CRM .NET

Customer Relationship Management System
A Project for Components Programming and Middleware, SS 2004
07/01/2004

Christian Kochs
Jan Schaffner
Jan Schulz-Hofen
Gaurav Singh

Customer Management: Challenges

- Manage distributed Customer Data
- Serve multiple Users
- Ensure Consistency
- Store different Aspects for each Customer Profile
Structure of Presentation

- Requirements and Scenario of Application
- Architecture of System
- Technologies used
- Live Demonstration with Technology Workshop
- Development Process and Experiences
- Outlook: Conceivable Extensions

Use Case Diagram (UML 2.0)
Architecture of System

- Multi-User-System
- Follows Model-View-Controller (MVC) Pattern
- Own Implementation of Naming Service (Registry)
- Distributed Architecture (.NET Remoting)

- Flexible Creation of Components (offered Functionalities)
  - Dynamic management of heterogeneous and distributed components
  - Component Dynamics on both Client and Server
  - GUI is deployed together with specific component, not in the client
  - Each Component is responsible for its individual Data Storage

Compositional Structure (FMC)
Featured Technologies

- Distribution and Communication via .NET Remoting:
  - Server Activated Object (SAO)
  - Singleton Pattern
  - Binary Formatter (Performance)
- Dynamic Assembly Loading (of Components’ UIs)
- Data Storage: MS SQL Server, ADO .NET, J2EE-Entity-Beans
- Components implemented in C# and J#
- COM-Interop: Microsoft Office Integration
- Web services via SOAP / XML
Customer Profile Data Structure: E/R-Diagram (FMC)

- Modular Architecture of CRM .NET requires modular Design of Data Structure

- CRM Base manages Customer Base Data (Profile Set) and object specific Security Privileges
- Each Profile consists of one or more Customer Aspects
- Aspects are managed individually by CRM .NET Components
- ContentType is Linking Element between Model, View and Controller
Dynamic Assembly-Loading in .NET

```csharp
if (!ControlExists) 
{ 
    Assembly MyAssembly = Assembly.Load(new byte[] { GuiExtensions.sosFileList.GetByIndex(1) }); 
    GuiExtensionInterface MyGuiExtension = (GuiExtensionInterface) Activator.CreateInstance(MyAssembly.GetType("Emp.GuiExtension"));
    this.tabControl.Controls.Add((TabPage)MyGuiExtension);
}
```

### Framework Comparison

#### Extending the GUI dynamically in Java

- Defining a Classloader extending the abstract ClassLoader Class

```java
// Define a GuiExtension ClassLoader 
public class GuiExtensionClassLoader extends ClassLoader 
{ 

    public Class loadClass(String name, boolean resolve) 
    { 
        // ...
        
        // tell ClassLoader to create a "class object" from the byte array 
        Class MyClass = defineClass(name, 
                byte[] GuiExtensionByteCodes, 0, 
                GuiExtensionByteCodes.length, 
                null);
        return MyClass;
        // ...
    }
    //....
    
}
```
Framework Comparison (contd.)

- Extending the GUI dynamically in Java
  - Using the defined GuiExtensionLoader to load a Class

```java
// instantiate our GuiExtension ClassLoader
GuiExtensionLoader myClassLoader = new GuiExtensionLoader(GuiExtensionByteCode);

// tell ClassLoader to load and resolve Class
Class MyGuiExtensionClass = myClassLoader.loadClass("GuiExtension", true);

// create an instance of the loaded class
Object MyGuiExtensionObject = MyGuiExtensionClass.newInstance();
```

Multi-User Facility

- Global Data Synchronization
- Let’s open up a second View!
COM Interop für Einbindung von MS Outlook (FMC)

Create an Instance of the referenced COM-Component
- Use of the Type Library Importer (tlbimp.exe) to create a .NET Interop Assembly containing the Metadata of the COM Type Library
  - tlbimp <COM-Component> <.NET Interop Assembly> / <Options>
- At Runtime, the .NET CLR will create an RCW for each COM Object
- RCW serves as a Bridge to the Unmanaged Code
  - Translating calls – consumes COM Interfaces
  - Marshalling data
  - Managing lifetime of the wrapped object

In Source Code:
```csharp
// Transparetnly create a RCW for the COM-Object
// at Runtime from the Interop Assembly
Outlook.Application objOutlook = new Outlook.ApplicationClass();
objOutlook.NameSpace objNameSpace = objOutlook.GetNameSpace("MAPI");

// Prepare access to Outlook-Contacts
Outlook.MAPIFolders skjFolder = objNameSpace.GetDefaultFolder(
    Outlook.OlDefaultFolders.olFolderContacts);
```
Security Concept

- User- and Group-based Authentication
- Item-based Authorisation

Authentication and Authorization (FMC Petri-Net)

- Dynamics of Controller’s Session Management

Diagram:
- CRM.NET Client Application (View)
- CRM.NET Base (Controller)
- ControllerInterface
- ReadProfile(in ID : int, in Ticket : AuthTicket) : Profile
- ModifyProfile(in ID : int, in ProfileToModify : Profile, in Ticket : AuthTicket) : void
- AddProfile(in ProfileToAdd : Profile, in TitleToAdd : string, in Ticket : AuthTicket) : void
- RemoveProfile(in ID : int, in Ticket : AuthTicket)
- Authenticate(in username : string, in password : string) : AuthTicket
- GetFilteredCustomerList(in Ticket : AuthTicket)
- GetProfileCustomerList(in Ticket : AuthTicket)
- GetProfile(in ID : int, in Ticket : AuthTicket) : Profile
- Generate and provide unique session ticket
- Store session ticket in RAM
- Provide authentication info and request connection
- Read reference auth info from storage
- Store ticket and corresponding user identity in list in RAM
- CRM .NET Client Application View
- CRM .NET Base Controller


10
Webservices: Talking to the J2EE-eRasm-Gallery

- Client Proxy is required to access Webservice
- Proxy Class is generated from WSDL-File by a tool
- Proxy is instantiated in Client

Development Process and Experiences

- Process Model applied: Extreme Programming (XP)
- Testing with NUnit
- Experiences with Tools
  - Microsoft Visual Studio 2003 .NET
  - Subversion and Tortoise Client
  - Microsoft (R) Web Services Description Language Utility
  - Type Library Importer (tlbimp.exe) for COM-Interop
Outlook: Conceivable Extensions

- **Functionality**
  - Stronger diversification within component hierarchy
  - Additional administration frontend

- **Security**
  - Extend our item-based authorization: Access Control Lists
  - Possible to implement encryption-algorithm in additional remoting-layer

- **Performance**
  - Improve caching algorithms
  - Increase efficiency of in-memory data structures

Sources

- MSDN Library: msdn.microsoft.com
- The Code Project: www.codeproject.com
- Thai: .NET Framework Essentials, O’Reilly, 2001
- The C# Corner: www.c-sharpcorner.com
- Hamilton, MacDonald: ADO.NET in a Nutshell, O’Reilly, 2003