The Spiral Model

The spiral model, illustrated in Fig. 1.7, combines the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model, therein providing the potential for rapid development of incremental versions of the software. In this model the software is developed in a series of incremental releases with the early stages being either paper models or prototypes. Later iterations become increasingly more complete versions of the product.

As illustrated in Fig 1.7, the model is divided into a number of task regions.
Depending on the model it may have 3-6 task regions (/framework activities) our case will consider a ‘6-task region’ model.

These regions are:

1. The **customer communication** task – to establish effective communication between developer and customer.
2. The **planning** task – to define resources, time lines and other project related information.
3. The **risk analysis** task – to assess both technical and management risks.
4. The **engineering** task – to build one or more representations of the application.
5. The **construction and release** task – to construct, test, install and provide user support (e.g., documentation and training).
6. The **customer evaluation** task – to obtain customer feedback based on the evaluation of the software representation created during the engineering stage and implemented during the install stage.

The evolutionary process begins at the centre position and moves in a clockwise direction. Each traversal of the spiral typically results in a deliverable. For example, the first and second spiral traversals may result in the production of a product specification and a prototype, respectively. Subsequent traversals may then produce more sophisticated versions of the software.

An important distinction between the spiral model and other software models is the explicit consideration of risk. There are no fixed phases such as specification or design phases in the model and it encompasses other process models. For example, prototyping may be used in one spiral to resolve requirement uncertainties and hence reduce risks. This may then be followed by a conventional waterfall development.

- Note that each passage through the planning stage results in an adjustment to the project plan (e.g. cost and schedule are adjusted based on the feedback from the customer, project manager may adjust the number of iterations required to complete the software....)
- Each of the regions is populated by a set of work tasks called a task set that are adapted to characteristics of the project to be undertaken. For small projects the number of tasks and their formality is low. Conversely, for large projects the reverse is true.
Advantages of the Spiral Model

• The spiral model is a realistic approach to the development of large-scale software products because the software evolves as the process progresses. In addition, the developer and the client better understand and react to risks at each evolutionary level.

• The model uses prototyping as a risk reduction mechanism and allows for the development of prototypes at any stage of the evolutionary development.

• It maintains a systematic stepwise approach, like the classic life cycle model, but incorporates it into an iterative framework that more reflect the real world.

• If employed correctly, this model should reduce risks before they become problematic, as consideration of technical risks are considered at all stages.

Disadvantages of the Spiral Model

• Demands considerable risk-assessment expertise

• It has not been employed as much proven models (e.g. the WF model) and hence may prove difficult to ‘sell’ to the client (esp. where a contract is involved) that this model is controllable and efficient. [More study needs to be done in this regard]