



**ProObject's Technical Education Series Presents...**

***“Patterns, Frameworks, and Middleware:  
Their Synergistic Relationships”***

**By: Doug Schmidt, PhD.  
Professor at Vanderbilt University**

**When: August 11, 2010 (Wednesday)**

**Time: 8:00 AM – 5:00 PM**

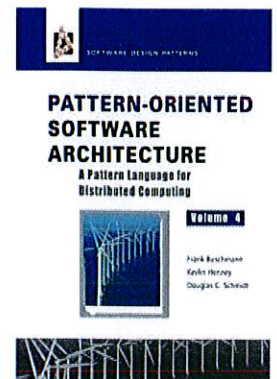
**Where: ProObject's Training Facility  
7467 Ridge Road, Suite 310  
Hanover, MD**

**POC: Dave Gray / John Paul Howell**

**RSVP: Mindy Ward [maward@proobject.com](mailto:maward@proobject.com)**

**Or 410-993-1699**

**RSVP by Wednesday, 8/4/2010**



**This is a free event!  
Come celebrate ProObject's 15 Year Anniversary!**

There will be a “meet the speaker” reception immediately following the seminar.  
Breakfast, lunch and snacks will be provided.

***Pattern-Oriented Software Architectures  
Patterns and Frameworks for  
Concurrent and Networked Software***

**Dr. Douglas C. Schmidt**  
**Vanderbilt University**

Developing concurrent and networked software is hard; developing high quality reusable concurrent and networked software is even harder. The principles, methods, and skills required to develop such software are best learned by understanding how to create and apply patterns and frameworks. A pattern is a recurring solution to a standard problem. When related patterns are woven together they form a “language” that provides a process for the orderly resolution of software development problems. Frameworks can be viewed as concrete realizations of patterns that facilitate direct reuse of design and code.

This tutorial describes how to apply patterns and frameworks to alleviate the complexity of developing concurrent and networked software. These patterns and frameworks have been used successfully by the speaker on production systems at hundreds of commercial companies for telecom/datacom, network management, electronic medical imaging systems, real-time avionics and aerospace systems, and computational finance. The tutorial illustrates by example how patterns and frameworks simplify and enhance the development of concurrent and networked software via the use of:

- Object-oriented design techniques—such as patterns, layered modularity, and data/control abstractions
- Object-oriented language features—such as inheritance, dynamic binding, and parameterized types
- Middleware—such as object-oriented frameworks for host infrastructure middleware (such as ACE and the Java Virtual Machine) and Service-Oriented Architecture (SOA) and component middleware (such as Enterprise Java Beans, the CORBA Component Model, and Web Services)

Much of the material presented in this tutorial is based on the “Pattern-Oriented Software Architecture” books, which are described at <http://www.cs.wustl.edu/~schmidt/POSA>.