MODELING FUNCTIONAL REQUIREMENTS WITH USE CASES: FREQUENTLY ASKED QUESTIONS (FAQs)

What is a use case? A use case describes an interaction between an actor (an external entity) and your system or business. As such, it describes a function that your system or business must provide to an actor. In an order processing application, for example, you might have an Enter Order use case that describes the order entry interaction between a customer (an actor) and the application.

Does a use case differ from a functional specification? You can employ use cases to model business processes, a system's functional requirements, or even the internal workings of a system. When used to model functional requirements, a use case describes one function required of your system or application. As such, your use cases constitute a functional specification.

Do use cases describe all requirements? No. While use cases represent a complete functional requirements model, they may not capture non-functional requirements such as usability and general performance requirements.

Why do functional requirements matter? A Standish Group study found that 37% of all project failures stemmed from inadequate or non-existent requirements analysis activities. Jerry Weinberg observed that 60% of all errors originate in the requirements phase, while Alan Davis found that errors are 10 times and 200 times more costly to correct during implementation and maintenance, respectively, than during requirements analysis. If these studies are anywhere close to accurate, the failure to correctly model a system's requirements has drastic negative effects on a project's chances of success.

What form does a use case take? The Unified Modeling Language (UML) includes a use case diagram, which serves as a high-level "roadmap" of actors, use cases, and their relationships. Each use case has accompanying text that provides the details of that use case. Typical descriptions include what triggers the use case and the use case's steps, among other things.

Why are use cases a preferred way to write functional requirements? With use cases, your functional requirements model is centered around the individual required functions of the system. Furthermore, use case descriptions are typically episodic in nature. As a result, use case-centered functional requirements are easier to read and understand than many other styles of requirements documents. Use cases can also drive subsequent development activities. For example, you can walk through use cases to develop a design of the system. Because use cases describe required functions, you can use them to identify and design system tests. In addition, because use cases describe how actors interact with the system, technical writers can use them as a basis for user documentation.

Where do use cases fit in the software development life cycle? Writing use cases for functional requirements occurs during a project's requirements analysis phase. This is typically very early in the development cycle, although you can apply an incremental development model in which you write use cases during each "build" cycle (specifically, for the requirements to be addressed during that cycle).

Do use cases make sense if I'm using a "lightweight" process model such as eXtreme Programming (XP)? Yes. The advocates of the "agile" process models such as XP stress the importance of up-front (although incremental, or "adaptive") planning. Identifying and modeling use cases are key components of that planning activity.

Are use cases object-oriented? No. Because they originated as one part of an object-oriented development method, use cases are often associated with object technology. There is nothing inherently object-oriented about use cases, however. You could employ use cases for modeling the functional requirements of any system or application.

Who should be a part of developing use cases? In some organizations, requirements and/or business analysts identify and write use cases. (Writing good, effective use cases does not require an understanding of software design and implementation.) In some settings, the software developers are also responsible for modeling the requirements and therefore write the use cases.

What kinds of problems am I likely to encounter? While use cases offer an effective vehicle for representing requirements, they do not eliminate the many conditions that make requirements modeling a difficult undertaking: human communication is imprecise, customers don't always know exactly what they want, the requirements change during development, etc. In addition, use cases present some challenges of their own. One problem many projects encounter in their initial use case effort is, what is the correct level for a use case? When is a use case too broad or too narrow? By applying some simple guidelines, use case writers can minimize these kinds of problems.

