Lecture: 9
Software Prototyping

Verification answers the question, "Did we build the right system?" while validations addresses, "Did we build the system right?"

Prototyping is the rapid development of a system.
Software prototyping refers to the process of visualizing a software product before it has been created. Creating software from scratch requires a great investment in the form of time, money and effort. That is why most clients prefer to have a visual prototype developed before work is put into the development of the actual product.
The prototype acts as a "model closely replicating the appearance and sometimes the functionality of the product that the client has in mind.

Why Software Prototyping?
The principal use is to help customers and developers understand the requirements for the system.

- **Requirements Elicitation:** users can experiment with a prototype to see how system supports their work.
- **Requirements Validation:** the prototype can reveal errors and omissions in the requirements.

Prototyping lets you gain valuable feedback from the users before the final product is delivered to the clients. While you run your prototypes, there are lots of inputs that can be undertaken which improves the overall workflow of the process, too.

Prototyping can be considered as a risk reduction activity which reduces requirements risks.

Prototyping Benefits:
- Misunderstanding between software users and developers are exposed.
- Missing services may be detected and confusing services may be identifies.
- A working system is available early in the process
- The prototype may serve as a basis for deriving a system specification.
- The system can support user training and system testing.
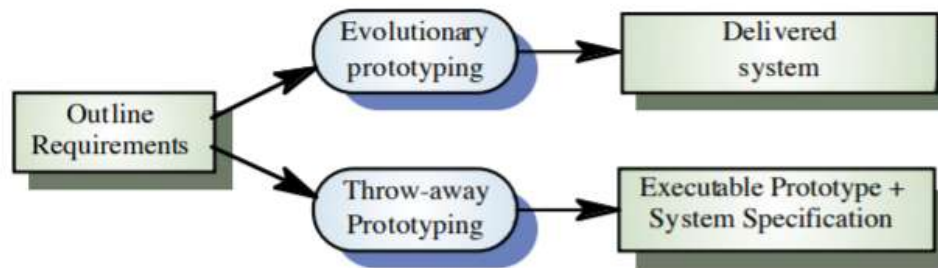
Types of Prototyping:

Two major types of approaches to prototyping are:

- **Evolutionary Prototyping:** an initial prototype is produced and refined through a number of stages to the final system.
- **Throw-away Prototyping (rapid prototyping):** a practical implementation of the system is produced to help discover requirements problems and then discarded. The system is then developed using some other development process.

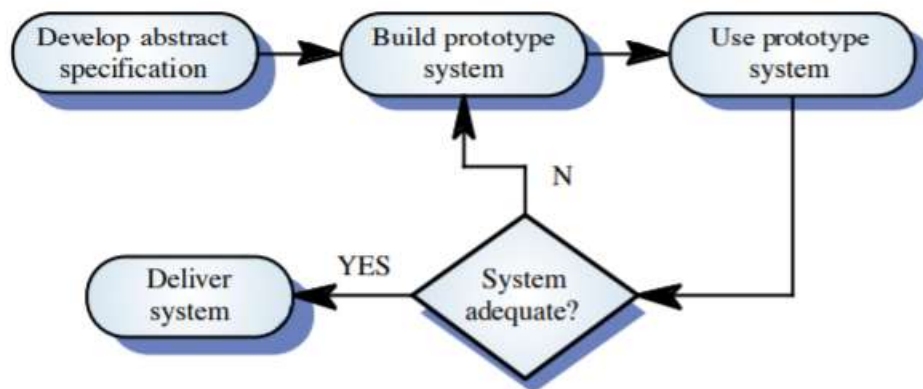**Evolutionary Prototyping (breadboard prototyping):**
The objective of evolutionary prototyping is to deliver a working system to end-users.
The development starts with those requirements which are best understood.

Evolutionary prototyping is used for the systems where the specification cannot be developed in advance e.g. AI systems and user interface systems.
It is based on techniques which allow rapid system iterations.
The system is developed as a series of increments that are delivered to the customer.

Advantages of Evolutionary Prototyping:
- Accelerated delivery of the system: Rapid delivery and deployment are sometimes more important than functionality or long-term software maintainability.
- User engagement with the system: Not only is the system more likely to meet user requirements, they are more likely to commit to the use of the system.

Disadvantages of Evolutionary Prototyping:
- Continual change tends to corrupt system structure so long-term maintenance is expensive.
- Specialist skills are required which may not be available in all development teams.

Throw-away Prototyping:

The objective of throw-away prototyping is to validate or derive the system requirements. i.e. to ensure that the system requirements are validated and that they are clearly understood.

The prototyping process starts with those requirements which are poorly understood. It is used to reduce requirements risk. The prototype is developed from an initial specification, delivered for experiment then discarded.

The prototype is simply there to aid understanding and reduce the risk of poorly defined requirements. The full system is being developed alongside the prototypes and incorporated the changes needed.

The throw-away prototype should not be considered as a final system as
- some system characteristics may have been left out.
- there is no specification for long-term maintenance.
- the system will be poorly structured and difficult to maintain.