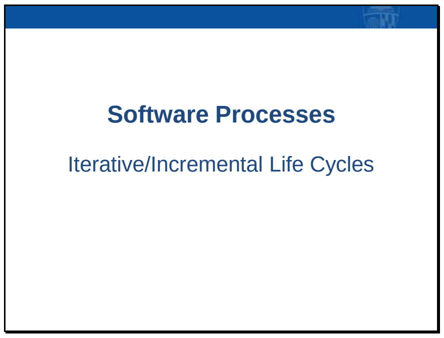
1

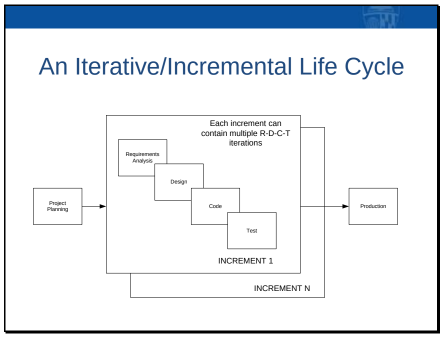


Software Processes

Iterative/Incremental Life Cycles

In this lecture we'll discuss some examples of iterative/incremental life cycles.

2



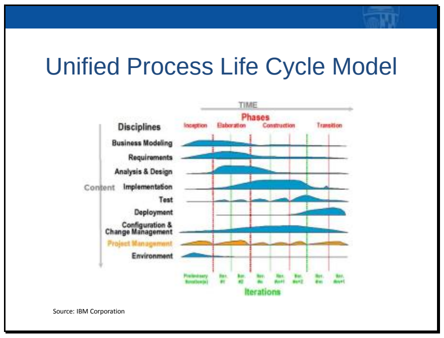
An Iterative/Incremental Life Cycle

Yet another type of life cycle that combines some of the characteristics of the life cycles discussed in earlier lectures is what some refer to as an iterative/incremental life cycle.

Iterative and incremental sometimes mean different things to different people, so it's worthwhile defining them. Iterative means that life cycle phases like requirements, design, and so forth are repeated multiple times. Incremental means that each release contains additional functionality…like in the staged delivery life cycle.

This is a simple schematic of an iterative/incremental life cycle I developed for a bank client. In this model, the planning phase contains a high-level requirements definition activity. The requirements are then bundled into delivery cycles…or increments. During each increment, several iterations of detailed requirements analysis, design, code, and test steps are performed…and then integrated into a delivered product. Each increment adds new or refined functionality. Within each increment, each iteration implements a subset of requirements through the test phase, and each iteration adds new functionality and/or refines existing functionality.

3

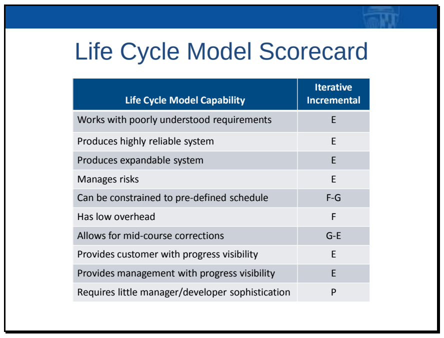


Unified Process Life Cycle Model

Source: IBM Corporation

Here's another example of an iterative/incremental life cycle. It's IBM's Unified Process model. This is a very complex model that provides more of a framework rather than a template for work activities. As can be seen from the diagram, this model provides for overlapping activities. For example, analysis and design for some functionality can be going on at the same time as implementation and test for other functionality…and each delivered set of functionality has most likely gone through several iterations.

4



Life Cycle Model Scorecard

Life Cycle Model Capability	Iterative Incremental
Works with poorly understood requirements	E
Produces highly reliable system	E
Produces expandable system	E
Manages risks	E
Can be constrained to pre-defined schedule	F-G
Has low overhead	F
Allows for mid-course corrections	G-E
Provides customer with progress visibility	E
Provides management with progress visibility	E
Requires little manager/developer sophistication	P

Here's a scorecard for the iterative/incremental life cycle models. The two models I discussed are lumped into one, since they have pretty much the same scores…which are basically the same as the spiral life cycle model…with better adaptability to mid-course corrections.

Like the prototyping and spiral models, iterative/incremental models are useful when the requirements are not well-understood, and give the customer good visibility into project progress…and they work well a little better for projects that have pre-defined schedule constraints (if more requirements work is done up front). They are pretty complex models and require quite a bit of management and developer sophistication and constant management attention. They are also very scalable, and can be used for both small and large projects.

It should be evident from our discussions in this course module that there is no best "one size fits all" life cycle model. Ideally, a life cycle model should be selected based upon the specific project, management, developer, and customer characteristics. And…for many projects there might be multiple life cycle models that are equally appropriate.