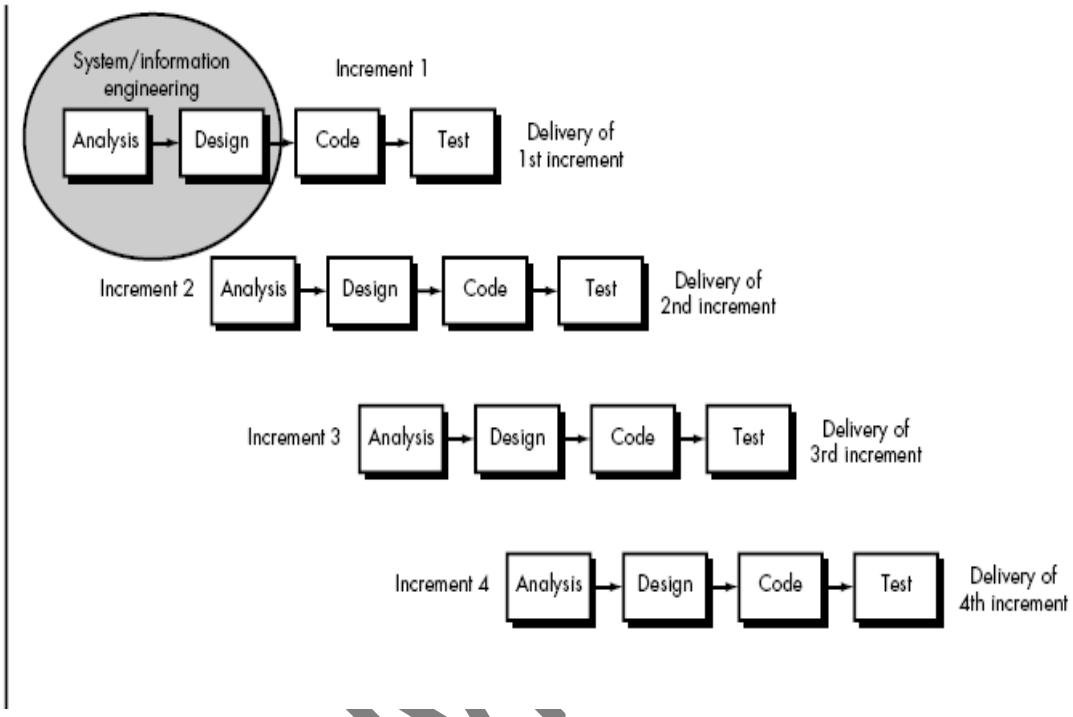
THE INCREMENTAL MODEL

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. For example, word-processing software developed using the incremental paradigm might deliver basic file management, editing, and document production functions in the first increment; more sophisticated editing and document production capabilities in the second increment; spelling and grammar checking in the third increment; and advanced page layout capability in the fourth increment. It should be noted that the process flow for any increment can incorporate the prototyping paradigm.

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.