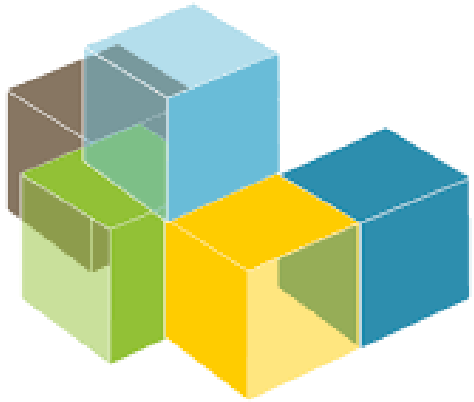
Software Engineering

Lecture 4:
Incremental Model
Component-Based Development

Asst. Instructor

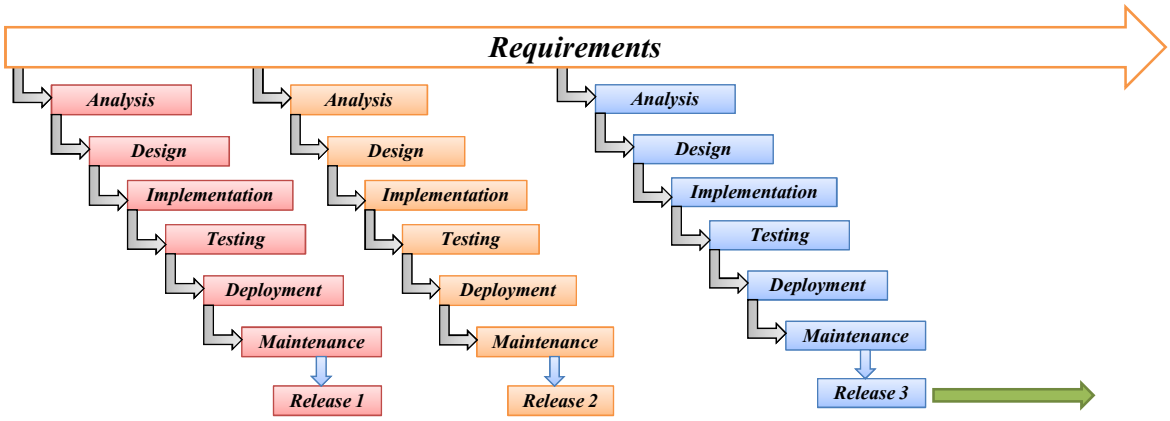
Ali A. Al-Ani



ESPM: Incremental Model

- *Incremental development is based on the idea of developing an initial implementation, exposing this to user comment and evolving it through several versions until an adequate system has been developed.*
- ***The incremental model*** *(also known as Iterative Enhancement model) is an approach to software development where some of the developed increments are delivered to the customer and deployed for use in an operational environment.*
- *This model combines elements of the (**linear sequential model with the iterative philosophy of prototyping**).*

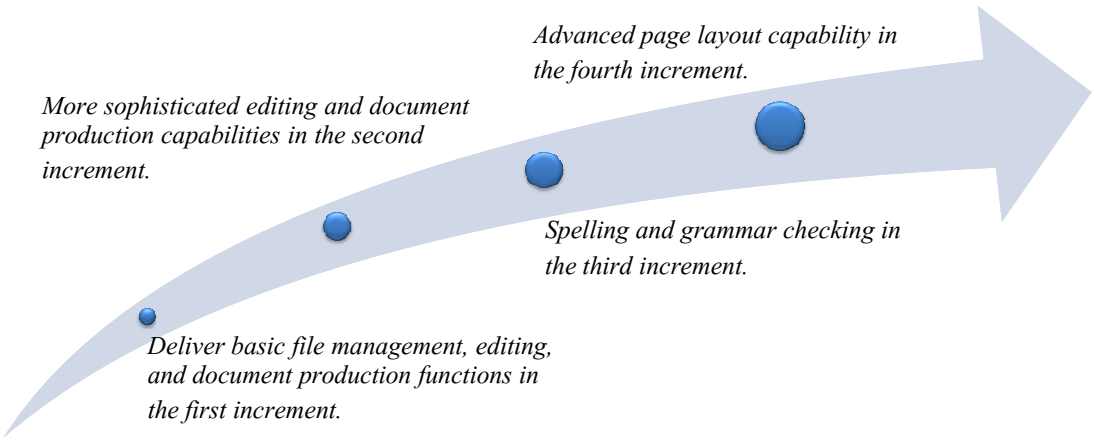
ESPM: Incremental Model



ESPM: Incremental Model

- *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. 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**. **For example, word-processing software developed using the incremental paradigm might:***

ESPM: Incremental Model



ESPM: Incremental Model

- ***Incremental development is particularly useful when:***

1. ***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.*
2. ***Increments can be planned to manage technical risks****. For example, a major system might require the availability of new hardware that is under development. It might be possible to plan early increments in a way that avoids the use of this hardware, thereby enabling partial functionality to be delivered to end-users without inordinate delay.*

ESPM: Incremental Model

- ***Incremental development and delivery should not be used for (suffer from the same problems of uncertain and changing requirements)***

1. *Very large systems where development may involve teams working in different locations.*
2. *Some embedded systems where software depends on hardware development.*
3. *Some critical systems where all the requirements must be analyzed to check for interactions that may compromise the safety or security of the system.*

ESPM: Incremental Model

- ***Advantage of Incremental model***

1. *Customers get usable functionality earlier than with waterfall.*
2. *Getting early feedback improves likelihood of producing a product that satisfies customers.*
3. *Reduces market risk: if customers hate the product, find out early before investing too much effort and money.*
4. *The quality of the final product is better.*
5. *The core functionality is developed early and tested multiple times (during each release)*
6. *Only a relatively small amount of functionality added in each release: easier to get it right and test it thoroughly.*

ESPM: Incremental Model

- ***Disadvantage of Incremental model***

1. *Needs a clear and complete definition of the whole system before it can be broken down and built incrementally.*
2. *Informal requests may lead to confusion and hard to identify common facilities needed in all increments.*
3. *Cost may increase as work done in one iteration may have to be undone later.*

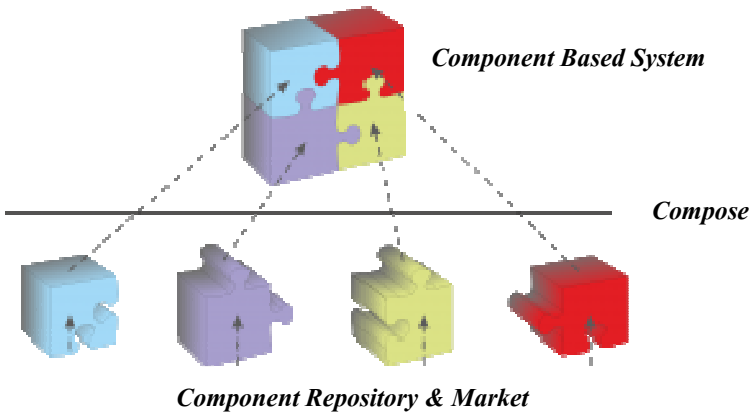
Component-Based Development

- *Specialized process models take on many of the characteristics of one or more of the traditional models presented in the preceding sections.*
- ***Component-Based Development***
- *The component-based development model incorporates many of the characteristics of the spiral model. It is evolutionary in nature, demanding an iterative approach to the creation of software.*
- *The component-based development model constructs applications from prepackaged software components.*

Component-Based Development

- ***In-fact**, now a day, all the simple and complicated systems are **built using the component approach** where some components are **already developed** by some developer and they are stored in the library for their re-use.*
- ***For example** Object-oriented technologies provide the technical framework for a component-based process model for software engineering. The object-oriented paradigm emphasizes the creation of **classes**.*
- *object-oriented classes are **reusable** across different applications and computer-based system architectures.*

Component-Based Development



Component-Based Development

- *Regardless of the technology that is used to create the components, the component-based development model incorporates the following steps (implemented using an evolutionary approach):*

1. ***Available component-based products** are researched and evaluated for the application domain in question.*
2. ***Component integration** issues are considered.*
3. ***A software architecture** is designed to accommodate the components.*
4. ***Components are integrated** into the architecture.*
5. ***Comprehensive testing** is conducted to ensure proper functionality.*

Component-Based Development

- *The component-based development model leads to **software reuse,** and reusability provides software engineers with a number of measurable benefits.*
- *The software engineering team can achieve a **reduction in development cycle time** as well as a **reduction in project cost** if component reuse.*
- *Based on studies of reusability, component assembly leads to a **70 percent reduction in development cycle time**; an **84 percent reduction in project cost.***

Component-Based Development

- ***Advantages of component-based development***

1. ***Flexibility****: runtime components can work independently if properly designed and they are less dependent on the environment*
2. ***Reuse****: Once developed, it can be used everywhere regarding the programming language and OS. But domain engineering should be kept in mind*
3. ***Easy to maintain****: because ideally the functionality is implemented once*
4. ***Reduce the cost****: development cost is much lower*
5. ***Reduce the time****: Lesser time required to build the software*

15

Component-Based Development

- ***Disadvantages of component-based development:***

1. ***Difficulty to build:*** *it is very difficult to build the environment that is fitted to component.*
2. ***Concept of reuse:*** *Ideally speaking, standards are needed regarding middleware in which the component is supposed to work. (Middleware: A communication layer which enables components to interact with higher level component in a network).*

16