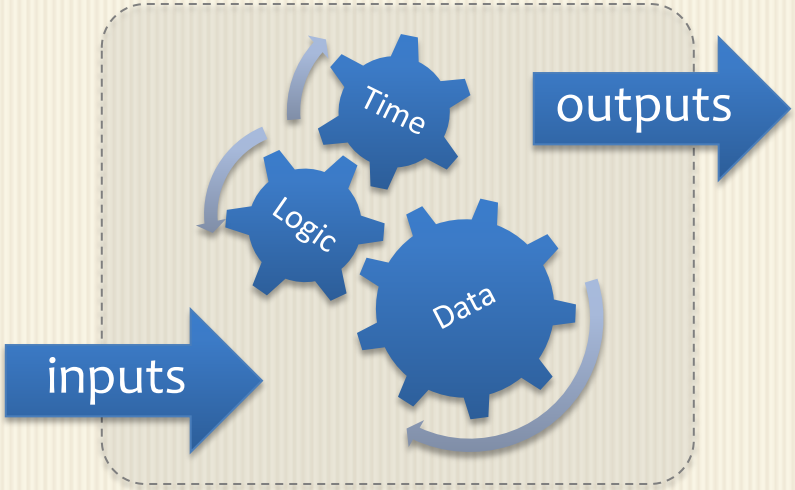
Application Architecture Guide

Don Smith
Community Liaison
Microsoft patterns & practices

Microsoft patterns & practices



The Application

users

expectations

boundaries

goals

ARCHITE - STRUCTURE

supports solution objectives

... in the simplest way possible

The

diplomat

team-member

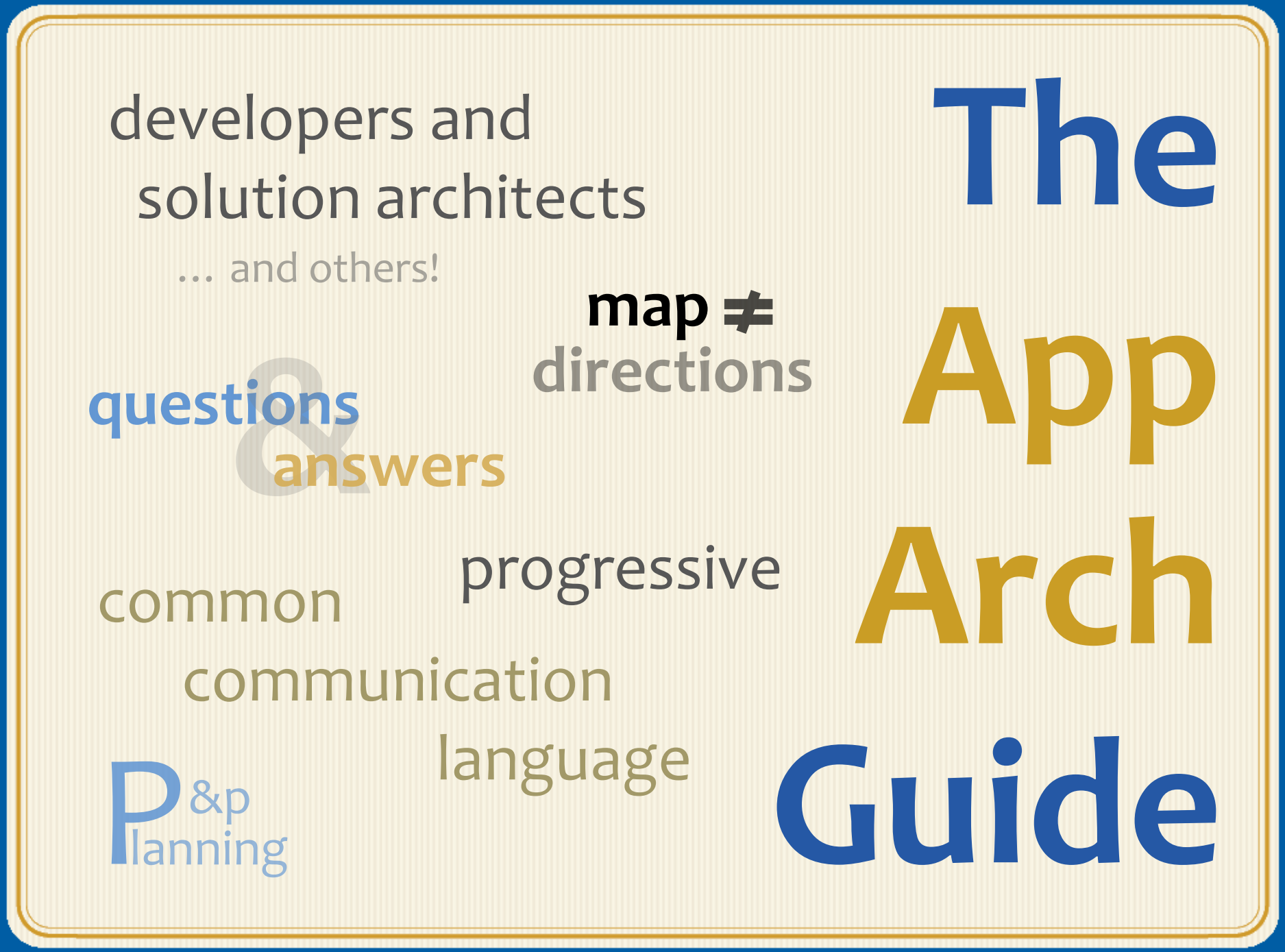
Architect

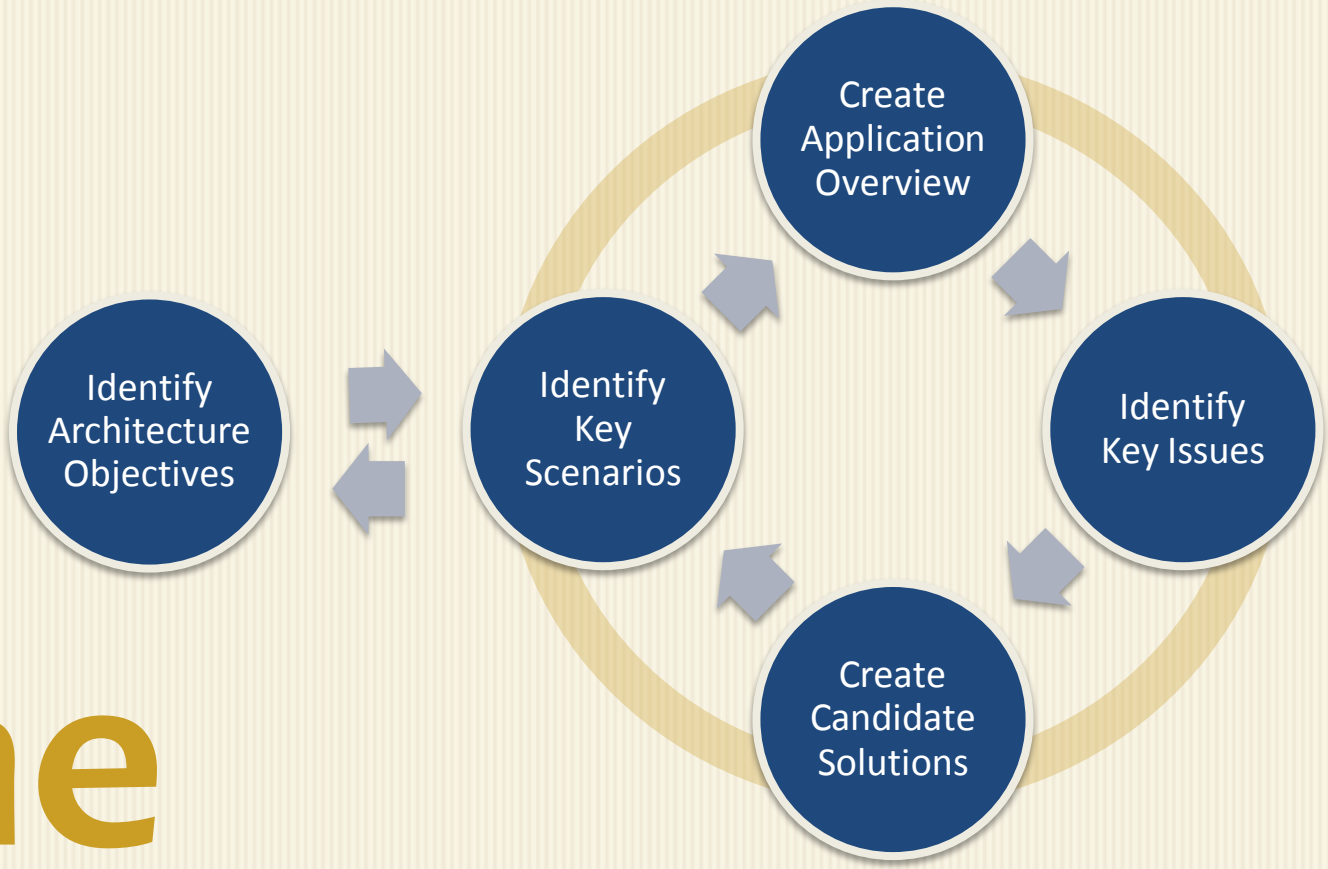
influencer

decision maker

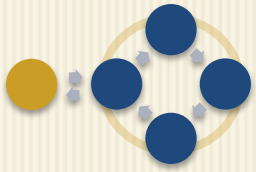
consultant

LEADER


developers and
solution architects
... and others!
map ≠
directions
questions
&
answers
progressive
common
communication
language
P&p
lanning
The
App
Arch
Guide



One Technique



scope

time

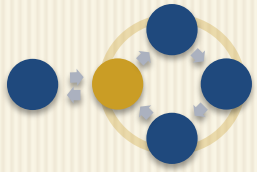
audience

goals

principles

how to communicate it

Architectural Objectives



architecturally
significant
use cases

risk

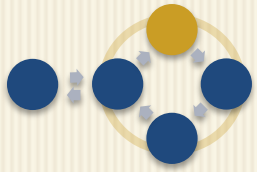
revenue

high impact

KEY

process

scenarios



Application Overview

choices choices choices choices choices choices choices choices

Application Type

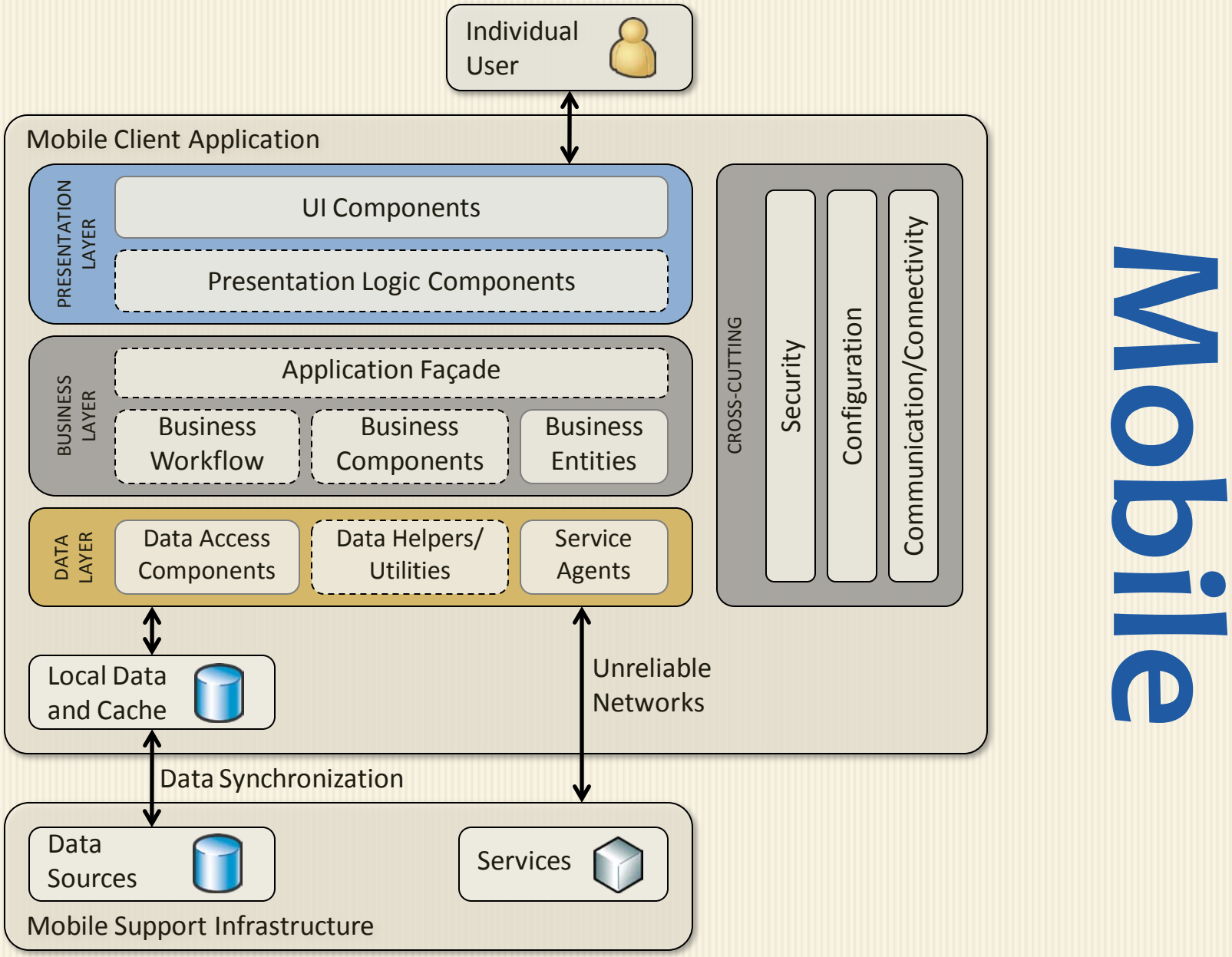
rich client

rich internet

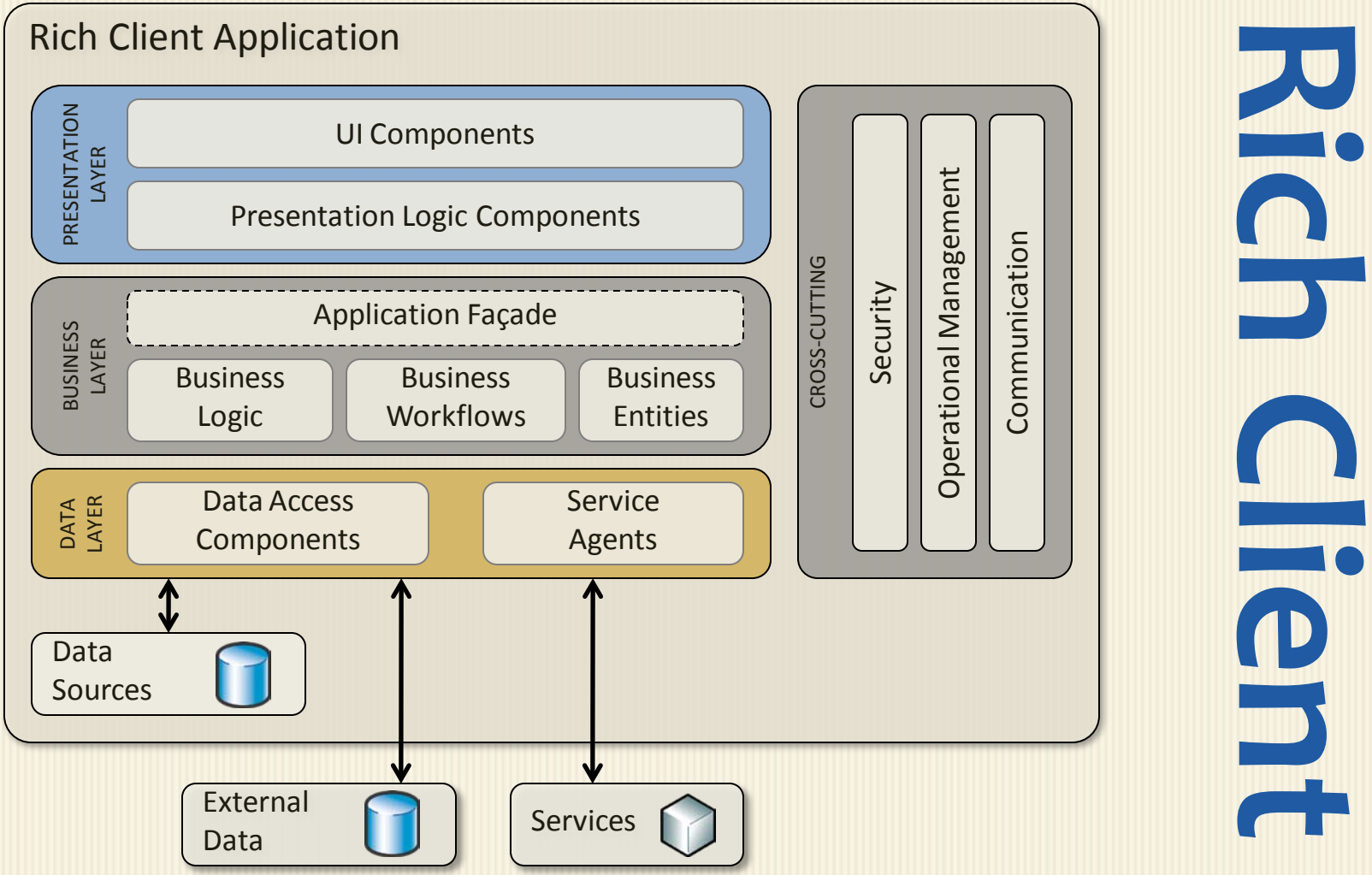
mobile

service

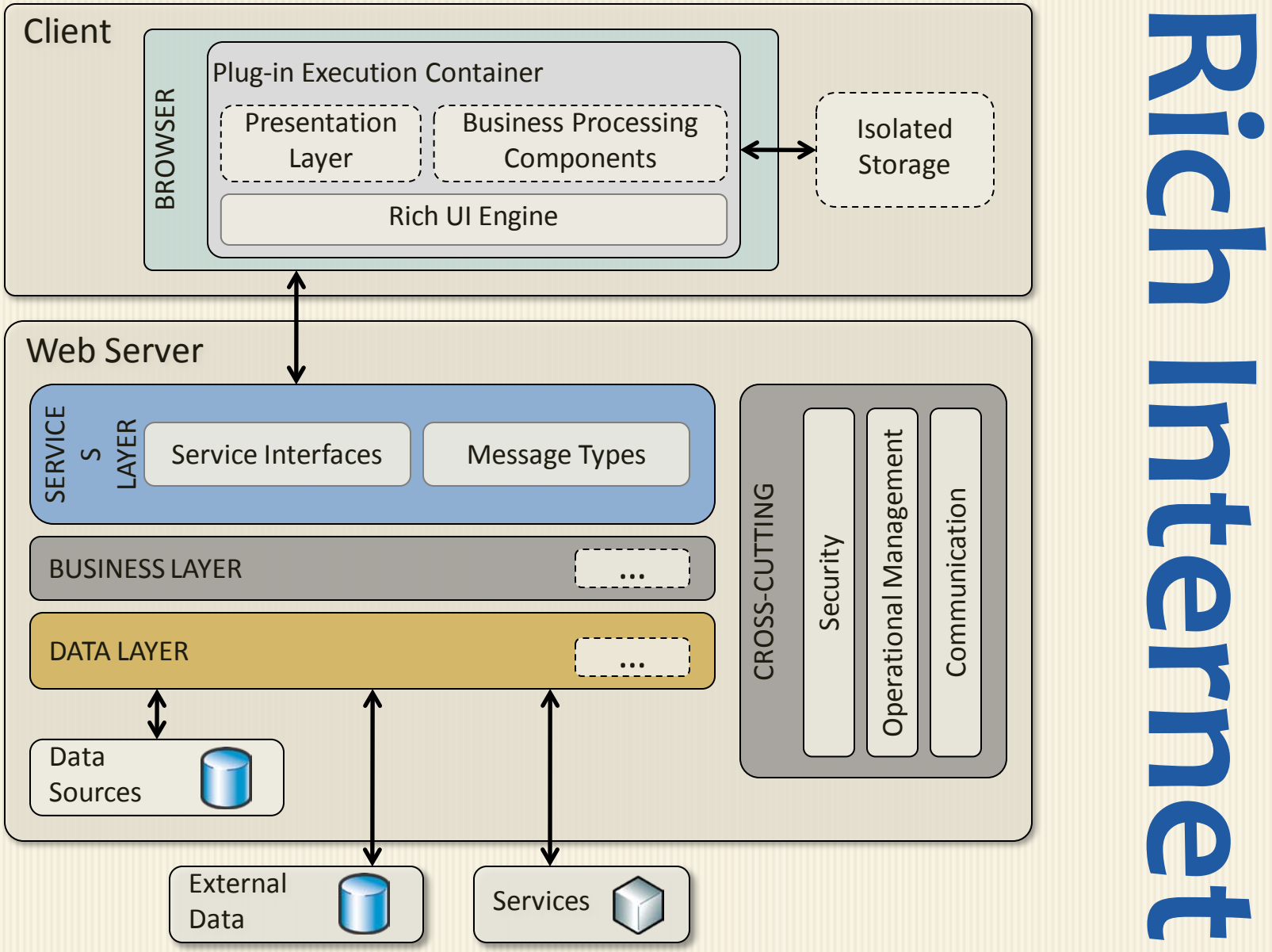
web

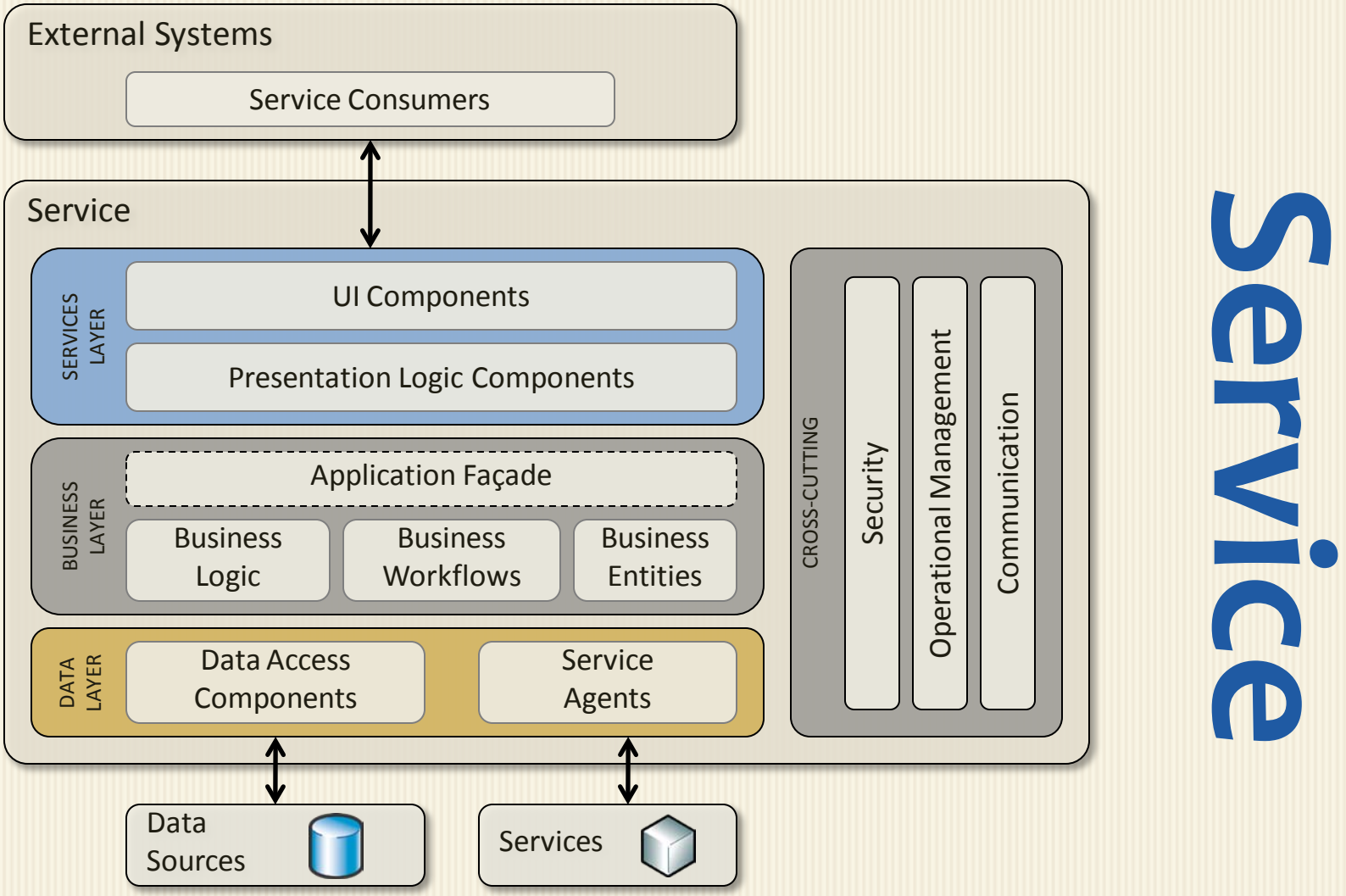

Individual User
Mobile Client Application
PRESENTATION LAYER
UI Components
Presentation Logic Components
BUSINESS LAYER
Application Façade
Business Workflow
Business Components
Business Entities
DATA LAYER
Data Access Components
Data Helpers/ Utilities
Service Agents
CROSS-CUTTING
Security
Configuration
Communication/Connectivity
Local Data and Cache
Unreliable Networks
Data Synchronization
Data Sources
Services
Mobile Support Infrastructure
Mobile

Rich Client

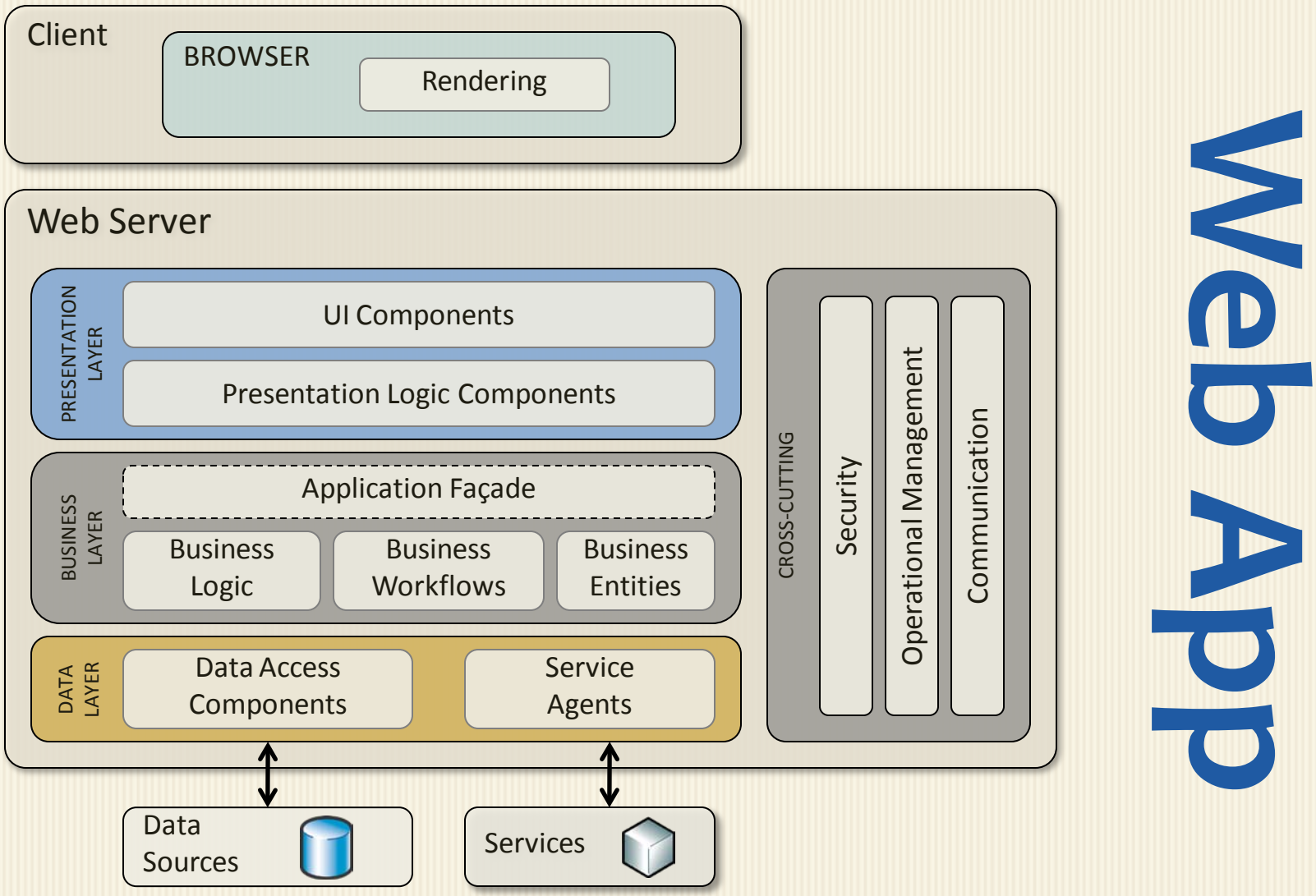

Rich Client Application
PRESENTATION LAYER
UI Components
Presentation Logic Components
BUSINESS LAYER
Application Façade
Business Logic
Business Workflows
Business Entities
DATA LAYER
Data Access Components
Service Agents
CROSS-CUTTING
Security
Operational Management
Communication
Data Sources
External Data
Services

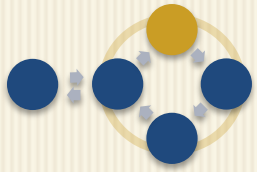
Rich Internet




External Systems
Service Consumers
Service
SERVICES LAYER
UI Components
Presentation Logic Components
BUSINESS LAYER
Application Façade
Business Logic
Business Workflows
Business Entities
DATA LAYER
Data Access Components
Service Agents
CROSS-CUTTING
Security
Operational Management
Communication
Data Sources
Services

Service


Client
BROWSER
Rendering
Web Server
PRESENTATION LAYER
UI Components
Presentation Logic Components
BUSINESS LAYER
Application Façade
Business Logic
Business Workflows
Business Entities
DATA LAYER
Data Access Components
Service Agents
CROSS-CUTTING
Security
Operational Management
Communication
Data Sources
Services
Web App



Application Overview

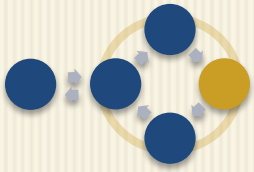
more choices ...

technologies

deployment constraints

Architecture Style

whiteboard it



Key Issues

Quality Attributes (the "ilities")

interoperability
maintainability
manageability
reliability
scalability
security
testability
usability

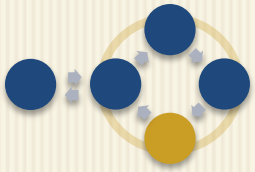
Crosscutting Concerns

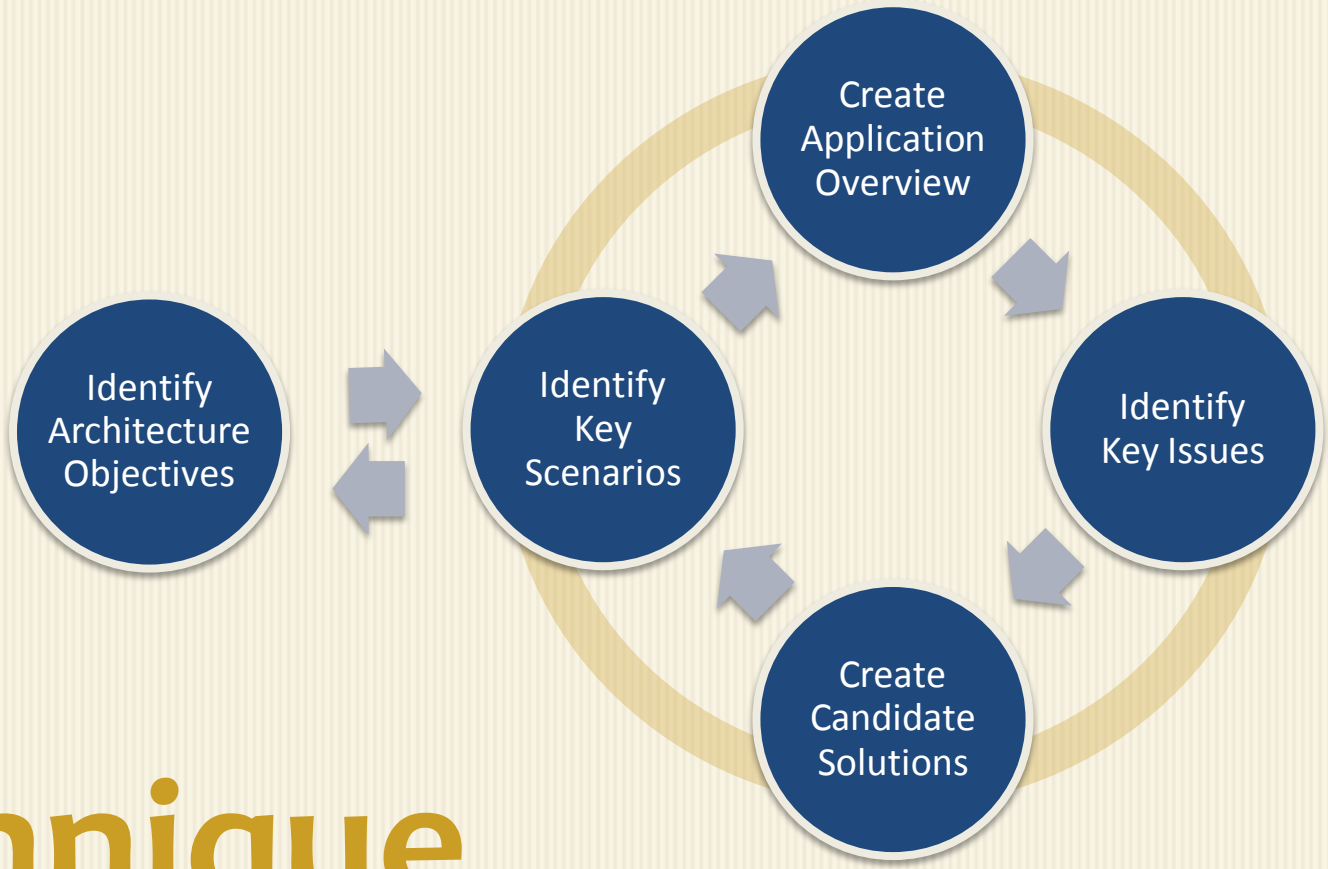
security (authn/authz)
caching
configuration
exception management
logging
validation

Candidate Solution

... to learn

It's time to **build** something





Technique Overview

Just one more Thing

REST & SOAP

Follow-up Links

- **Application Architecture Guide**
  - http://msdn.microsoft.com/en-us/library/dd673617.aspx

- p&p's site: http://msdn.com/practices
- don's email: don.smith@microsoft.com
- don's blog: http://blogs.msdn.com/donsmith

Microsoft®

Your potential. Our passion.™

The Appendix

Server Iconography

ISA Server

SQL

Commerce Server

Content Management Server (CMS)

Mobile Information Server (MIS)

Exchange Server

BizTalk

RTC Server

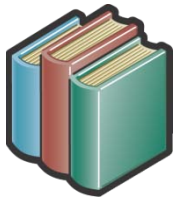
Host Integration Server (HIS)

Application Server

Server Running XML Web service

Information Iconography

Code


libraries


GXA


BizTalk Analysis


BizTalk Developer


Template


Document


Key


BizTalk Analysis


BizTalk Developer

Information Iconography

Generic Application

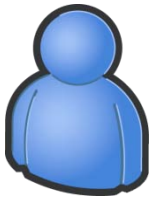
Database blue

Database purple

Database green

Databases

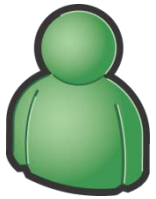
Tools



User 1

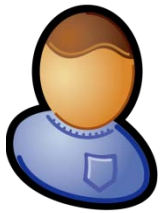
User 2

User 3



User 4

User casual



User casual man

User business woman

User business casual

User business man

Information Iconography

Folder

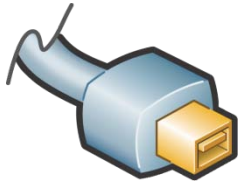
Folders

Firewall

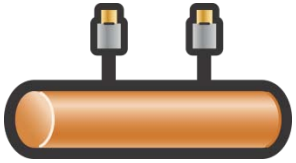
Commerce

Email/Package

CD



Connector



Message Bus network connection

Hardware Iconography

Server

PC

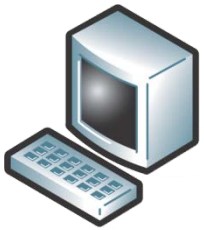
PC (with XML Web service)

Notebook

Pocket PC

Mobile Phone

Pager



Dumb Terminal

Dumb Phone

Tablet PC

PC blank screen

Monitor

iMac – Apple newer

iMac - Apple

LCD flat panel monitor

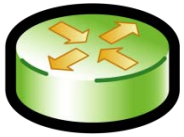
XML Web Services Iconography

XML Web service

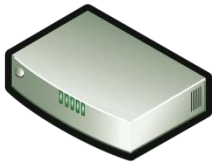
XML Web service and
Binary Code

XML Web service
(black and white)

Icons



Router-Logical



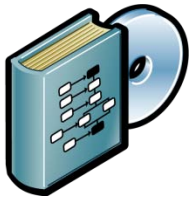
Router-Physical

Storage Array

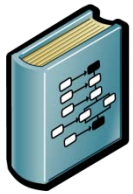
Switch

Multi-Layer Switch

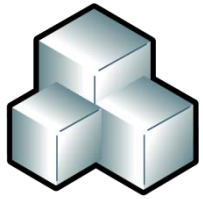
Internet cloud



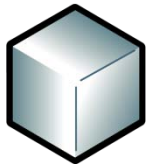
Pattern Book with CD



Pattern Book

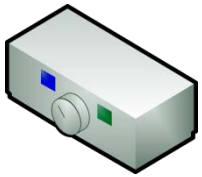


Services

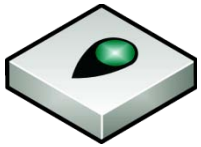


Service

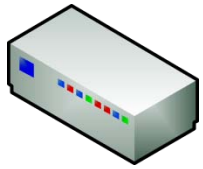
Monitor



AB switchbox



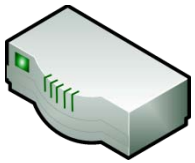
Biometric reader



Bridge

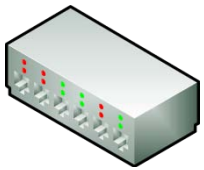
Camera

Information Iconography

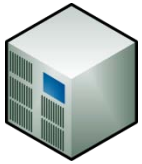


router

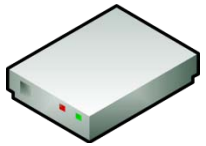
Patch panel
switch box



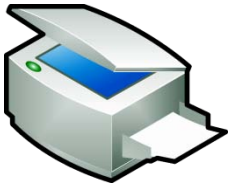
Hub



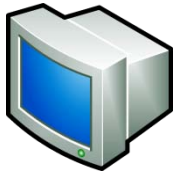
PBX box



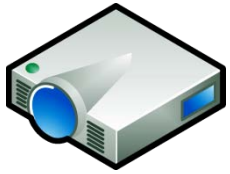
modem



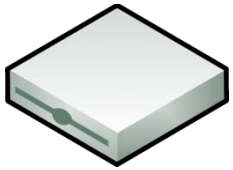
Copier



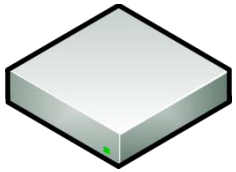
CRT monitor



CRT projector



External media device

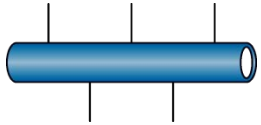


External hard drive

Fax machine
phone

User 1

Digital video
camera



Ethernet cable
network

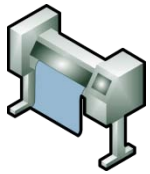
Fiber optic transmitter

Information Iconography

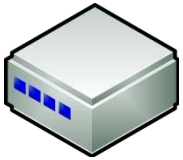

Projection screen

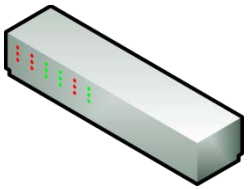

Printer

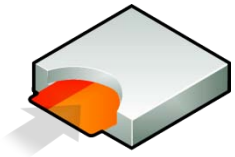

Printer, Copier, Fax


Plotter printer


Phone


Repeater

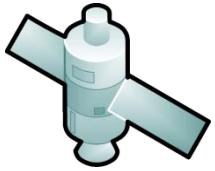

Switch

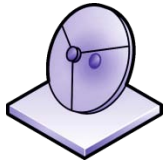

Smart card reader

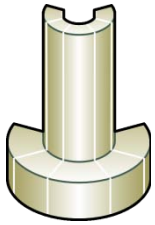

Scanner


Wireless access

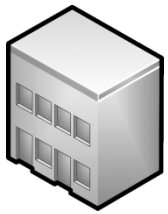

Tablet PC

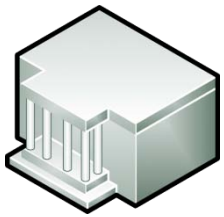

Satellite


Satellite dish



Buildings


Building


University


House yellow

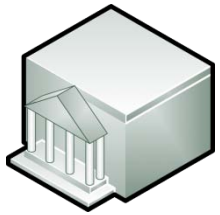

House tan


House purple


House green


House blue

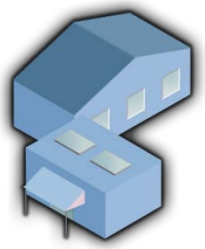

House red


Government building


Enterprise sand


Factory red


Factory yellow


Factory blue


Enterprise blue


Enterprise red

Buildings

Building brick

Building blue

Building red

Building gold

Building teal

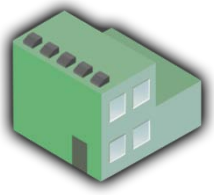
Building purple

Building white with tree

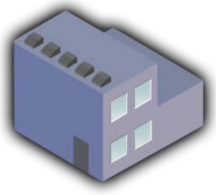
Building purple with tree

Building green with tree

Building blue with tree



Small business green



Small business purple

Small business sand

Small business rose

Small business yellow

Buildings

Manufacturer
green

Manufacturer
blue

Manufacturer
olive

Manufacturer
tan

Manufacturer
blue

Icons

Policy rukes

claim

Claim status

Xml

Claim form

Payment

Patient Data

2009软件模式与实践高峰论坛

Patterns & Practices Summit 2009

12.10 - 12.11,2009

中国 · 深圳

Build Better WPF & Silverlight applications using Prism v2

David Hill
Principal Architect
Microsoft patterns & practices

Client Application Challenges

- The Problem:
  - Client Applications can be Difficult!
  - How Do You Make The Application Dynamic, Customizable, Extensible, Testable?

- The Solution:
  - Break Application Into Modules
  - Manage Dependencies & Interactions
  - Re-assemble Application From Modules

- Prism = Patterns For Composite Client Apps

What's In The Box?

- Prism – Composite Client Application Guidance for WPF and Silverlight
  - Library
  - Reference Implementation
  - Documentation
  - Quick-Starts & How-To's
  - Community – CodePlex

- Prism 1.0 – WPF
  - Released July 2008
- Prism 2.0 – WPF & Silverlight
  - Released Feb 2009
- Prism 4.0 – WPF & Silverlight 4.0
  - Coming Soon!

Prism Core Concepts

Bootstrapper

DI/IoC Container

Modules

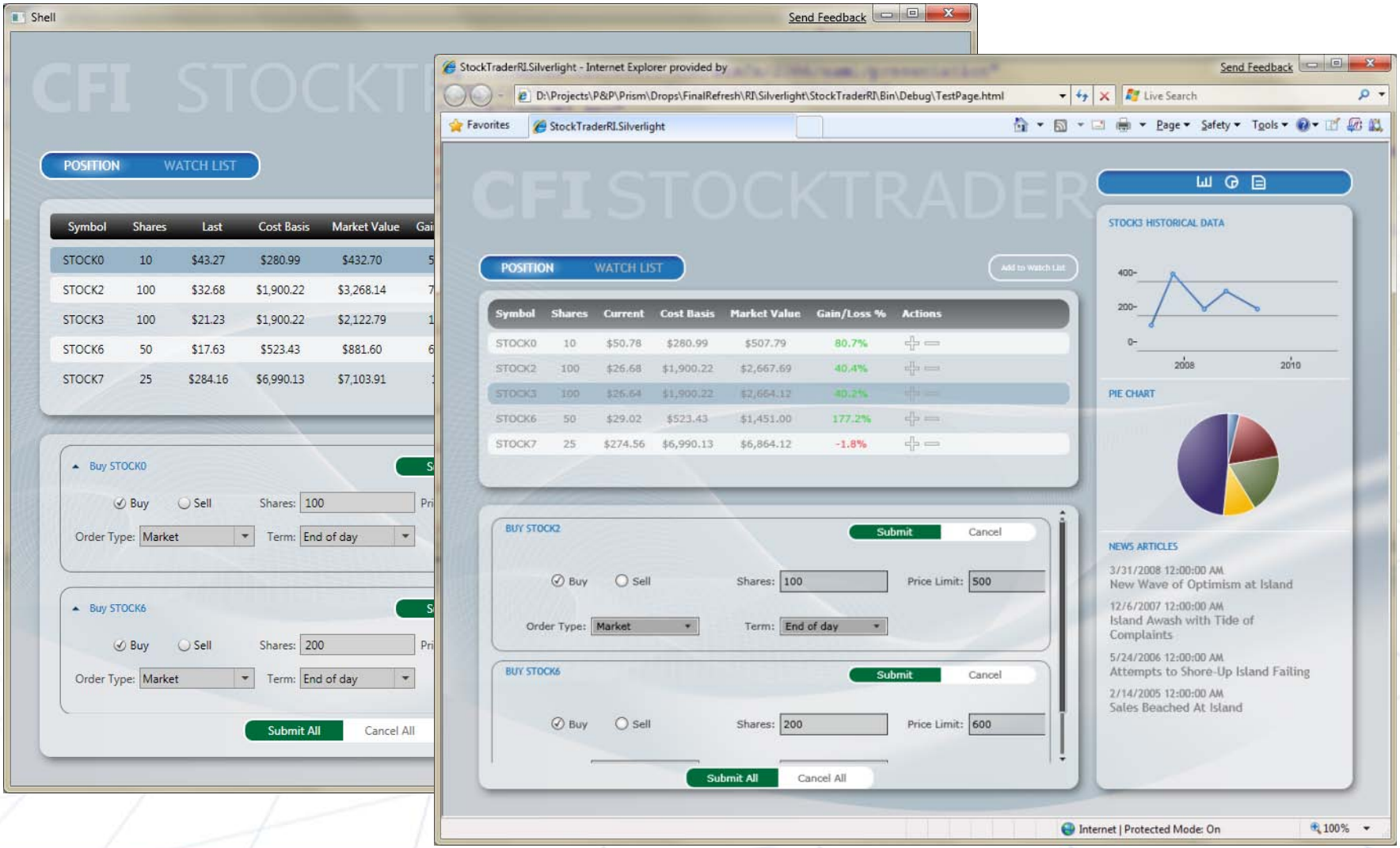
UI Composition

Separated Presentation

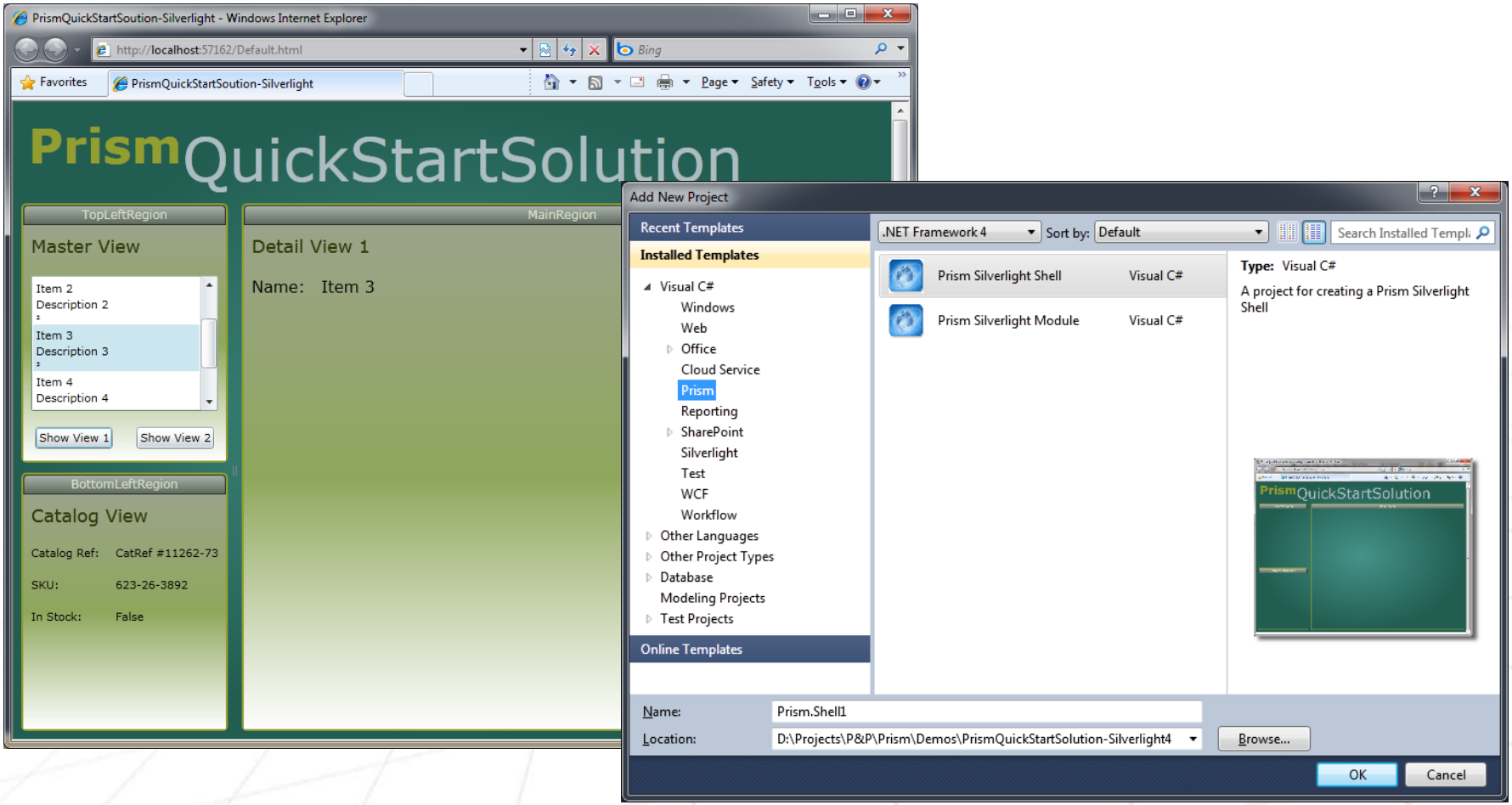
Events/Commands

Multi-Targeting

Reference Implementation



Prism Quick Start Templates



Available From : http://blogs.msdn.com/dphill

- StockTrader Reference Implementation

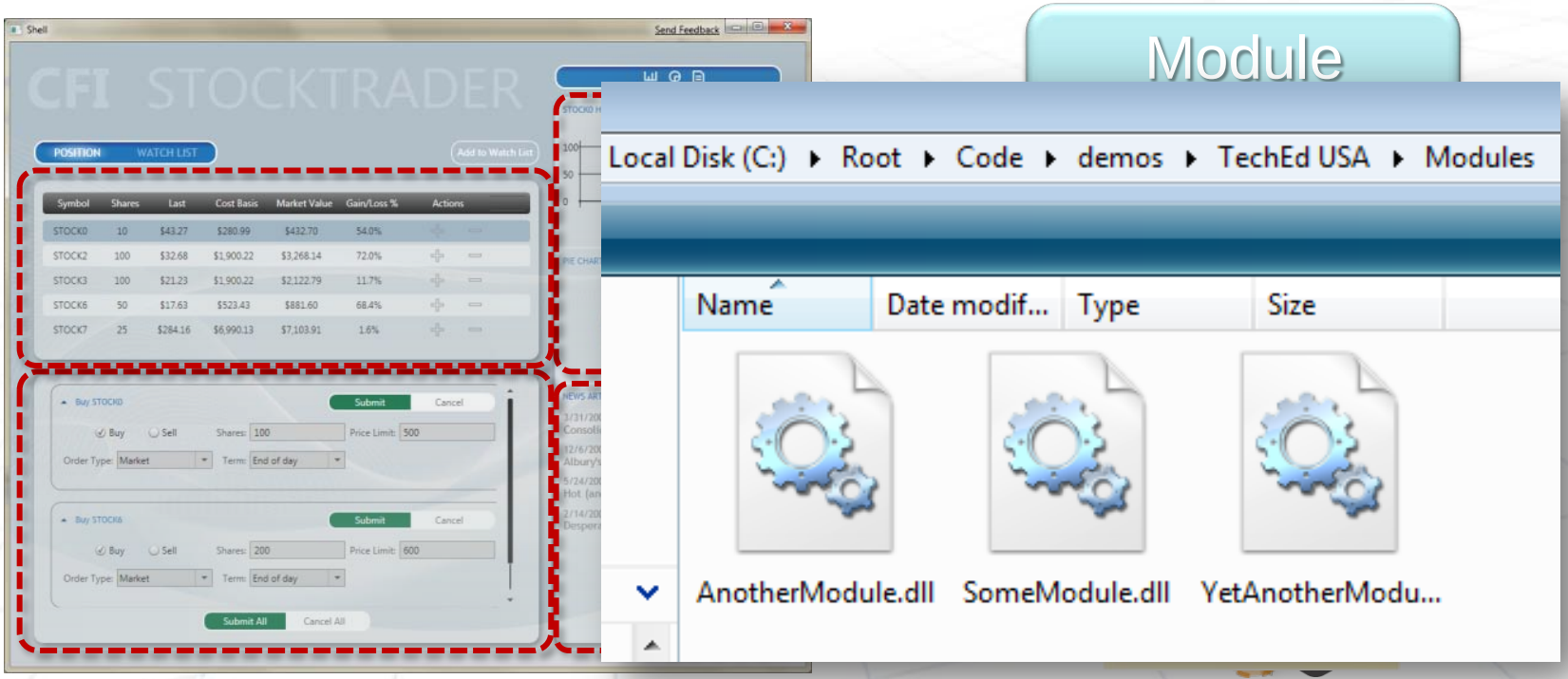
- Quick Start Templates

What Are Modules?

- Unit Of Application Assembly
  - Collection of Related Components
  - Feature, Services, Views, Data Access
  - Slice & Dice: Mandatory, Optional, Role Specific
- Unit Of Development
  - Independent Development
  - Independent Testing
- Unit Of Deployment
  - Up-Front, Background or On-Demand

Finding & Loading Modules

- Module Discovery
  - Pluggable Catalogs
- Module Loading
  - Background or On-Demand



Building the UI

- Shell – Application Host Window
- Regions – Named Areas For View Placement
- Presentation Logic

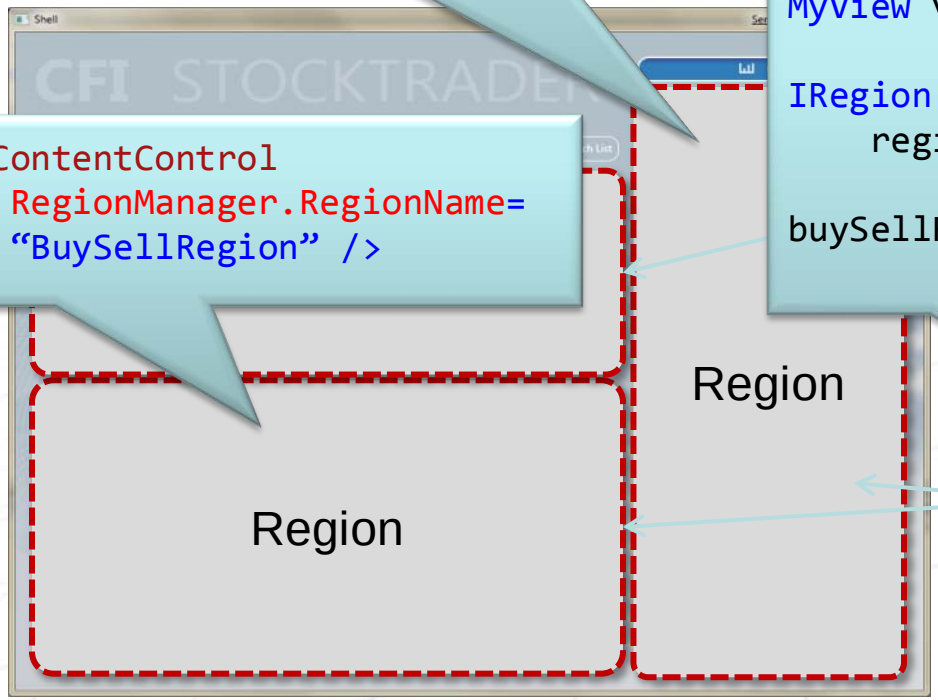
```
<ItemsControl
    RegionManager.RegionName=
    “DataRegion">
```

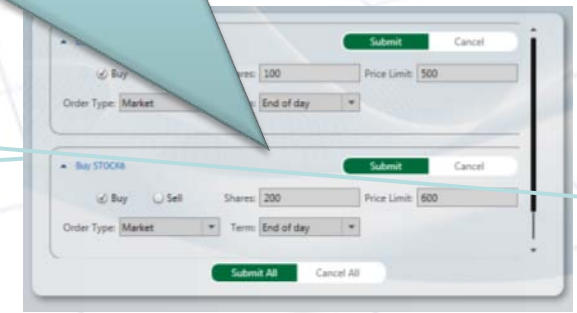
```
<ContentControl
  RegionManager.RegionName=
  “BuySellRegion” />
```

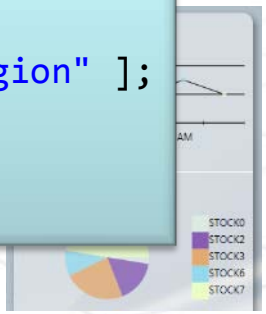
```
MyView view = …;

IRegion buySellRegion =
    regionManager.Regions[ “BuySellRegion" ];

buySellRegion.Add( view );
```







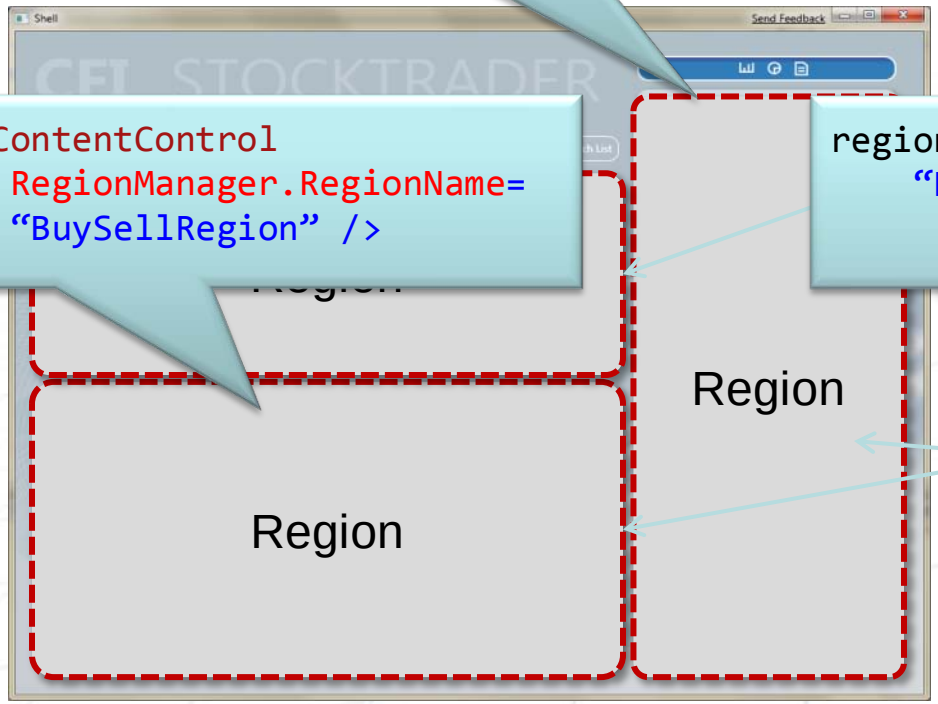
Building the UI Automatically

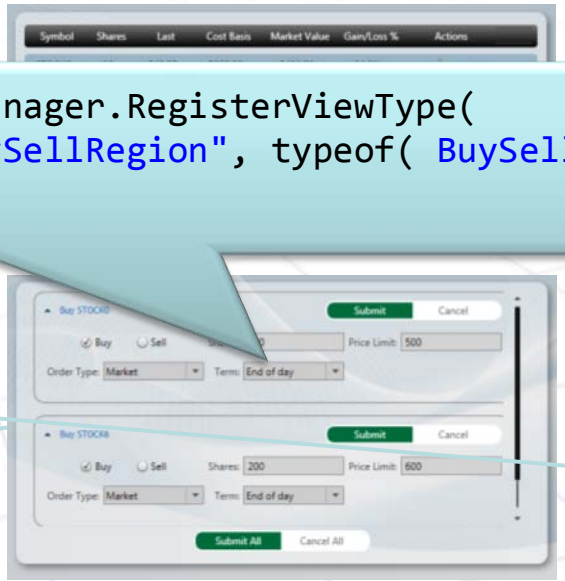
- View Discovery:
  - Less Complex
  - mbly' Composition
  - to Region

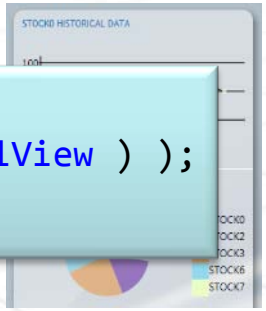
```
<ItemsControl
    RegionManager.RegionName=
    "DataRegion">
```

```
<ContentControl
  RegionManager.RegionName=
  "BuySellRegion" />
```

```
regionManager.RegisterViewType(
    "BuySellRegion", typeof( BuySellView ) );
```

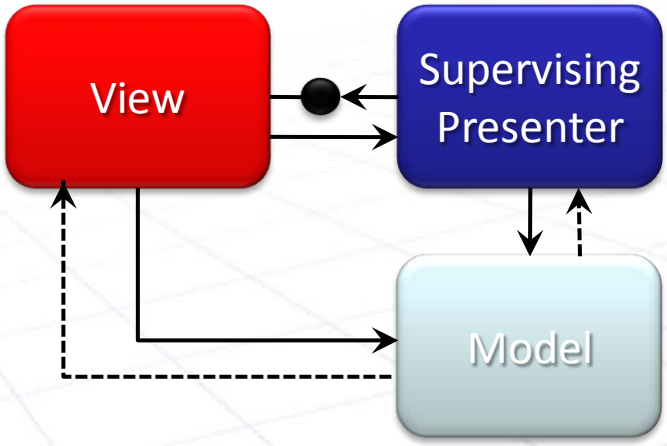






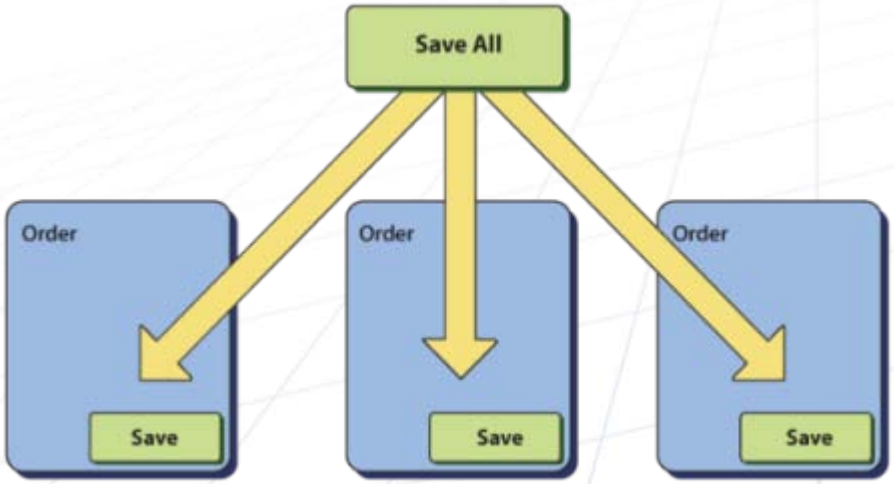
Separated Presentation

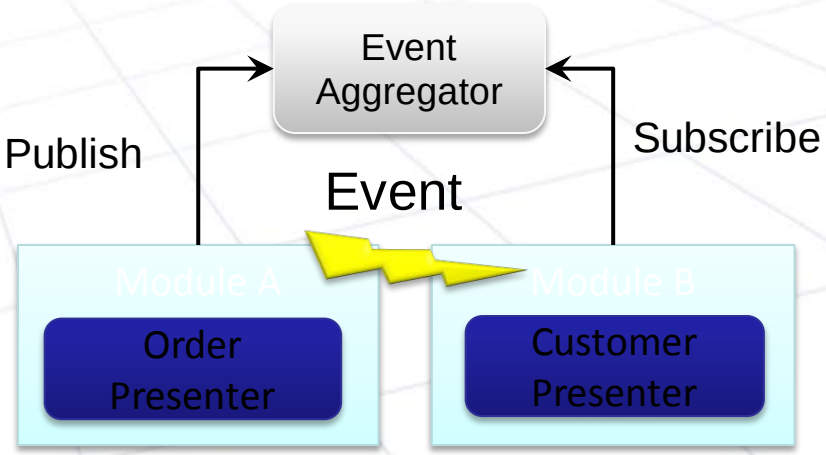
- Common UI Patterns
  - Supervising Presenter
  - Presentation Model (“Model-View-ViewModel”)
- Views:
  - Encapsulate UI & UI Logic
  - Minimal Code Behind
  - UI Designer Friendly
- Presenters:
  - Presentation Logic & State
  - Unit Testable
  - View Independent



Commands and Events

- Delegate Commands
  - Delegate Based Commanding Pattern
- Composite Commands
  - Multiple Handler Command Routing
- Event Aggregator
  - Loosely Coupled Pub/Sub Events

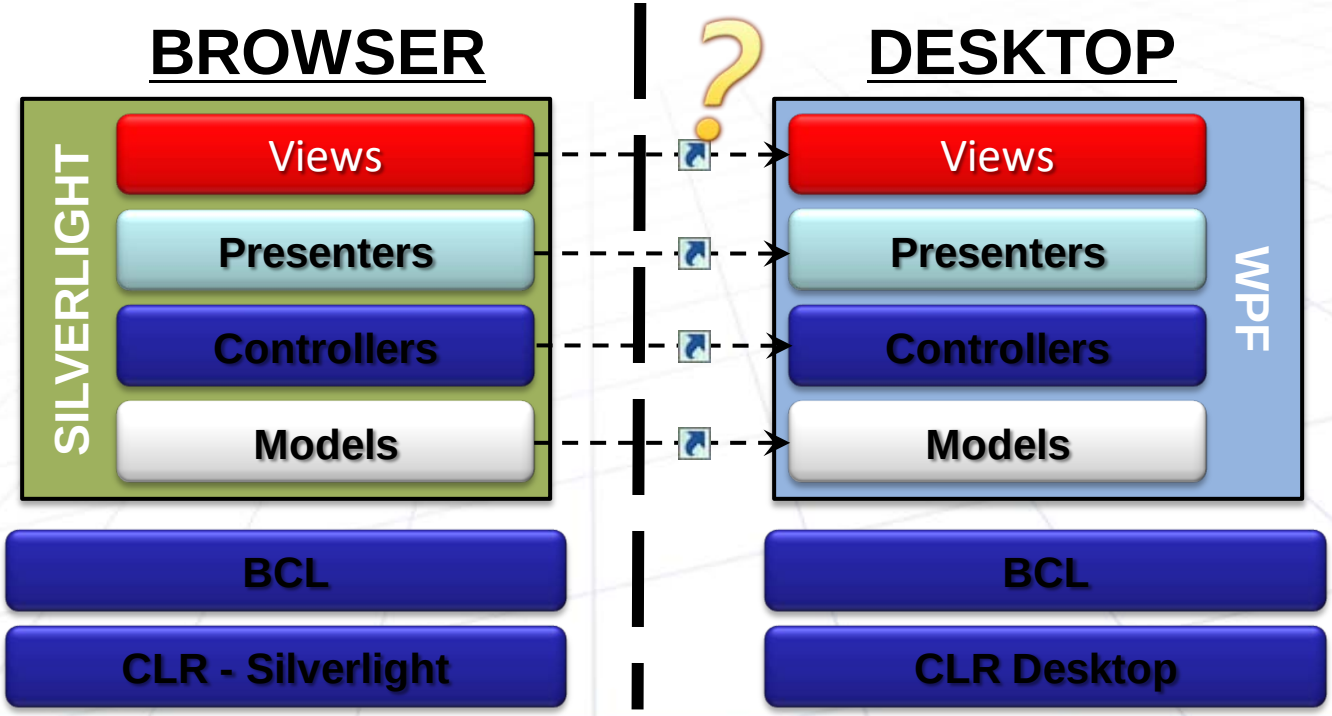




- Separated Presentation
- Commands

Multi-Targeting

- Deployment For User Experience
  - Desktop: In the Office, Full Functionality Offline Capable
  - RIA: Out of the Office, Functional Subset, Online Only

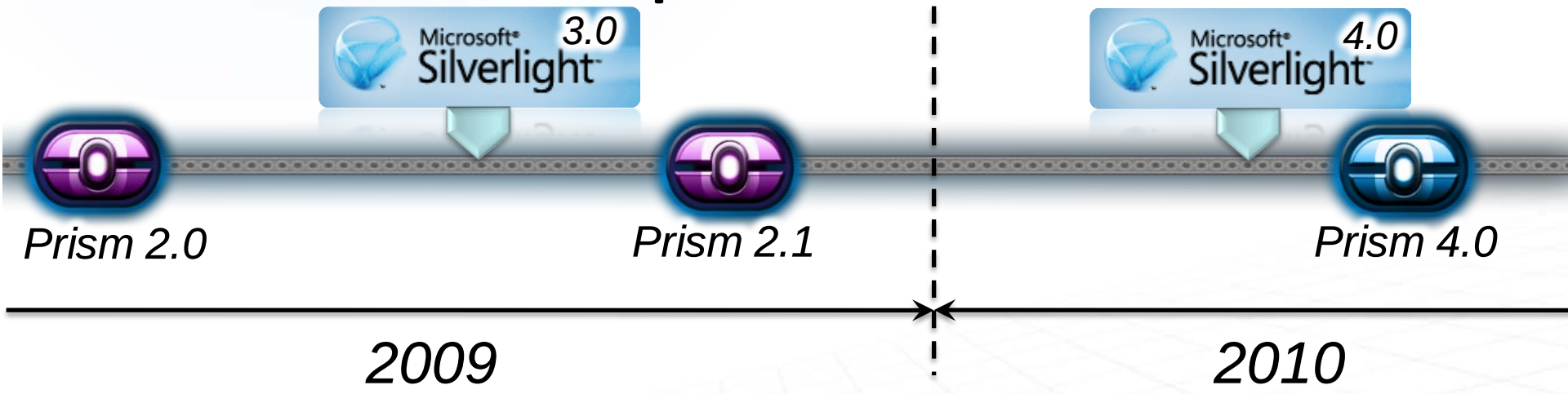


- Multi-Targeting

Summary

- Library of Patterns for Client & RIA Applications
  - Modularity, Composition, Separated Presentation…
- Solid but Flexible Architectural Foundation
  - Promotes Re-use, Unit Testing, Independent Development
  - Supports Design-Time & Run-Time Extensibility
- Multi-Targeting
  - Promotes Re-use across WPF & Silverlight
  - Support Multi User Experience or Migration
- Code, Reference Implementation, Documentation & Quick-Starts Available on MSDN & CodePlex
- What's Next?
  - Next Release for WPF & Silverlight 4.0 Spring 2010
  - Send us feedback & ideas for Prism 4.0!

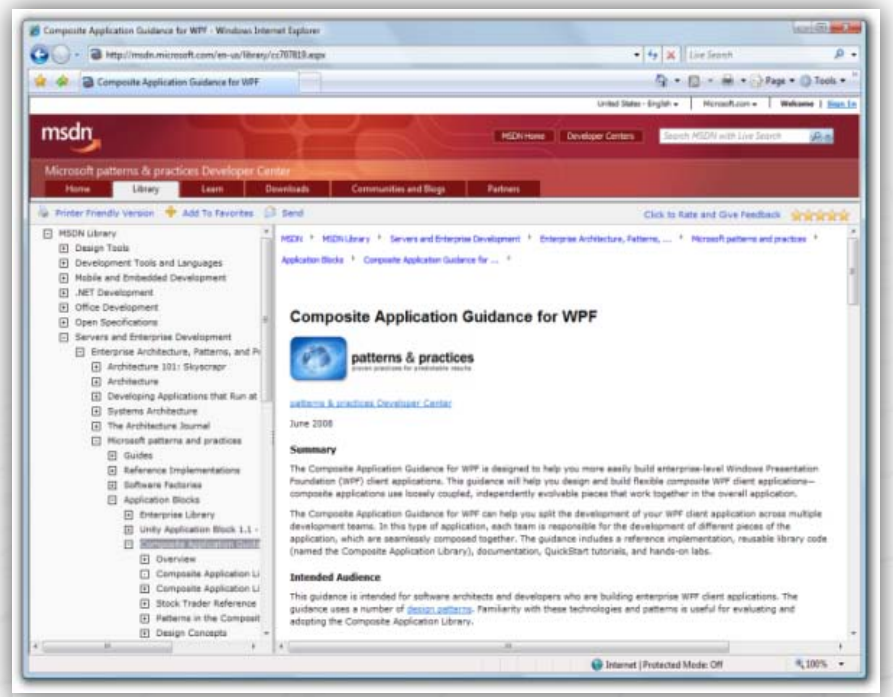
Prism Roadmap

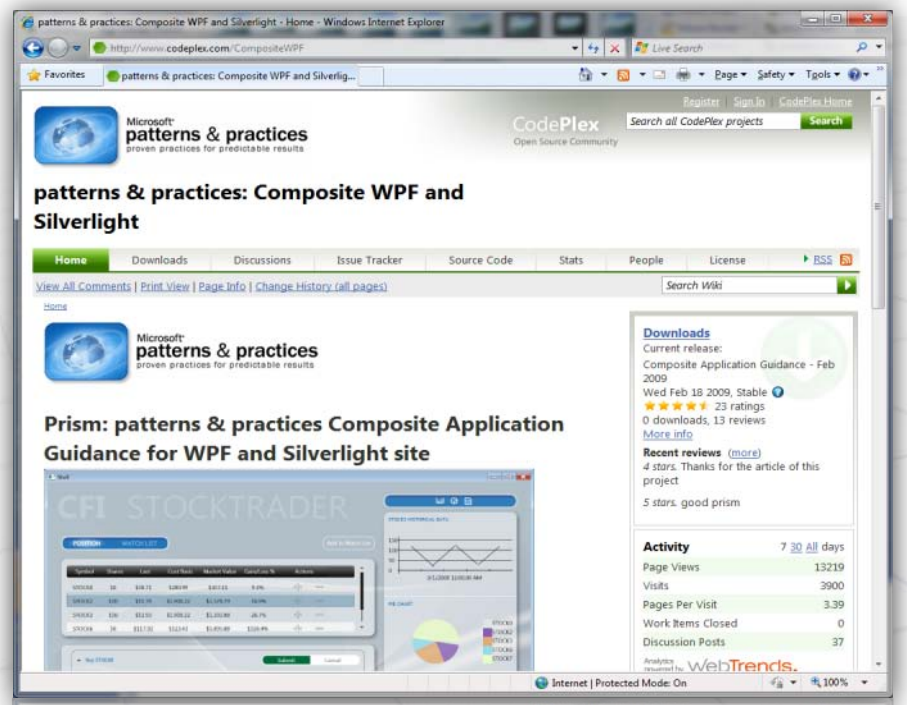


- What's Coming in Prism 4.0?
  - Silverlight 4.0 and WPF 4.0 Support
  - More ViewModel Pattern Guidance
  - Managed Extensibility Framework (MEF) Support
  - Guidance For Out of Browser Applications
  - ???

Where Can You Find It?

- www.microsoft.com/prism
- www.codeplex.com/prism





12.10 - 12.11,2009

中国 • 深圳

Making your life easier with Enterprise Library

Don Smith
Community Liaison
Microsoft patterns & practices

Microsoft patterns & practices

Motivation

Customers want to:

- → leverage **proven practices**
- → drive **productivity** up, improve **economics**
- → more effectively **jump-start** junior developers
- → achieve **consistency** & **compliance**
- → drive **quality: maintainability**
- → drive **quality: mature** code **components**

EntLib is this, in **actionable** form

Documentation
(PDF, HTML, CHM)

Pluggable Binaries

Quickstarts

Acceptance Tests

Source
Code

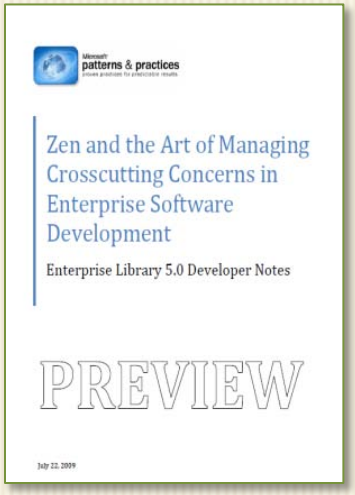
Unit Tests

in the Box

Out of the BOX

FAQs

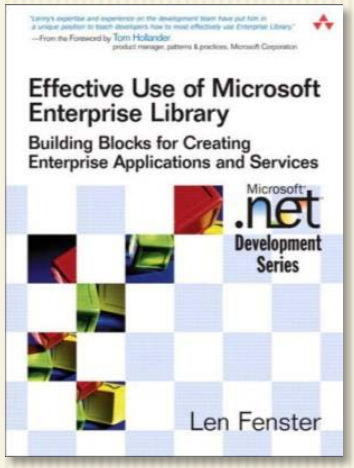
Dev Guides



Community extensions

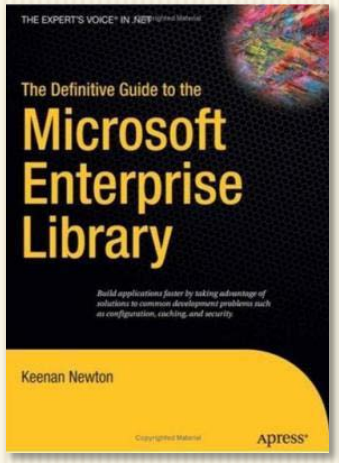
videos & demos

Discussion Forums

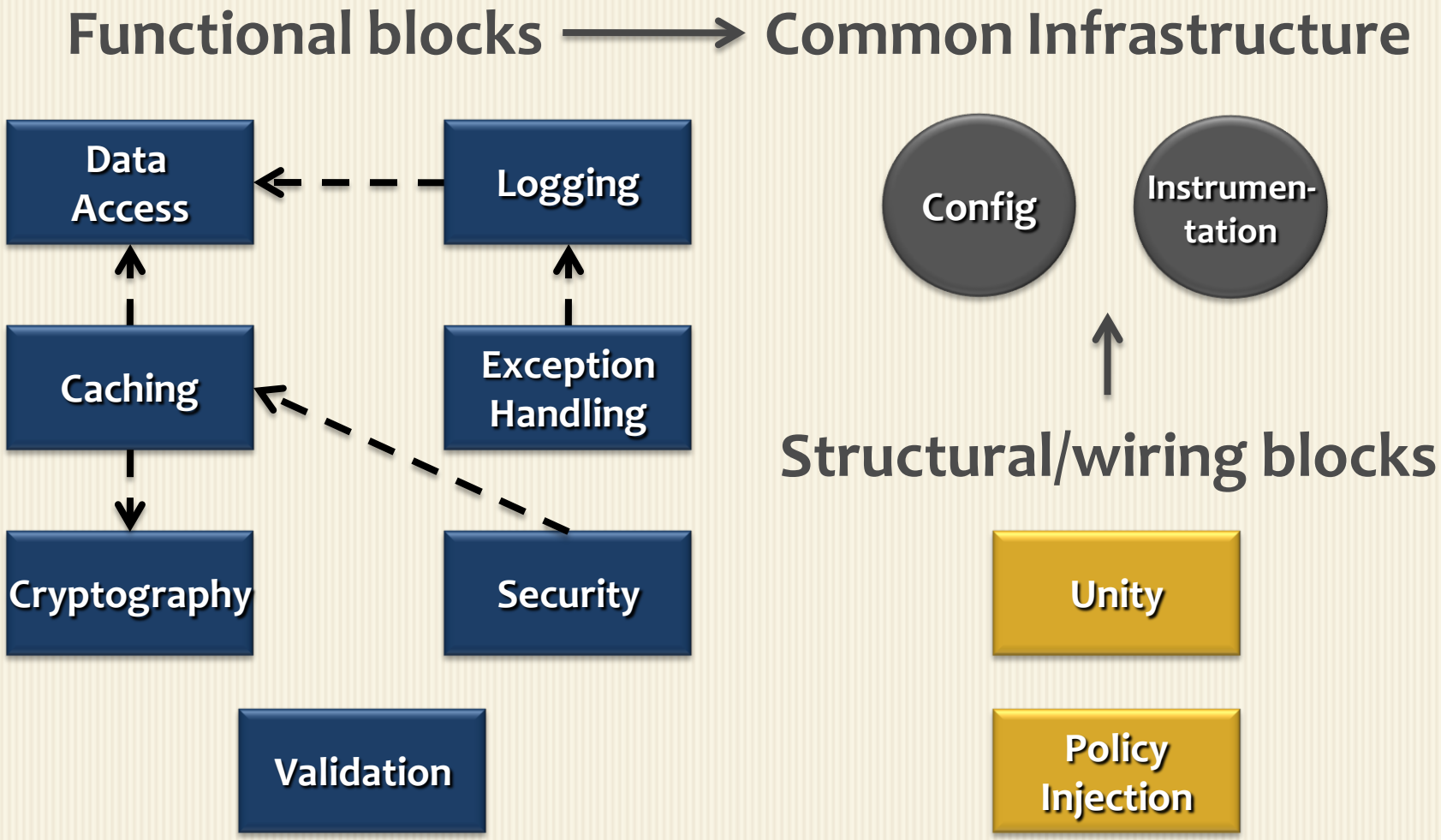


Hands-on Labs

Industry Books



Conceptual Architecture



Design Goals

Consistency

Blocks are written with and used in common patterns

Extensibility

Pluggable extension points

Ease of use

Configuration tool, tons of documentation, simple interfaces, hands-on labs, webcasts

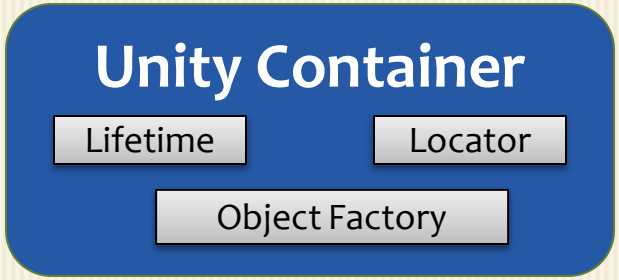
Integration

Work well together or separately

Dependency Injection

- The Hollywood principle
- Decouples dependencies
- Improving testability
- Dependency injection container will…
  - figure out which constructor to call
  - figure out which objects need to be created to pass to the constructor
  - create the actual objects passed as constructor parameters
  - map interfaces to concrete implementations

  … and it's recursive!
- Unity is one such container



→ demo

Logging Application Block

patterns & practices

Logging Application Block

- Provides a simple model for logging events
  - Strongly typed, extensible log schema
- Built on top of System.Diagnostics
- Configuration driven – you decide what messages are logged where at runtime
- Use any .NET TraceListener, including supplied formatter-aware listeners:
  - EventLog, Database, Flat File, Rolling Flat File, MSMQ, E-mail, WMI, XML or create your own
- Tracer class lets you time key activities and correlate any enclosed events

→ demo

Exception Handling Application Block

patterns & practices

Exception Handling Application Block

- Provides **simple mechanism** that allows you to consistently deal with exceptions throughout your application
- Define "**Exception Policies**" which link an exception to an action, e.g.
  - Exceptions of type ApplicationException should be logged
  - Exceptions of type SqlClientException should be caught and wrapped with an exception of type DataLayerException and re-thrown
  - Exceptions of type SecurityException should caught and replaced with an AccessDeniedException which will be thrown
- Actions provided include
  - Logging
  - Wrapping one exception with another
  - Replacing one exception with an other
  - Map to WCF Fault Contract

→ demo

Data
Access
Application
Block

Data Access Application Block

- Provides simplified access to the most often used features of ADO.NET with applied proven practices
- Improve Consistency
  - Write code that works against multiple database brands
  - Integrate with System.Transactions functionality
- Improve ease of use
  - Easily call a stored procedure with one line of code
  - Easily consume results from a sproc call
  - Let the block manage the lifetime of database connections
  - Work with database connection strings stored in configuration or specified in code
- NOT an Object Relational Mapper (O/RM) like EF

LINQ Style Result Processing

- Basic idea:
  - Instead of a DataReader, get back an IEnumerable<T>
  - LINQ to Objects
- Introducing Accessors
  - Provide a higher level abstraction that combines input mapping, output mapping, and result set management into a single object
- Accessors can be easily injected

Summary

- You cannot afford not to reuse
- Enterprise Library is a mature free set of pluggable components
- A-la-carte approach
- DI-style development for the win!
- Focus on testability
- Engage with us!
- Hands-on labs are a good start

- Happy coding!

Resources

- Released versions and related resources:
  - msdn.microsoft.com/entlib
- Pre-releases for version 5.0:
  - entlib.codeplex.com
  - unity.codeplex.com
- Community sites:
  - entlib.codeplex.com
  - entlibcontrib.codeplex.com
- patterns & practices home page
  - msdn.microsoft.com/practices
- Grigori's blog:
  - blogs.msdn.com/agile/

12.10 - 12.11,2009

中国 • 深圳