IDDS

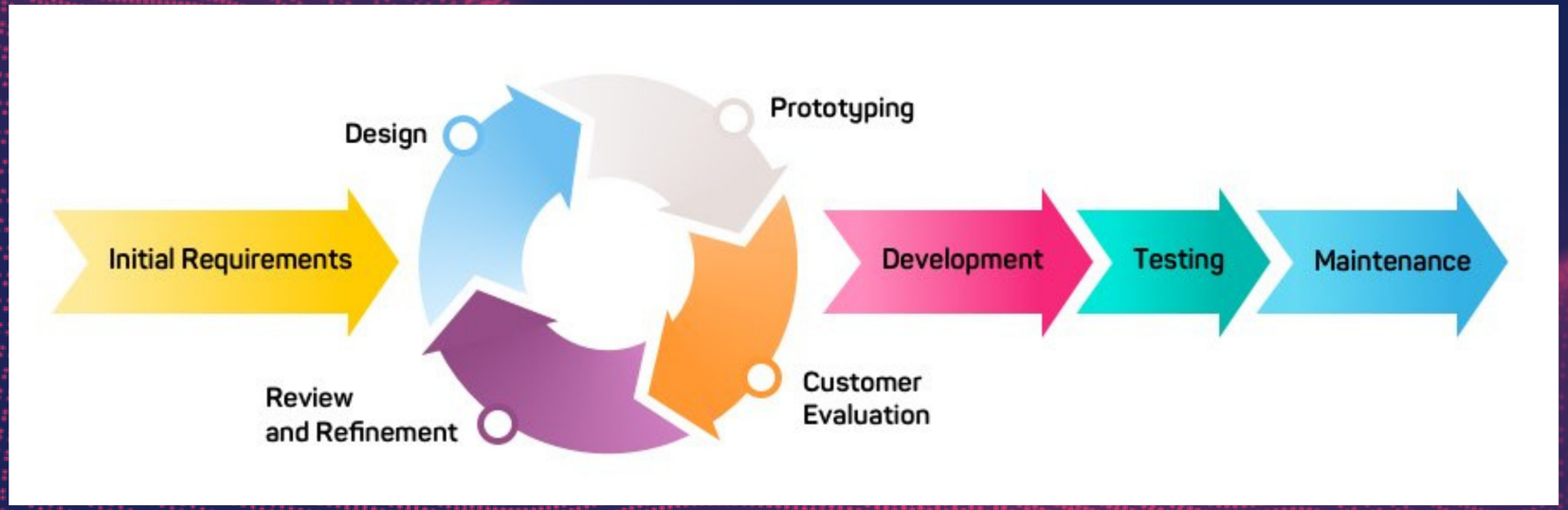
Prototype MODEL

Presented by : Gehad Safwat

01 - Definition

- The Prototyping Model involves creating an initial version of the software with limited functionality and features. This prototype is used for gathering user feedback, refining requirements, and improving the final product.

- A prototype usually turns out to be a very crude version of the actual system, possible exhibiting limited functional capabilities, low reliability, and inefficient performance as compared to actual software.



02 - Pros

- Early user involvement and feedback.
- Identification and resolution of design flaws.
- Enhanced communication
- Reduced development time by addressing issues early on.
- Rapid exploration of potential solutions.

03 - Cons

- Difficulty in managing evolving requirements.
- Time-consuming and costly if not properly planned.
- User confusion if the prototype differs significantly from the final product.
- Require extensive customer collaboration

04 - Use Cases

- The Prototyping Model should be used when the requirements of the product are not clearly understood or are unstable.
- The prototyping model can also be used if requirements are changing quickly.
- This model can be successfully used for developing user interfaces , high-technology software-intensive systems, and systems with complex algorithms and interfaces.
- The prototyping Model is also a very good choice to demonstrate the technical feasibility of the product.

IDDS

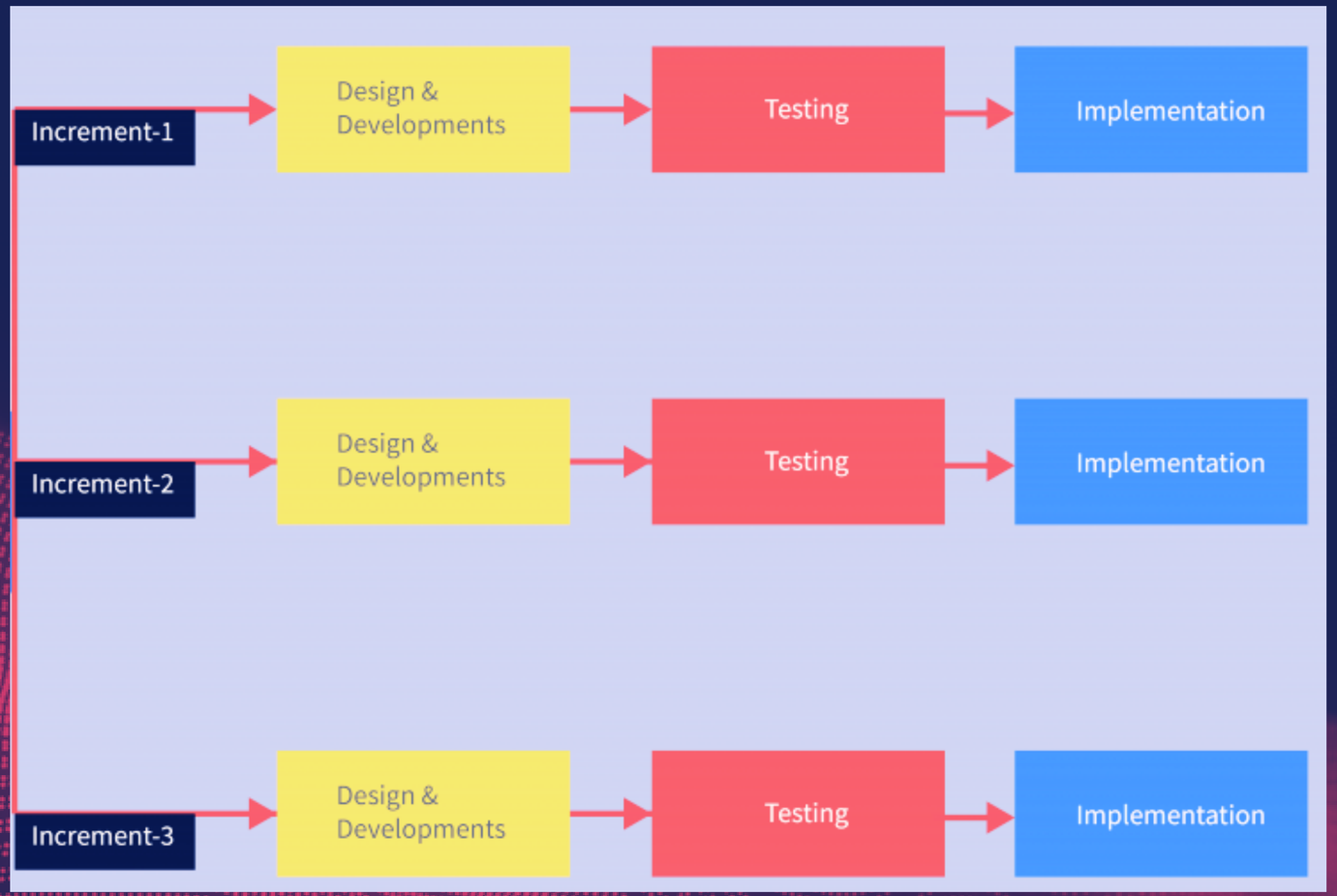
Incremental MODEL

Presented by : Gehad Safwat

01 - **Definition**

- The Incremental Model involves dividing the software development process into multiple increments or phases. Each increment delivers a subset of functionality , and requirements divided into multiple standalone modules of the SDLC.

- **Every subsequent release of the module adds function to the previous release. The process continues until the complete system achieved.**



02 - Pros

- Errors are easy to be recognized.
- Easier to test and debug
- More flexible.
- Simple to manage risk because it handled during its iteration.
- The Client gets important functionality early.

03 - Cons

- Requires careful planning and coordination of increments.
- Increased complexity due to interdependencies.
- Challenges with integration and compatibility.
- Potential for higher costs if increments are not well-defined.

04 - Use Cases

- When the requirements are superior.
- A project has a lengthy development schedule.
- When Software team are not very well skilled or trained.
- When the customer demands a quick release of the product.
- You can develop prioritized requirements first.

IDDS

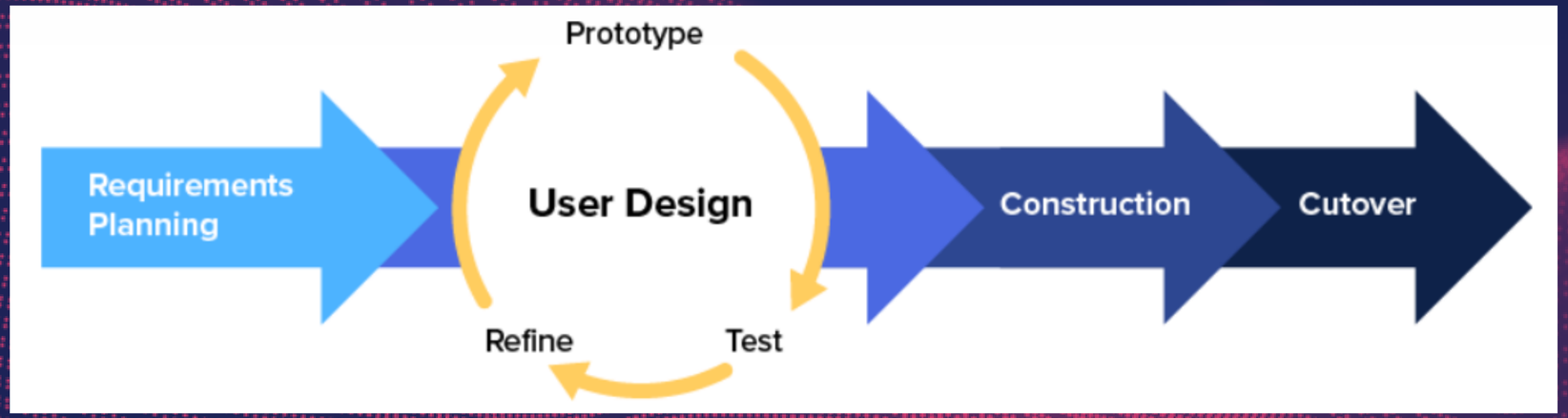
RAD MODEL

Presented by : Gehad Safwat

01 - Definition

- The Rapid Model emphasizes quick development cycles and frequent iterations. It involves breaking down the project into smaller modules and delivering functional components in short timeframes.

- RAD (Rapid Application Development) is a concept that products can be developed faster and of higher quality through:
  a. Gathering requirements using workshops or focus groups
  b. Prototyping and early, reiterative user testing of designs
  c. The re-use of software components



02 - Pros

- This model is flexible for change.
- Each phase in RAD brings highest priority functionality to the customer.
- It reduced development time.
- It increases the reusability of features.
- Reduced risk

03 - Cons

- Active and continuous user involvement required.
- Challenges with large-scale projects.
- It requires highly skilled designers.
- Small Apps aren't compatible with RAD.

04 - Use Cases

- When the system should need to create the project that modularizes in a short span time (2-3 months).
- When the requirements are well-known.
- When the technical risk is limited.
- When there’s a necessity to make a system, which modularized in 2-3 months of period.
- It should be used only if the budget allows the use of automatic code generating tools.