CONSTRUX SEMINARS GET YOU TO ALL THE RIGHT DESTINATIONS.

2011

MANAGEMENT
AGILE DEVELOPMENT
REQUIREMENTS
DESIGN & CONSTRUCTION
QA & TEST
METHODS & PROCESSES

Construx®
SOFTWARE DEVELOPMENT BEST PRACTICES

Let's go!

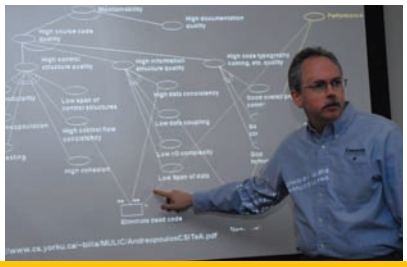
TWO GREAT WAYS TO IMPROVE QUALITY, PREDICTABILITY AND TIME-TO-MARKET.

Private, On-site Seminars

at your facility

If tight schedules and budgets are making it a challenge to find time for training, you can take your team to new heights with a Construx onsite seminar — and do it on your schedule:

- **Eliminate the travel time and expense**
- **Candid, confidential discussions**
- **Focus on your unique issues**
- **A great shared experience for your team**



Public Seminars

at our facilities in Bellevue, WA

Seminars at the Construx Training Center are a great way to have a concentrated learning experience in a stimulating environment, with peers from other organizations who face similar and diverse challenges:

- **Held at our offices in Bellevue, Washington, near Seattle**
- **Taught by experienced Construx consultants with deep knowledge of current best practices**
- **Other Construx experts, including founder, Steve McConnell, are often available to address your specialized issues**

TRY BEFORE YOU BUY.

Attend a Construx public seminar FREE!

If you're considering hosting a Construx private, on-site seminar for 15 people or more, we invite you to have a representative attend the public seminar of your choice as our guest at our offices in Bellevue, Washington — a value of up to $1995.

We're confident that once you've experienced the quality and effectiveness of Construx training, you'll appreciate the positive impact an onsite seminar could have for your organization. We can help you decide which seminar would be the most valuable for your team. Contact us at **training@construx.com** or call **866-296-6300**.

PUBLIC SEMINARS AT CONSTRUX
January–June 2011

Book early and save! Register for any Construx seminar up to 30 days before the starting dates indicated here and *save up to $300*. For more information about seminars go to: **www.construx.com/calendar**.

JANUARY

- **Software Estimation in Depth**
  January 10-11 – $1,395 *(page 5)*
- **Software Project Management Boot Camp**
  January 12-14 – $1,995 *(page 4)*
- **Requirements Boot Camp**
  January 17-19 – $1,995 *(page 13)*
- **Agile Requirements in Depth** **NEW!**
  January 20-21 – $1,395 *(page 14)*
- **Peer Mentoring: Effective Knowledge Transfer** | January 20 – $695 *(page 20)*
- **Configuration Management Essentials**
  January 21 – $695 *(page 19)*
- **10x Software Engineering**
  January 24-26 – $1,995 *(page 23)*
- **Scrum Boot Camp**
  January 31-February 2 – $1,995 *(page 8)*

FEBRUARY

- **Developer Testing Boot Camp**
  February 3-4 – $1,395 *(page 18)*
- **Professional Tester Boot Camp**
  February 7-8 – $1,395 *(page 21)*
- **Advanced Agile: Beyond Basic Scrum** **NEW!**
  February 7-9 – $1,995 *(page 9)*
- **Design Boot Camp**
  February 9-11 – $1,995 *(page 16)*
- **Scrum Product Owner Boot Camp** **NEW!**
  February 14-16 – $1,995 *(page 12)*
- **Developer Boot Camp**
  February 14-16 – $1,995 *(page 17)*
- **How to Be Agile Without Being Extreme**
  February 28-March 1 – $1,395 *(page 12)*
- **Object-Oriented Requirements Analysis and Design Using the UML**
  February 28-March 2 – $1,995 *(page 15)*

MARCH

- **Risk Management in Depth**
  March 3-4 – $1,395 *(page 7)*
- **Advanced Quality Boot Camp**
  March 7-9 – $1,995 *(page 22)*
- **Use Cases in Depth**
  March 10-11 – $1,395 *(page 14)*
- **Code Complete Essentials**
  March 14 – $695 *(page 19)*

APRIL

- **Software Estimation in Depth**
  April 11-12 – $1,395 *(page 5)*
- **Requirements Modeling** **NEW!**
  April 11-12 – $1,395 *(page 15)*
- **Scrum for the Enterprise** **NEW!**
  April 13-15 – $1,995 *(page 10)*
- **10x Software Engineering**
  April 13-15 – $1,995 *(page 23)*
- **Software Project Management Boot Camp**
  April 18-20 – $1,995 *(page 4)*
- **Requirements Boot Camp**
  April 26-28 – $1,995 *(page 13)*

MAY

- **Scrum Product Owner Boot Camp** **NEW!**
  May 2-4 – $1,995 *(page 12)*
- **Configuration Management Essentials**
  May 2 – $695 *(page 19)*
- **Advanced Quality Boot Camp: Beyond Testing**
  May 4-6 – $1,995 *(page 22)*
- **Peer Mentoring: Effective Knowledge Transfer** | May 9 – $695 *(page 20)*
- **Advanced Agile: Beyond Basic Scrum**
  May 16-18 – $1,995 *(page 9)*
- **Developer Boot Camp**
  May 18-20 – $1,995 *(page 17)*

JUNE

- **Professional Tester Boot Camp**
  June 6-7 – $1,395 *(page 21)*
- **Requirements Modeling** **NEW!**
  June 6-7 – $1,395 *(page 15)*
- **Developer Testing Boot Camp**
  June 9-10 – $1,395 *(page 18)*
- **Design Boot Camp**
  June 13-15 – $1,995 *(page 16)*
- **Scrum Boot Camp**
  June 27-29 – $1,995 *(page 8)*
- **Agile Requirements in Depth** **NEW!**
  June 30-July 1 – $1,395 *(page 14)*

Software Project Management Boot Camp [18 PDUs]

Leading any project can be a challenge. Leading a software project can be even more challenging if you're new to project management or new to software. This seminar will help you make the transition to solid software project leadership. Software Project Management Boot Camp teaches you the concepts and techniques necessary to manage projects successfully. This seminar closely follows the Project Management Institute's (PMI) Project Management Body of Knowledge (PM-BOK) and shows how to apply these best practices to a typical small to medium sized software project. This course involves extensive hands-on practice with real-world case studies. Three days.

Introduction

- Defining software project success
- Understanding the challenges on a software project
- Typical software project outcomes
- Construx's path to software project success

Basic Survival Concepts

- Understanding labor rate, burn rate, capital vs. expense
- Taking advantage of the upstream/downstream effect
- Recognizing the intellectual phases of a software project
- Fundamentals of software project estimation

Project Initiation

- Chartering the project
- Assessing risks to software project success
- Recognizing software project assets

Project Close Out

- Typical close out tasks
- Using a project retrospective to learn from the experience

Project Planning

- Using a software project plan template
- Developing work breakdown structures (WBS)
- Simplifying the WBS with a project matrix
- Building the WBS dictionary
- Choosing a project organization
- Choosing a project lifecycle
- Typical software project effort allocations
- Creating an activity network (PERT chart)
- Finding the critical path
- Developing a realistic project schedule
- Scheduling to fixed end dates
- Addressing uncertainty using rolling wave planning
- Tuning the plan to the specifics of your project

Execution, Checking, and Correcting: Succeeding in Stages

- Controlling change
- Using earned value to objectively track project status
- Conducting effective status meetings
- Creating useful project status reports
- Refining the project plan based on actual progress
- Capturing valuable project history in a project log
- Sanity checking the project using planning checkpoint reviews

INSTRUCTOR SPOTLIGHT

Jerry Deville has nearly three decades of experience in business management, engineering process and management, requirements analysis, and testing. As a Senior Fellow at Construx Software, he creates and leads seminars and provides consulting services to help organizations optimize business management, product management, engineering management, engineering processes, and requirements. Jerry is also certified as a Project Management Professional by the Project Management Institute and as a Scrum Master from the Scrum Alliance.

Software Estimation in Depth [12 PDUs]

This course provides many useful rules of thumb and procedures for creating software estimates ("the art of estimation") and briefly introduces mathematical approaches to creating software project estimates ("the science of estimation"). This course features extensive lab work to give you hands-on experience creating many different kinds of software estimates—for large, medium, and small projects—as well as calibrating estimates to be accurate for your specific development environment. Two days.

Estimation Background

- Estimation "art" vs. estimation "science"
- Estimates, targets, and commitments
- Kinds of estimates: macro vs. micro, top-down vs. bottom up, algorithmic vs. heuristic
- State of the art and limits on estimation accuracy
- Surprise: Estimation's real role on software projects

Estimation Process

- Basic steps in creating a software estimate
- Estimating agile projects vs. estimating plan-driven projects
- Best estimation approaches by project phase
- Estimate refinement
- Standardized estimating procedures for agile and plan-driven projects

Estimation Error

- The good, the bad, and the ugly: evaluating estimates
- Errors in the estimation process
- Sources of project uncertainty
- Software's Cone of Uncertainty

Popular Estimation Methods

- Off-the-cuff estimation
- Using expert judgment successfully
- Wide-band Delphi
- Cocomo II

Better Estimation Methods

- Estimation by analogy
- Decomposition
- Proxy-based estimation
- Estimation by function points
- The PERT formula
- Putnam's Method

Special Issues in Scope Estimation

- Counting, Computing, and Judgment
- Fuzzy Logic
- T-Shirt Sizing
- Software's diseconomy of scale

Special Issues in Effort Estimation

- Productivity variations across types of software
- Calibration
- Industry data
- Historical company data
- Project data
- Estimating individuals' work

Special Issues in Schedule Estimation

- The schedule equation
- Effect of schedule compression and expansion
- The "impossible zone"

Automated Estimation Support

- Product demos
- Tool capabilities
- Interplay of the art and science of software estimation

Human Roles in Estimation

- Estimate presentation techniques
- How to explain and defend an estimate
- Estimation and negotiation

INSTRUCTOR SPOTLIGHT

Steve McConnell is CEO and Chief Software Engineer at Construx Software where he authors books and articles, blogs, consults, leads seminars, and oversees Construx's software development practices. Steve is the author of several bestsellers on software best practices including *Code Complete* and *Software Estimation: Demystifying the Black Art*. He also serves as Editor in Chief Emeritus of IEEE Software magazine. Steve's areas of expertise include project estimation, software construction practices, Agile and rapid development methodologies, and outsource software management.

Software Estimation Master Class

This course is an expanded version of Construx's popular *Software Estimation in Depth* seminar. This course dives deeper into mathematically intensive methods than that seminar and explores more variations in estimation approaches between Agile and plan-driven projects. This seminar features extensive lab work to give you hands-on experience creating many different kinds of software estimates—for agile projects and traditional projects and for large, medium, and small projects. *See complete course description at www.construx com.*

Project Outsourcing Essentials [6 PDUs]

Software is increasingly being outsourced. In this seminar, project managers will learn how management of outsourced projects differs from management of in-house projects. They will see the numerous ways in which outsourced projects can go wrong and learn strategies that will steer them toward success. Lectures, case studies, and lab work build on professional project management practices to cover the key considerations project managers must deal with on outsourced software projects. *See complete course description at www.construx com.*

Managing Projects Using Earned Value [6 PDUs]

The Earned Value project management technique has been in use for more than 50 years. It has been shown on thousands of projects to be the most objective indicator of real status. This one-day, hands-on seminar teaches you everything you need to know to apply earned value in order to successfully manage software projects. This seminar is specifically tailored to applying earned value to software projects. While the basic theory and calculations are identical to generic earned value, the case study used in the seminar is a software project.

See complete course description at www.construx com.

Software Project Survival Boot Camp [12 PDUs]

Many people in the software industry are thrust into positions of responsibility for software project outcomes, but few are given formal or informal training in how to make that happen. In this two-day course, the workshop leader describes specific steps that small and medium-sized development teams should take to succeed on projects from three to 12 months long. This Software Survival Process is based both on the instructor's direct experiences on software projects and on experience gained from Construx Software's project assessment and project recovery consulting business.

See complete course description at www.construx com.

"Fantastic in-depth review of software estimation techniques."

Arjun Kapur, Intel

Risk Management in Depth [12 PDUs]

The project was a guaranteed success—until the subcontractor announced a 3 week delay and your chief architect quit to go hiking in Nepal. If you don't attack project risks, they will attack you! Learn intermediate and advanced strategies you can use on both general and project-specific risks. Discover how to identify, address, and eliminate sources of risk before they grow into major problems. This two-day course focuses on intermediate and advanced strategies you can use to manage general risks and details practical techniques you can use to control your project's specific risks.

Introduction
- A definition of risk
- What is risk management?
- The need for risk management

Risks in Detail
- The scope of risks
- Risk as cause-effect
- Ultimate causes and ultimate effects
- Risk timeframes
- Assets

Risk Identification
- Categories of risks
- Common project risks
- Practical techniques to identify risks and assets

Risk Analysis/Prioritization
- Risk probabilities
- Risk severities
- Techniques for accurately estimating risk probabilities and severities
- Determining risk exposure
- Prioritizing risks
- Analyzing/prioritizing assets

Risk Response Planning
- Risk response strategies
- Prevent
- Mitigate
- Transfer/share
- Contingency plan
- Risk reserve/provision ignore
- Planning risk response

Risk Responses
- Responses to requirements problems
- Responses to inadequate project management
- Responses to inattention to upstream quality
- Responses to misunderstood project dynamics
- Strategies for maximizing common assets

Risk Response Control
- Response control through project tracking
- Ongoing risk reassessment

INSTRUCTOR SPOTLIGHT

Earl Beede is a Senior Fellow at Construx Software, where he designs and leads seminars and provides consulting services on early project-lifecycle practices, estimation, requirements, quality assurance, contract management, and software methodologies. With more than 20 years experience in quality assurance, systems analysis, process architecture, and management, Earl has designed and written software development processes for a wide variety of industries. He is a member of the IEEE Computer Society and a coordinator of the Seattle Area Software Process Improvement Network.

Scrum Boot Camp [18 PDUs]

This seminar provides everything you need to know to ensure your transition to Scrum is successful. In the years since the Agile Manifesto, Scrum has emerged as the most popular Agile process for managing software development projects. More companies are switching to Scrum, but many are struggling. This three-day course combines key aspects from Certified Scrum Master- and Certified Scrum Product Owner-specific training plus specific best practices based upon the instructor's direct experience on Scrum transitions and experience gained from Construx's consulting engagements.

What is Scrum?

- Agile origins, principles, and benefits
- A brief history of Scrum
- Scrum philosophy and theory
- What makes Scrum different?

Why Scrum Works

- A simple process
- A committed, self-managed team
- Transparency: nowhere to hide
- Finishing what you start
- Continual improvement

Scrum Roles

- The Scrum Master
- The Product Owner
- The Team Member
- Stakeholders
- Levels of Commitment

Scrum Processes and Meetings

- Release Planning
- Sprint Planning
- The Daily Standup
- The Sprint Review
- The Sprint Retrospective

Scrum Artifacts

- The Product Backlog
- The Sprint Backlog
- The Release Burndown
- The Sprint Burndown

Scrum Best Practices

- Timeboxing: Nothing Concentrates The Mind...
- Commitment: "Either Do or Do Not; There Is No Try"
- Working Agreements: This Is How We Do It
- Acceptance Criteria And The Definition of Done
- There Is No 'I' in Team

Spinning Up Scrum

- Selecting the Product Owner
- Creating a product backlog
- Setting up the Scrum team
- Planning the release
- Launching the first sprint

Life During Sprint Time

- A day in the life of a Scrum team
- A day in the life of a Scrum Master
- A day in the life of a Product Owner

Tracking Progress: Scrum Metrics

- Whiteboards or software?
- How is your iteration going?
- Exposing and removing impediments
- When are you going to release?
- Are you improving?

Something's Rotten: Scrum Smells

- Scrum (Task) Masters
- The Product Dictator
- The Tyranny of the Urgent
- They're Just Not That Into It: When the Team Fails to Meet Commitments
- Self-Unmanaged Teams
- Just give me the fish!
- When burndowns don't
- The plague known as 'Scrum-But'

Learning to Fly

- Scrum is simple but not easy
- Best practices are still applicable
- The importance of sprint retrospectives
- Hyperperformance: Scrum team dynamics
- Scaling Scrum

Advanced Agile: Beyond Basic Scrum

NEW! FOR 2011

After experiencing initial success with Scrum on small projects, companies are now looking to take Scrum to the next level. For some this means scaling Scrum to larger teams, multiple teams, or multiple sites. For others it means better integrating Scrum with existing business practices. For some it means pushing Scrum out to teams that weren't initially ready for Scrum. And for others it means simply building on success—seeing how far Scrum can be optimized after its initial implementation.

This intensive seminar shares insights from Construx's experience helping companies successfully scale Scrum beyond the individual project level. Extensive case studies and hands-on exercises provide practical advice for adopting Scrum planning and management practices and integrating them with other business processes to address the full software and product development lifecycles.

The Challenges of Scaling Scrum

- Why Scrum is successful on small projects
- Changes needed to achieve project scale
- Changes needed to support organizational scale
- Complex features that span iterations, teams, and products
- Initiatives and programs that span releases
- Scrum vs. Agile: Is Scrum still Scrum at scale?

Scaling Scrum

- Maintaining architectural integrity
- Sharing information across teams and the organization
- Prioritizing work across multiple products
- Managing utilitization of key people
- Techniques for estimating epics and large features
- Grooming the product backlog to support multi-team release planning
- Techniques for driving work across multiple teams

Using Strategy to Drive Agile Plans

- Using multiple levels of planning to succeed on larger projects
- Using milestone driven development to align teams
- Achieving flexibility in planning
- Understanding what drives business value
- Three often-overlooked categories of value

Scrum/Agile and Corporate Governance

- Scrum's ability to support governance models
- Working with the PMO
- Working with the PMI
- Working with the CMM
- Achieving predictability
- Achieving transparency

Scrum/Agile in a Stage/Gate Environment

- Attributes of Stage/Gate models
- Understanding the real goals and requirements of process models
- Finding the "hard" and "soft" points
- General strategy for meeting stage gate requirements
- Meeting stage-gate product-definition requirements
- Meeting stage-gate planning requirements
- Meeting stage-gate delivery requirements

Scrum/Agile in Regulated Industries

- Understanding real regulatory requirements
- Example organizations that have implemented Agile in regulated environments
- Identifying the "hard points" and "soft points"
- Overcoming common challenges

Scrum/Agile in Combined Hardware/Software Environments

- Combining Scrum/software teams with hardware teams
- Managing the software/hardware interface
- Managing requirements flow
- Systems engineering with Scrum

Advanced Scrum/Agile Team Issues

- Large team in one location
- Geographically distributed team members on a single team ("virtual team")
- Teams in multiple locations
- Teams in multiple countries
- Teams from multiple companies
- Working with vendors
- Integrating Scrum projects with non-Scrum projects
- Combining all of the above

See complete course description at www.construx com.

Scrum for the Enterprise

NEW! FOR 2011

Interested in Scrum but not sure how to implement it in a large organization? Based on Construx's popular Scrum Boot Camp, this seminar provides everything you need to know to ensure a successful Scrum implementation in an enterprise setting. This three-day course combines key aspects from Certified Scrum Master- and Certified Scrum Product Owner-specific training plus specific practices and insights needed to make Scrum successful within a large organization.

What is Scrum?

- Agile origins, principles, and benefits
- A brief history of Scrum
- Scrum philosophy and theory
- What makes Scrum different?

Why Scrum Works

- A simple process
- A focus on delivering value
- A committed, self-managed team
- Transparency: nowhere to hide
- Finishing what you start
- Continual improvement

Scrum Roles

- The Scrum Master
- The Product Owner
- The Team Member
- Stakeholders
- Levels of Commitment

Enterprise-Level Scrum Processes and Meetings

- Enterprise-Level Release Planning
- Sprint Planning
- The Daily Standup
- The Sprint Review
- The Sprint Retrospective

Enterprise-Level Scrum Artifacts

- The Enterprise-Level Product Backlog
- The Sprint Backlog
- The Release Burndown
- The Sprint Burndown
- The Release Roadmap
- The Product Roadmap

Scrum Best Practices

- Timeboxing: Nothing Concentrates The Mind...
- Commitment: "Either Do or Do Not; There Is No Try"
- Working Agreements: This Is How We Do It
- Acceptance Criteria And The Definition of Done
- There Is No 'I' in Team

Initiating an Enterprise-Level Scrum Project

- Establishing the Project Vision
- Creating a Release Roadmap
- Enterprise-Level Release Planning
- The Enterprise-Level Product Backlog
- Launching the Project

Populating the Product Backlog

- Using stories to drive value
- The attributes of an effective story
- Common agile estimation techniques
- Using story maps to find stories
- Single backlog or multiple backlogs?
- Overcoming common challenges

Life During Sprint Time

- A day in the life of a Scrum team
- A day in the life of a Scrum Master
- A day in the life of a Product Owner

Tracking Progress: Scrum Metrics

- Whiteboards or software?
- How is your iteration going?
- Exposing and removing impediments
- Burndown charts
- When are you going to release?

See complete course description at www.construx com.

"I learned a lot more from your course than I did from my Certified Scrum Master class!"

Sivapriya Mohan, Microsoft

Enterprise Agile: Planning, Managing, and Scaling Agile Projects Using Scrum [18 PDUs]

After experiencing initial success with Agile development on small projects, many companies are now looking to scale Agile development into multiple teams, multiple sites, and larger programs. Companies also want increased predictability and transparency and integration with existing business processes—without giving up Agile's responsiveness. In the years since the Agile Manifesto, Scrum has emerged as the Agile process that is "best of breed" for industrial-strength software development. This intensive seminar shares insights from Construx's experience helping companies successfully adopt Scrum and other Agile planning and management practices.

Agile At the Enterprise Level

- Scaling Scrum/Agile for larger projects
- Integrating Agile projects with non-Agile projects
- Aligning Agile with enterprise governance
- Integrating Agile with existing product lifecycles
- Overcoming common challenges

Using Strategy to Drive Agile Plans

- What is Strategy-Driven Development?
- Use multiple levels of planning to succeed on larger projects
- Achieve flexibility in planning
- Understand what drives business value
- Three often-overlooked categories of value

Agile Teams

- How are teams structured?
- What are the attributes of effective team members?
- Managing large Agile teams
- Managing multi-site/distributed Agile teams
- Managing combination Agile/non-Agile teams
- Identifying common challenges

Stories as Planning Tools

- Creating effective stories
- Estimating using stories
- Release planning and stories
- Sprint planning and stories
- Overcoming common challenges

Release Planning

- Creating Product Roadmaps and Release Roadmaps
- Achieving predictably and responsiveness
- Addressing technical debt
- Combining short-term and long-term planning
- Overcoming common challenges

Sprint (Iteration) Planning

- Creating Sprint Plans
- Details of the Sprint Plan
- Four key criteria for selecting stories for the sprint plan
- Balancing demand with capacity

Keys to Implementing Agile

- Three most important keys to successful implementation
- Achieving predictability and transparency
- Working within PMI and CMM environments
- Overcoming challenges in planning
- Overcoming challenges in estimation
- Overcoming challenges in scaling Agile
- Back to Work Action Plan

"I value the immediate applicability of the subject matter to my personal situation – I now have a huge list of to-do's."

Carla Chandler, Tom Tom

Scrum Product Owner Boot Camp

NEW! FOR 2011

The Product Owner role is arguably the most important role in Scrum—and the most challenging. The Product Owner is part project manager, part product manager, and part customer advocate. This person must ensure the customer's wants and needs are understood while ensuring that the team delivers the greatest-value features as quickly as possible—all while responding to ever-changing requirements. Ultimately, the Product Owner owns a Scrum project's success or failure. Scrum Product Owner Boot Camp drills down into the detailed information needed to successfully plan releases, reflect stakeholder priorities, ensure the team builds the right product, and communicate with marketing, sales, executives, and other project stakeholders.

It All Starts With a Vision

- The Product Vision

The Product Owner's View of Scrum

- What Is Scrum?
- How It Works
- Scrum Roles, Processes and Artifacts

The Successful Product Owner

- Technical…enough
- Product Owner, Project Manager, or Product Manager?
- Attributes of successful Product Owners

What Are We Trying To Build?

- What is a requirement?
- The Product Vision as the top-level requirement
- Three purposes of requirements
- Using requirements to manage risk
- Product versus project requirements
- User stories and acceptance criteria
- The Definition of Done and why it matters

How Will We Build It?

- Successful Scrum projects require planning
- The Product Backlog
- Creating a product roadmap
- Release planning
- User Story Mapping
- Incremental Delivery via Sashimi Implementation

During the game

- Product backlog grooming
- Sprint planning
- Sprint reviews and retrospectives
- Dealing with change
- Scrum project tracking

How to Be Agile Without Being Extreme [9 PDUs]

Agile software development promises low overhead, high flexibility, and satisfied customers, but how do you separate the hype from the reality? Leading organizations have benefited from Agile development practices for many years. Learn how to select and deploy today's most powerful Agile practices. Apply the essentials of Scrum, Extreme Programming, and other Agile methods. This intensive seminar presents modern practices combined with decades of time-tested, low-risk methods—all with a track record of proven results. This seminar is based on Construx's experience working with companies that have successfully deployed Agile practices—and our experiences with companies whose Agile projects have failed.

See complete course description at www.construx com.

"When to apply Agile and when not to."

Robert Orr, Northrup Grumman

Requirements Boot Camp [18 PDUs]

What is the most frequently reported cause of software project failure–regardless of project size or type of software? Requirements challenges. Discover how leading-edge companies use requirements engineering to support successful software projects. Learn the three purposes of requirements and how to distinguish between requirements fantasies and requirements reality. Practice practical techniques for exploring user needs, capturing requirements, controlling changes, and building highly satisfactory software.

Software Requirements: What and Why

- Requirements: fantasies and real world
- What is a requirement?
- Three purposes of requirements
- Product and project requirements
- Levels and types of requirements
- Characteristics of good individual requirements
- Characteristics of good sets of requirements
- The Vision Statement as the top-level requirement
- Requirements as a risk management activity
- Knowing when you're done

The Requirements Process

- Comprehensive strategies for defining requirements
- Iterative elicitation, analysis, specification, and validation
- Breadth-first approaches
- Depth-first approaches
- Spiral approaches
- Tools: chartering workshop, collaborative development, risk management, parallel development

Requirements Elicitation

- Who has requirements?
- Eliciting requirements from people
- Eliciting requirements from other systems
- Eliciting requirements from the environment
- Finding the decision maker
- Incorporating business rules
- Dealing with ambiguity
- Tools: interviews, context-free questioning, brainstorming, JAD workshops, prototyping, task analysis, use cases, competitive benchmarking, document archeology, project charter, vision statement

Who Defines Requirements

- The requirements engineer
- Requirements engineering roles
- Skills needed to develop requirements effectively
- How the requirements engineer relates to the rest of the project
- Checklist for requirements leads

Requirements Analysis

- Classification and prioritization schemes
- Requirements negotiation
- Tools: prototypes, use cases, essential systems modeling, data dictionary

Requirements Specification

- Characteristics of a good requirement specification
- Models as specification
- Guidelines for writing requirements
- Picking the right model for the task at hand
- Quantifying qualitative attributes
- Writing effective use cases

See complete course description at www.construx com.

INSTRUCTOR SPOTLIGHT

Dave Wright is a Senior Fellow at Construx Software. He has more than 30 years of experience in project management, program management, requirements, software architecture, design, construction, and configuration management. Throughout his career, Dave has shipped numerous large commercial products spanning Web, Windows desktop, client-server, and embedded technologies. He has also led numerous software process improvement teams and established software engineering process standards, including ISO 9001 certification.

Agile Requirements In Depth

NEW! FOR 2011

Agile development shifts traditional requirements work to a "just in time" approach. How does this affect good requirements practices? This class explains Agile approaches to traditional requirements sources including MRDs, PRDs, feature lists, and user scenarios. It dives into techniques for developing requirements on Agile projects, including the Agile Work Breakdown Structure (WBS), using story mapping to define the scope of the project, writing user stories, sizing stories (agile estimation), and developing acceptance criteria for user stories. Concepts are illustrated through extensive use of hands-on labs.

What Are We Trying To Build?

- It all starts with the Product Vision
- Characteristics of good Product Visions

Software Requirements: What and Why

- What is a requirement?
- The Product Vision as the top-level requirement
- Three purposes of requirements
- Using requirements to manage risk
- Product versus project requirements
- Potentially useful requirements artifacts
- Working software as the ultimate requirements specification

Initial requirements gathering

- Envisioning the high level requirements
- What can I do with it: high-level user stories and story mapping
- How it works: business rules and the domain model
- How it looks: low-fidelity UI models and workflows

Just-in-time Requirements Elaboration

- No requirement before its time: the concept of the last responsible moment
- Requirements elaboration during iteration planning

Requirements Change Management

Requirements Validation

- Acceptance criteria
- The Definition of Done and why it matters

Use Cases in Depth

Use cases are a powerful technique for gathering, organizing and verifying users' business-systems requirements. In this two-day seminar, leading software expert Meilir Page-Jones will show you what use cases are and how to use them to understand, model, and validate user requirements quickly and precisely. He will explain how to distinguish a good use case from a bad one. He will delve into the highly effective technique of business-event modeling and explain how a thorough understanding of business events simplifies creating sound use cases.

See complete course description at www.construx com.

INSTRUCTOR SPOTLIGHT

John Clifford is a Senior Fellow at Construx Software where he focuses on software development, project management, and team management practices with an emphasis on Agile practices. With more than three decades of IT experience, John has held leadership roles as a development engineer, product feature team manager, group QA manager, group project manager and development director. John holds Certified Scrum Master, Certified Scrum Product Owner, and Certified Scrum Practitioner certifications from the Scrum Alliance.

Object-Oriented Requirements Analysis and Design Using the UML

[4 PDUs]

This three-day seminar provides a programming language-independent overview of object-oriented requirements analysis and design using the UML. Special attention is given to relating OOA and OOD concepts to real-world software. No prior knowledge of object-oriented development is required.

See complete course description at www.construx com.

Requirements Modeling

NEW! FOR 2011

This course gives you hands-on experience using five basic requirement models to more efficiently and effectively elicit and analyze functional requirements. You'll create Context Diagrams, Activity Models (aka workflow models), Domain Models (aka E-R models, class models, data models), Use Cases and State Models. You'll gain proficiency at using these models in practical and agile ways to more precisely and concisely capture requirements without getting caught up in modeling semantics. You'll see how these five models can enable you to gather more requirements earlier in the project, and why model-based requirements exhibit greater stability than those that are interview-based. The course will also sharpen your instincts for knowing when you've done enough requirements work to proceed, and where requirements risks remain.

Define Our Model Toolkit

- Define model
- Requirements modeling toolbox

Model Your Context

- State the boundary
- Identify actors
- Classify into sets
- Name the data flows
- Prioritize

Define Use Cases

- Define tasks
- Select the primary actor
- Confrim value proposition
- Develop post-conditions
- Develop pre-conditions
- Write a description
- Select a normal course
- Separate actor and system steps
- Create alternate courses
- Define exception courses
- Add specific information

Create Activity Models

- Examine the activities
- Start from the end
- Align actors and sub-tasks
- Sequence the activities
- Show decision points
- Fork parallel activities
- Sync activities
- Identify technology artifacts

Start a Class Model

- Find the nouns
- Search other models
- Examine data cohesion
- Name the classes
- List important attributes
- Run the pit test
- Look for data coupling
- Associate the classes
- Assign cardinality

Utilize State Models

- Find "status"
- Describe a state
- Borrow from activity models
- Bring an instance into existence
- Transition to a new state
- Record transition rules
- Delete an instance

Find the Requirements

Design Boot Camp

Different designers will create designs that differ by at least a factor of 10 in the code volume produced. How do you invent simple, straightforward designs and avoid complex, error-prone designs? Understand the fundamental design principles that lead to high-quality designs requiring low implementation effort. Learn both Agile and traditional approaches to create great designs quickly and economically.

What is "Design"?

- Design as an activity vs. a product
- Design as a tool for communicating
- Managing complexity with design
- Characteristics of excellent designs

Fundamental Design Principles

- Use abstraction
- Encapsulate design decisions
- Maximize cohesion; minimize coupling
- Design to invariants
- Avoid premature optimization
- Beware of Fisher's Fundamental Theorem

Managing Design Complexity

- Dimensions of design (interface, data, function)
- Measures of design complexity in each dimension
- Balancing local and global complexity

Design attributes: The "-ilities"

- ISO/IEC 9126 Quality Model
- Non-ISO/IEC 9126 "-ilities"
- Managing conflicts among the "-ilities"

Architectural Design

- Pipes & filters
- Model-view-controller
- Layered architectures
- Service-oriented architectures
- Blackboard

Design Paradigms

- Aspect-oriented design
- Object-oriented design
- Structured design
- Design patterns

See complete course description at www.construx com.

Design Pattern Essentials

Design patterns made understandable! Design patterns are powerful, predefined solutions to common software design problems. Patterns provide a powerful form of reuse because they are transportable across different languages and software architectures. This seminar introduces and explains the highest-leverage design patterns in a clear way that's easy to understand. You will learn how to use specific design patterns to improve your designs, and you will practice applying design patterns to sample design problems. You will also see how design patterns can improve your design process overall.

See complete course description at www.construx com.

INSTRUCTOR SPOTLIGHT

Steve Tockey is the author of *Return On Software*, a guide for companies that want to maximize their investment in software development. As a Principal Consultant at Construx, Steve focuses on software project management, estimation, software quality, object-oriented development, and distributed object computing. He is a Certified Software Development Professional (CSDP), and chairs the CSDP Certification Committee of the IEEE Computer Society, and is a charter member of the OMG, the group that oversees development of the UML.

Developer Boot Camp

In this intense hands-on seminar, you will learn dozens of proven tips, techniques, and principles to produce clean, dependable, and maintainable code. Capturing the body of knowledge available from research, academia, and everyday commercial practice, this seminar synthesizes the most effective techniques and must-know principles into clear, pragmatic guidance. This seminar illustrates the principles with numerous concrete examples of good and bad code in a variety of languages. In depth labs allow you to practice applying the principles.

Managing Complexity

- Essential and accidental difficulties
- Modularity
- Encapsulation
- Information hiding
- Coupling
- Cohesion
- Abstraction
- Interfaces
- Design by contract
- Knowledge as data

Designing for Change

- Variability analysis
- Typical changes
- Design to invariants
- Extension & intension
- Association versus inheritance
- Delay binding times
- Composability
- Open / closed principle

Key Construction Skills

- Design principles
- Design patterns
- Structured programming
- Object-oriented programming
- Functional programming
- Improving productivity
- Working on a team
- Writing legible code
- Tools, techniques, and practices

Increasing Your Software's Value

- Effective requirements
- Prototyping
- Modeling
- User interface design
- Read time versus write time convenience
- Characteristics of high value software
- Transparency

Error Handling

- Assertions
- Exceptions
- Diagnostics
- Defensive programming

Ensuring Correctness

- Functional unit testing
- Structural unit testing
- Automated feature testing
- Domain analysis
- Equivalence class partitioning
- Testing strategies
- Measuring complexity
- Minimize the lag between error insertion and error detection
- Avoiding common pitfalls
- XUnit frameworks
- Mutation testing
- Static analysis
- Debugging
- Developer testing tools

INSTRUCTOR SPOTLIGHT

Melvin Perez-Cedano is a Senior Fellow at Construx Software where he focuses on software design, software construction, software process, and configuration management. He has a successful track record at transforming dysfunctional and immature organizations into performance and quality-oriented environments. Melvin has deep experience in Agile practices including code cleaning, refactoring, and unit testing, and is also well versed in UML, RUP, and CMMI. He has lectured extensively throughout the Americas and is a former IEEE Distinguished Lecturer for Latin America.

Developer Testing Boot Camp [12 PDUs]

Developer testing is a critical component of Agile development—yet studies show that developer tests typically achieve only 50-60% test coverage. Learn how to achieve higher test coverage with your unit tests, how to create better test case designs, and how to make the software itself more testable. See how to plan and carry out an efficient developer testing strategy. Avoid common testing pitfalls, and learn to determine how much developer testing is enough.

Key Concepts in Developer Testing

- Test self-assessment—how good a tester are you?
- Test concepts: white box vs. black box, regression testing, positive and negative test cases, testing as "proof" of quality, test coverage
- Relating testing to other QA and development activities

Developer Testing Basics

- Kinds of tests: single-step testing, bench testing, unit testing, integration testing, as well as system test and acceptance test
- Stubs, drivers, and mock objects

Test-First Development

- Advantages of writing test cases before writing code
- Test-first development cycle described
- Use of unit test frameworks
- Most useful test strategies

Test Planning

- Scoping unit and integration testing to match your project's needs
- Design for testability
- Maximizing efficiency of unit testing and integration testing
- Validating your unit tests and integration tests

Testing Databases

- Database interface testing
- Internal database testing
- Obtaining test data
- Tools for database testing

Black-Box (Functional) Testing

- Finding domain specifications
- Common domain defects
- Designing input domain unit tests
- Designing output domain unit tests
- Equivalence classes
- Boundary value analysis
- Pairwise testing

White-Box (Structural) Testing

- Control-flow testing
- Tools for structural coverage
- Statement coverage
- Path coverage
- Decision (branch) coverage
- Modified condition/decision coverage

Special Testing Issues

- Mock objects
- Inheritance issues
- Testing embedded software
- Writing tests based on specifications
- Integration testing strategies

Test Tools

- Unit test frameworks
- Memory leak detectors
- Performance analyzers
- Coverage monitors

Testing the Tests

- Peer Reviewing test cases
- Defect seeding
- Mutation testing

Effective Concurrency [18 PDUs]

This course covers the fundamental tools that software developers need to write effective concurrent software for both single-core and multi-core/many-core machines. To use concurrency effectively, we must identify and solve four key challenges:

- Leverage the ability to perform and manage work asynchronously
- Build applications that naturally run faster on new hardware having more and more cores
- Manage shared objects in memory effectively
- Engineer specifically for high performance

See complete course description at www.construx com.

Configuration Management Essentials [6 PDUs]

Configuration management is a lynch pin of successful software projects. Effective software teams manage change well; they know what was changed, when, and by whom. In this class you will learn practical, real-world techniques for controlling and stabilizing your software, and you will gain an understanding of how to draw from both Agile and traditional configuration management practices to create a process that is "right-weight" for you. You will also learn to identify and avoid many classic pitfalls encountered when implementing configuration management.

See complete course description at www.construx com.

Code Complete Essentials

In this intense one-day seminar you will learn dozens of proven tips, techniques, and principles to produce clean, industrial strength code. Capturing the body of knowledge available from research, academia, and everyday commercial practice, this seminar synthesizes the most effective techniques and must-know principles into clear, pragmatic guidance. This seminar uses dozens of examples of good and bad code in Java, C++, C#, and Visual Basic to explain how to shorten development time, reduce errors, and make debugging easier.

This seminar is taught by Steve McConnell, the coding guru who wrote the best-selling *Code Complete*, a computing industry classic that won the Jolt Excellence award for best programming book of the year and has been translated into more than a dozen languages.

Introduction

- Construction's critical role in software development
- Technology knowledge vs. principles knowledge
- Dealing with "accidental" and "essential" difficulties

Defensive Programming

- Error processing
- Effective strategies for anticipating change
- Code stepping
- Offensive programming

Creating High Quality Designs

- Differences in design effectiveness
- Attributes of great designers
- The Primary Technical Imperative: Managing Complexity
- Managing technical debt
- The relationship between naming and design
- Design guidance: Information hiding, abstraction, encapsulation, modularization, cohesion, separation of concerns

High-Quality Routines

- Coding Horror: examples of low-quality routines
- Program layout techniques
- Low-effort, high-payoff commenting techniques
- The Pseudocode Programming Process

High-Quality Classes

- Good and bad reasons to create classes
- Designing interfaces

Code Optimization

- A defensive strategy for code optimization
- Three optimization approaches that don't work
- Example of intensive optimization

Quality Practices

- Debugging by superstition
- A scientific approach to debugging
- Tips for finding defects
- Tips for fixing defects
- Defect cost dynamics
- Error prone modules

Key Principles

- Actively manage essential difficulties
- Keep accidental difficulties from increasing needlessly
- Minimize complexity
- Differentiate between complexity inherent in the problem vs. complexity created by the solution
- Minimize needless variations
- Favor read-time convenience to write-time convenience
- Consider whether you should be programming "into" your language rather than "in" it
- Minimize the lag between error insertion and error detection

Peer Mentoring: Effective Knowledge Transfer

This one-day seminar teaches people how to deliver on-the-job training. Originally developed for engineers at Microsoft, the course has evolved over the last ten years to help thousands of individuals and teams cross-train, transfer knowledge, and get new staff up to speed quickly and efficiently. This is a fast-paced, one-day program that uses plain language and practical tools to develop communication and training skills.

Roles in Peer Mentoring

- Outline job descriptions for manager, mentor, and apprentice
- Consider the benefits of being a better peer mentor
- Provide tips for how to be a successful apprentice
- Write goals that help guide the relationship and the work
- Conduct a "first meeting" to set expectations and discuss a plan for moving forward

Managing Communication

- Cover practical communication techniques for all three roles and role responsibilities
- Say how to communicate in the way that works best for the mentor (email, open door, time of day, interruptions)
- Use regular status meetings and status reports to consistently stay in touch
- Develop practical advice on how to ask a well-thought-out, problem-solving question

Focusing on the Most Important Information

- Learn to quickly build a foundation before teaching a skill
- Define the "air, food, and water" that must be covered at the beginning
- Paint a "big picture" to provide context
- List and prioritize the skills needed

Telling What You Know

- Pick up basic teaching techniques
- Determine the "least amount of information necessary"
- Move skills and information from short-term memory to long-term memory
- See how to do the most effective demonstration

Leveraging Learning Styles

- Define four different learning styles
- Consider how learning styles affect teaching styles
- Use different teaching styles with different learners
- Help apprentices identify their own learning style

Assessing Knowledge

- Use open-ended questions to assess the apprentice
- Figure out what they already know before starting
- Check in to make sure they're learning
- Ensure they have clear priorities before turning them loose

Giving and Getting Feedback

- Look for opportunities for peer-appropriate feedback
- Define the characteristics of good feedback
- Learn how to focus on the goal, not the person
- Discuss how to ask for feedback

Developing an Action Plan

- Create a peer mentoring plan
- Identify obstacles to success
- Discuss ways to bring peer mentoring tools back to a larger group

PMI Registered Education Provider — Project Management Institute

Construx is a recognized provider with the PMI Registered Educational Provider Program (PMI R.E.P.). Construx accepts and adheres to all PMI R.E.P. Program policies, requirements and rules concerning the provision of professional education activities and materials. Attendees receive Professional Development Units (PDUs) for completing each PMI Registered course.

IIBA Endorsed Education Provider

Construx is a recognized provider with the IIBA Endorsed Educational Provider Program (IIBA EEP®). Construx accepts and adheres to all IIBA program policies, requirements and rules concerning the provision of professional education activities and materials. Attendees receive Continuing Development Units (CDUs) for completing each IIBA Endorsed course.

Professional Tester Boot Camp [4 PDUs]

How do professional testers test software? This course teaches the techniques, tips, tricks, and strategies used by test professionals. You will learn and apply numerous detailed test design strategies for black-box (functional) testing and system testing. You will also learn practical techniques for planning, designing, and executing effective testing on real software projects. Determine how much testing is enough for your project, whether your test cases are adequate, and how to minimize wasted testing effort. *This seminar emphasizes black box (functional) testing and system testing.*

Test Planning

- How to make your testing more effective
- Making your testing more efficient
- Evaluating test case designs
- When to plan for testing
- Relating testing to other software development activities
- Evaluating test case designs
- Determining how much testing is enough

Domain Testing

- Where to find domain specifications
- Analyzing domain specifications
- Common domain defects you should test for
- Designing input domain tests
- Designing output domain tests
- Analyzing domain boundaries

Testing from Requirements

- How to ensure that every requirement is tested
- Deciding the number of test cases necessary for each requirement
- Creating test cases from use cases
- Creating test cases from relationship models
- Creating test cases from state-transition models

Tool Support for Testing

- Determining which tools are useful for supporting testing activities

Pairwise Testing

- When is pairwise testing useful?
- Learn how to reduce the number of test cases when you have to test all combinations. For example:
  - Multiple operating systems
  - Multiple web browsers
  - Plug-ins
  - Web server software
  - Server operating systems
- Downloadable software to figure out your test cases
- Commercial tools that can help with pairwise testing

Test Automation

- Improving the efficiency of testing with automation
- Why is test automation harder than just recording and playing back keystrokes?
- How to get started with data driven test automation
- Advantages of keyword driven test automation
- Commercial tools/frameworks to support keyword driven test automation

Testing the Tests

- How do you know your tests are reliable?
- How do you know your tests are effective?
- How do you uncover and improve low quality tests?

INSTRUCTOR SPOTLIGHT

Eric Rimbey is a Senior Fellow at Construx Software where he focuses on software design, requirements, quality, and testing. He has held a variety of leadership roles including business analyst, technical lead, test lead, and QA lead, and has developed a wide range of Web 2.0 applications for social networking, wireless systems, manufacturing, large-scale military programs, and the retail industry. Eric is a Certified Software Development Professional (CSDP) and a member of IEEE.

Advanced Quality Boot Camp: Beyond Testing [15 PDUs]

From project inception, a focus on quality through planning, execution, and delivery can help improve software project cost, schedule, and functionality. This three-day seminar shows you how to define quality in specific terms with a focus on how to decide what is "good enough" for a particular project. You will learn to align project activities to achieve quality throughout the project lifecycle—including numerous alternatives to end-of-project testing.

Defining Quality
- Textbook Quality Definitions
- QA vs. QC

Quality Goals
- The ISO 9126 definition of quality
- Characterization
  - Scaling a characteristic
  - Prioritization of characteristics
- Establishing what it means to be "good enough"

General Strategy
- The basic philosophy of software quality
- Early removal of defects
- Insulating from impact of defects
- Using the PDCA cycle

Finding Faults
- Common faults in software and where they are created
- Identifying the defects
- Tracking defects

Detection Toolbox
- Peer reviews
- Quality attribute workshops
- Dynamic testing
- Prototyping
- Detection effectiveness

Approaching Quality
- Basic approach
- Lifecycles and characteristics

Implementing Quality in Requirements
- Five whys
- Task analysis
- Vision & goals
- Gists
- Fit criteria

Implementing Quality in Design
- Facilitated workshops
- Architecture tradeoff analysis method
- Design impact estimation
- Personas

Implementing Quality in Code
- Satisfice and softgoals
- Optimization

Analytical Toolbox
- Measures
- Comparing discrete classes
- Trending a class
- Process control
- Collecting data

Managing Software Quality
- Organizational level quality
- Process improvement

Establishing a Quality Culture
- Growing capability
- Selecting practices
- Audits

See complete course description at www.construx com.

Software Inspections: Intensive [4 PDUs]

Upstream inspections can eliminate up to 90 percent of the defects before testing while simultaneously producing net cost and schedule savings of 10 to 30 percent. Each hour of upstream inspection on large projects can save an average of 33 hours of maintenance. Inspections reduce the number of defects encountered by your customers, improve individual skills and performance, and help build stronger teams. Used skillfully, inspections can be one of the best investments your organization makes in the improvement of its software development process.

See complete course description at www.construx com.

10x Software Engineering [16 PDUs]

Decades of research have found at least a ten-fold (10x) difference in productivity and quality between the best developers and the worst—and between the best teams and the worst. Discover the 8 Key Principles of 10x Engineering. Gain a deeper understanding of the factors that affect productivity, and avoid the productivity traps of "minus-x" engineering. Learn and apply critical techniques that will turn your team into a high performing, 10x Team.

Defining 10x

- 10x Differences in Productivity and Quality
- 10x Principles

10x Principle: Avoid Minus-x Engineering

- Classic Mistakes
- Brute Force Quality
- Multi-Tasking
- Typical Minus-x Project
- Your Minus-x Project

Basic Engineering

- Mastery of Fundamentals and Excellent Execution
- Technical Fundamentals
- Technical Management Fundamentals,
- Quality Fundamentals
- Your Organization's Bare Essentials

10x Principle: Seek Ground Truth

- Daily Build & Smoke Test
- Project Tracking
- Communicating Status
- Root Cause Analysis
- Change Control–Agile and Plan driven
- Gates and Checkpoints
- Evidence-Based Case Study

10x Principle: Base Decisions on Data

- Plan-Do-Check-Act
- Measurement
- Iteration and Incrementalism
- How Much Agility is Enough?
- Customer Involvement
- Power of Interactive Workshops

10x Principle: Tailor the Solution to the Problem

- Streamlining Work with Intellectual Phase Profiles
- Phase and Activity Variations
- Life Cycle Models
- Efficient Information Capture
- Documents and Digital Cameras
- Toolboxes

10x Principle: Set Direction

- Project Charter
- Vision Statements

Feature Selection

- Pareto Analysis
- T-Shirt Sizing
- Rolling Wave Planning
- Product Backlog

See complete course description at www.construx com.

Rapid Development in Depth

Time to market is a constant issue for software projects. This seminar presents specific, practical techniques for dramatically improving time to market on real-world software projects. Case studies based on high-profile projects as well as projects drawn from Construx's consulting experience are interspersed throughout the workshop to illustrate development principles and give students experience applying best practices in rapid development.

This seminar is based on Steve McConnell's best-selling book, *Rapid Development.*

Attendees will gain an in-depth understanding of the major issues that affect software development speed. Through numerous case studies, they will practice rapid development techniques that they can apply to their own projects. They will have time to discuss their specific issues with the instructor. And they will develop a Back-to-Work Action Plan.

See complete course description at www.construx com.

ANNOUNCING THE NEW SEMINARS FOR 2011

- Advanced Agile: Beyond Basic Scrum
- Agile Requirements
- Requirements Modeling
- Scrum for the Enterprise
- Scrum Product Owner Boot Camp

Details inside

INSIDE:

Two offers that make NOW the best time for training

PUBLIC SEMINARS

Register now for great early-bird savings!

PRIVATE SEMINARS

Be our guest to preview the seminar of your choice!

PHONE
1-866-296-6300
ONLINE
www.construx.com/2011
EMAIL
training@construx.com

Construx®
10900 NE 8th Street
Suite 1350
Bellevue, WA 98004