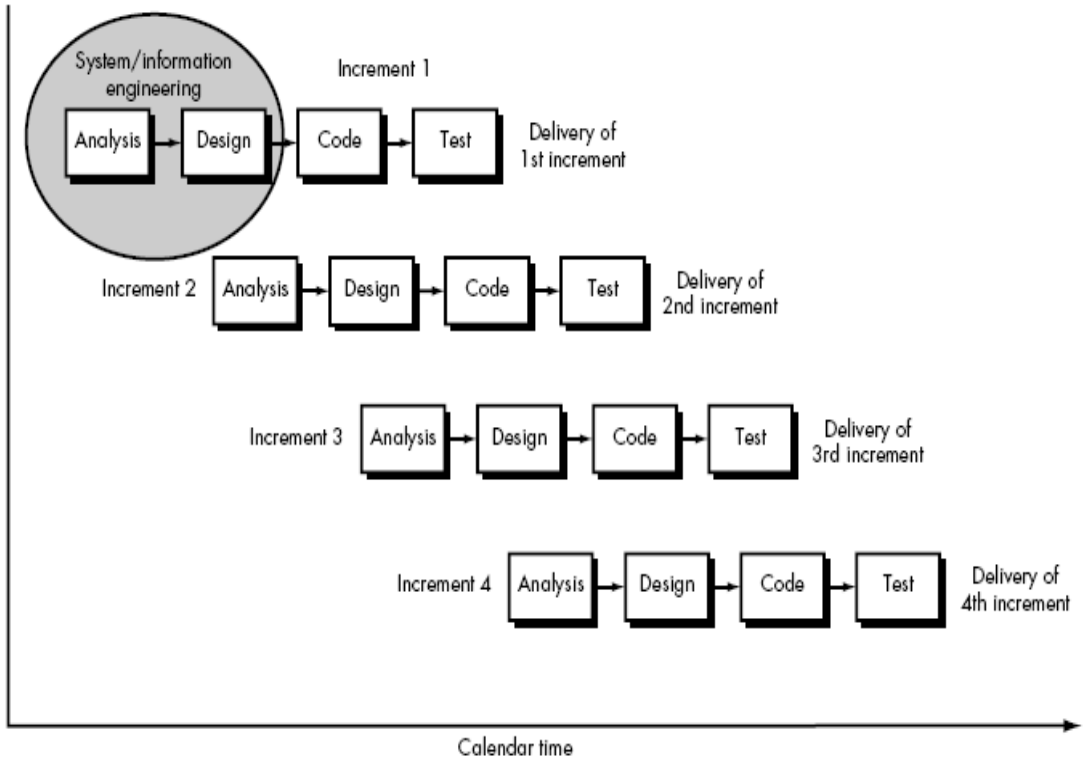
THE INCREMENTAL MODEL

The incremental model combines elements of the linear sequential model (applied repetitively) with the iterative philosophy of prototyping. Referring to Figure below, the incremental model applies linear sequences in a staggered fashion as calendar time progresses. Each linear sequence produces a deliverable "increment" of the software.

When an incremental model is used, the first increment is often a core product. That is, basic requirements are addressed, but many supplementary features (some known, others unknown) remain undelivered. The core product is used by the customer (or undergoes detailed review). As a result of use and/or evaluation, a plan is developed for the next increment. The plan addresses the modification of the core product to better meet the needs of the customer and the delivery of additional features and functionality. This process is repeated following the delivery of each increment, until the complete product is produced.

The incremental process model, like prototyping and other evolutionary approaches, **is iterative** in nature. But unlike prototyping, the incremental model focuses on the delivery of an operational product with each increment. Early increments are stripped down versions of the final product, but they do provide capability that serves the user and also provide a platform for evaluation by the user.

Incremental development is particularly useful when staffing is unavailable for a complete implementation by the business deadline that has been established for the project. Early increments can be implemented with fewer people. If the core product is well received, then additional staff (if required) can be added to implement the next increment.



Advantages

- As product is to be delivered in parts, total cost of project is distributed.
- Limited number of persons can be put on project because work is to be delivered in parts.
- As development activities for next release and use of early version of product is done simultaneously, if found errors can be corrected.

- As functionality is incremented in steps, testing also becomes easy and customers or end users get the chance to see the useful functionality early in the software development life cycle.
- As a result of end user's feedback requirements for successive releases become more clear.
- Risk of failure of a product is decreased as users start using the product early.

Disadvantages

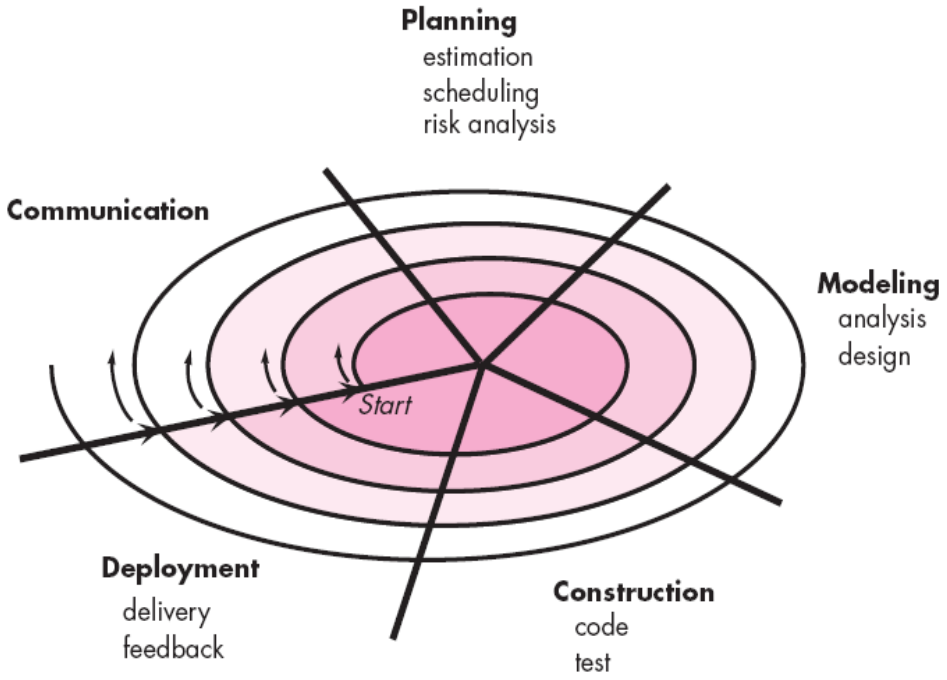
- As product is delivered in parts, total development cost is higher.
- The model requires well defined project planning schedule to distribute the work properly and well defined interfaces to connect modules developed with each phase.
- Testing of modules also results into overhead and increased cost.

THE SPIRAL MODEL

The spiral model is an evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the waterfall model. It provides the potential for rapid development of increasingly more complete versions of the software.

Using the spiral model, software is developed in a series of evolutionary releases.

During early iterations, the release might be a model or prototype. During later iterations, increasingly more complete versions of the engineered system are produced.



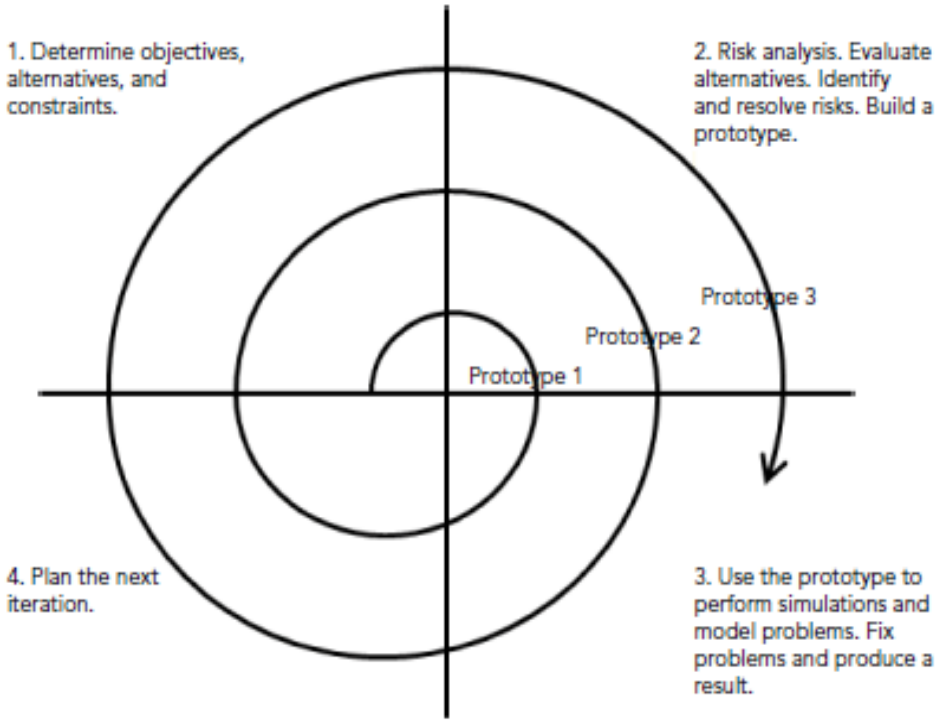
The first circuit around the spiral might result in the development of a product specification; subsequent passes around the spiral might be used to develop a prototype and then progressively more sophisticated versions of

the software. Each pass through the planning region results in adjustments to the project plan.

Cost and schedule are adjusted based on feedback derived from the customer after delivery.

In addition, the project manager adjusts the planned number of iterations complete the software.

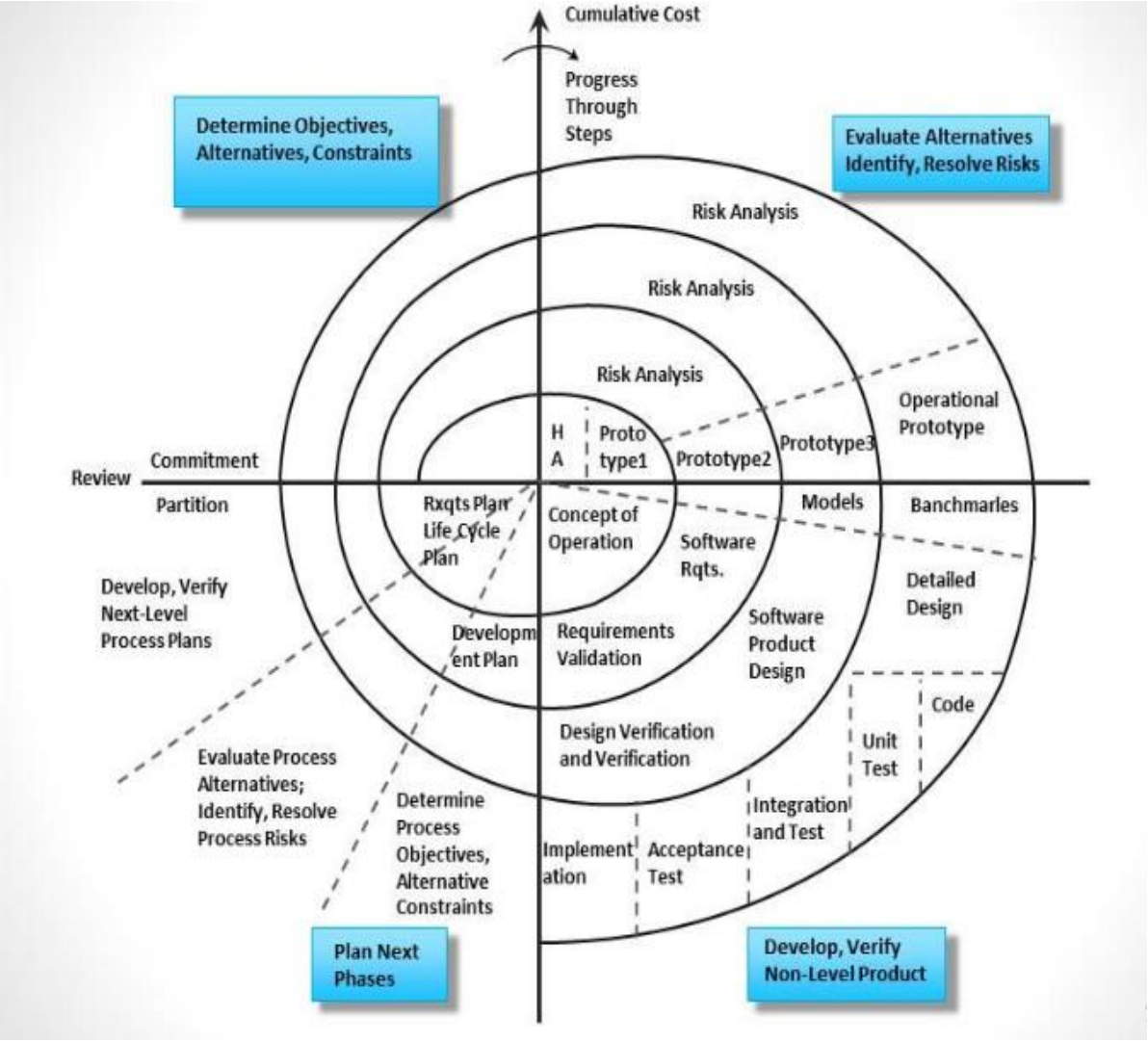
The spiral model uses a risk - driven approach to help project teams decide on what development approach to take for various parts of the project. For example, if you don’ t understand all the requirements, then you might use an iterative approach for developing them.



The spiral process uses four phases.

Each phase of the Spiral Model is divided into four quadrants as shown in the above figure. The functions of these four quadrants are discussed below-

1. **Objectives determination and identify alternative solutions:** Requirements are gathered from the customers and the objectives are identified, elaborated, and analyzed at the start of every phase. Then alternative solutions possible for the phase are proposed in this quadrant.
2. **Identify and resolve Risks:** During the second quadrant, all the possible solutions are evaluated to select the best possible solution. Then the risks associated with that solution are identified and the risks are resolved using the best possible strategy. At the end of this quadrant, the Prototype is built for the best possible solution.
3. **Develop next version of the Product:** During the third quadrant, the identified features are developed and verified through testing. At the end of the third quadrant, the next version of the software is available.
4. **Review and plan for the next Phase:** In the fourth quadrant, the Customers evaluate the so far developed version of the software. In the end, planning for the next phase is started.



Advantage

- Its spiral structure gives stakeholders a lot of points for review and making “go” or “no-go” decisions.
- It emphasizes risk analysis. If you identify and resolve risks correctly, it should lead to eventual success.
- It can accommodate change reasonably well. Simply make any necessary changes and then run through a cycle to identify and resolve any risks they create.

- Estimates such as time and effort required become more accurate over time as cycles are finished and risks are removed from the project.

Disadvantages:

- It’ s complicated.
- Because it’ s complicated, it often requires more resources than simpler approaches.
- Risk analysis can be difficult.
- The complication isn't always worth the effort, particularly for low - risk projects.
- Stakeholders must have the time and skills needed to review the project periodically to make sure each cycle is completed satisfactorily.
- Time and effort estimates become more accurate as cycles are finished, but initially those estimates may not be good.
- It doesn’ t work well with small projects. You could end up spending more time on risk analysis than you'd need to build the entire application with a simpler approach.

For those reasons, the spiral approach is most useful with large high - risk projects and projects with uncertain or changeable requirements.