eCommerce requires the rapid development of large systems. Two requirements for success are a lightweight process, such as eXtreme Programming; and avoiding the reinvention of skills (embodied in patterns) and software (embodied in components). In order to use components successfully, we have to define their interfaces carefully. Although this should be done with test code (in the XP style) in practice the tests form a very clumsy basis for thinking about the components' behaviours. UML provides a more readable basis, provided it is used at a high level of abstraction (not just pictures of the code) and provided it is directly connected to the interface tests. The use of UML to define tests contrasts strongly with the typical transformation of UML into program code. In recent times, a growing body of developers have reported disappointing results with 'reverse engineering', finding that UML diagrams drawn from complex code are just as hard to follow and re-engineer as the code itself. This tutorial will show how to use UML to make useful abstract descriptions of component and system requirements and interfaces; how to translate those to tests; and how to integrate component based development with eXtreme Programming.

**Audience**
Technical managers, architects and developers.

**Benefits**
- Make development more rapid and reliable.
- Avoid re-invention of solutions.
- Obtain early and continuous feedback on development.

*Alan Wills is a consultant in software development methods based in Manchester, UK. Since 1991, he has worked with clients in a wide variety of fields in UK, Europe, and the USA. He is a frequent presenter at conferences.*