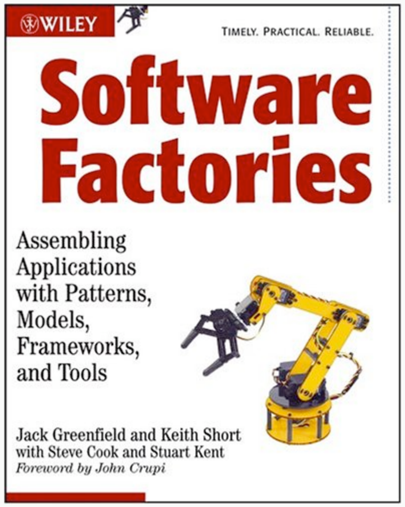
Software Factories & Model Driven Development

Jimmy Figueroa
Regional Director
.NET Platform
Microsoft Corporation



WILEY

Timely. Practical. Reliable.

Software Factories

Assembling Applications with Patterns, Models, Frameworks, and Tools

Jack Greenfield and Keith Short
with Steve Cook and Stuart Kent
Foreword by John Crupi

Common Challenges

- What kind of system do we need to build?
  - Where can we find a list of system types?
- What is the architecture of this kind of system?
  - What artifacts do we need to build and how are they related?
  - What technologies should we use and how should we configure each one?
- What is the best way to build this kind of system?
  - What is the best path from requirements to production?
  - What are the key decisions that need to be made?
- How do we go faster without compromising quality?
  - Why are there always so many changes late in the project?
  - Why is it so hard to estimate how long projects will take?

Models and Methods

- Weren’t they supposed to help answer those questions?
- Why are methodologies so abstract?
  - Why do they give the same advice whatever we’re building?
  - Where can we find concrete guidance for this project?
- Why aren’t models more effective?
  - How do I model structures that are larger than classes?
  - Why don’t models describe key design elements more clearly?
  - Why don’t the tools generate production quality code?
  - Why don’t the models stay synchronized with the code?

General Pain Points

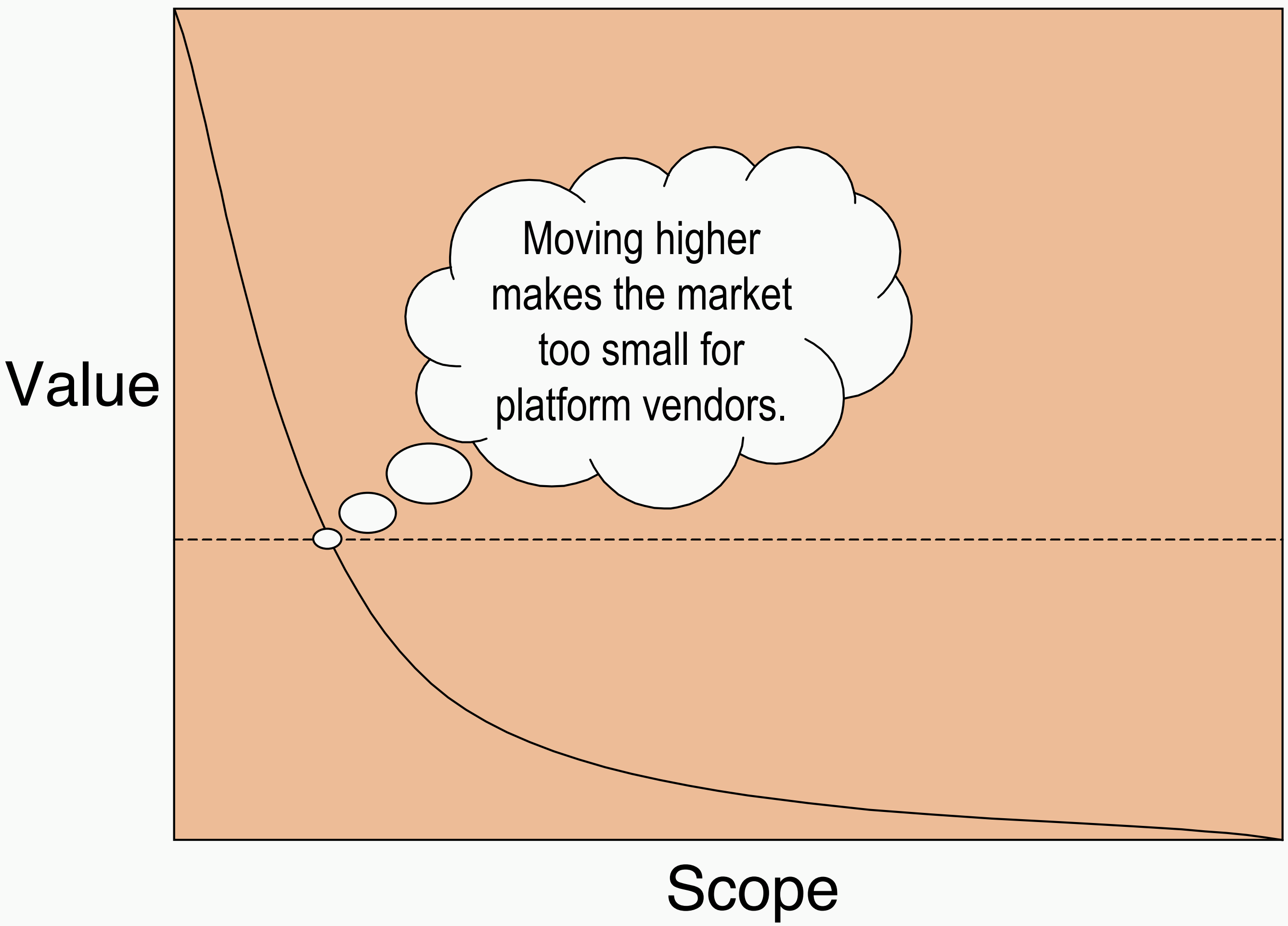
- Defect levels are too high
- Hard to predict the effects of new requirements
    - On design, implementation, deployment, operations, budget, schedule, process
- Junior developers need to learn from experts
- Not enough experts to give hands on guidance
- Takes too long to train new developers
- Many copies of the same thing are hard to maintain
- Similar systems have different quality attributes
    - Usability, reliability, performance, security

Software Development as Craftsmanship

- Labor Intensive
- Generic Tools
- Generic Processes
- One off applications
- Hand stitched from scratch
- Minimal reuse

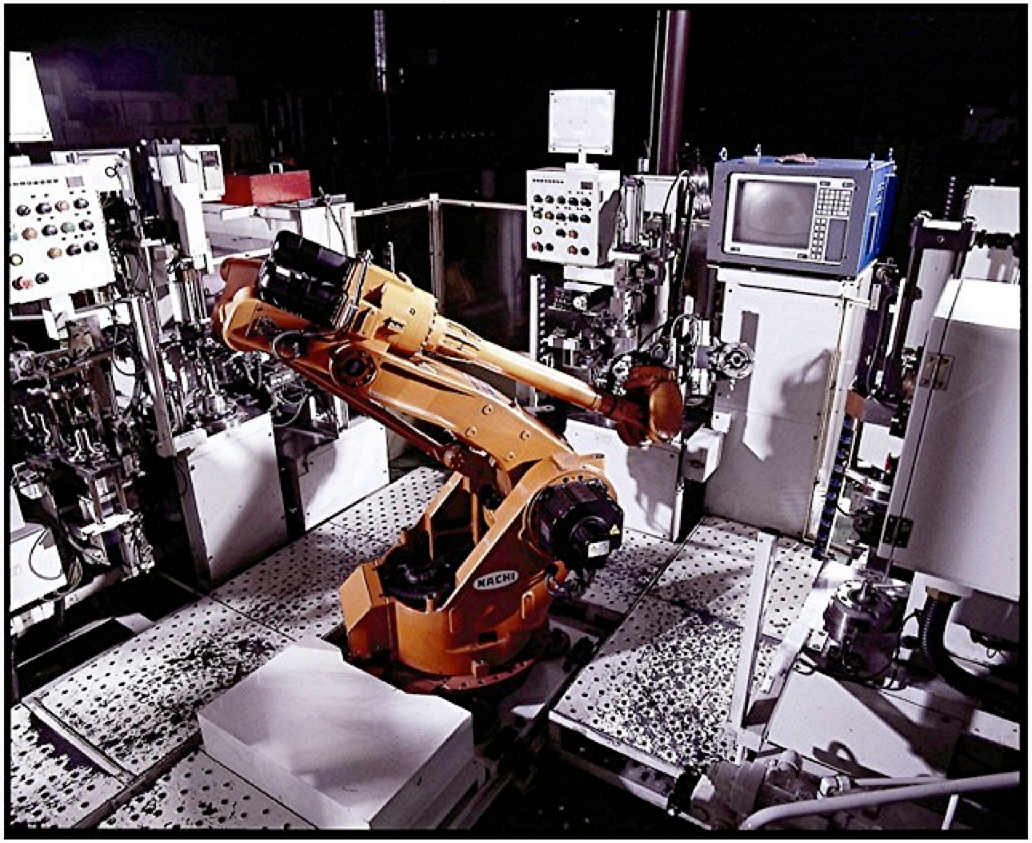
Overruns, defects, security holes, project failures

Value vs. Scope



Moving higher makes the market too small for platform vendors.

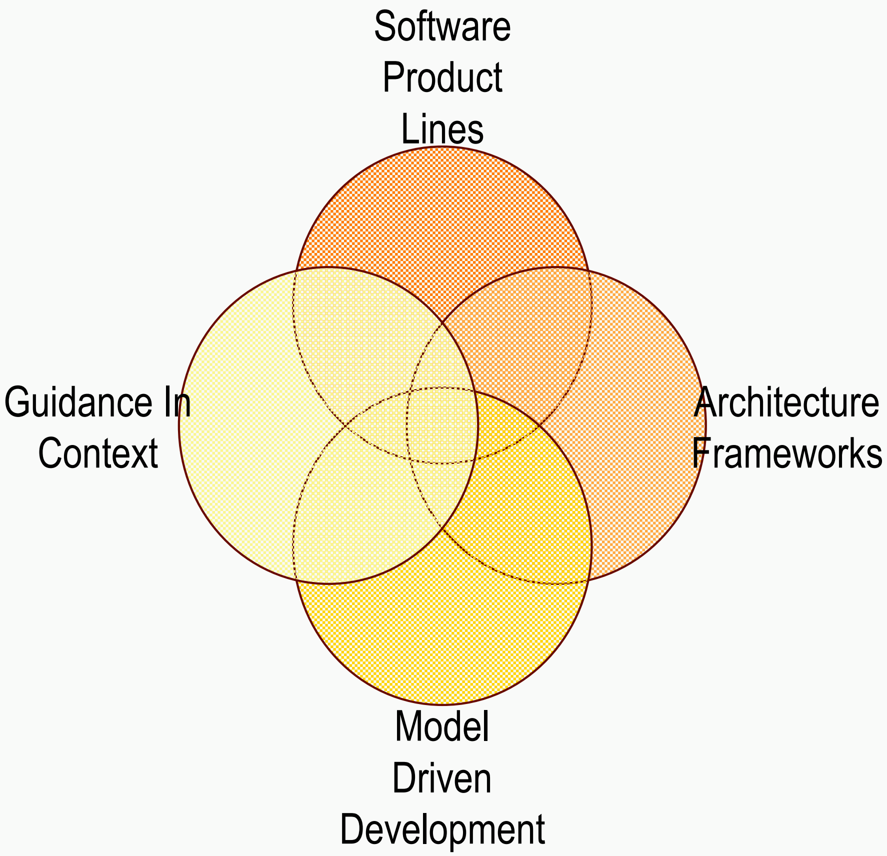
Software Factories



- *Domain-specific process*
- *Domain-specific tools & languages*
- *Domain-specific content*
- *Automate* rote and menial tasks
- *Define* Variability and product architecture
- *Improve* through experience and measurement

General-purpose IDEs become domain-specific software factories

Industrializing Software Development

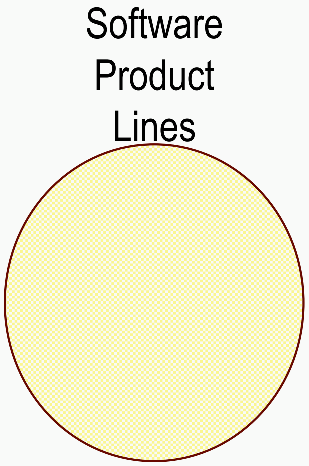


- Improve productivity and predictability across the software life cycle
- Make it easy to deliver a wide range of tailored solutions that satisfy needs of individual consumers

Every organization a packaged application developer

Software Product Lines

- Build new solutions by assembling partial solutions and/or configuring general ones
- Specify only the unique features of each solution and assume the common ones
- Variations in requirements map predictably to variations in artifacts and processes
- Reduce custom development by 40% to 80% for the typical solution



A set of systems sharing a set of managed features that satisfy the specific needs of a particular market segment and that are developed from a common set of core assets in a prescribed way.

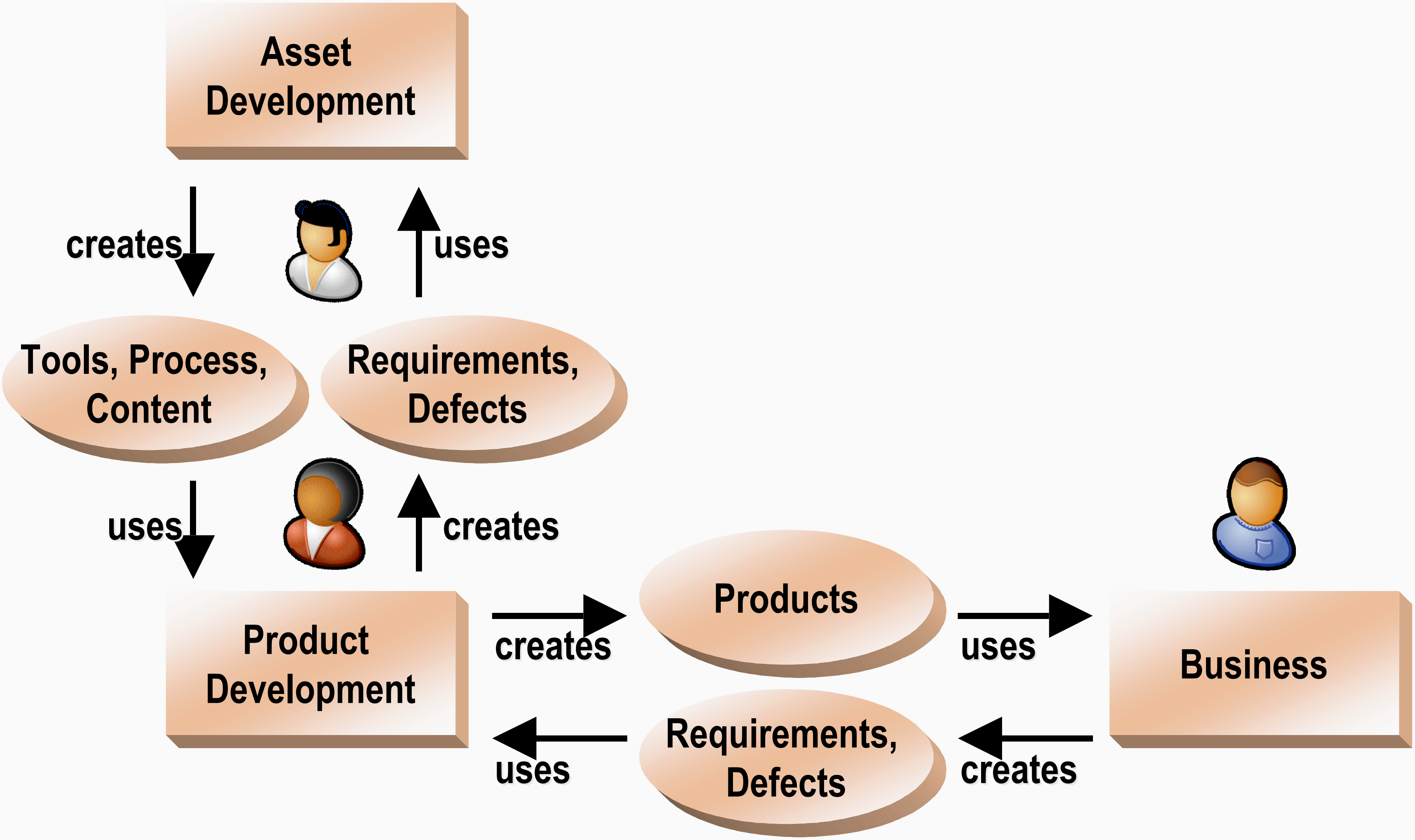
Clements and Northrop

Exploiting Commonality

- We can also exploit *economies of scope*
- Reuse *designs* & *components*
- Build many similar but distinct *prototypes*
- Key is supporting *variability*

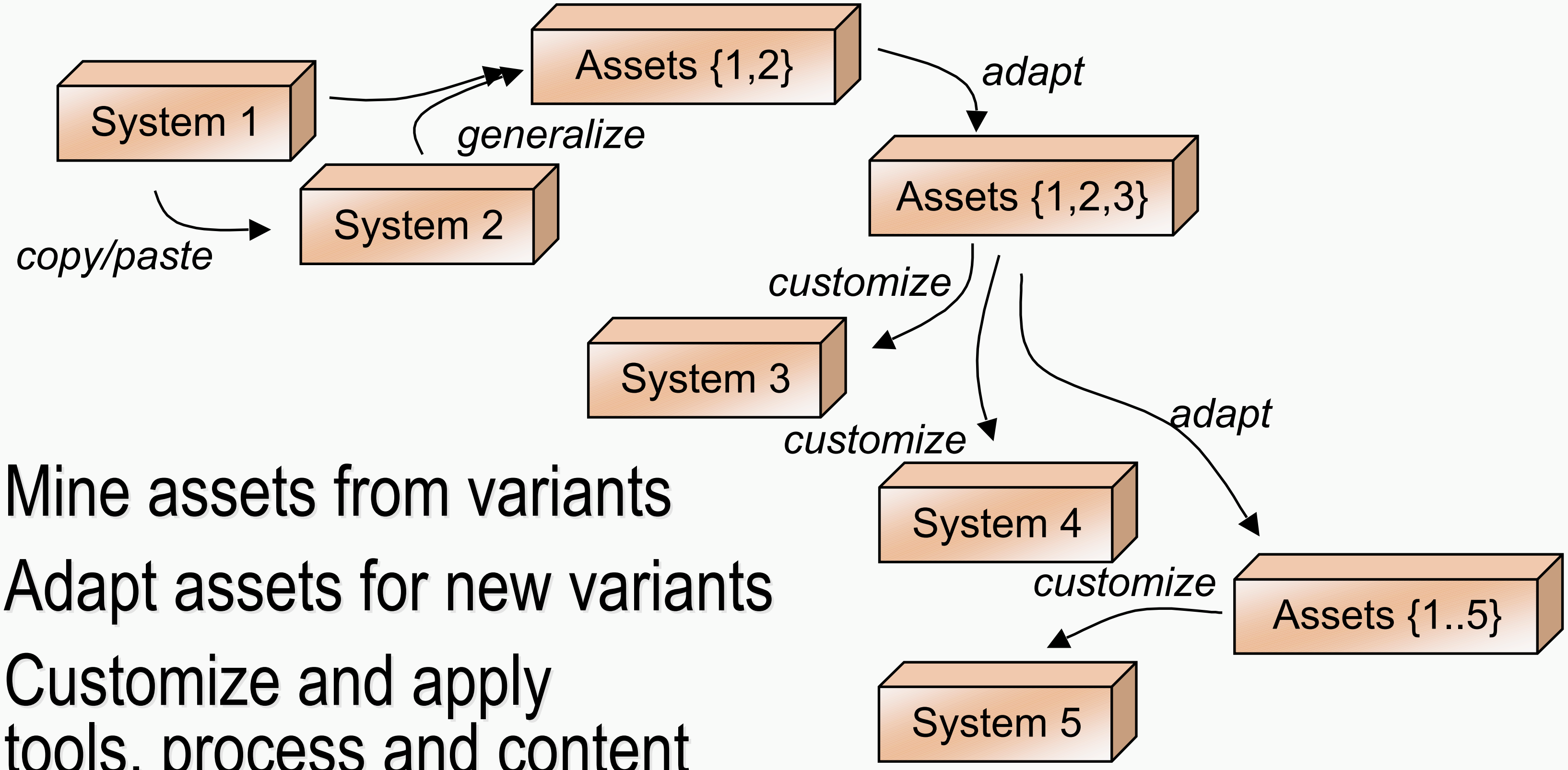
Define only the unique pieces of each system

Another Development Process



How Product Lines Form

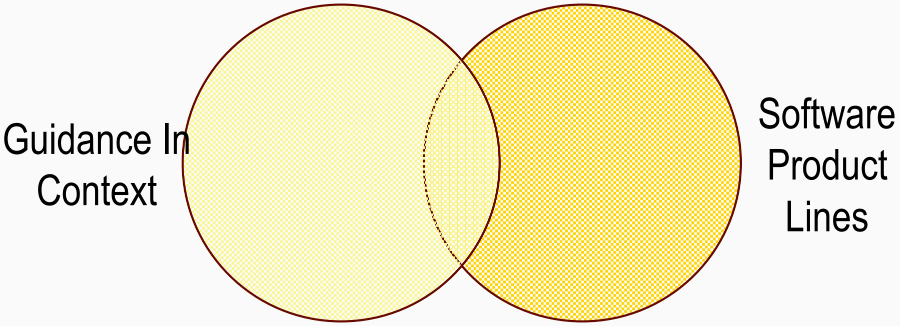
- Variants of a system
  - e.g. CRM system



- Mine assets from variants
- Adapt assets for new variants
- Customize and apply tools, process and content for each variant

Guidance In Context

- Provide guidance that helps practitioners know what to do and that helps them do it
- Build installable packages containing organized sets of configurable guidance assets for common use cases
- Attach guidance to steps in the process and parts of the architecture
- Scope process steps with pre and post conditions to let project work flow vary subject to constraints

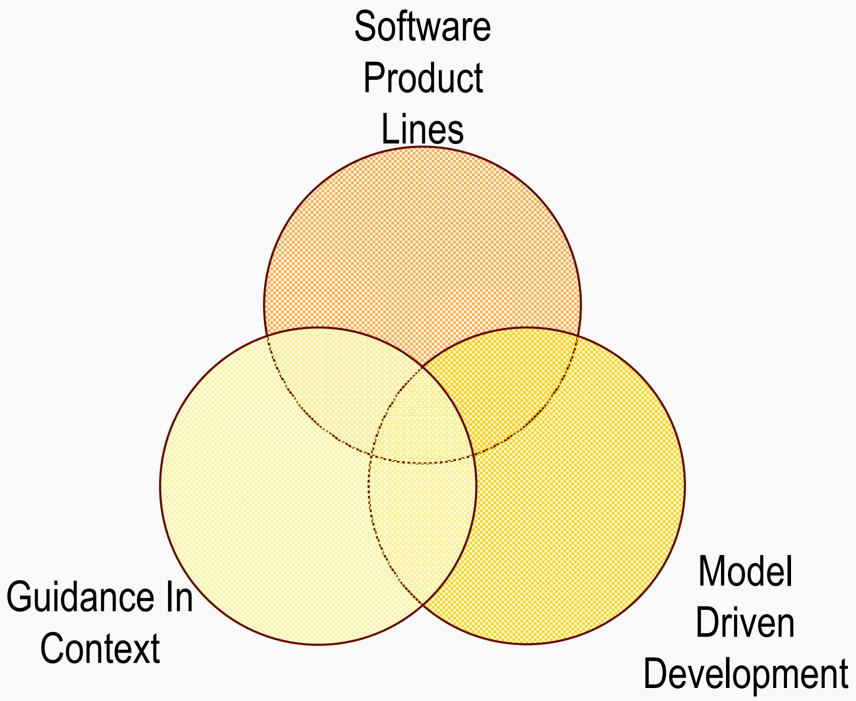


Guidance Packages

- Packages of guidance assets
  - Templates for text like ASP.NET pages
  - Templates for solutions, projects, items
  - Actions that modify solution artifacts
  - Scripts called recipes that invoke actions
  - Wizards that use the templates and recipes
- Simple user experience
  - Wizards gather information from user
  - Generate a solution OR inject artifacts into existing solutions
  - Recipes attach to solution artifacts
  - Recipes cue new recipes to unfold a process

Model Driven Development

- Create highly focused custom languages for specific problems, platforms or tasks
- Develop custom tools to support them
- Use metadata captured by models for automation



The history of programming is an exercise in raising the level of abstraction as language developers build new languages from lessons learned...

Smith and Stotts

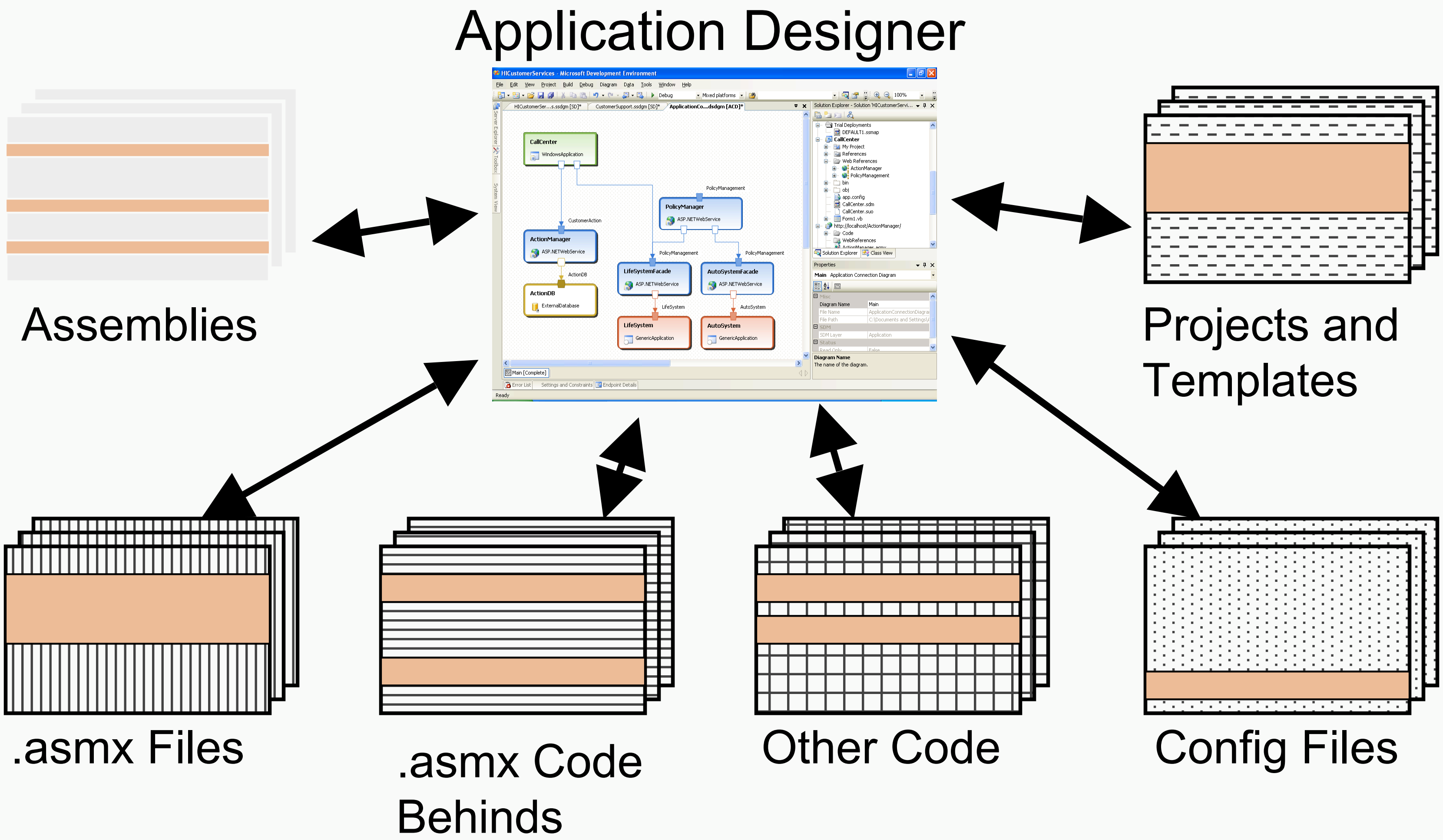
Domain Specific Languages

- Highly focused custom languages
  - Designed for specific problems, platforms or tasks
- Many proven examples
  - SQL, GUI builders, HTML, regular expressions
- Make solution easier to understand and maintain
  - Improve agility through rapid iteration

The good thing about bubbles and arrows, as opposed to programs, is that they never crash.

Bertrand Meyer

Raising The Level Of Abstraction



Application Designer

Assemblies

Projects and Templates

.asmx Files

.asmx Code Behinds

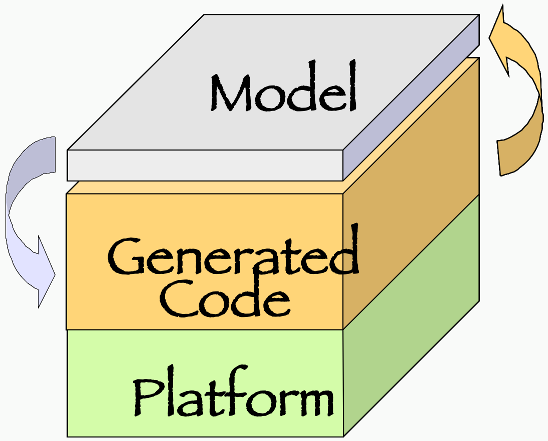
Other Code

Config Files

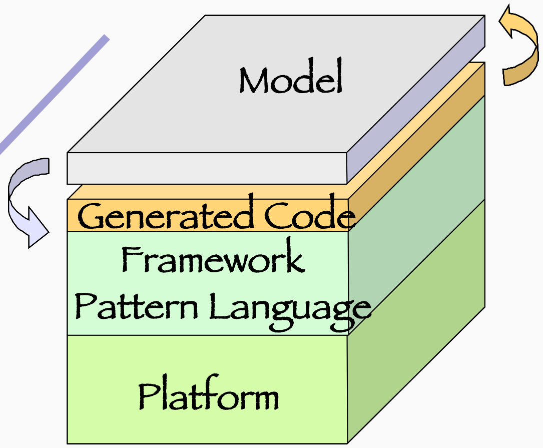
Diagram Name: Main
File Name: ApplicationConnectionDiagra
File Path: C:\Documents and Settings\
SDM Layer: Application
Read Only: False

Diagram Name
The name of the diagram.

Effective Transformations

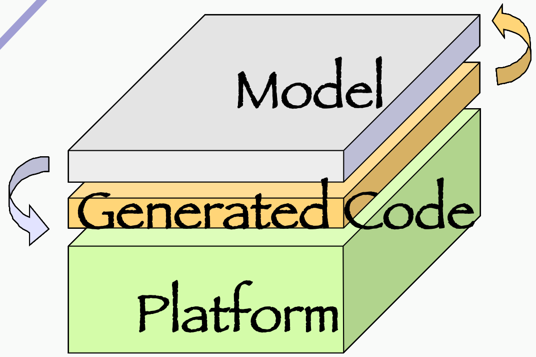


Old Case

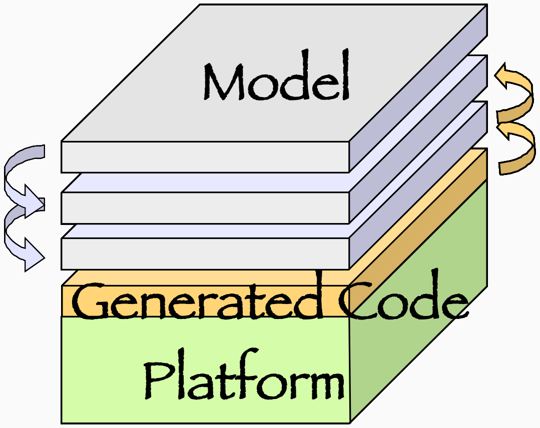


Generate to Framework

DSLs

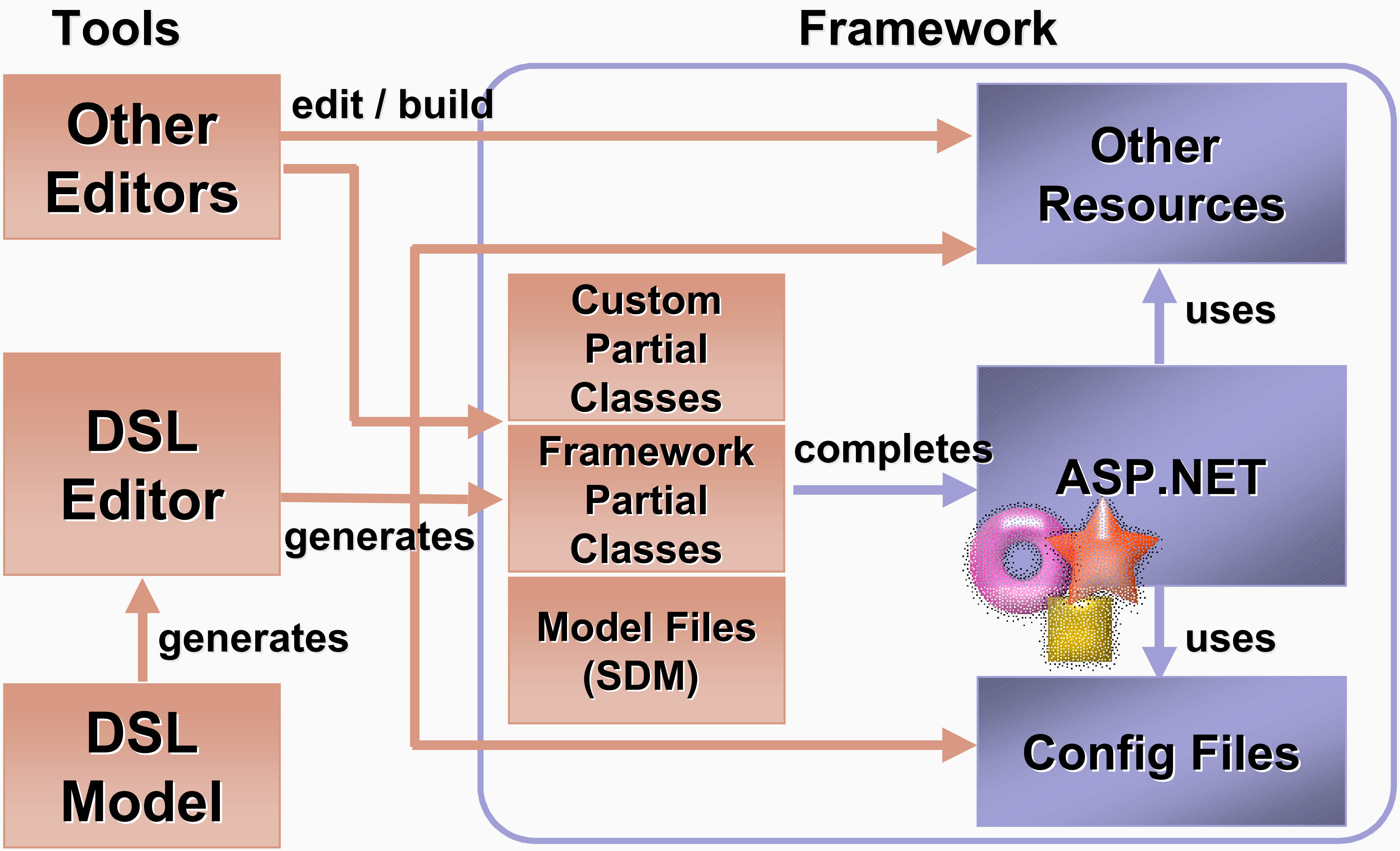


Generate To Platform

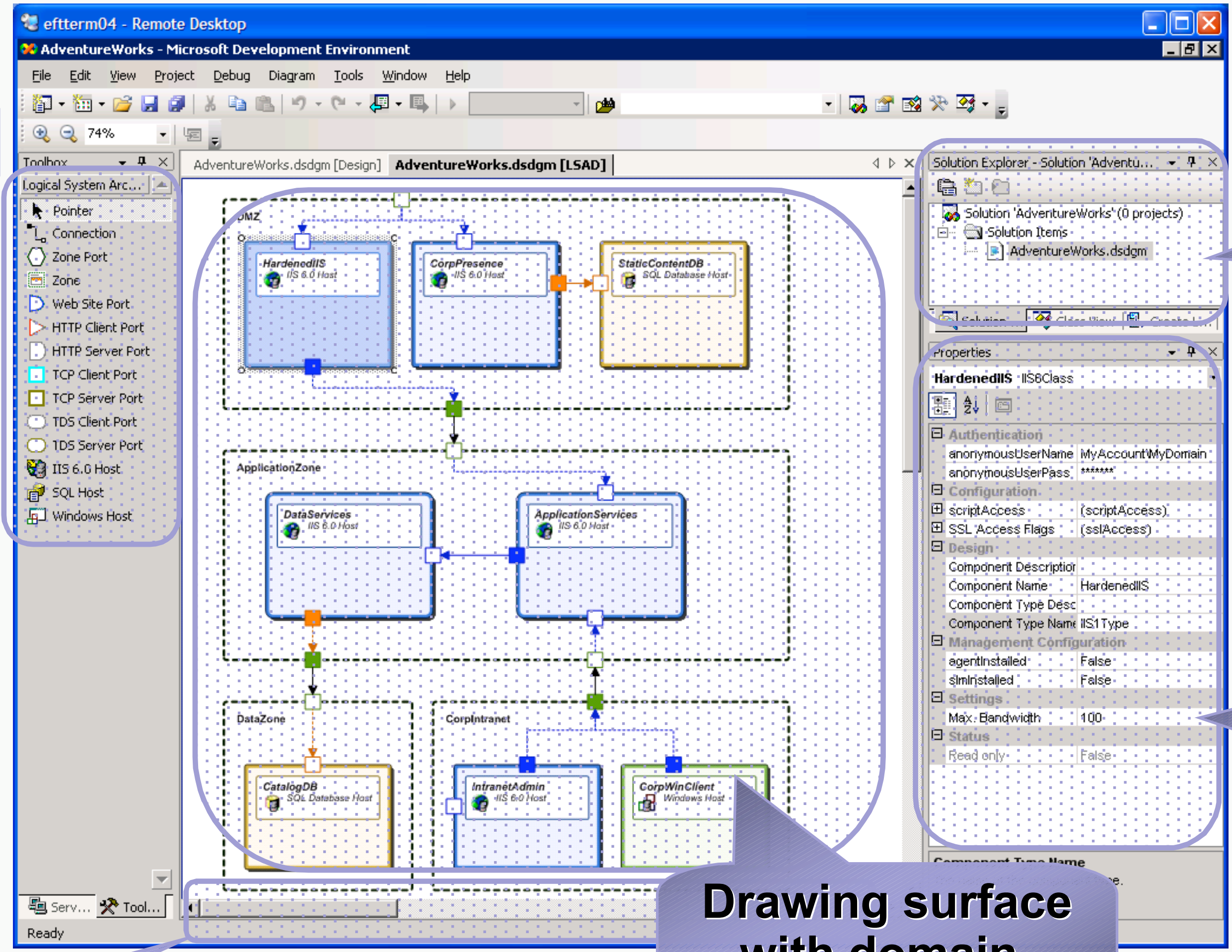


Interrelated Models

Models and Frameworks



Building a Designer for Visual Studio



HardenedIIS IIS6Class

Authentication	
anonymousUserName	MyAccount\MyDomain
anonymousUserPass	*******
Configuration	
scriptAccess	(scriptAccess)
SSL Access Flags	(sslAccess)
Design	
Component Descriptior	
Component Name	HardenedIIS
Component Type Desc	
Component Type Name	IIS1Type
Management Configuration	
agentInstalled	False
slmInstalled	False
Settings	
Max. Bandwidth	100
Status	
Read only	False

Toolbox

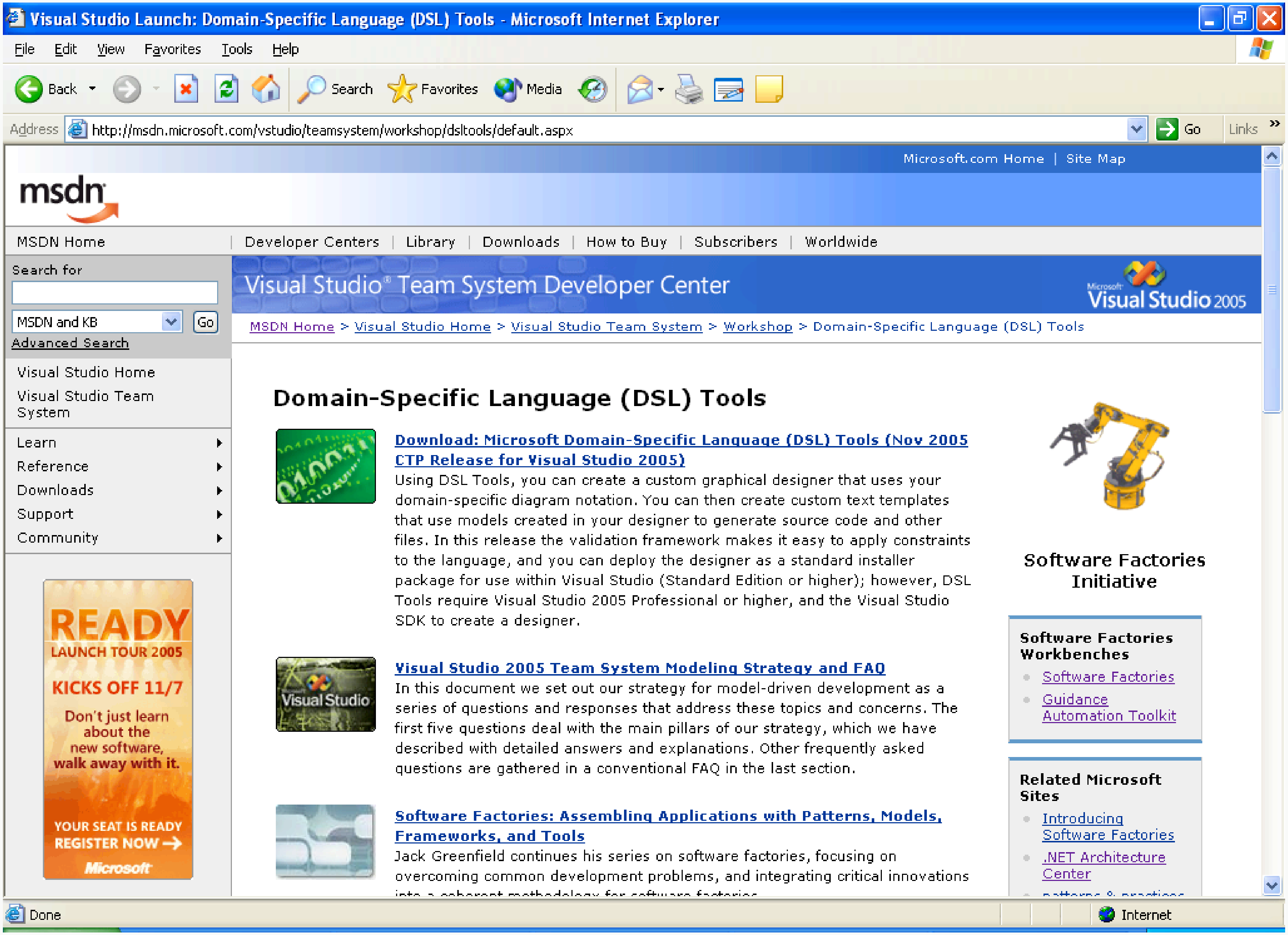
Model Explorer

Properties Window

Drawing surface with domain-specific notation

Validation

Portions copyright © 2003-2005 Microsoft Corporation. Used by permission. Portions copyright © 2003-2005 Jack Greenfield & Keith Short with Steve Cook & Stuart Kent. Used by permission of John Wiley & Sons, Inc. All rights reserved.


Visual Studio Launch: Domain-Specific Language (DSL) Tools - Microsoft Internet Explorer
File
Edit
View
Favorites
Tools
Help
Back
Search
Favorites
Media
Address
http://msdn.microsoft.com/vstudio/teamsystem/workshop/dsltools/default.aspx
Go
Links
Microsoft.com Home | Site Map
msdn
MSDN Home
Developer Centers
Library
Downloads
How to Buy
Subscribers
Worldwide
Search for
MSDN and KB
Go
Advanced Search
Visual Studio Home
Visual Studio Team System
Learn
Reference
Downloads
Support
Community
READY
LAUNCH TOUR 2005
KICKS OFF 11/7
Don't just learn about the new software, walk away with it.
YOUR SEAT IS READY
REGISTER NOW
Microsoft
Visual Studio® Team System Developer Center
Microsoft Visual Studio 2005
Visual Studio
Done
Internet

MSDN Home > Visual Studio Home > Visual Studio Team System > Workshop > Domain-Specific Language (DSL) Tools

Domain-Specific Language (DSL) Tools

Download: Microsoft Domain-Specific Language (DSL) Tools (Nov 2005 CTP Release for Visual Studio 2005)

Using DSL Tools, you can create a custom graphical designer that uses your domain-specific diagram notation. You can then create custom text templates that use models created in your designer to generate source code and other files. In this release the validation framework makes it easy to apply constraints to the language, and you can deploy the designer as a standard installer package for use within Visual Studio (Standard Edition or higher); however, DSL Tools require Visual Studio 2005 Professional or higher, and the Visual Studio SDK to create a designer.

Visual Studio 2005 Team System Modeling Strategy and FAQ

In this document we set out our strategy for model-driven development as a series of questions and responses that address these topics and concerns. The first five questions deal with the main pillars of our strategy, which we have described with detailed answers and explanations. Other frequently asked questions are gathered in a conventional FAQ in the last section.

Software Factories: Assembling Applications with Patterns, Models, Frameworks, and Tools

Jack Greenfield continues his series on software factories, focusing on overcoming common development problems, and integrating critical innovations into a coherent methodology for software factories

Software Factories Initiative

Software Factories Workbenches

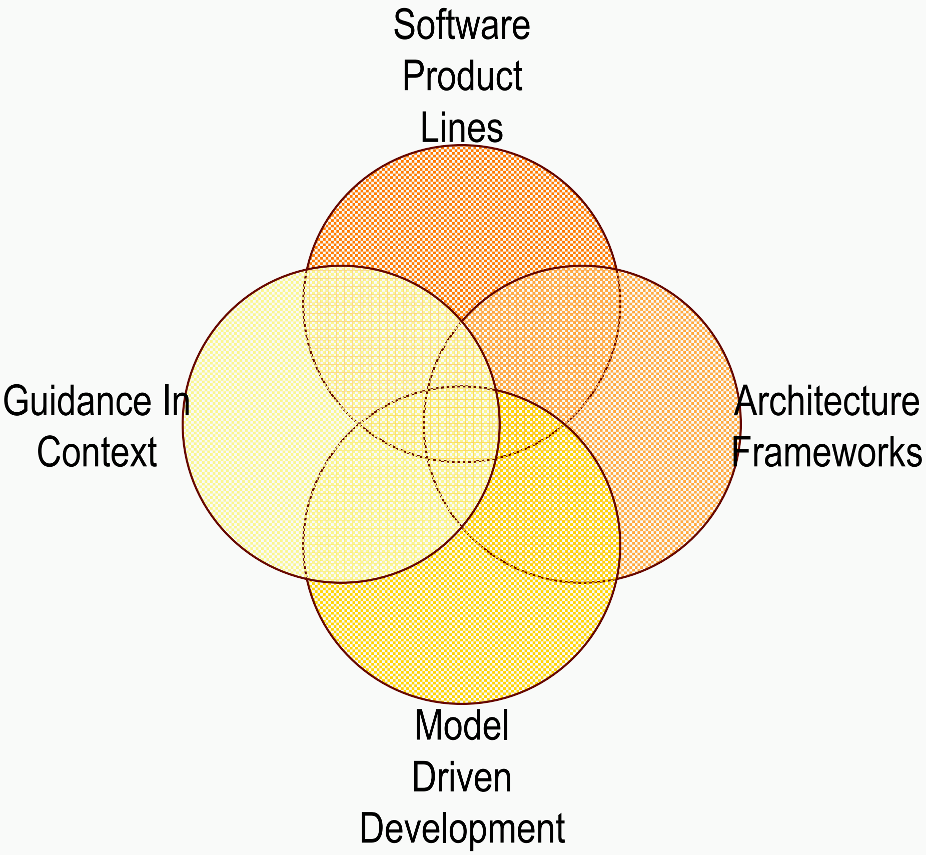
- Software Factories
- Guidance Automation Toolkit

Related Microsoft Sites

- Introducing Software Factories
- .NET Architecture Center
- patterns & practices

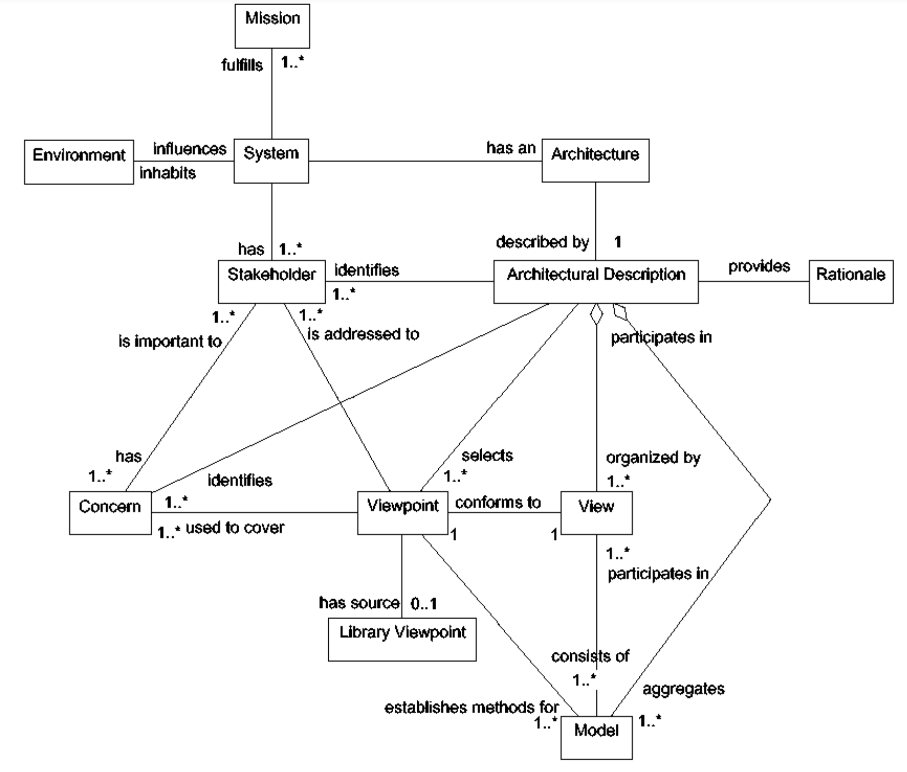
Architecture Frameworks

- Define viewpoints that identify and separate key stakeholder concerns
- Organize tools, process and content by viewpoint
- Relate and integrate life cycle phases, system components, and levels of abstraction



Software Architecture

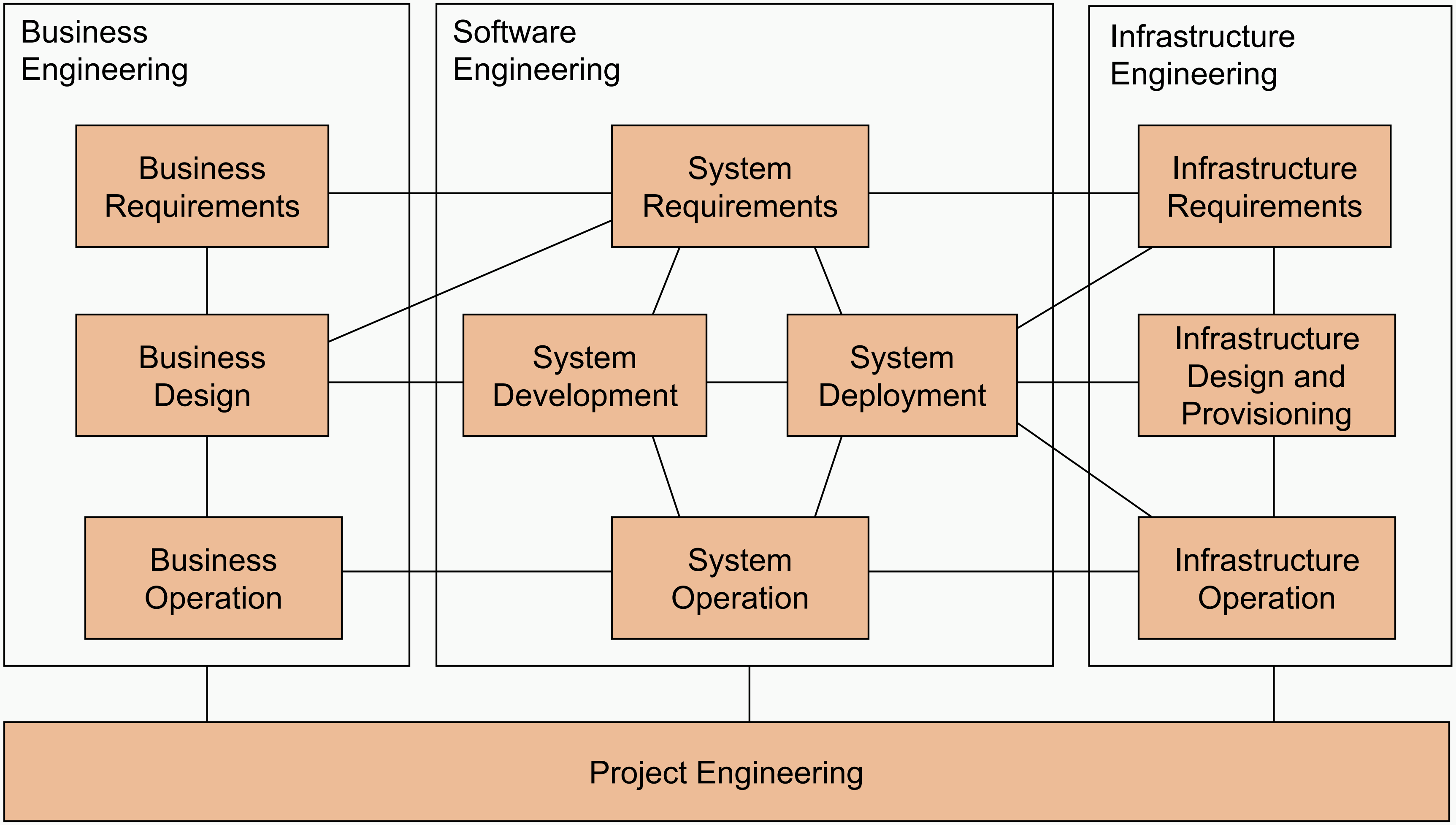
IEEE 1471 - Architecture Description Standards



Interface design and functional factoring constitute the key intellectual content of software and are far more difficult to create or recreate than code.

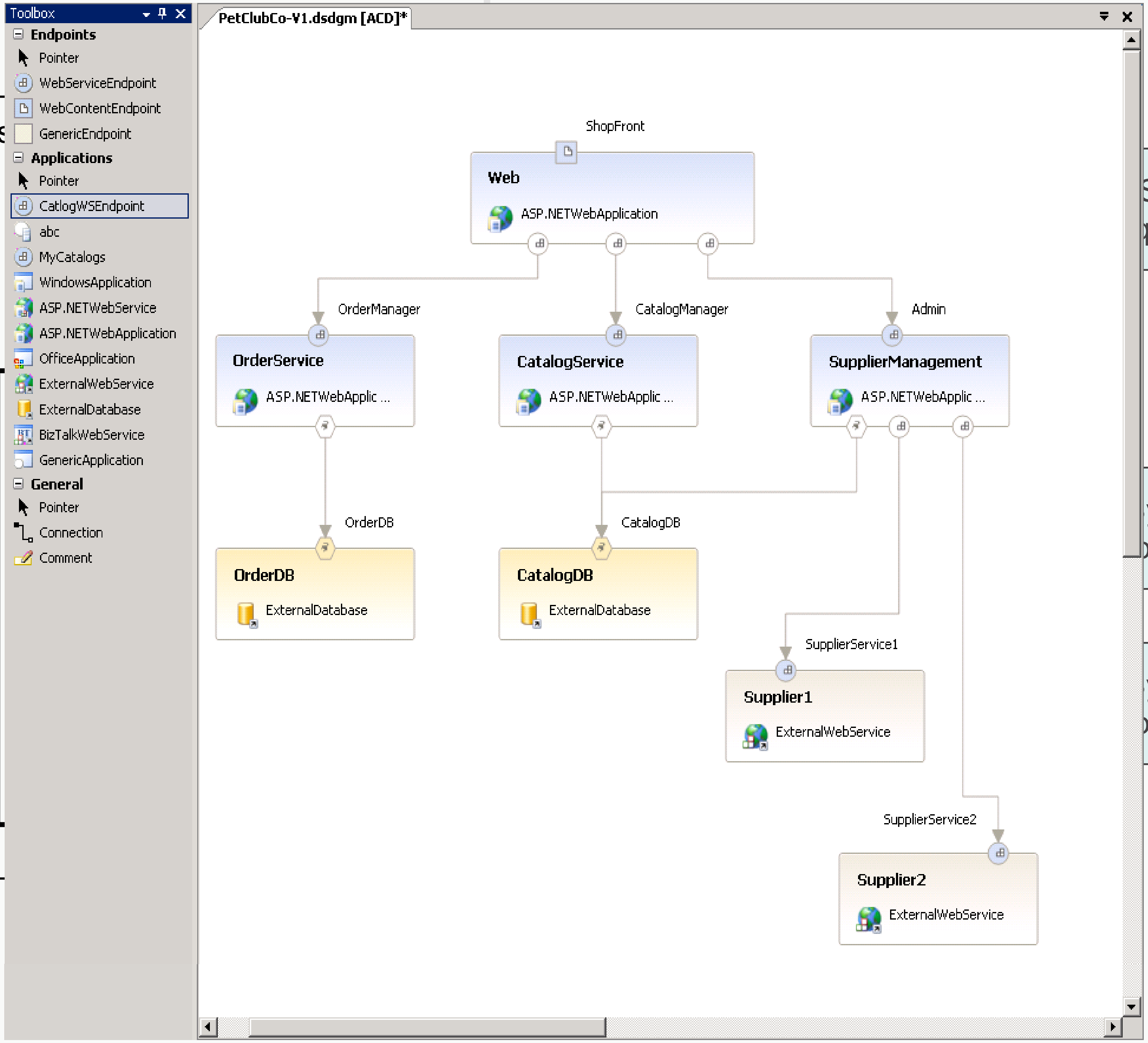
Peter Deutsch

A Factory Schema



Application Development

Sys

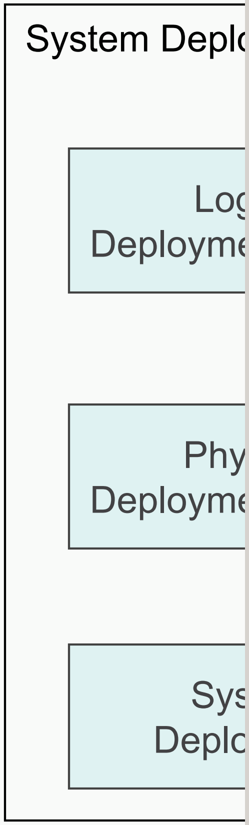


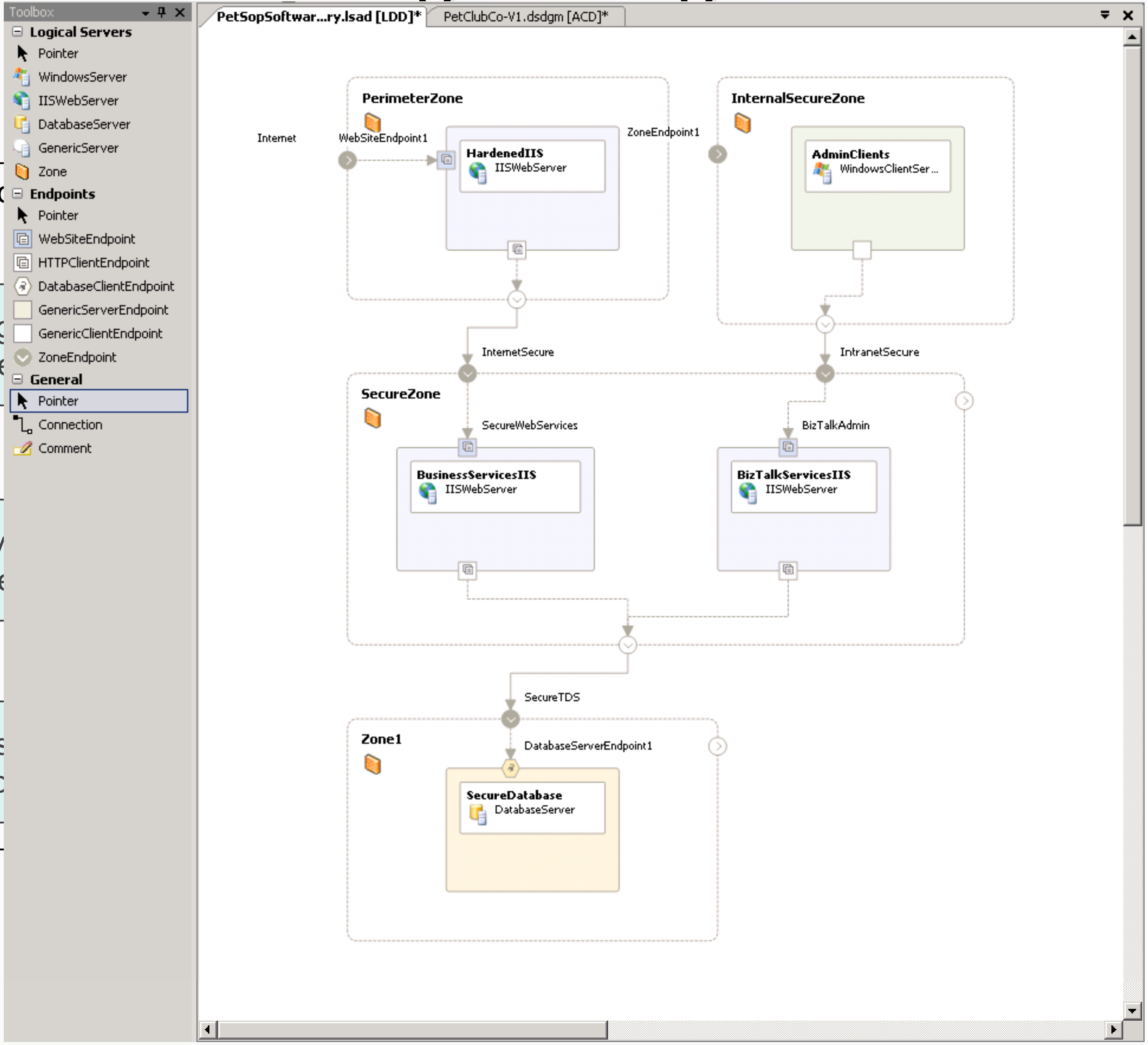
System
quirements

ystem
ployment

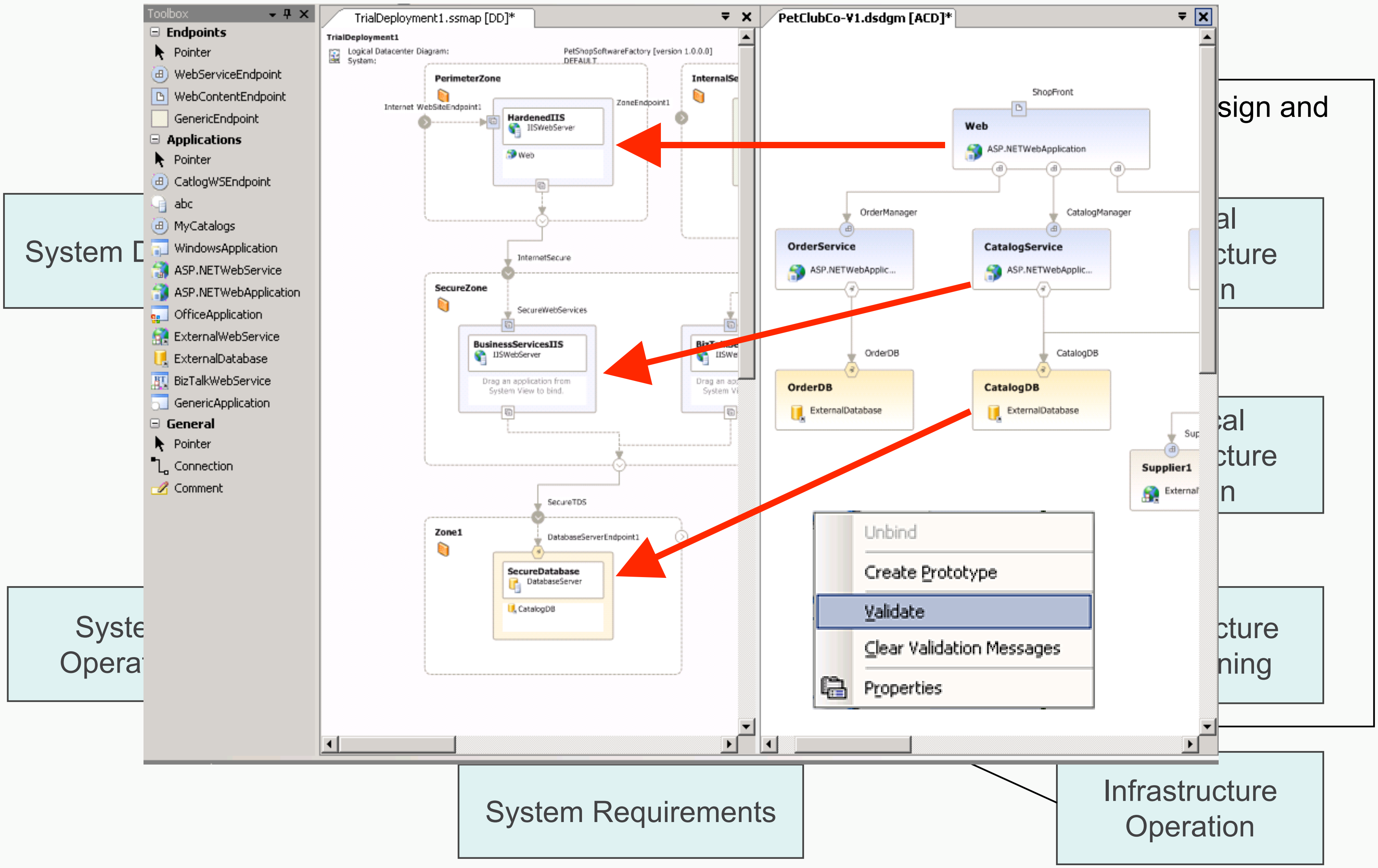
ystem
eration

Infrastructure Engineering





System Deployment



System D

Syste
Opera

sign and

al
cture
n

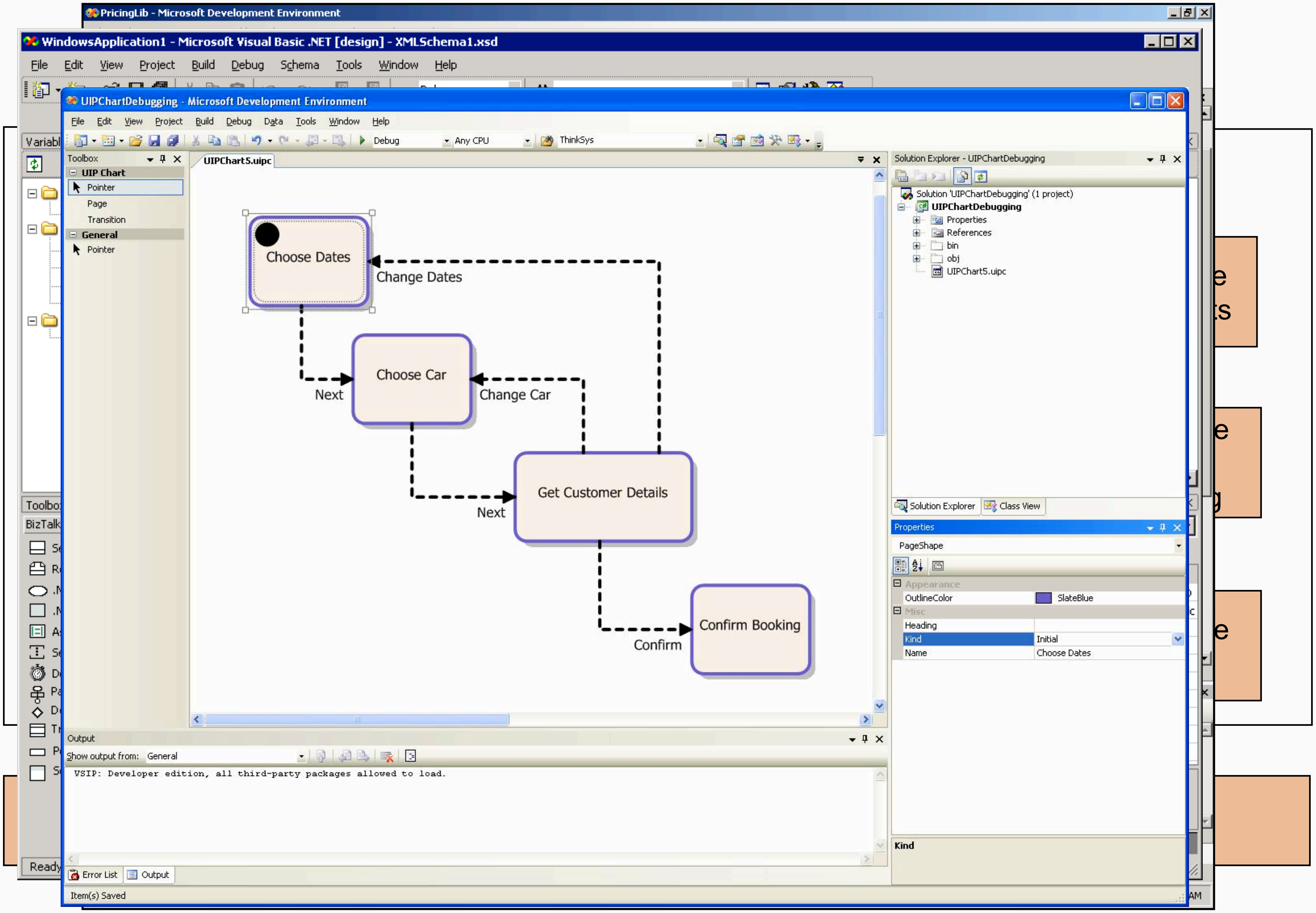
cal
cture
n

cture
ning

System Requirements

Infrastructure
Operation

A Factory Schema



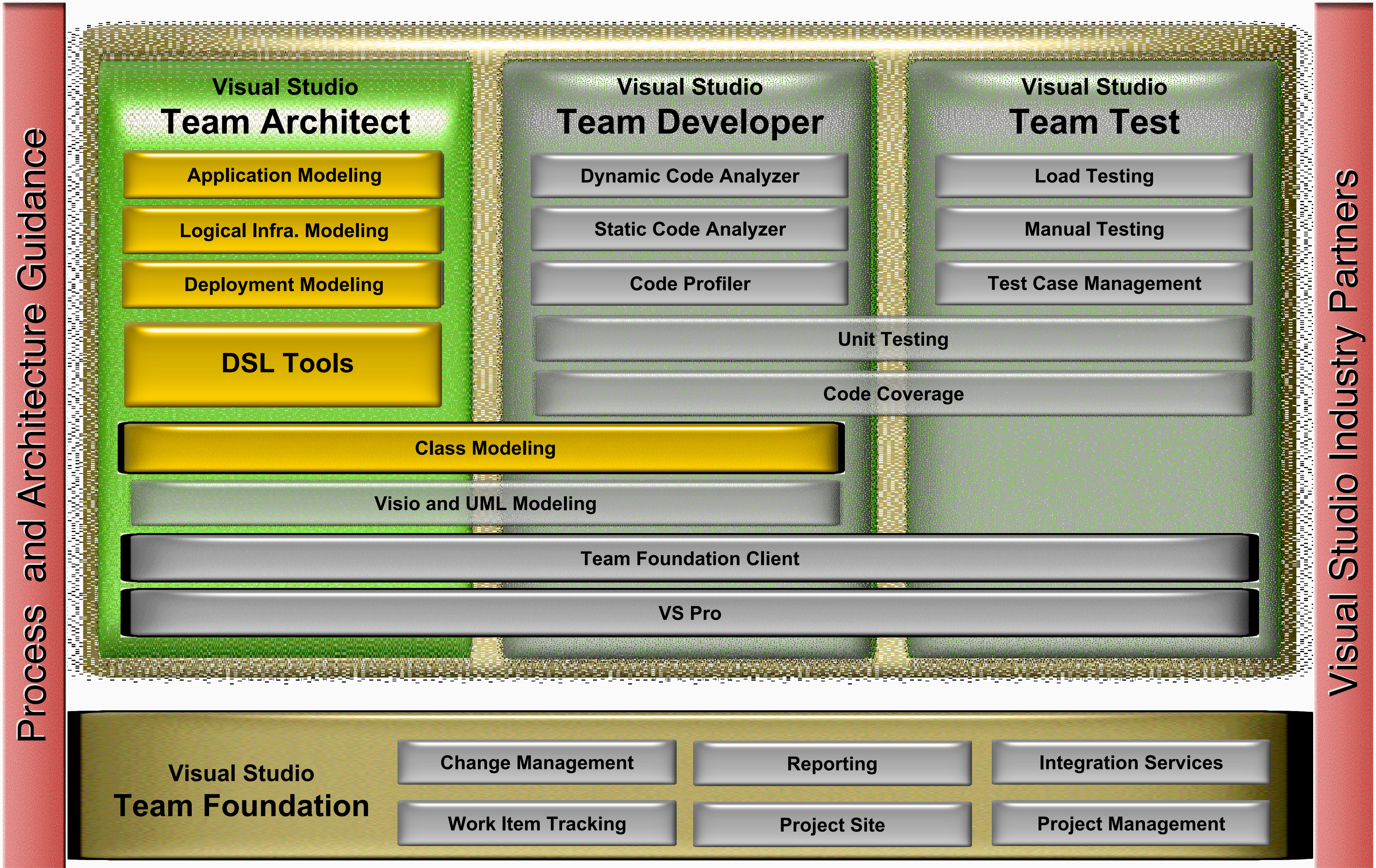
PricingLib - Microsoft Development Environment
WindowsApplication1 - Microsoft Visual Basic .NET [design] - XMLSchema1.xsd
File Edit View Project Build Debug Schema Tools Window Help
UIPChartDebugging - Microsoft Development Environment
File Edit View Project Build Debug Data Tools Window Help
Debug
Any CPU
ThinkSys
Toolbox
UIP Chart
Pointer
Page
Transition
General
Pointer
UIPChart5.uipc
Choose Dates
Change Dates
Next
Choose Car
Change Car
Get Customer Details
Next
Confirm Booking
Confirm
Solution Explorer - UIPChartDebugging
Solution 'UIPChartDebugging' (1 project)
UIPChartDebugging
Properties
References
bin
obj
UIPChart5.uipc
Solution Explorer
Class View
Properties
PageShape
Appearance
Misc
Output
Show output from: General
Error List
Output
Item(s) Saved
Ready
Toolbox
BizTalk

Properties	
OutlineColor	SlateBlue
Heading	
Kind	Initial
Name	Choose Dates

Kind

```
VSIP: Developer edition, all third-party packages allowed to load.
```

Visual Studio Team System



Process and Architecture Guidance

Visual Studio Team Architect

- Application Modeling
- Logical Infra. Modeling
- Deployment Modeling
- DSL Tools

Visual Studio Team Developer

- Dynamic Code Analyzer
- Static Code Analyzer
- Code Profiler

Visual Studio Team Test

- Load Testing
- Manual Testing
- Test Case Management

Unit Testing

Code Coverage

Class Modeling

Visio and UML Modeling

Team Foundation Client

VS Pro

Visual Studio Team Foundation

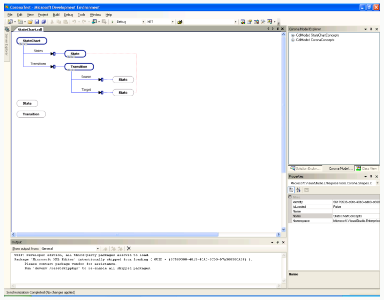
- Change Management
- Reporting
- Integration Services
- Work Item Tracking
- Project Site
- Project Management

Visual Studio Industry Partners

DSL Tools

Modeling Languag (tool) designer

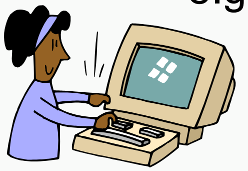
feeds in language / tool design



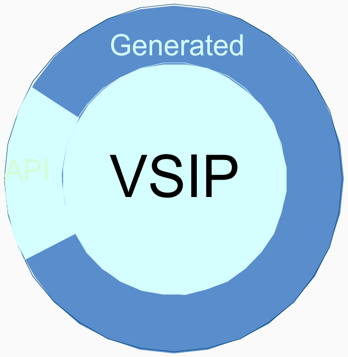
Developer

adds detail to design

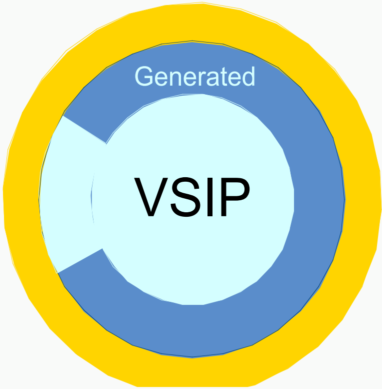
e.g. extra metadata to drive code generation



generate code



adds code (optional)



Software / systems designer

8/25/2005

Portions copyright © 2003-2005 Microsoft Corporation. Used by permission. Portions copyright © 2003-2005 Jack Greenfield & Keith Short with Steve Cook & Stuart Kent. Used by permission of John Wiley & Sons, Inc. All rights reserved.

What Is A Software Factory?

“A set of integrated tools, process and content assets used to accelerate life cycle tasks for a specific type of software component, application or system”

- Software Factories exploit innovations in three main areas
  - The use of models throughout the life cycle
  - A focus on component, application or system families (software product lines)
  - An emphasis on architecture-driven design
- Software Factories are described in an XML document called a software factory schema that defines one or more viewpoints and their interrelationships
  - Viewpoints include domain specific patterns, tools, process, and other assets

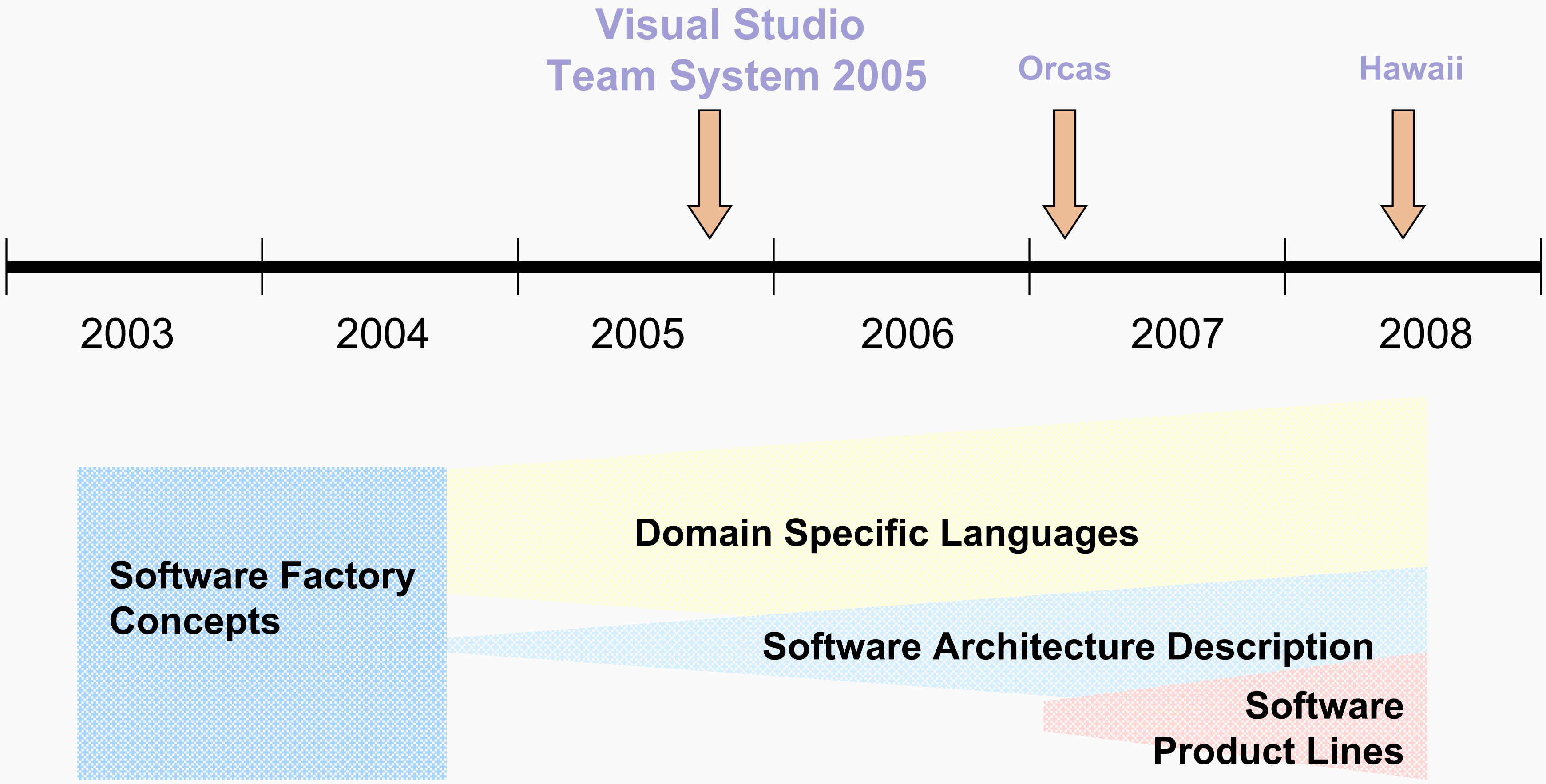
What’s In A Factory?

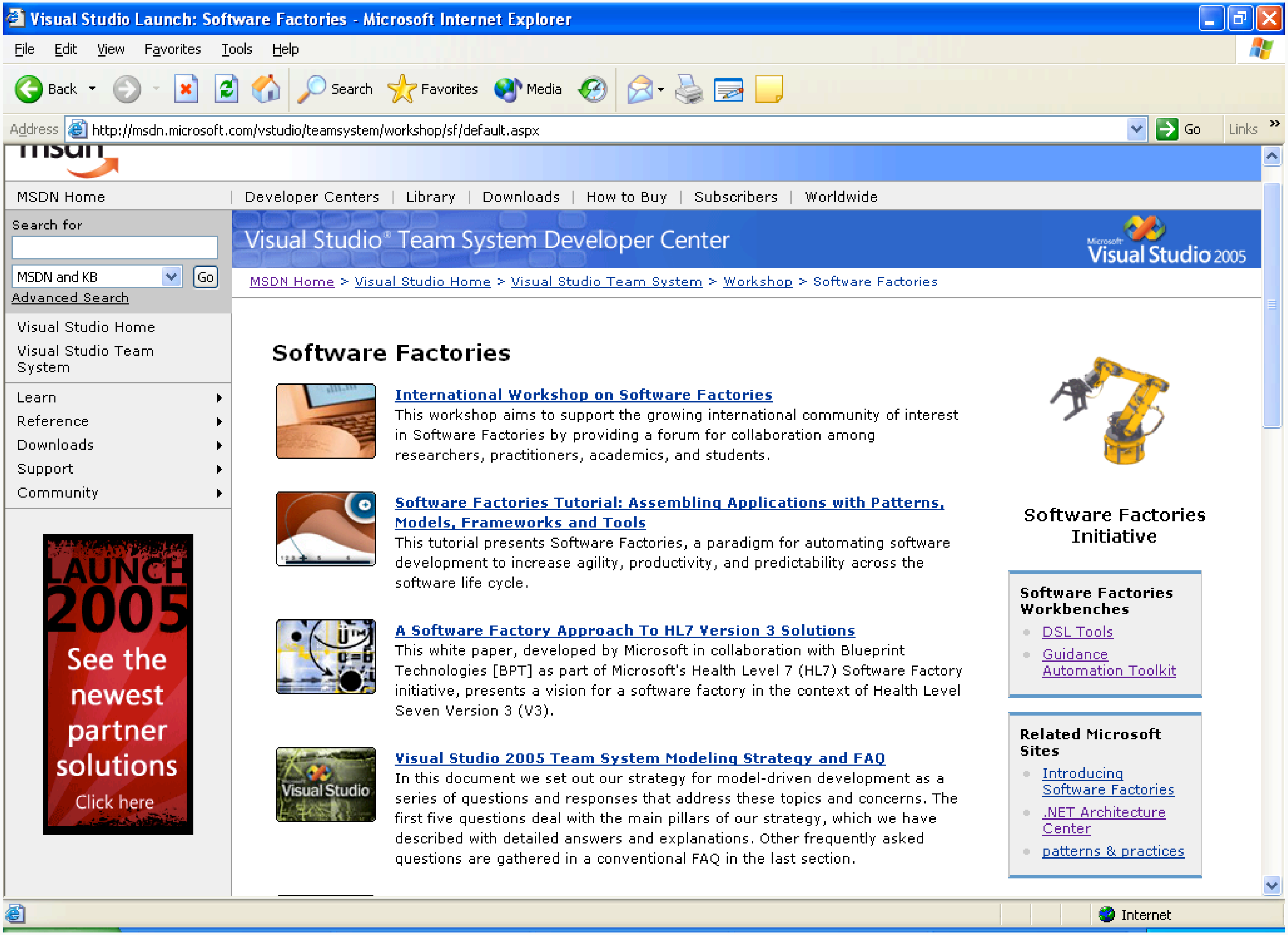
- A structured installable collection of customizable, integrated tool, process and content assets (software factory template)
  - Guidelines, patterns, code samples, snippets, templates, wizards, class libraries, frameworks, designers, models, configuration files
  - Developed as a set of Visual Studio solutions, delivered as a set of MSIs, installed on project team servers and/or team member workstations
- A description of the software factory (software factory schema)
  - Expressed as a model interpreted by users and tools
  - Describes the assets, how they are packaged into the factory, how they should be analyzed, customized and installed, and how they should be used

Software Factories in Visual Studio

- Visual Studio Team Architect 2005
  - Application Designer
  - System Designer
  - Deployment Designer
  - Logical Datacenter Designer
- All Visual Studio 2005 SKUs
  - Class Designer
- BizTalk 2004
  - Orchestration Designer
- Designer Framework SDK
  - Recipe Designer
  - DSL Tool Definition
- Visual Studio 2005 Team Foundation Server
  - Methodology templates, dynamic help
  - VSTF Database extensibility and linking
- Longer Term
  - Software Factory examples
  - Software Factory Factory
  - More DSLs and tools

Software Factory Roadmap




Visual Studio Launch: Software Factories - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Media
Address http://msdn.microsoft.com/vstudio/teamsystem/workshop/sf/default.aspx
Go
Links
msdn
MSDN Home | Developer Centers | Library | Downloads | How to Buy | Subscribers | Worldwide
Search for
MSDN and KB
Go
Advanced Search
Visual Studio Home
Visual Studio Team System
Learn
Reference
Downloads
Support
Community
LAUNCH 2005
See the newest partner solutions
Click here
Visual Studio® Team System Developer Center
Microsoft Visual Studio 2005
MSDN Home > Visual Studio Home > Visual Studio Team System > Workshop > Software Factories
Visual Studio
Internet

Software Factories

International Workshop on Software Factories

This workshop aims to support the growing international community of interest in Software Factories by providing a forum for collaboration among researchers, practitioners, academics, and students.

Software Factories Tutorial: Assembling Applications with Patterns, Models, Frameworks and Tools

This tutorial presents Software Factories, a paradigm for automating software development to increase agility, productivity, and predictability across the software life cycle.

A Software Factory Approach To HL7 Version 3 Solutions

This white paper, developed by Microsoft in collaboration with Blueprint Technologies [BPT] as part of Microsoft's Health Level 7 (HL7) Software Factory initiative, presents a vision for a software factory in the context of Health Level Seven Version 3 (V3).

Visual Studio 2005 Team System Modeling Strategy and FAQ

In this document we set out our strategy for model-driven development as a series of questions and responses that address these topics and concerns. The first five questions deal with the main pillars of our strategy, which we have described with detailed answers and explanations. Other frequently asked questions are gathered in a conventional FAQ in the last section.

Software Factories Initiative

Software Factories Workbenches

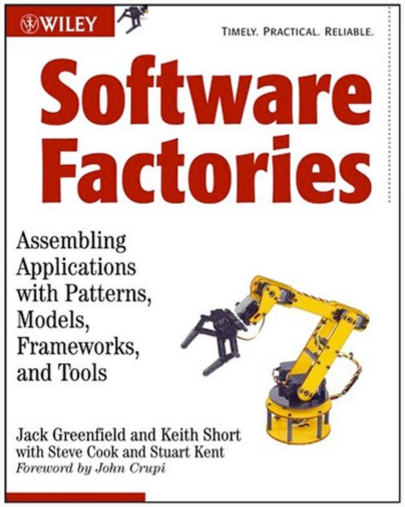
- DSL Tools
- Guidance Automation Toolkit

Related Microsoft Sites

- Introducing Software Factories
- .NET Architecture Center
- patterns & practices

Resources

- Books
  - Software Factories by Jack Greenfield and Keith Short
- Websites
  - http://lab.msdn.microsoft.com/teamsystem/workshop/sf/default.aspx
- Newsgroups
  - Microsoft.private.whidbey.teamsystem.architect
  - Microsoft.private.whidbey.teamsystem.architect.modeling
- Email
  - jackgr@microsoft.com
  - keithsh@microsoft.com
  - stcook@microsoft.com
- Blogs
  - http://blogs.msdn.com/jackgr/
  - http://blogs.msdn.com/keith_short/
  - http://blogs.msdn.com/stevecook/


WILEY
TIMELY. PRACTICAL. RELIABLE.
Software Factories
Assembling Applications with Patterns, Models, Frameworks, and Tools
Jack Greenfield and Keith Short
with Steve Cook and Stuart Kent
Foreword by John Crupi

Microsoft®

Your potential. Our passion.™

This presentation is for informational purposes only. Microsoft makes no warranties, express or implied, in this summary.