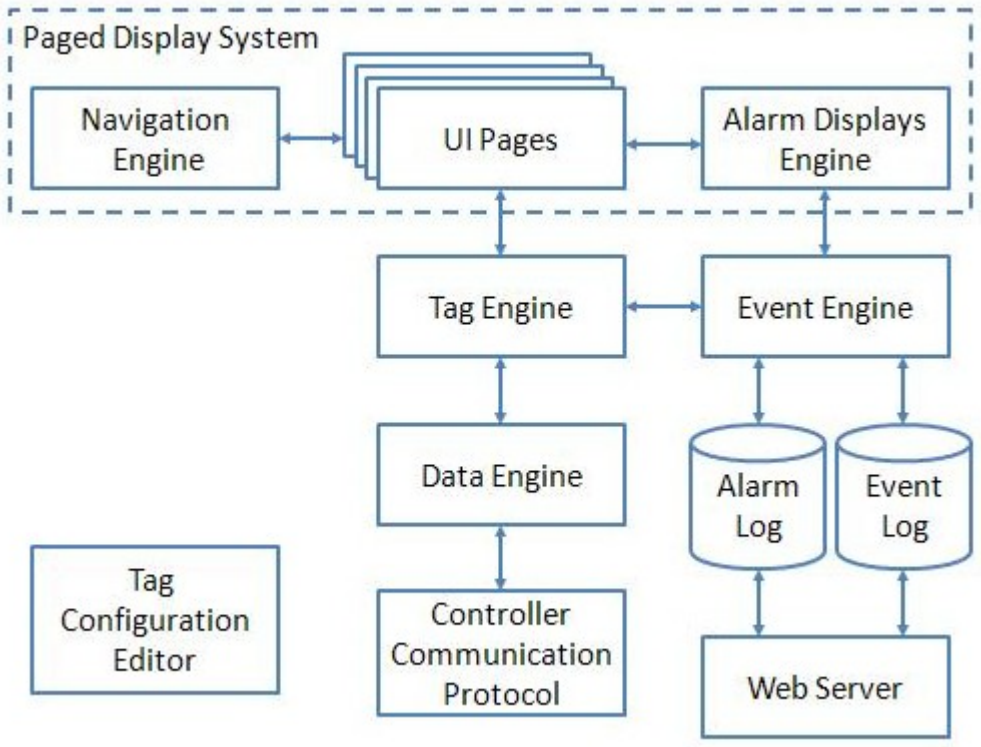
Jeffrey Habets
NI Certified LabVIEW Architect
www.vi-tech.nl

Agenda

- Recap part I & II
- Design patterns I
  - What are they and why use?
  - Design patterns in LabVIEW
- Object-Oriented Concepts
  - What is it?
  - Why use it?
- Design patterns II (Object Oriented)

What is Software Architecture

SA can be defined as a **high-level system design** that '*describes the **sub-systems** and **components** of a software-system and the **relationships** between the components*'.



Why software architecture?

- future extensions / elaboration
- readability / documentation
- development efficiency
- multiple developers
- reuse of code

You are not alone

- Others have thought about SA too, and documented their findings..
- We call these Design Patterns

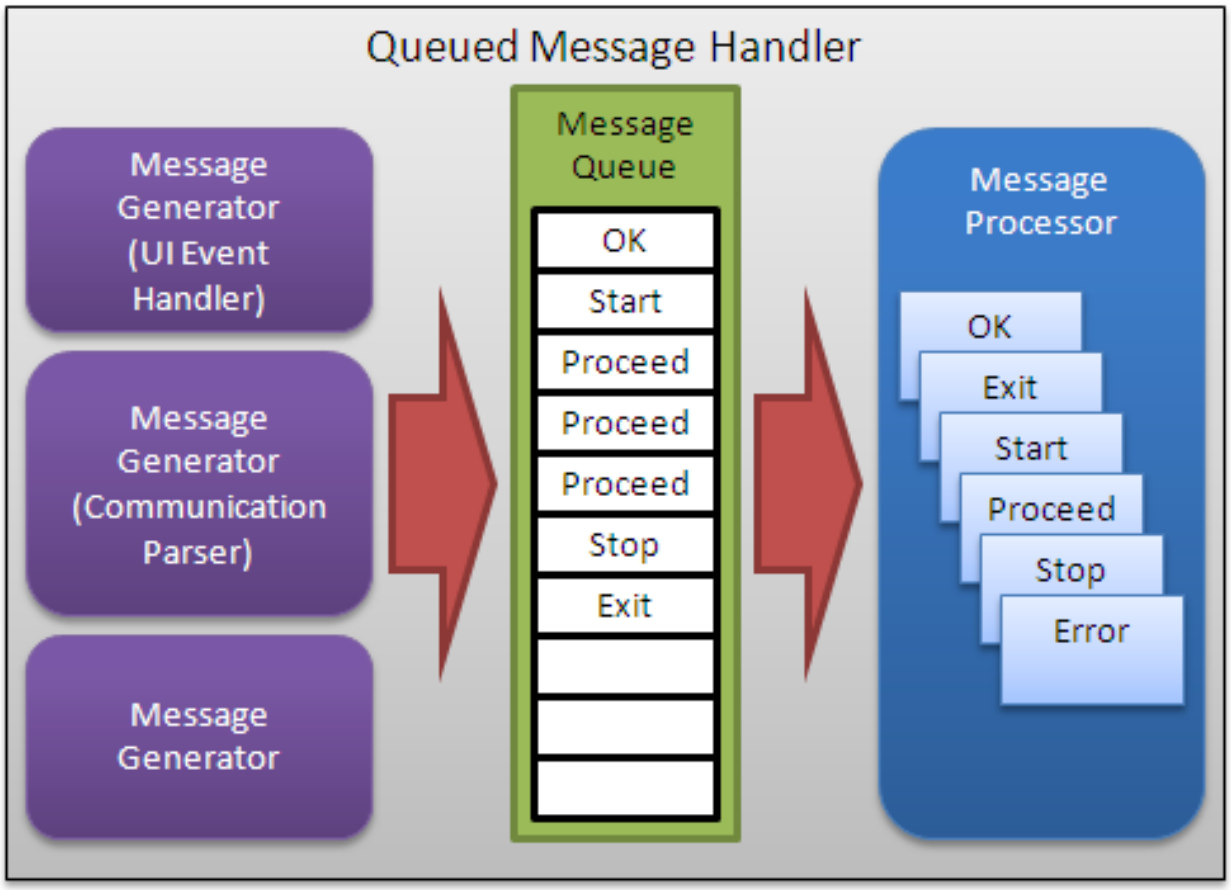
A design pattern is a general reusable solution to a commonly occurring problem in software design

Find some very basic design patterns in LabVIEW:
New... From template → Frameworks → Design patterns

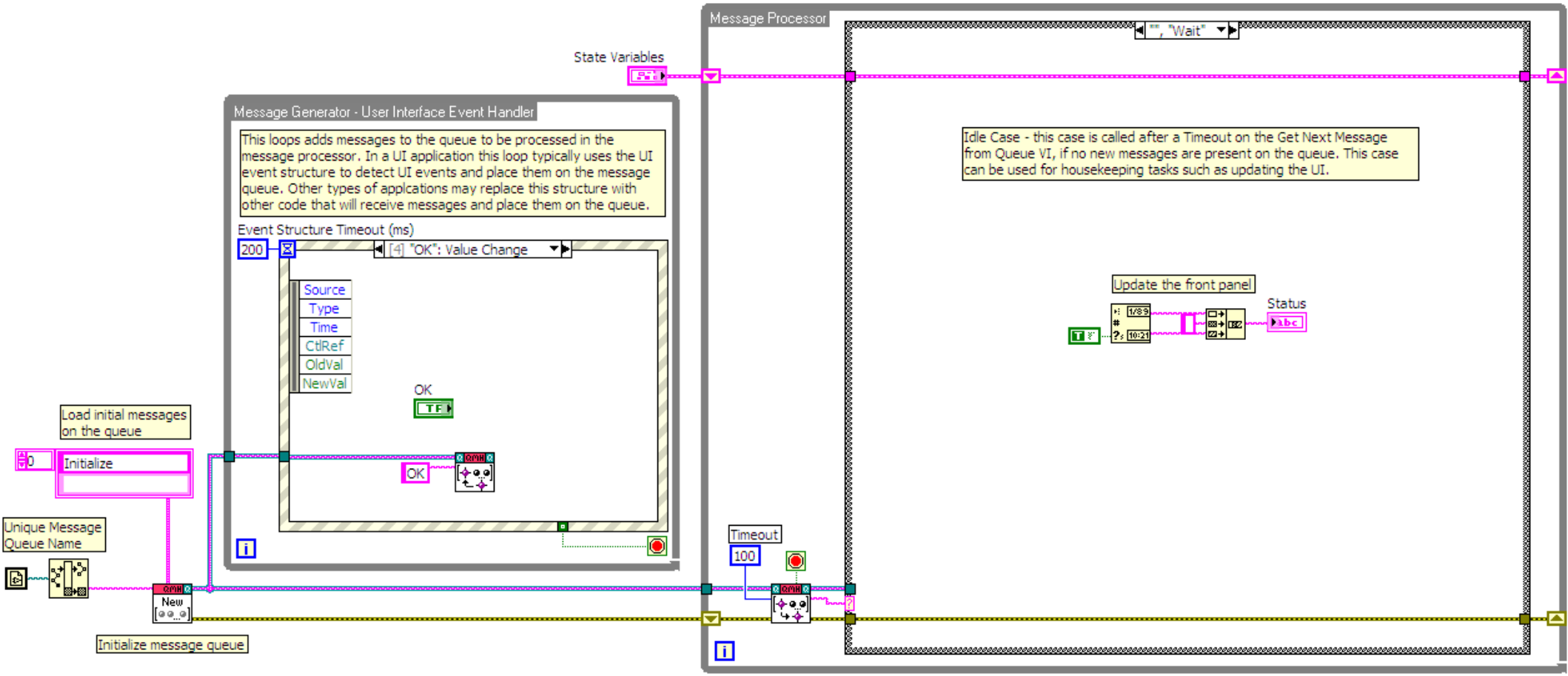
Design Patterns I

- State machine / Queued Message Handler (QMH)
- Functional global
- Active Object
- Observer / Publish-Subscribe
- Template method

Queued Message Handler

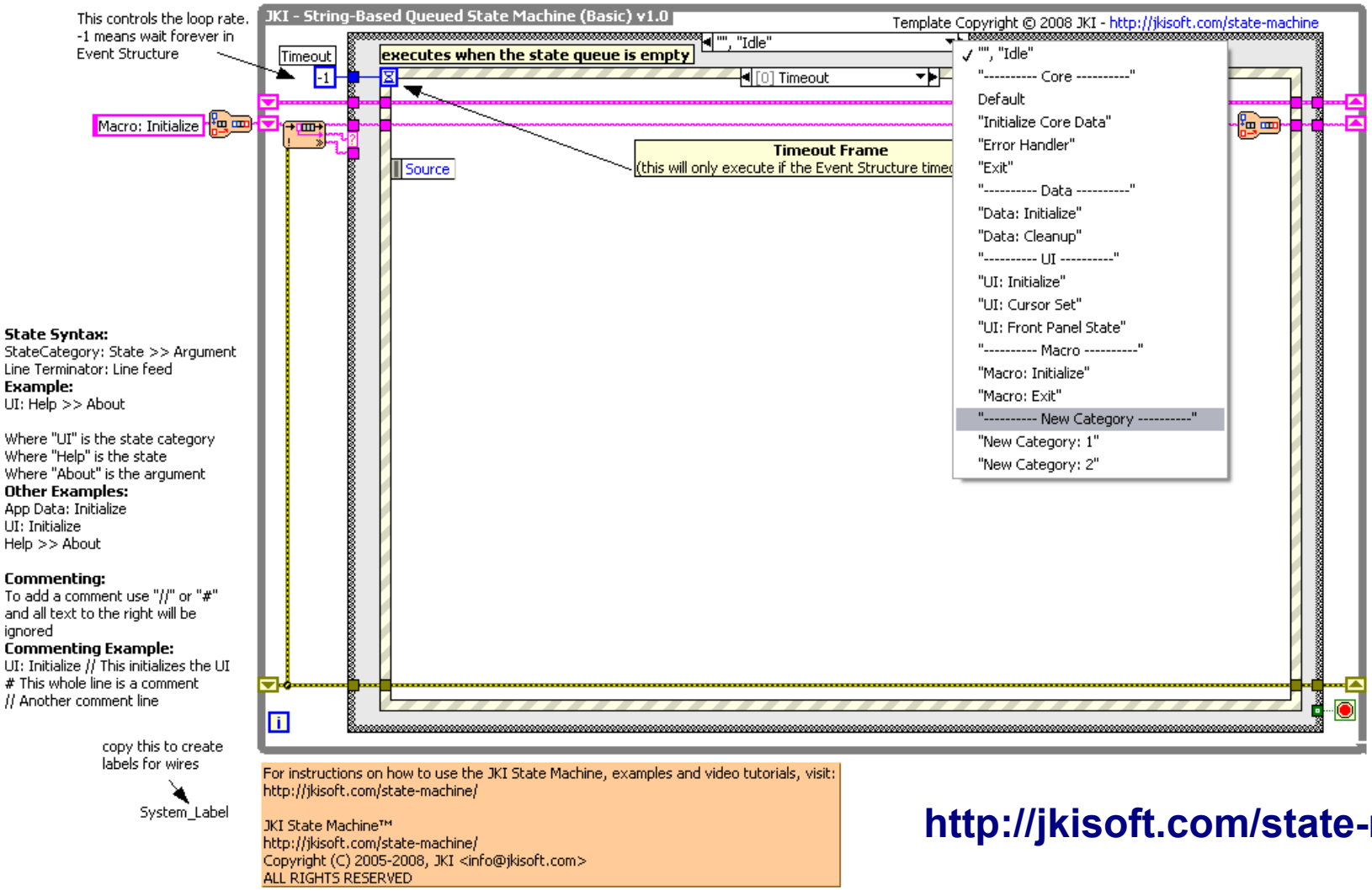


Queued Message Handler



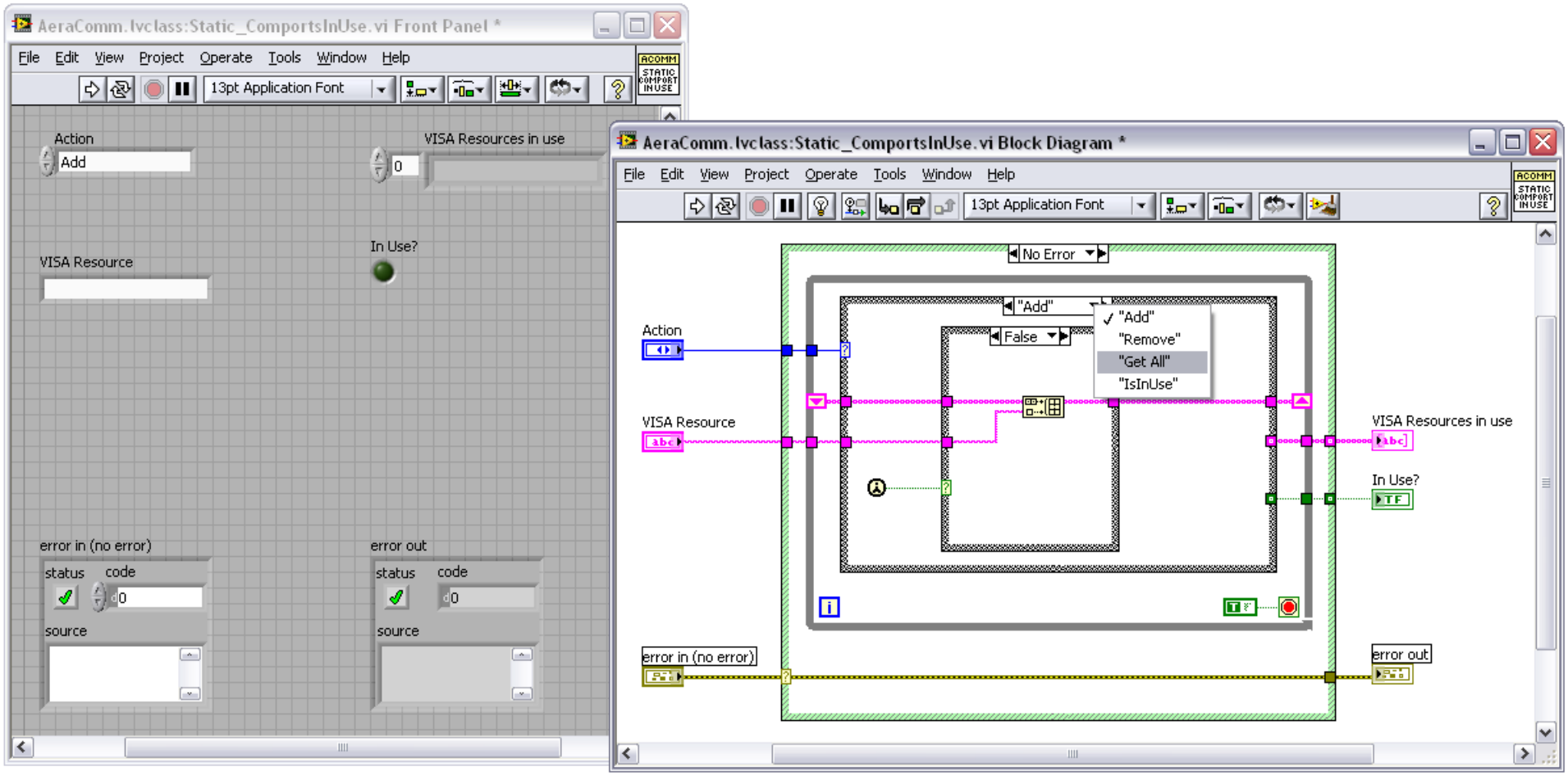
Queued Message Handler Design Pattern and Reference Design Library
http://zone.ni.com/devzone/cda/epd/p/id/6091

Queued Message Handler

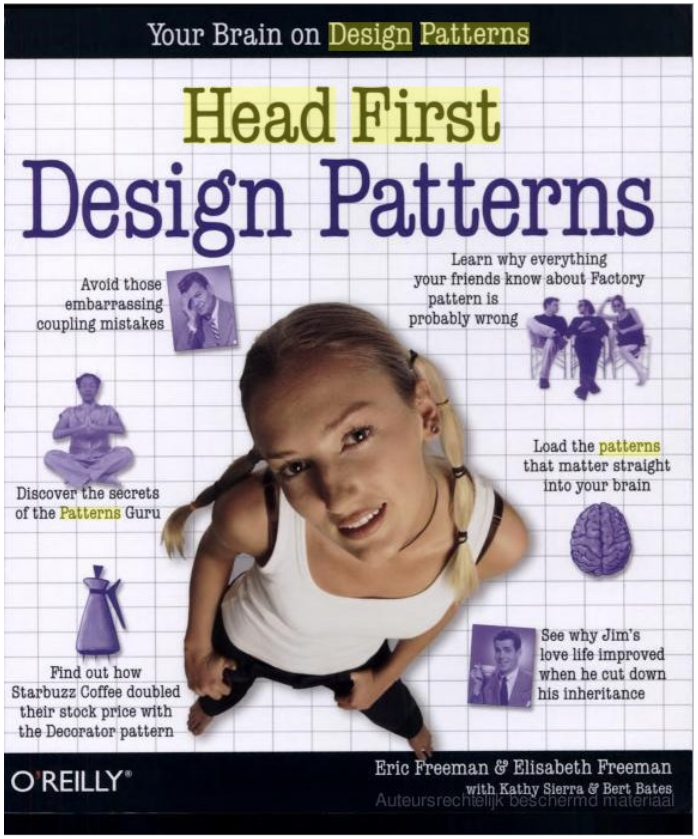


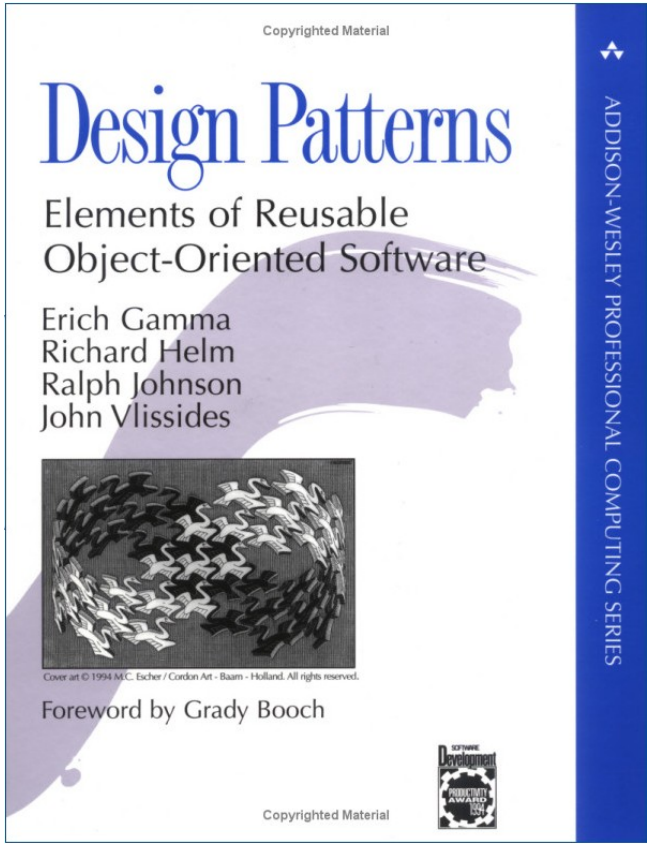
http://jkisoft.com/state-machine

Functional Global / LV2 Global / Action engine



Literature about design patterns





And of course the Internet, just Google “Design patterns”

What Is Object-Oriented Design?

- It's a way of structuring your software
  - OOD requires the programmer to think of a program in terms of objects, instead of procedures / VI's
- An object:
  - Encapsulated data and the methods for accessing that data
  - “Cluster + VI's”
  - Group of VI's with a common responsibility

What Is Object-Oriented Programming?

- OOP uses objects and their interactions to design applications
- OOP is bases on programming techniques such as encapsulation, inheritance and polymorphism

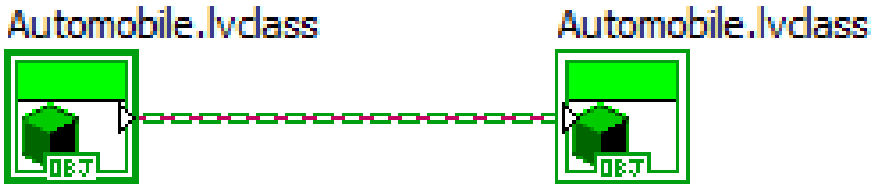
When and why to use Object-Orientation?

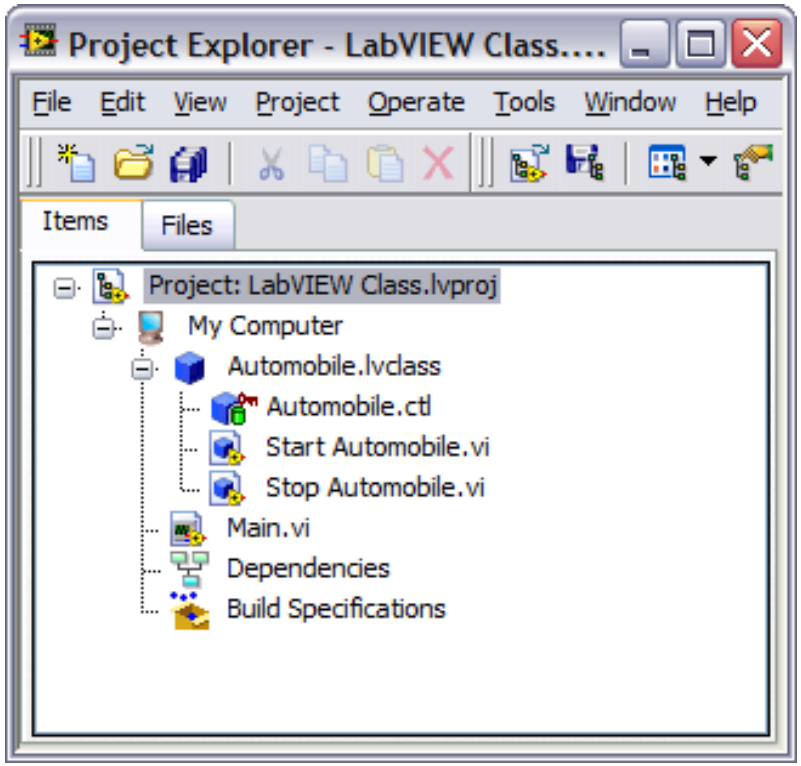
- Use it when you want
  - Encapsulation
  - Inheritance
  - Dynamic dispatching (polymorphism)
- Benefits of OOP
  - Easier to maintain your code
  - Easier to extend your code
  - Easier to test your code
  - Increase of code reuse
  - Benefits increase when the system grows

- Or when you want to start using OO Design Patterns!

What is a LabVIEW class?

- A glorified cluster
- A user-defined data type
- A type of Project Library





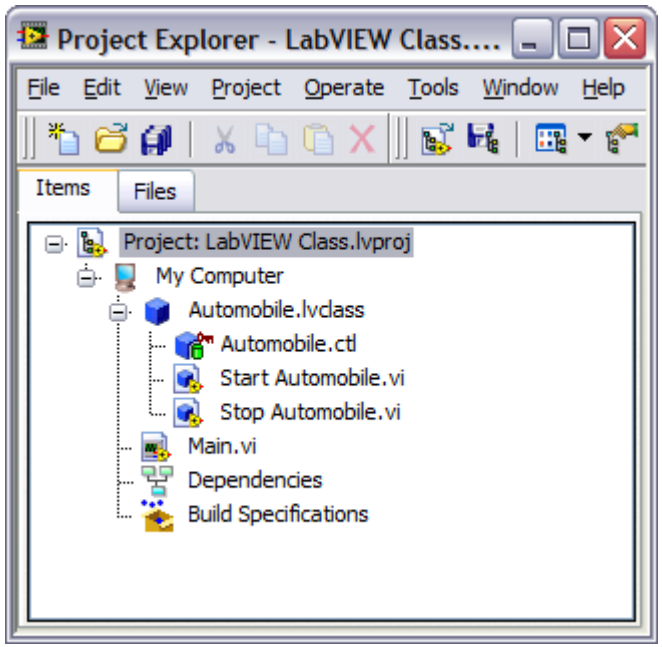
Anatomy of a class

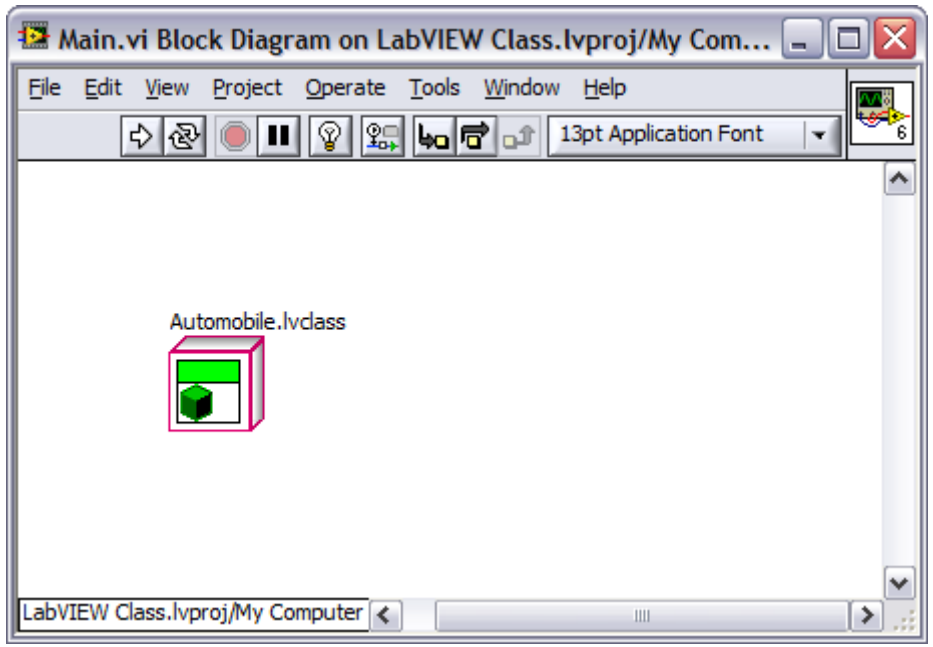
- Each LabVIEW class consists of:
  - A private data control (cluster)
  - Member VIs to access that data

- Class file (.lvclass) stores class information
  - Private data control definition
  - List of member VIs
  - Properties of member VI

What is an Object?

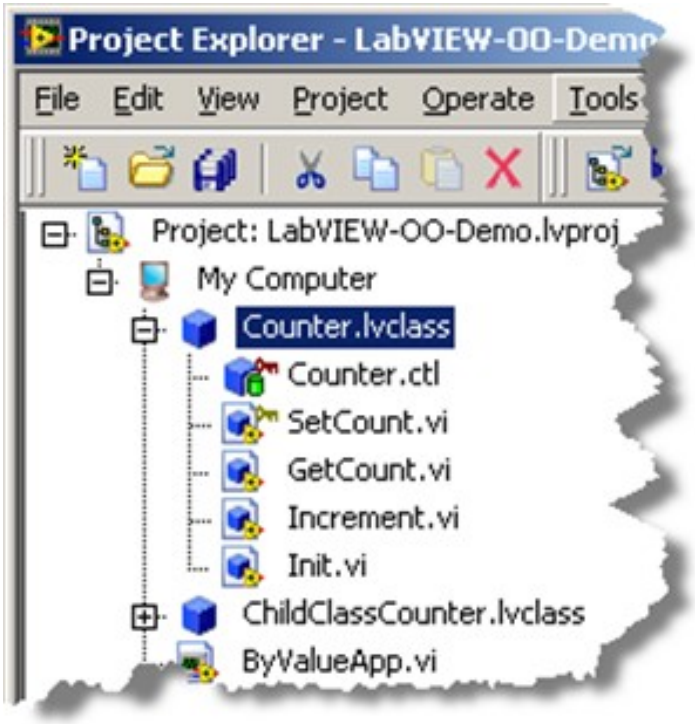
- An object is a specific instance of a class
- Object data and methods are defined by the class





DEMO: A class in LabVIEW

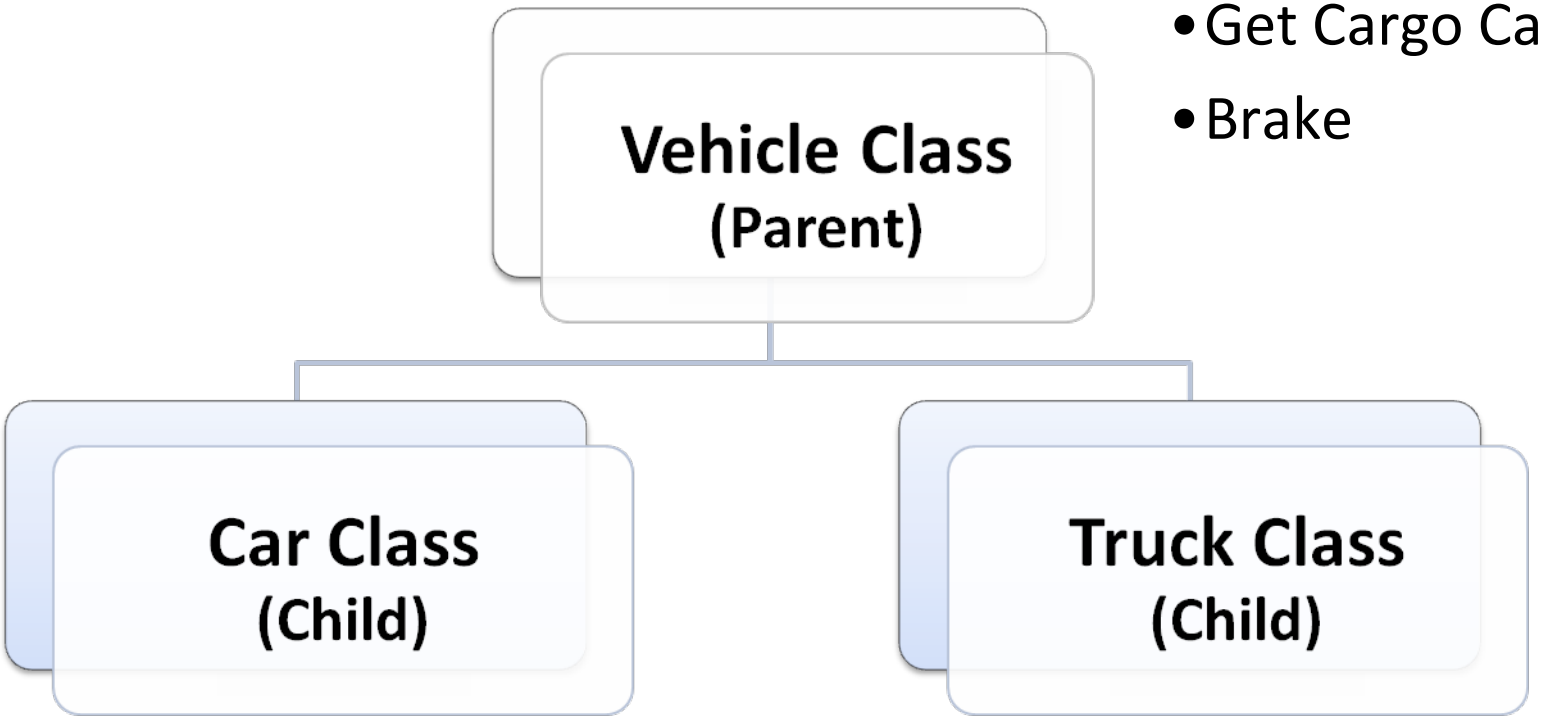
1. Create and explore a class
2. Class: Counter and the By ValueApp.vi
3. Class constant, read-write data
4. Class icon template and wire



What Is Inheritance?

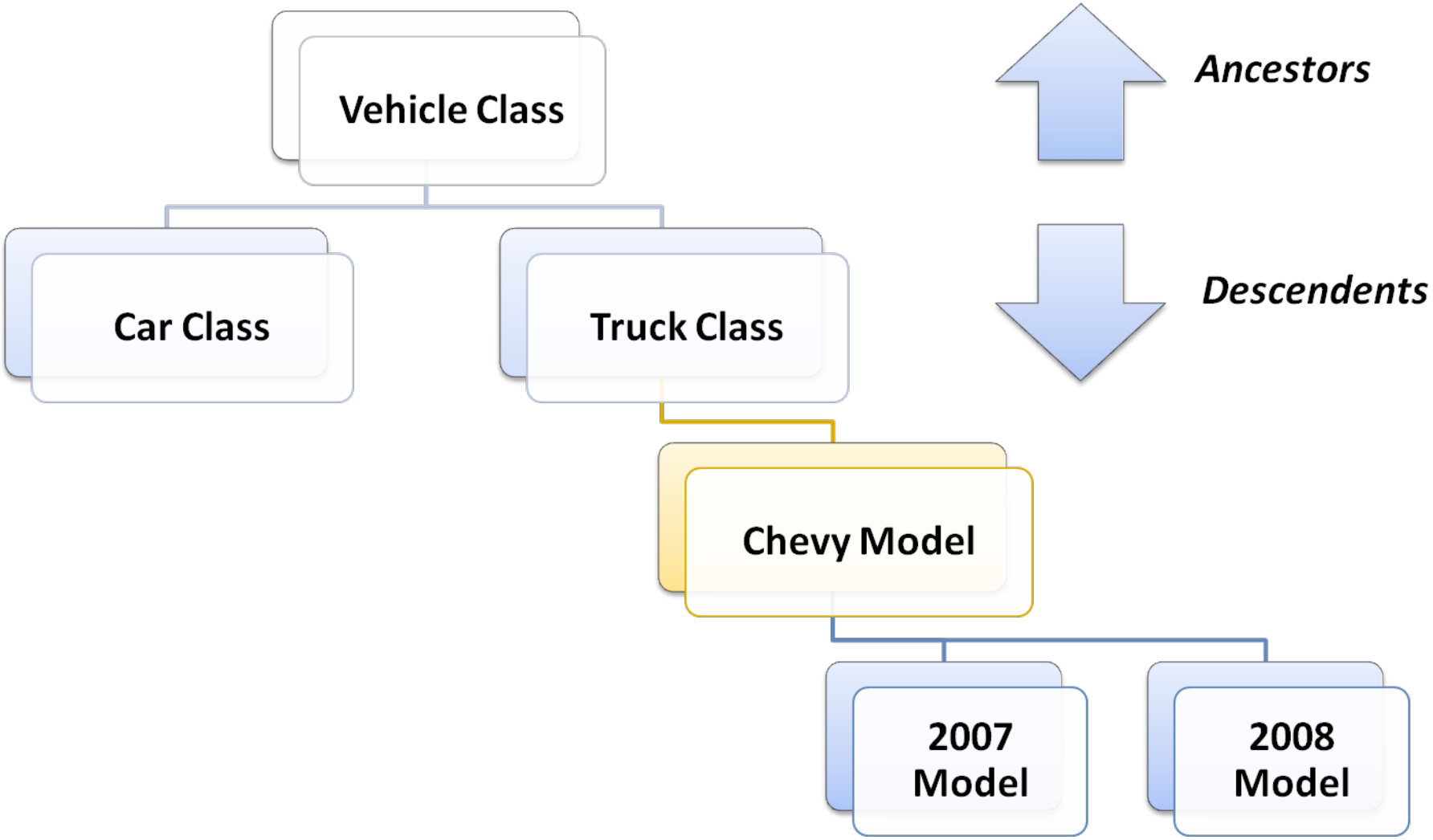
Example methods:

- Initialize
- Get Cargo Capacity
- Brake



A car is a type of vehicle. ***A truck is a type of vehicle.***

Inheritance example

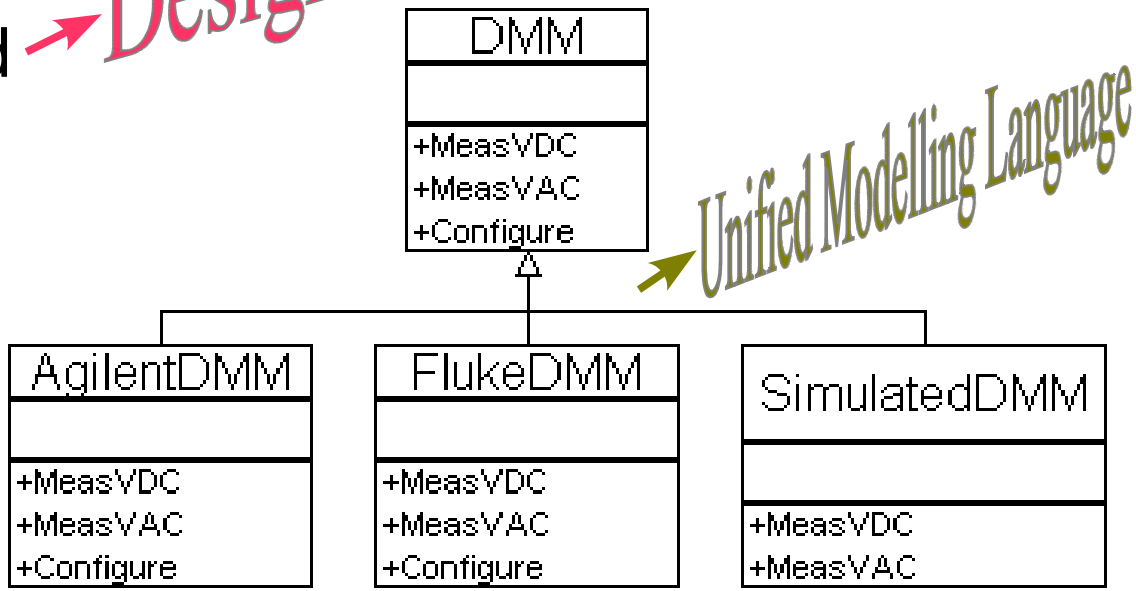


Inheritance

- Creates replaceability between classes which:
  - Inherit from the same ancestor
  - Have the same public VI's (methods)
- Benefits
  - Code reuse combined with specialization
  - Changes to parent propagate to children

Design Patterns!

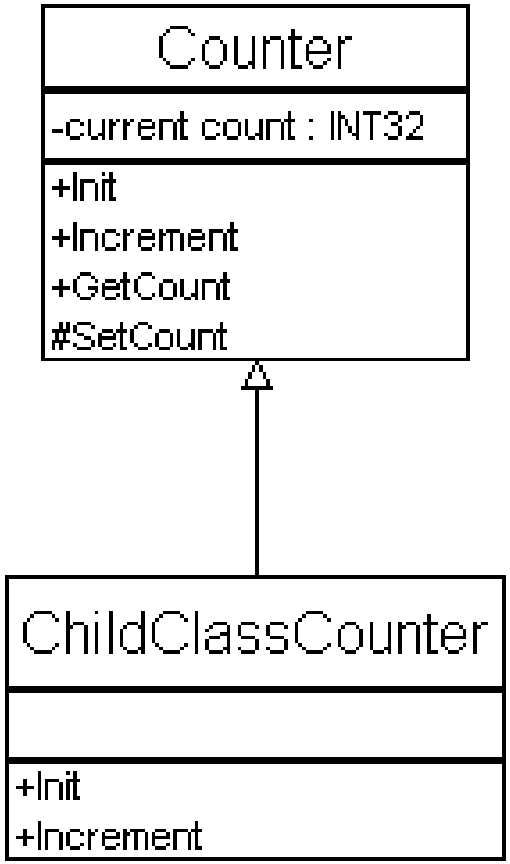
Unified Modelling Language



DEMO: Inheritance in LabVIEW

Init en Increment are “magic”
Dynamic dispatch VI's:

- Same VI name on each class
- Different block diagrams
- LabVIEW chooses which VI to run



DEMO: InheritanceApp.vi

Extension - GOOP

LabVIEW class + Reference

Instead of: Object in the wire

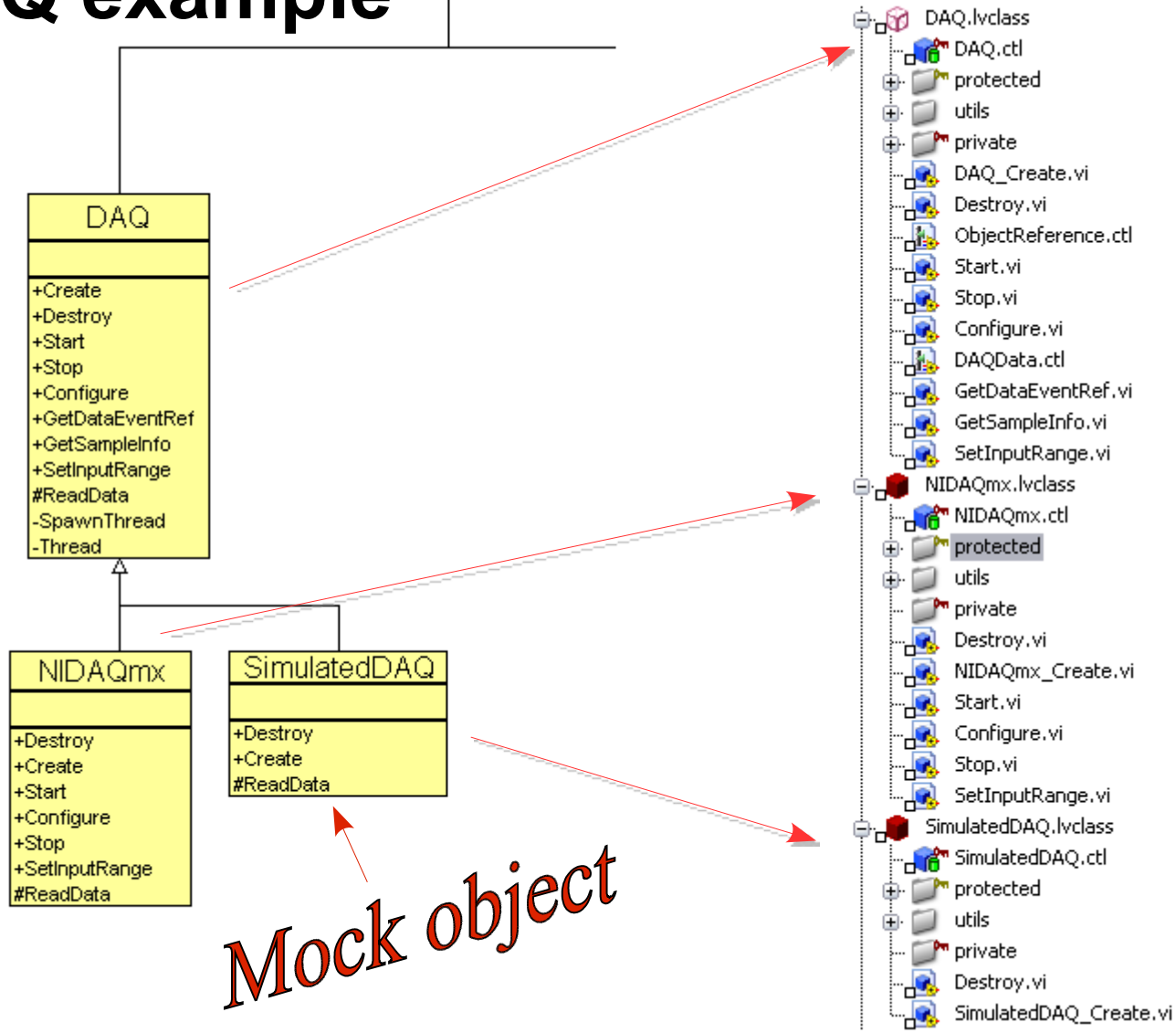
→ Reference in the wire

Gives us control of object creation and destruction

How?

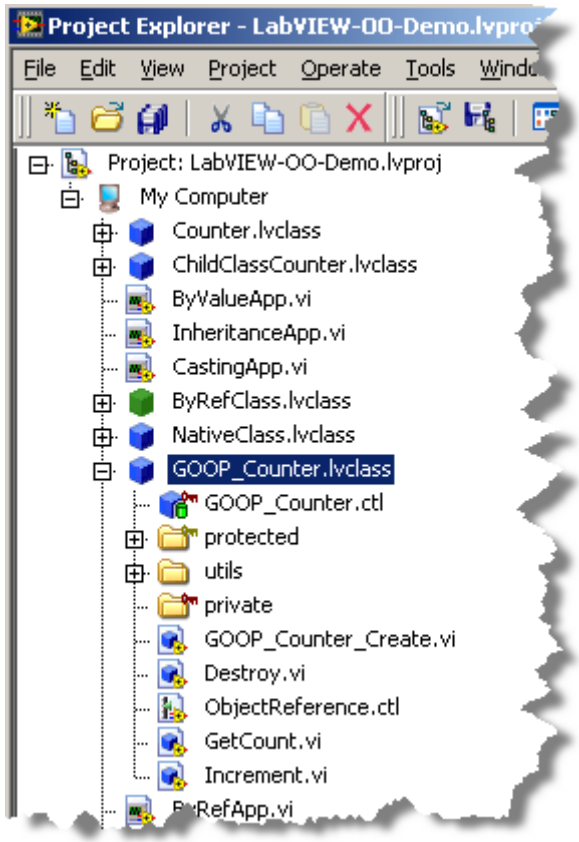
- NI Example Finder → Fundamentals → Object-Oriented → ReferenceObject.lvproj
- 3rd Party reference frameworks and/or tooling
- Future LV versions??

Real world DAQ example



DEMO: GOOP

- ByRefApp.vi
- Creation of a GOOP class
- Explore the tools



Use Case Summary

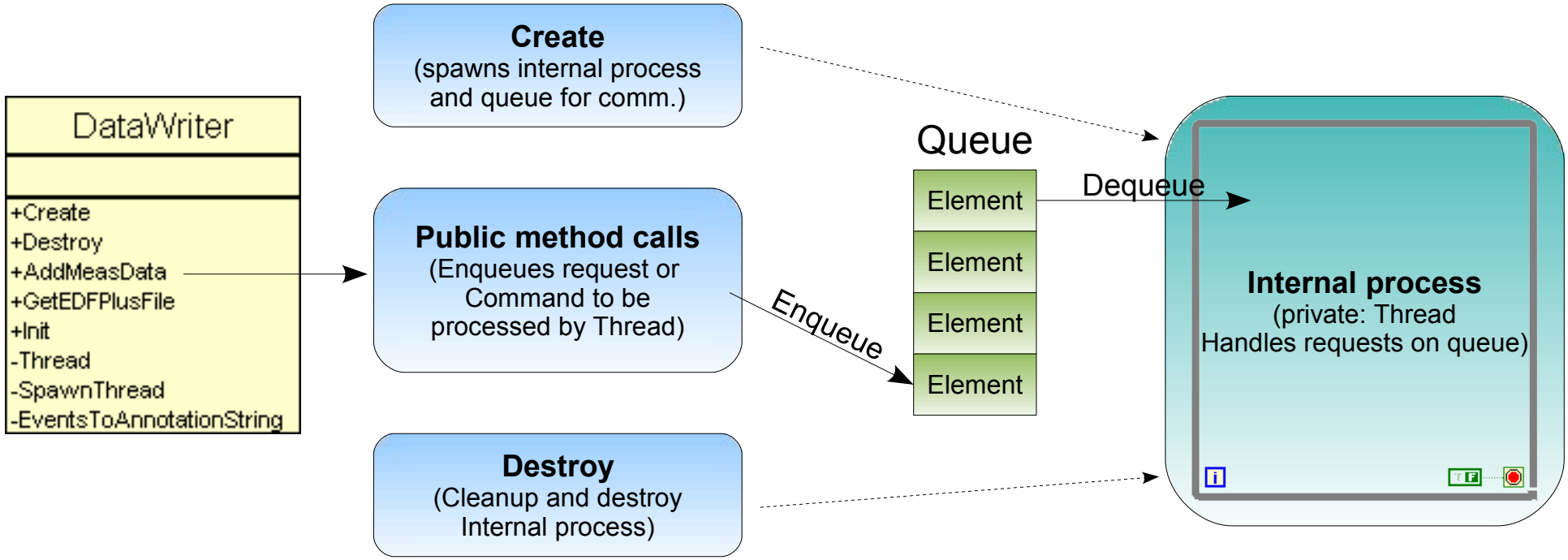
- GOOP
  - Modeling of systemresources / hardware
  - Parallel (R / W) access to object data
  - Tooling!
  - Common design patterns usage
- LVOOP
  - Parallel access to data (without semaphores)
  - Dataflow (replacement of clusters)
  - Native dynamic dispatching

Design Patterns II

- State machine / Queued Message Handler (QMH)
- Functional global
- Active Object
- Observer / Publish-Subscribe
- Template method

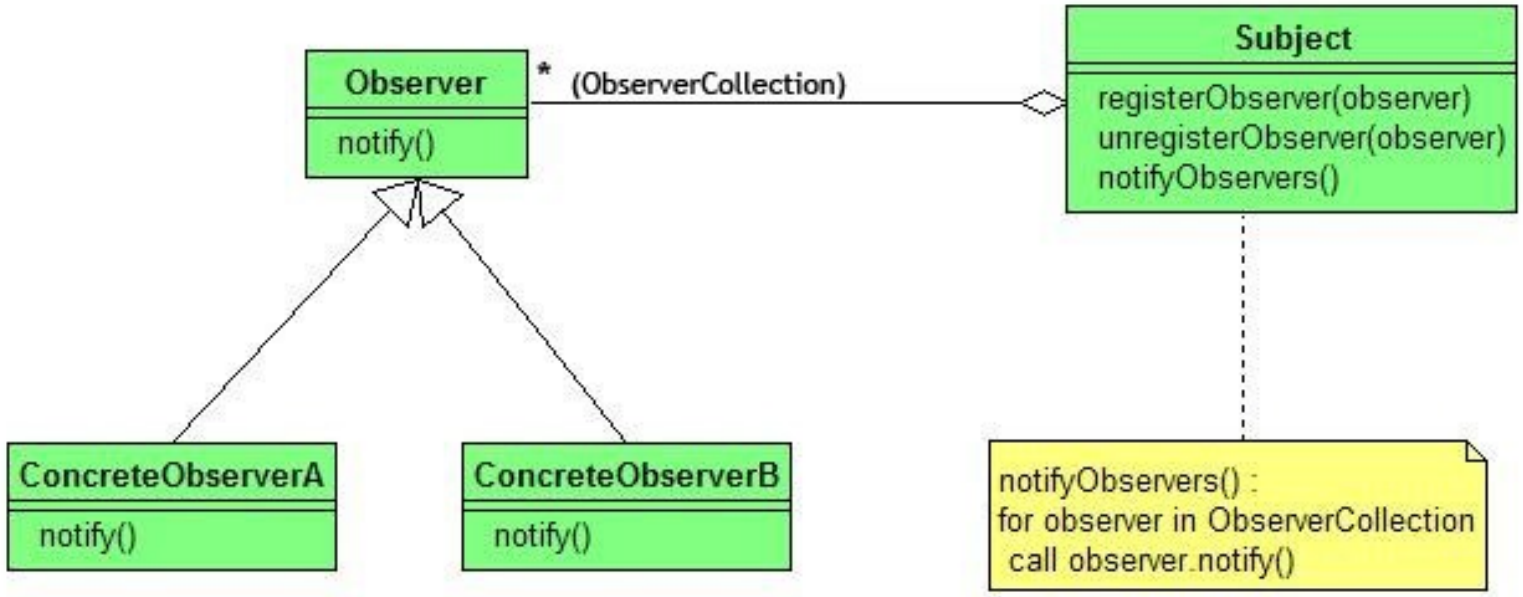
Active Object

Wikipedia: "The Active Object design pattern decouples method execution from method invocation that reside in their own thread of control. The goal is to introduce concurrency, by using asynchronous method invocation and a scheduler for handling requests."

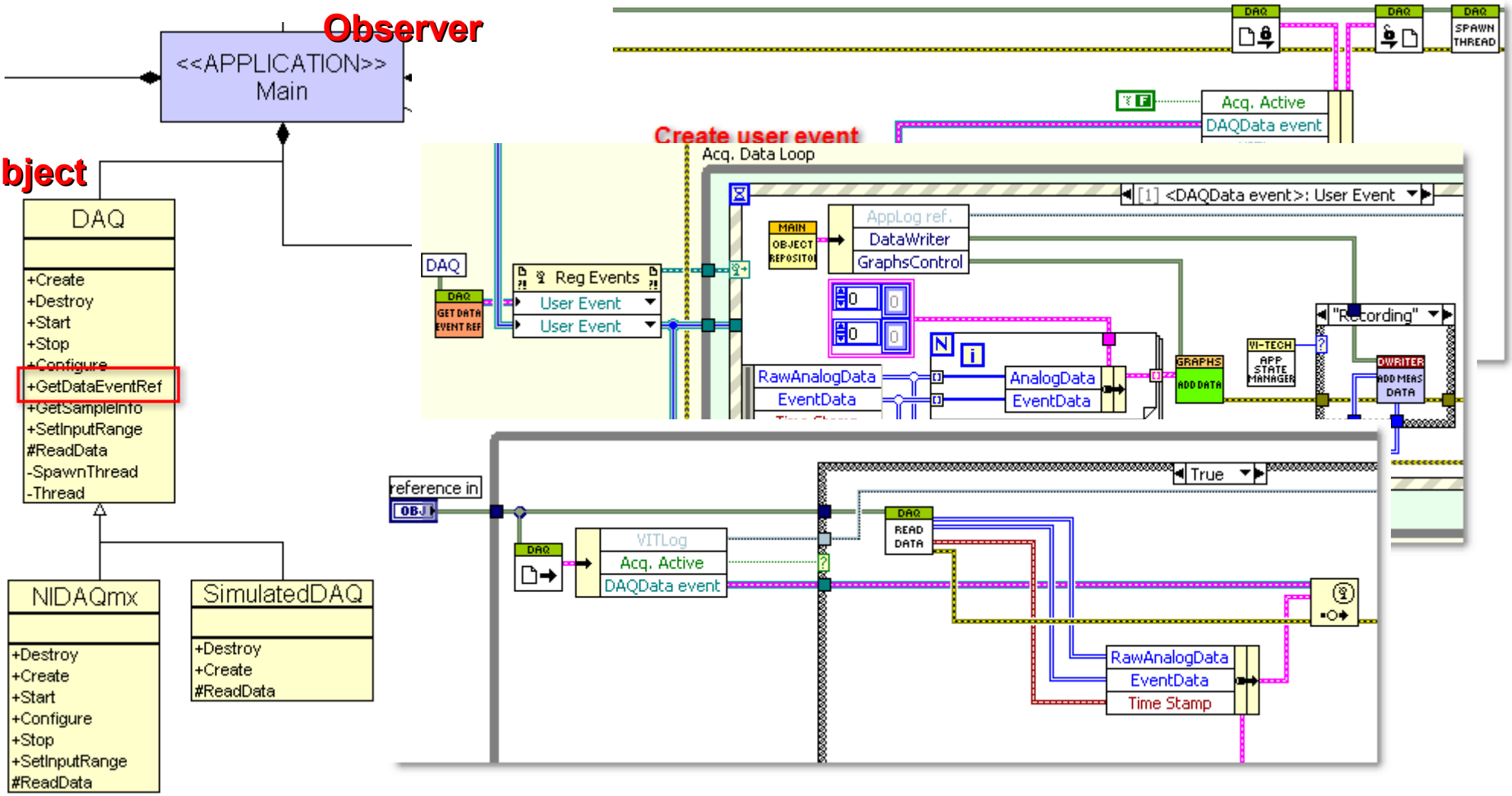


Observer / Publish-Subscribe

Wikipedia: "The observer pattern (sometimes known as publish/subscribe) is a software design pattern in which an object, called the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods. It is mainly used to implement distributed event handling systems."

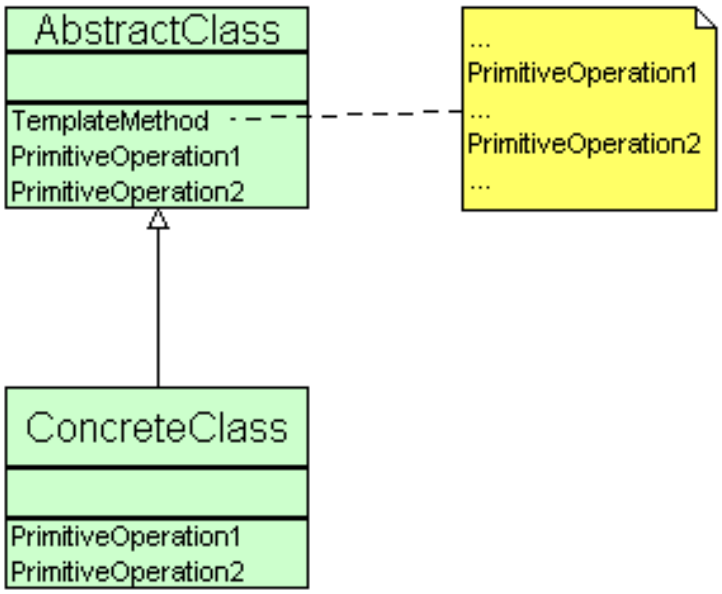


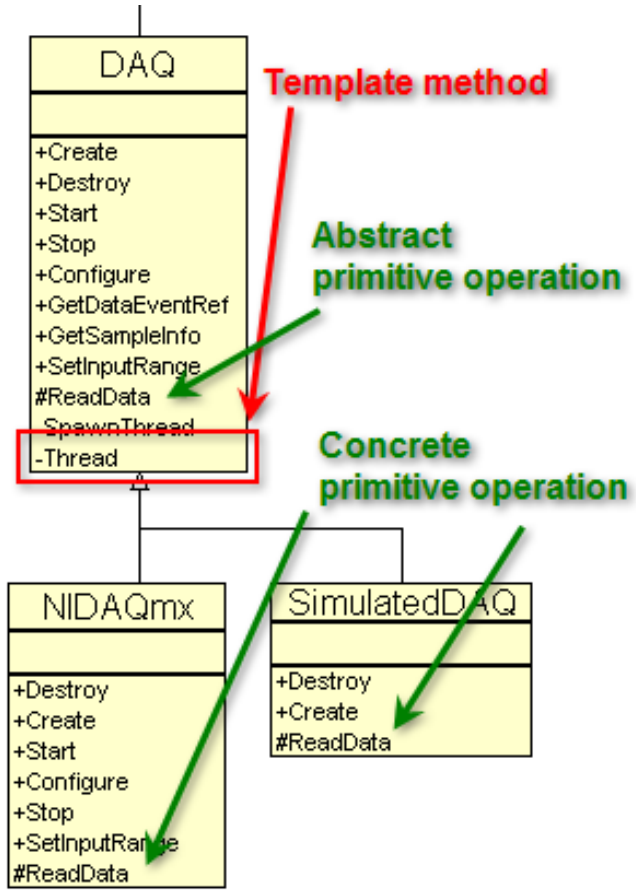
Observer / Publish-Subscribe



Template method

GoF: "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."





Other interesting patterns

- Abstract factory
- Composite (naturally!)
- Facade
- Proxy
- Command

Resources and acknowledgments

LabVIEW Object-Oriented Programming FAQ
http://zone.ni.com/devzone/cda/tut/p/id/3573

Expressionflow – Blog by Tomi Maila
http://expressionflow.com/

GOOP on LAVA
http://forums.lavag.org/GOOP-f68.html

Endevo – Makers of Goop Development Suite and UML Modeller
http://www.endevo.se/content/blogcategory/18/103/lang,en/

LabVIEW Examples – Fundamentals → Object-Oriented

VI Technologies (Training Graphical Object Oriented Programming 19/20-05-2009)
http://www.vi-tech.nl/

NI – Applying Text-Based Design Patterns Using LabVIEW Object-Orientation
http://decibel.ni.com/content/docs/DOC-2875