

The problems with the **Waterfall Model** created a demand for a new method of developing systems which could provide faster results, require less up-front information, and offer greater flexibility. With **Iterative Development**, the project is divided into small parts. This allows the development team to demonstrate results earlier on in the process and obtain valuable feedback from system users. Often, each iteration is actually a mini-**Waterfall** process with the feedback from one phase providing vital information for the design of the next phase. In a variation of this model, the software products which are produced at the end of each step (or series of steps) can go into production immediately as incremental releases.

**Figure 3. Iterative Development (5)**

### Problems/Challenges associated with the Iterative Model

While the **Iterative Model** addresses many of the problems associated with the **Waterfall Model**, it does present new challenges.

- The user community needs to be actively involved throughout the project. While this involvement is a positive for the project, it is demanding on the time of the staff and can add project delay.
- Communication and coordination skills take center stage in project development.
- Informal requests for improvement after each phase may lead to confusion -- a controlled mechanism for handling substantive requests needs to be developed.
- The **Iterative Model** can lead to "scope creep," since user feedback following each phase may lead to increased customer demands. As users see the system develop, they may realize the potential of other system capabilities which would enhance their work.

### Variations on Iterative Development

A number of **Process Models** have evolved from the **Iterative** approach. All of these methods produce some demonstrable software product early on in the process in order to obtain valuable feedback from system users or other members of the project team. Several of these methods are described below.

---

(5) Kal Toth, Intellitech Consulting Inc. and Simon Fraser University, from lecture notes: Software Engineering Best Practices, 1997.