**SPIRAL MODEL**

**Definition:**

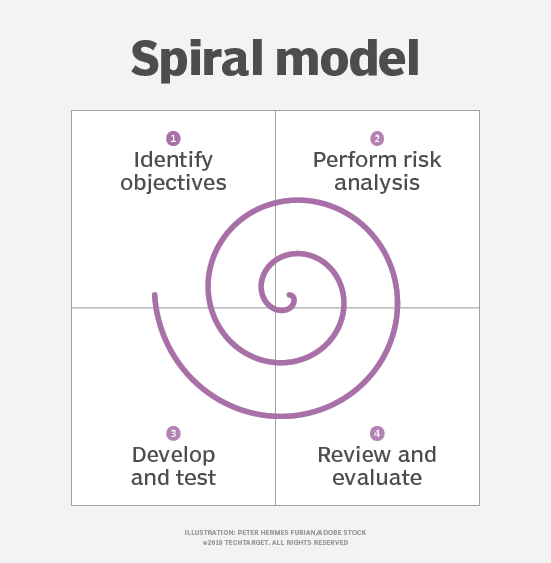
The spiral model is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model.

**Uses:**

As mentioned before, the spiral model is best used in large, expensive and complicated projects. Other uses include:

* projects in which frequent releases are necessary;
* projects in which changes may be required at any time;
* long term projects that are not feasible due to altered economic priorities;
* medium to high risk projects;
* projects in which cost and risk analysis is important;
* projects that would benefit from the creation of a prototype; and
* projects with unclear or complex requirements

**Spiral Model Phases:**



When looking at a diagram of a spiral model, the radius of the spiral represents the cost of the project and the angular degree represents the progress made in the current phase. Each phase begins with a goal for the design and ends when the developer or client reviews the progress.

Every phase can be broken into four quadrants: identifying and understanding requirements, performing risk analysis, building the prototype and evaluation of the software's performance.

Phases begin in the quadrant dedicated to the identification and understanding of requirements. The overall goal of the phase should be determined and all objectives should be elaborated and analyzed. It is important to also identify alternative solutions in case the attempted version fails to perform.

Next, risk analysis should be performed on all possible solutions in order to find any faults or vulnerabilities -- such as running over the budget or areas within the software that could be open to cyber attacks. Each risk should then be resolved using the most efficient strategy.

In the next quadrant, the prototype is built and tested. This step includes: architectural design, design of modules, physical product design and the final design. It takes the proposal that has been created in the first two quadrants and turns it into software that can be utilized.

Finally, in the fourth quadrant, the test results of the newest version are evaluated. This analysis allows programmers to stop and understand what worked and didn’t work before progressing with a new build. At the end of this quadrant, planning for the next phase begins and the cycle repeats. At the end of the whole spiral, the software is finally deployed in its respective market.