# xtUML / BridgePoint Training

Ericsson has expressed interest in a tailored approach to training engineers in the xtUML method and the use of the BridgePoint tool suite. The audience and goals, along with an outline of each component of this training regimen are described below.

## Audience and Goals

Programmers of varying levels of experience with little or no exposure to model-based development represent the core of the audience. The goal is to produce graduates who can build a working model, given only a functional specification as input.

## Background

In the 1980s and 1990s, Project Technology was very successful teaching the Shlaer-Mellor predecessor to the xtUML method. During this period the method was taught in three separate courses, each nominally four days in length. The exercises were done with paper and pencil, so syntactic correctness and the ability to execute and automatically translate the resulting models were unimportant. Instead, the focus was on learning proper use of the various modeling constructs and creating good abstractions. The three four-day courses in this series were (translating S-M nomenclature to xtUML):

* Components, Interfaces, and Classes
* State Models and Action Language
* Model Compiler Design and Construction

However, this was an era in which managers valued training. The following decade saw a shift away from knowledge and skill acquisition toward tooling. Rather than educate their engineers, managers sought to arm them with tools with the hope that doing so would increase productivity. Over time, market pressure drove the compression of the first two modeling classes into a single, five-day affair. Several modeling constructs and idioms were eliminated from the new one-week modeling class for lack of time to cover them.

Tool usage was taught during a separate three-day course, with the modeling course as a prerequisite. “Eight days of training?!!!” “We simply cannot afford to have our engineers take off eight full days for training!!” “The release is due out next month!!”

Eventually, market forces demanded a combined course that covered modeling and tool usage in less than a week. The result is the current four-day course that attempts to cover what was in the past eleven days of material.

The current audience for these training courses has a quite different educational background and experience base from that of the students in the 1980s and 1990s. However, these two generations of engineers share an important trait. Nearly all of them have been trained to write procedural code rendered in imperative languages like C, C++, and Java before having been introduced to the notion of first creating abstractions around which the code will eventually be written. The proposal below leverages the much broader range of technology available while recognizing that the fundamental challenge of teaching object-oriented, model-based development remains much the same as it was in 1990.

## Proposal

This training regimen includes several components, each of which is a prerequisite for the one that follows it. Each of these components is briefly described in a sub-section below. The outline for the program is:

1. Motivational Discussion (half day)
2. Self-paced Tool Introduction (half day)
3. Basic xtUML Modeling (four days)
4. Self-paced Tool Training (TBD)
5. Self-paced Completion of Case Study Model (TBD)
6. Advanced xtUML Modeling (four days)

### Motivational Discussion

Before embarking on a new approach to any significant effort it is useful first to explore the need for a change and the rationale behind the new way of working. Such an exploration is best done in full duplex, preferably in person. This session leverages presentations, but it must not be a monologue in which the presenter does nothing more than dictate to the audience how things will be. Instead, the presentation must be interactive, seeking to engage the audience, encouraging questions, and facilitating discussions of the technical aspects as well as the inevitable fear, uncertainty and doubt that accompanies change of this sort.

Topics covered include:

* Motivation for change: issues with traditional development approaches
* Overview of model-based development and testing methods
* Automation of these methods through tooling

### Self-paced Tool Introduction

* Prerequisite: Motivational Discussion

After the motivational discussion most engineers crave a look at the tooling. Therefore, the next step in the training sequence is a brief overview of the BridgePoint tool suite to expose the student to the look and feel of the tool and xtUML models. After completing this component the student clearly understands:

* The collection of UML notation used when constructing executable models
* That xtUML models can be executed, tested, and debugged
* That xtUML models can be compiled automatically into executable code

### Basic xtUML Modeling

* Prerequisites:
  + Motivational Discussion
  + Self-paced Tool Introduction

With a focus on modeling this instructor-led course covers the xtUML method, teaching both the language of modeling and the techniques of object-oriented abstraction. Exercises are completed on paper to avoid the distractions associated with the tooling.

Upon completion of this component the student is capable of building an xtUML model, given only a functional specification as input. This includes:

* Defining use cases and activity diagrams to clarify requirements
* Defining component structure and interfaces
* Building class and state models
* Writing action language
* Connecting models to externally-produced code
* Creating modelled test cases

Given limited time (four days) and the student’s finite capacity to absorb new concepts, this component necessarily omits a number of modeling constructs, idioms, and techniques. However, the material covered in this component is sufficient for building working models.

### Self-paced Tool Training

* Prerequisites:
  + Motivational Discussion
  + Self-paced Tool Introduction
  + Basic xtUML Modeling

With a solid foundation in xtUML modeling the next step in the training sequence is comprehensive tool training. While this component provides a guided tutorial presenting all the tooling capabilities in a logical sequence along with a quiz for each section, it can also be used as an on-demand reference manual.

Upon completion of this component the student understands the mechanics of:

* Creating workspaces and projects
* Creating every type of model element and diagram supported by the xtUML editor
* Connecting models to externally produced code
* Leveraging version control and configuration management
* Comparing and merging branches
* Searching for model elements within a model
* Executing and debugging a model with Verifier
* Compiling a model and compiling and executing the resulting generated code

### Self-paced Completion of Case Study Model

* Prerequisites:
  + Motivational Discussion
  + Self-paced Tool Introduction
  + Basic xtUML Modeling
* Co-requisite: Self-paced Tool Training

This component includes no instruction or guidance but instead is an assignment. The student is provided with a set of requirements for a particular application, most likely the same application used as a case study for the basic xtUML modeling course. The student is then expected to do the following with BridgePoint:

* Build a complete executable model of the application
* Build a modeled test suite to test and verify the application model
* Debug the model using Verifier
* Generate code for the application and test suite using a specified model compiler
* Test and verify the generated code

### Advanced xtUML Modeling

* Prerequisites:
  + Motivational Discussion
  + Self-paced Tool Introduction
  + Basic xtUML Modeling
  + Self-paced Tool Training
  + Self-paced Completion of Case Study Model

This instructor-led course covers all the modeling constructs, and several idioms and techniques omitted from the basic modeling course while focusing on the creation of good abstractions. With a goal of teaching students to build models that are easy to understand, extend, and maintain, upon completion of this component each student:

* Knows all of the modeling constructs available in the xtUML method
* Has been exposed to the most useful idioms and patterns
* Understands the range of abstraction levels available
* Can evaluate models along several dimensions including:
  + Accuracy of the abstraction
  + Understandability
  + Level of abstraction
  + Degree of invariance
  + Coupling and cohesion

## Summary

Note that this training regimen cannot promise to produce engineers capable of building high-quality models. For some, producing excellent abstractions is a gift while others must work years to acquire a reasonable level of competency. However, even poorly abstracted models are more accessible and hence more valuable than poorly abstracted code.

Accordingly, this training program produces engineers competent in the xtUML method. How well the method is then employed will vary from one engineer to the next in much the same way as code quality varies across an organization. Though, unlike code, models expose abstractions, encouraging and facilitating improved quality along all the dimensions enumerated above.