

# Development Considerations for ArcGIS Desktop Add-Ins

David A. Howes  
Integral GIS, Inc.

May 11, 2011



# Objective

Provide suggestions to help improve programming practices using as an example an ArcGIS Add-in for network data development

# What are Add-Ins?

- ArcMap demo
  - An upgrade of existing ArcGIS tools
  - Bookmark manager
  - Attribution manager
  - Alert manager

# Development Process

- Understand the existing tools
- Design and build new forms
- Incorporate existing functionality
- Test and fix
- Deliver

# Events

- Set up a procedure to run when something happens
- Can be tricky to manage
  - be careful
  - don't be over-ambitious

# Form Flow

- Start with an empty container
- Add a form user control
- Do things with the form
- Close it
- Event fires in response
  - Removes form control from container
  - Adds the next form to the container

# Error/Alert Handling

- It's more than just try/catch
- What do you want to happen if an error occurs?
  - Let the program crash?
  - Let the user address the error (if they can) and then continue?
  - Show/log the details of the error?
  - Provide some instructions on how to address the error?

# Incorporating Existing Code

- Re-use as much code as possible

# Add-In Construction

- Visual Studio demo
  - Solution components
  - Configuration file

# Programming Considerations

- Use events wisely
- Handle errors cleanly
- Adopt a modular approach
- Separate logic into (logical) groups
  - E.g., User interface vs. underlying functionality
- Consider a test-driven design approach
  - Integrated testing
  - Unit testing

# General Suggestions

- Make clarity your ultimate goal
- Keep things simple
- Be realistic
- Resist the temptation to start coding too soon
- If you're under pressure to start coding sooner than you'd like, stand your ground, but always be clear that you're acting in the best interests of your boss and your client

# General Suggestions

- Keep plenty of backups
- Think carefully about the implications of changes
- Blind alleys aren't always a bad thing, as long as you learn from them

Copyrighted Material

Microsoft

# CODE COMPLETE

2  
Second Edition



A practical handbook of software construction

**Steve McConnell**

Two-time winner of the *Software Development Magazine* Jolt Award

Copyrighted Material

# McConnell - Selected Key Points

1. The overarching goal of preparing for software construction is risk reduction.
2. If a good problem definition hasn't been specified, you might be solving the wrong problem during construction.
3. Part of a programmer's job is to educate bosses and coworkers about the software development process, including the importance of adequate preparation before programming begins.

# McConnell - Selected Key Points

4. The kind of project you're working on significantly affects construction prerequisites - many projects should be highly iterative, and some should be more sequential.
5. Good design is iterative; the more design possibilities you try, the better your final design will be.

# McConnell - Selected Key Points

6. Ask yourself whether the programming practices you're using are a response to the programming language you're using or are controlled by it.

Program *into* the language, rather than program *in it*.

7. Software's Primary Technical Imperative is *managing complexity* - focus on simplicity.

# Questions?

[david@integralgis.com](mailto:david@integralgis.com)

