

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

The Problem

- SDI's customers need to update "master files" used by SDI applications
 - f 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:
f Customer puts the file(s) "late"
- Second Problem:
f Customer needs to add an event; i.e. update their files before the next scheduled time

The Problem (3)

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

The Design

- Goal was to make this update process more "automatic"
- f Less dependent on fixed schedules
- f 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:
 - f Mechanism example was written in VB
 - f "Translation" to ObjectRexx
 - f 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
 - f Most of my usage of OO to that point was as "extended" BIFs; programs were still procedural
 - Had done some enhancement 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:
 - f Did not use any formal OOD methodology (problem was too simple)
 - f Didn't really focus on the data but on the processing required
 - f Decided on three principal objects:
 - Watcher - implements the Windows mechanism
 - Sender - implements the Sockets communication

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