

Practical Software Engineering for Architects and Developers

An Introduction

By Enricos Manassis

Enricos.Manassis@ProClaris.com



01/09/2006

Agenda

– Part I

- Context: measures of success in Software Development - The CHAOS report
- Software Engineering:
 - What is it about?
 - Are we mature: what do we need and what do we have?
 - How to be practical
- Issues and Impediments

– Part II

- A Comprehensive Example



Part I

Concepts

Measures of Success in Software Development

- Successful Software Development:
 - Delivery on time
 - Within budget
 - To the purpose, with measured quality
 - Functional: appropriate automation of business function to respect user requirements
 - Non-Functional: measurable system performance to the expectations of the organization (availability, robustness, performance, scalability, maintainability/evolution, manageability, security)

Reality check: the CHAOS report

- Three quarters of software development projects are still:
 - Delivered late
 - Over budget
 - Not adequately addressing user needs
 - Scrapped altogether
- See Standish Group: CHAOS report
 - http://www.standishgroup.com/chaos_resources/index.php
- -> Need for software engineering

What is Software Engineering?

IEEE definition:

Software engineering is the application of a systematic, disciplined, quantifiable approach to development, operation, and maintenance of software; that is, the application of engineering to software.

What is Software Engineering?

A conceptual definition:

- Refinement of the knowledge through successive abstraction layers
- Traceability of each and every item of information between abstraction layers

Practical implication:

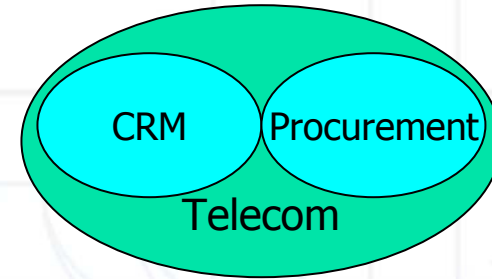
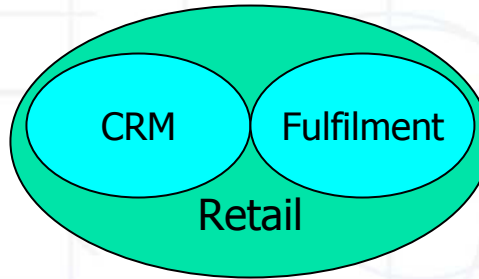
- Comprehensive set of:
 - Process
 - Methods and Techniques
 - Tools

Abstraction Layers

Domain Space

Business Domain

- Industries
- Functions



Solution Space

Business Problem

System Vision and Features

System Specification

Requirements:
Functional and Non
Functional

Use Cases

System Design/Integration

Analysis Model

Test Cases

Security Model

System Development and Configuration

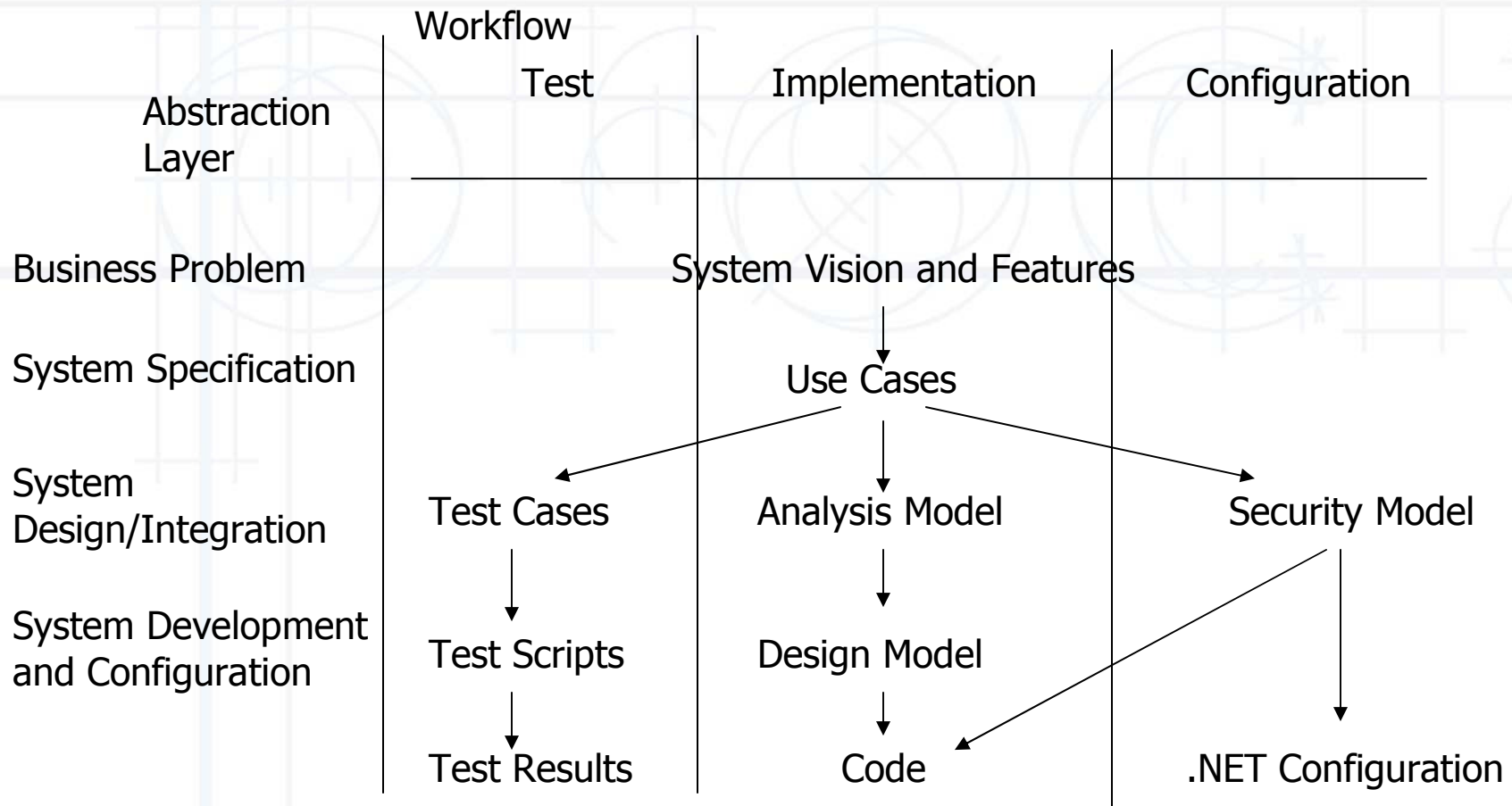
Design Model

Code

Test Scripts

.NET Configuration

Traceability



Traceability: → : Is a Technique, Method or Tool

How to implement Traceability

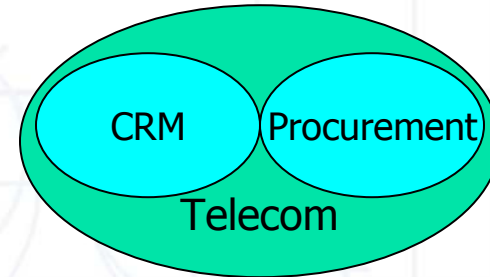
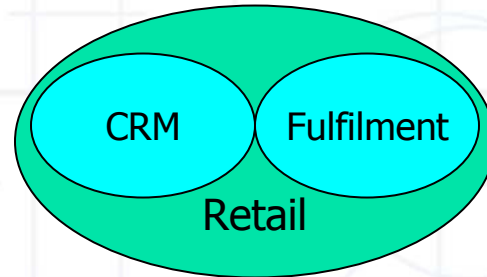
- Abidance by a clearly defined **Process**
- Description of a **Technique or Method**
 - Test Case traceability matrix
 - Role Based security traceability matrix
- Usage of a **Tool**
 - Rational XDE for traceability from model to code

The Whole Picture

Domain Space

Business Domain

- Industries
- Functions



Solution Space

Business Problem

System Vision and Features

System Specification

Requirements:
Functional and Non
Functional

Use Cases

System Design/Integration

Analysis Model

Test Cases

Security Model

System Development and Configuration

Design Model → Code

Test Scripts

.NET Configuration

What Maturity do we need?

- Technology
- Tools
- Methods and Techniques
- Process
- People

Technology

- Languages: C#, Java
- Application Servers: system services
 - Transactional Monitoring
 - Security Control, Role Based Security
 - Connection Pooling
 - Thread and Object Pooling
 - Queued Components
 - Loosely Coupled Events
- J2EE vs. .NET

Tools

- Basic elements of an integrated development toolkit:
 - IDE (Eclipse, Visual Studio.NET)
 - Modelling (Rational XDE)
 - Testing (Application Center Test)
- Extended elements:
 - Application composition from functional blocks
 - Application composition from Use Cases

Methods and Techniques

- Methods
 - Use Case analysis
 - OOAD
- Techniques
 - Test Case traceability matrix
 - Role Based security matrix

Process

- The Unified Software Development Process
- Rational Unified Process (RUP)
- RUP .NET/J2EE Developer's Configuration
- Iconix
- Catalysis
- eXtreme Programming
- ProClaris Practical Process™
(www.ProClaris.com)

People

- Understanding and using Technologies (+/-)
- Exposure and experience with Tools (+/-)
- Knowledge and experience of Methods and Techniques (+/-)
- Comprehension of Process (+/--)

How to be Practical

- Do not try to apply all the theory, all of the process. Instead select what you need at a specific time. Use more if needed.
- Think in terms of techniques to apply, nurture a collection of the techniques you find the most effective for you.
- Free your mind, follow your own path

Future Vision

Functional component suites

- Integrate Use Cases and case studies
- Integrate UML diagrams, test cases and usage pattern
- Components
 - Application framework specific components (.NET, J2EE)
 - WEB Service interface specifications (with their Service Level Agreements)

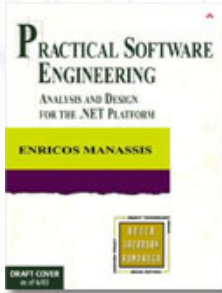
Issues and Impediments

Functional Packages Availability

- Application Server Integration
- UML Object Models
- Use Case Integration

References

- Practical Software Engineering: Analysis and Design for the .NET Platform
Enricos Manassis (ISBN: 0321136195)



- The Rational Unified Process: An Introduction
Phillipe Kruchten (ISBN: 0201707101)
- Objects, Components, and Frameworks With Uml : The Catalysis Approach
Desmond Francis D'Souza, Alan Cameron Wills (ISBN: 0201310120)
- Applying Use Case Driven Object Modeling with UML
Doug Rosenberg, Kendall Scott (ISBN: 0201730391)

Questions

Enricos.Manassis@ProClaris.com

Part II

A Practical Example: Roles Based Security Design



Agenda

- What is Role Based Security?
 - Concepts
- Approach
 - Overview
 - Identify Roles
 - Analysis Security Model
 - Design Security Model
- Implementation
- Other Security Considerations
 - Programmatic Security
 - Infrastructure Security

Application Servers

- J2EE / .NET
- Services
 - Transaction Monitoring
 - Connection Pooling
 - Thread and Object Pooling
 - Queued Components
 - Loosely Coupled Events
 - Access Control and Authorization

Concepts of Role Based Security

- Application designers plan the system without any security feature in the design
- Application developers identify roles that can access each object/component. Role is a category of users.
- Granularity to either:
 - Component
 - Member functions

Concepts of Role Based Security

- Each role is associated to one or more registered users of the systems.
Conversely one user can be in multiple roles.
- Applied through configuration of app server and not within application implementation

Role Based Security Challenge

- The challenge for the application developers
 - Decide which roles are appropriate for the application
 - Decide which objects and member methods can be accessed by each role
- The proposed approach
 - Takes out the guesswork of the process
 - Introduces systematisation
 - Achieves consistent quality

Role Based Security Challenge

- Limitations
 - Use case analysis for system specifications
 - All objects of the system should be modelled with their collaborations (e.g. sequence diagrams)

Approach Overview

- From UML model review the actors model
- Actors are mapped to roles in app server
- Hierarchy of actors may or may not map to hierarchy of roles in app server. Non abstract actors will map to app server roles.
- Create object catalogue
- Review the class and component diagrams and identify all public interfaces.

Approach Overview

- Map actors to objects and member methods
- Review sequence diagrams and identify which user has access to each public interface

Security Model Template

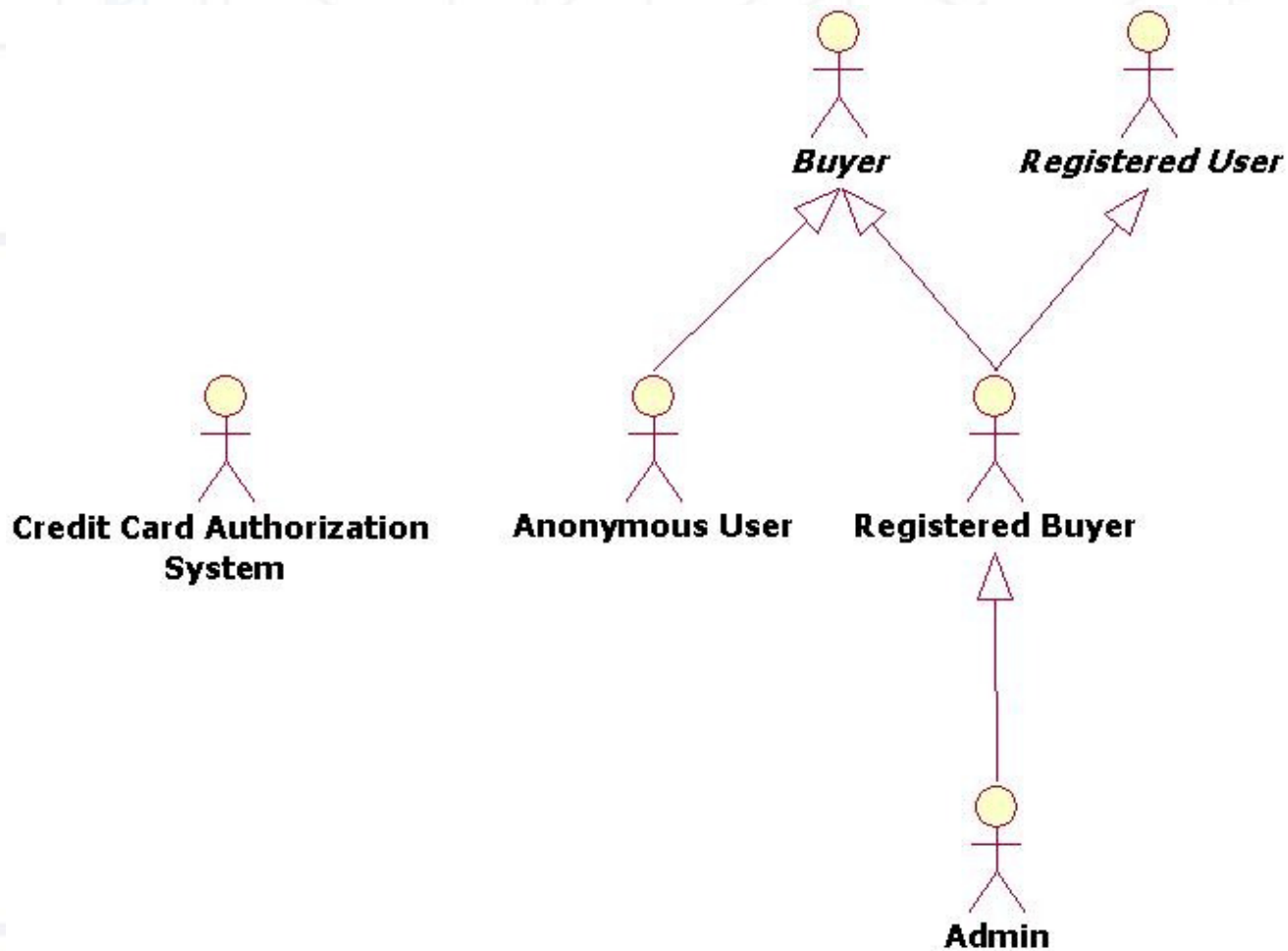
- Analysis

Analysis Classes	Actor A	Actor B	Actor C
Class 1	X	X	
Class 2		X	X

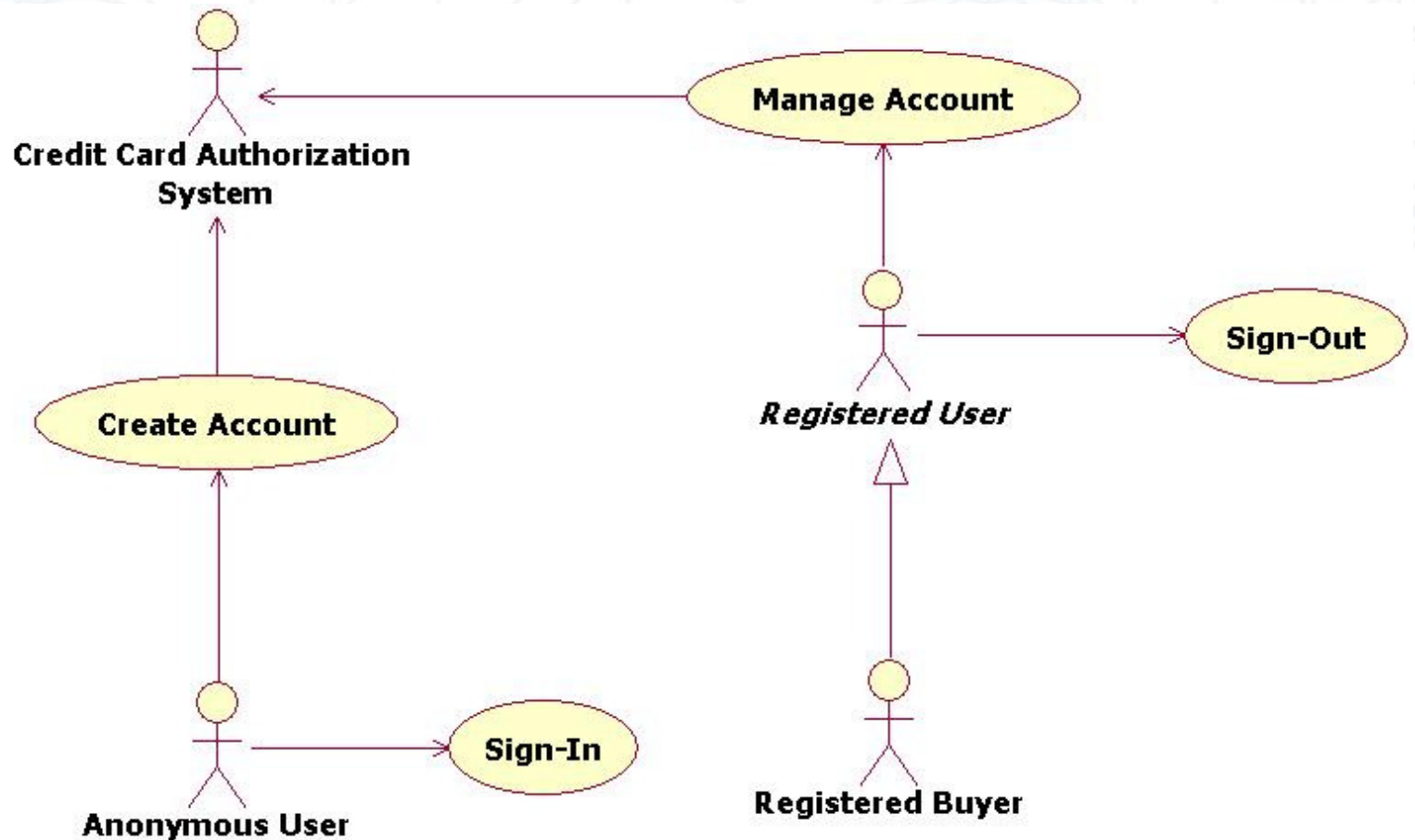
- Design

Implementation Classes	Operation	Actor A	Actor B	Actor C
Class 1	Operation 1	X		
Class 1	Operation 2	X	X	
Class 1	Operation 3	X		X
Class 2	Operation 1	X		
Class 2	Operation 2		X	

Identify Roles : Use Case Actors



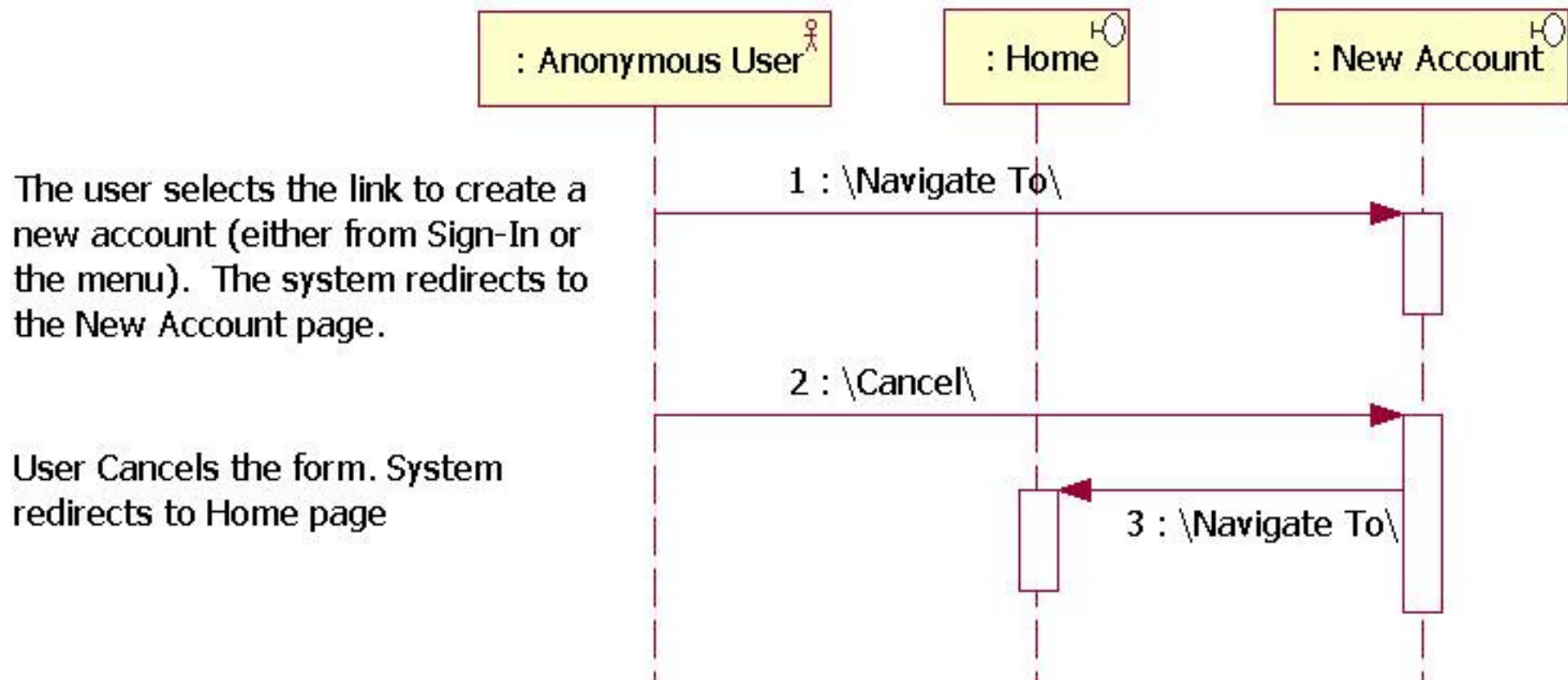
Identify Roles : Anonymous Users



The Security Model

- Analysis level
 - Shows the interactions of the actors with boundary type of classes
- Design level
 - Shows the interactions of the actors with classes of:
 - The presentation layer in the case of an interactive user
 - The business layer in the case of an external system
 - Could be used as input for a tool module that generates app server configuration (Visual Studio.NET extension).

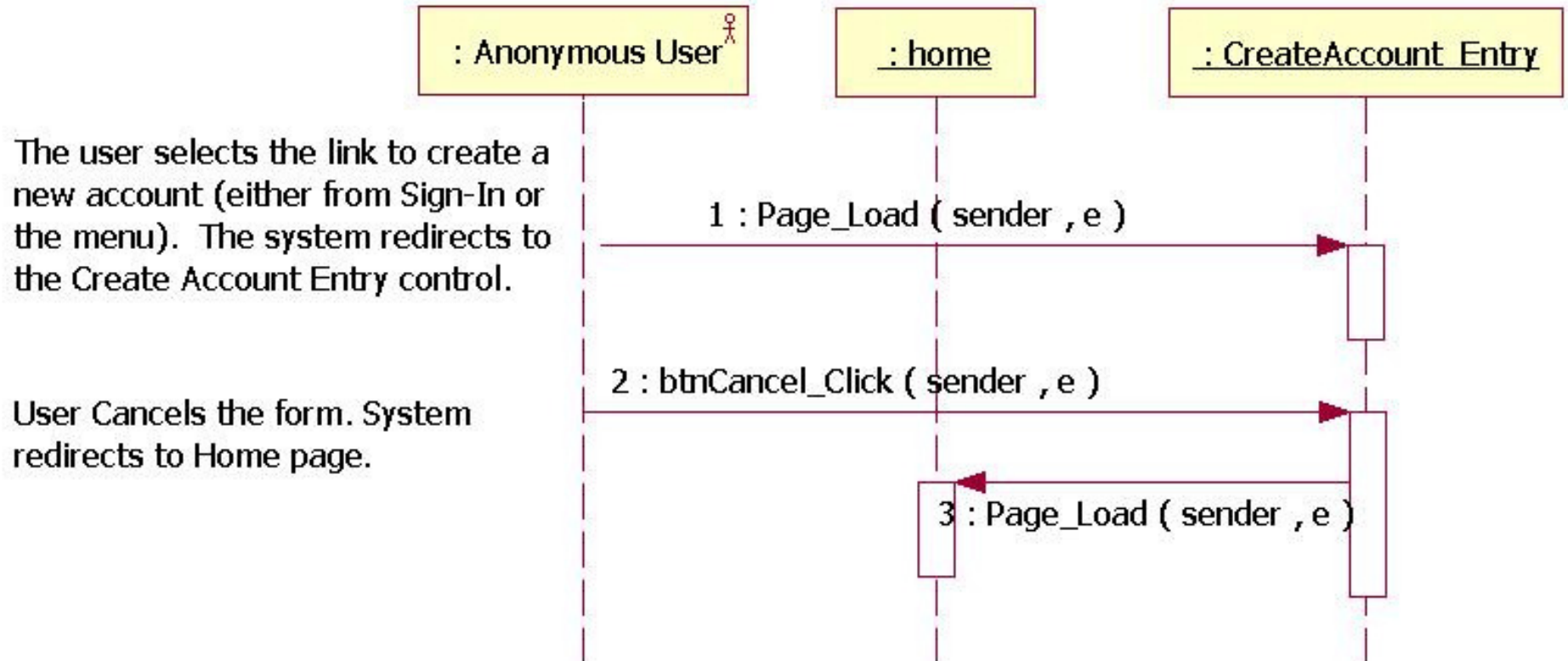
Analysis Security Model-Sequence Diagram



Analysis Security Model Matrix

Analysis Classes	Anonymous User	Registered Buyer	Admin
New Account	X		
New Account Verify	X		
Account Home		X	X
Home	X		
Create Account Dispatcher	X		
Credit Card System	X		
Account Manager	X		
User Account	X		
Credit Card	X		
User Group	X		

Design Security Model- Sequence Diagram



The user selects the link to create a new account (either from Sign-In or the menu). The system redirects to the Create Account Entry control.

User Cancels the form. System redirects to Home page.

Design Security Model Matrix

Class	Operation	User Role		
		Anonymous User	Registered Buyer	Admin
CreateAccount_Entry	Class Definition	X		
CreateAccount_Verify	Class Definition	X		
Home_Account	Class Definition		X	X
Home	Class Definition	X		
CreateAccountDispatcher	ProcessEvents	X		
AccountManager	ActivateUserAccount			
	CreateAccount	X		
	DeactivateUserAccount			
	GetCreditCardInfo			
	getCreditCardInfoByCardNumber			
	GetUserAccountByUniqueId			
	GetUserAccountByUserId	X		
	GetUserGroupByUniqueId			
	SetUserUIDToGroupId	X		
	UpdateAccount			
	ValidateAccount			
CreditCardManager	ChargeTransactionFee			
	IsCreditCardValid	X		

.NET Configuration

SignIn.ascx.cs

```
/// <summary>
///   SignIn: Page for entering the user name and password for signing
in.
/// </summary>
[PrincipalPermissionAttribute(SecurityAction.Demand, Role =
"Anonymous")]
public abstract class SignIn : System.Web.UI.UserControl
```

SignInDispatcher.cs

```
/// <summary>
/// processEvents: Determines the next step to take in the sign in use
case.
/// </summary>
/// <param name="ctrl">Current sign in usecase ASCX control</param>
[PrincipalPermissionAttribute(SecurityAction.Demand, Role =
"Anonymous")]
public void processEvents (UserControl ctrl)
```

AccountManager.cs

```
/// <summary>
/// getUserAccountByUserId: Locates the user account with the specified
userId.
/// </summary>
/// <param name="userId">userId of the user to retrieve user account
information for.</param>
/// <returns>Typed UserInfo dataset that contains the UserAccount
record with a matching userId.</returns>
[PrincipalPermissionAttribute(SecurityAction.Demand, Role =
"Anonymous")]
public UserInfo getUserAccountByUserId (string userId)
```

Summary

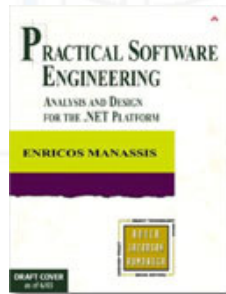
- Role based security is a powerful control mechanism
 - Application level
 - Easy to implement
 - Independent of operating system security configuration
 - Flexible
- Advantages of applying a method
 - Systematisation
 - Brings rational behind the app server configuration
 - Relates the functional (use cases) with the technical (configuration of app servers)
 - Supports the generation of app server configuration

More Security Considerations

- Programmatic Security
 - Application architecture: designed to support role based access control to use cases
 - .NET role based security reinforces and validates application security
- Infrastructure Security
 - Firewalls
 - Demilitarised network zones
 - Web server configuration
 - File access configuration
 - Database security

References

- From Use Cases to Role-Based Security Components
 - Enricos Manassis
(http://www.therationaledge.com/content/feb_01/t_usecase_em.html)
- Practical Software Engineering: Analysis and Design for the .NET Platform
 - Enricos Manassis (ISBN: 0321136195)



Questions

Enricos.Manassis@ProClaris.com