ABSTRACT

The rapid growth of technology coupled with the tightened development time and production cost constraints have imposed a tremendous pressure and desire for software enterprises to create new and innovative designs to respond to a rapidly changing business environment. Enterprises must invest in building stable architectures that can be ready to be adapted in many different ways to meet the new challenges. These kinds of architectures are called architectures on demand as they can be adapted accordingly to meet the future requirements and changes in the system. The primarily focus of this tutorial is to show how software stability concepts are used to develop on-demand architectures.

The tutorial focuses on three key aspects: (1) Enduring Business Themes (EBTs) or business goals and transformations which we call stable analysis patterns, (2) Business Objects (BOs) or business process design which we call stable design patterns, and (3) Industrial Objects (IOs) or application objects. Both EBTs and BOs form a stable core, and hence, provide architectures on demand for any domain. We call these architectures, stable architectural patterns.

Our approach is based on an iteratively refined framework that we call it “software stability model” which consisting of three major phases:

1. **Enduring Business Themes (EBTs)** that represent the goals of the business and govern the business transformation. We also call them “Stable Analysis Patterns”. Their impact on business design includes a technique for dividing an enterprise into self-contained business components in order to identify opportunities for innovation and improvement, and prioritize business transformation activities. These components may span a single department of an enterprise, multiple departments inside an enterprise or an ecosystem of partner enterprises. This will be achieved through the external adaptation of stable analysis patterns.

2. **Business Objects (BOs)** that are the capabilities or the work-horses of the enterprise. BOs are called “Stable Design Patterns”. This contains the workflow of the business, business process design and infrastructure; such as hooks, a gang of four patterns, and similar
concepts. BOs focus on the operational aspects of business transformation. They seek to design and implement flexible processes that integrate cross-divisional and cross-enterprise processes as well as core process activities. BOs are key to business and support processes such as finance and human resources. Flexibility is realized by separating business logic, application logic and data management in order to improve the time take to evaluate the new or modified processes. This phase focus only on the business logic and has extension points called hooks.

3. Industrial Objects (IOs) that represent the application logic. The application logic can benefit from standards-based virtualization, automation and integration technologies through hooks that are offered by the on demand-operating environment. Such technologies enable the alignment of IT processes with business priorities, improve asset utilization and reduce the total cost of ownership of the infrastructure through the instantiation of many applications using the architecture on demand concept.

EBTs and BOs are stable software patterns and a combination of EBTs and BOs form the core knowledge for a given domain. The core knowledge for any domain is called a Stable Architectural pattern that can be extended and adapted through hooks. The quality of stable architectural patterns creates competitive advantages through differentiation and productivity. It also integrates partners in order to increase adaptive capabilities.

The tutorial answers the following questions:
1. How can we achieve software stability over time and extend the life span of software products in any domain?
2. How can we build a stable architecture that can be adapted on-demand to meet future changes and evolving technologies?
3. What are the relationships between software architecture and software that has been stable over time?
4. What are the relationships between software that has been stable over time and business objects?
5. How does one build timeless architectures for any field of knowledge?

The tutorial also discusses several issues related to building systems from patterns. This tutorial answers the following questions: 1. Are the various claims related to building any system from patterns stand? 2. What do we mean when we say “systems of patterns”? 3. What are the various claims related to patterns composition, are they true? 4. If you would like to build a system from patterns, how do you select patterns? 5. What kind of patterns does one select for building your system from patterns? 6. How do you compose patterns? 7. Are there any guidelines for patterns composition? 8. What other ways will help you build any system from patterns?

TABLE OF CONTENTS

1. Software Stability Model 90 Minutes
  1.1 Why Software Stability Modeling?
  1.2 The Importance of Software Stability
  1.3 Problems Facing Today’s Software Development
  1.4 What is Software Stability Modeling?
    1.4.1 Enduring Business Themes
    1.4.2 Business Objects

© M.E. Fayd, PhD, 2000-2004
1.4.3 Industrial Objects
1.5 Software Stability Advantages
  1.5.1 Contributes to Stable Architectures
  1.5.2 Aids in Understanding the Problem?
  1.5.3 Provides Accurate Problem Modeling
  1.5.4 Aids Engineering of a System of Patterns
1.6 Software Stability Challenges
1.7 Documentation Template
1.8 Pattern Specifications
1.9 Case Studies

2. Design Session-I 30 Minutes

Break 30 Minutes

3. Stable Analysis Patterns 90 Minutes
  3.1 Patterns: Background and definition
  3.2 Analysis Patterns: Overview and History
  3.3 General Classifications of Analysis Patterns
  3.4 Our Classification – 3 Groups
  3.5 Major Challenges Facing Analysis Patterns
  3.6 Stable Analysis Patterns
  3.7 Stable Analysis Patterns vs. Meta Models & Data Models
  3.8 Samples of Analysis Patterns: Negotiation, Automation, Searching, etc.

4. Design Session-II 30 Minutes

Break 30 Minutes

5. Stable Design & Architectural Patterns 90 Minutes
  5.1 Problem Space vs. Solution Space
  5.2 Why Design & Architectural Patterns?
  5.3 The Importance of Design and Architectural Patterns
  5.4 Design Patterns Classifications
  5.5 Problems Facing Today’s Design Patterns.
  5.6 What are Stable Design and Architectural Patterns?
  5.7 Samples of Design Patterns: AnyAccount, AnyProject, AnyParty, etc.

Design Session-III 30 Minutes

DURATION: Full-Day

LEVEL: Intermediate
PREREQUISITES/REQUIRED PREVIOUS KNOWLEDGE

Familiarity with basic notions of software engineering, UML notation and models, and design patterns

CONTEXT: WHAT EXISTING WORK WILL BE USED AS A BASIS FOR THE TUTORIAL? WHERE DOES THIS TOPIC FIT INTO THE LARGER SCHEME OF THINGS?

YES!!!!

RELATED BOOKS:


RELATED JOURNAL ARTICLES:


**RELATED CONFERENCE ARTICLES:**


**RELATED CACM THINKING OBJECTIVELY COLUMNS:**


**RECENT AND RELATED TUTORIALS:**

1. *Object-Oriented Design Patterns and Frameworks*, First Hong Kong Conference on Quality Software Development, 2-day tutorial, Hong Kong, April 1996.

2. *Object-Oriented Experiences and Future Trends*, a half-day tutorial, Portugal's Object-Oriented Programming (OOP '96), Lisbon, Portugal, June 1996.

3. *Object-Oriented Experiences and Future Trends*, a half-day tutorial, OOPSLA '96, San Jose, CA, Oct. 96.

5. **Object-Oriented Experience & Future Trends**, ECOOP '97, a half-day tutorial, JYVASKYLA, Finland, July 1997.

6. **Object-Oriented Enterprise Frameworks**, Sixth European on Software Engineering Conference (ESEC '97), a half-day tutorial, Zurich, Switzerland, Sept. 1997.

7. **Object-Oriented Enterprise Frameworks**, a half-day tutorial, Nokia, Helsinki, Finland, 1999.


11. M.E. Fayad. **Stable Design Patterns: True Solution Space**. A half-day tutorial, The 9th International Conference on Object-Oriented Information Systems (OOIS’ 03), Geneva, Switzerland, Sept. 2003 Accepted.


RELATED SEMINARS AND SHORT COURSES:


2. **Object-Oriented Design: UML in action**, Tandil, Argentina, 40-Hour Seminar, June 01.

3. **Stability-Oriented Software Engineering**, Tandil, Argentina, 40-Hour Seminar, June 01.


COURSE DEVELOPMENT:

- "Object-Oriented Modeling & Applications Using UML" University of Nevada, Fall 1996.
- “Object-Oriented Analysis and Design” San Jose State University, Fall 2002.
- “Software Patterns” San Jose State University, Spring 2003.
- “Software Reuse” San Jose State University, Fall 2004.
• “Software Architectures” San Jose State University, Spring 2005

THEME ISSUES (GUEST EDITOR)


A DESCRIPTION OF TEACHING METHODS: WILL THE COURSE CONSIST OF PRESENTATIONS, DEMONSTRATIONS, GROUP WORK, DISCUSSION, ETC. OR A COMBINATION?

A combination. In addition, course participants will form teams and be involved in hands-on design sessions lasting 30 minutes to one hour each. A half-day tutorial will have one design session and a full-day tutorial will have two design sessions and a three-day seminar will have 6 design sessions.

WHO SHOULD ATTEND

This tutorial is targeted both at computer and software researchers, pattern researchers and users, framework developers, and computer professionals. A short overview on current trends on software stability and creation of architecture on demand for any domain will be given as an introduction. The presentation will then focus on defining the various constituents of these two amazing subjects. These definitions will first take the form of enduring concepts, and their properties in a targeted domain, and then be transformed as Stable Software Patterns. Although the ability to understand basic object-oriented concepts, software engineering principles, software modeling techniques, (OMT, UML, or any object-oriented method) and software architecture would be of great help when observing the realization of Stable Software Patterns, no particular knowledge will be needed in order to follow the conceptual and/or theoretical developments.

REQUESTED AUDIO/VIDEO EQUIPMENT.
An overhead projector to connect to my *notebook* and a large easel with notepad and three different colored markers

A BIOSGAPHY OF EACH SPEAKER, DESCRIBING CLEARLY WHY THE SPEAKER IS QUALIFIED TO PRESENT THE TUTORIAL. INCLUDE REFERENCES TO EXPERIENCE PRESENTING OTHER TUTORIALS (INCLUDE URLS).

PRESIDENT'S BIOGRAPHY

**MOHAMED FAYAD** is a Full Professor of Computer Engineering at San Jose State University from 2002 to present. He was a J.D. Edwards Professor, Computer Science & Engineering, at the University of Nebraska, Lincoln, from 1999 to 2002, and an associate professor at the computer science and computer engineering faculty at the University of Nevada, from 1995 - 1999. He has 15+ years of industrial experience.

Dr. Fayad is a Senior Member of the IEEE, a Senior Member of the IEEE Computer Society, a Member of the ACM, an IEEE Distinguished Speaker, an Associate Editor, Editorial Advisor, and a Columnist for The Communications of the ACM and his column is Thinking Objectively, and a columnist for Al-Ahram Egyptians Newspaper (2 million subscribers), an Editor-In-Chief for IEEE Computer Society Press - Computer Science and Engineering Practice Press (1995-1997), IASTED Technical Committee member on Software Engineering (2001-2004), a general chair of IEEE/Arab Computer Society International Conference on Computer Systems and Applications (AICCSA 2001), Beirut, Lebanon, June 26-29, 2001, and he is the president of Arab Computer Society (ACS) from April 04 to present.


TUTORIAL RESUME
Has the tutorial been given before?  YES

© M.E. Fayad, PhD, 2000-2004
When and Where was it given?

- Several local, national and international companies and universities. The most recent two are:


(Please check tutorial, seminars, and courses).