The Watcher

An OO Development Case Study

Gil Barmwater
(gil_b@bellsouth.net)
Independent Consultant

Overview

- Background
- The Problem
- The Design
- The Implementation

Background

- Consulting Client: Safe Data, Inc.
  - Business: Data Collection and Data Reporting
  - Data Exchange: Fax,BBS -> FTP
- Personal
  - Long programming career
    - Witnessed evolution of technology
  - New to OOP/OOD
    - All of my experience was procedural

The Problem

- SDI's customers need to update "master files" used by SDI applications
  - Normally a "scheduled" event:
    - Customer is expected to put the file(s) on a predefined schedule
    - SDI retrieves those files after a "grace period" and processes them

The Problem (2)

- First Problem:
  - Customer puts the file(s) "late"
- Second Problem:
  - Customer needs to add an event; i.e. update their files before the next scheduled time

The Problem (3)

- First Problem Solution:
  - Failure to retrieve file(s) results in "rescheduled" processing until successful
- Second Problem Solution:
  - Phone call from customer causes "manual scheduling" of file retrieval and processing

The Design

- Goal was to make this update process more "automatic"
  - Less dependent on fixed schedules
  - Less resource consuming:
    - Computing resources
    - People resources

The Design (2)

- Lee Peedin found a Windows mechanism that could provide "notification" of file or folder modifications
- "Proof of concept" program developed:
  - Mechanism example was written in VB
  - "Translation" to ObjectRexx
  - Addition of Sockets code for notification
    - Security issue

The Design (3)

- My goal was to implement the functionality of Lee's program using the OO paradigm
  - Most of my usage of OO to that point was as "extended" BIFs; programs were still procedural
    - Had done some enhacement of OO sample programs but hadn't changed the basic design, the Objects themselves

The Design (4)

- Attempted to identify the objects needed to implement the desired functionality:
  - Did not use any formal OOD methodology (problem was too simple)
  - Didn't really focus on the data but on the processing required
  - Decided on three principal objects:
    - Watcher - implements the Windows mechanism
    - Sender - implements the Sockets communication
    - Historian - records the program activity

The Implementation

- Code development went very quickly
- Testing done by parallel operation of Lee's code with comparison of results
- Staff extremely happy with the result
- Customers happy as well