Software development stages: Requirements analysis, system design, program design, coding, unit, integration, system testing, delivery, maintenance

A generic process framework for software engineering defines five framework activities:

* Communication (project initiation, req. gathering).
* Planning (estimating, scheduling, tracking)
* Modeling (analysis, design)
* Construction (code, test)
* Deployment (delivery, support, feedback)

Process Flows:

1. Linear: all 5 activities in sequence.
2. Iterative: can repeats one or more of the activities before proceeding to the next.
3. Evolutionary: executes the activities in a “circular” manner. (repeats all activities in order, if needed).
4. Parallel: executes one or more activities in parallel with other activities.

Task set (Communication):

1. Inception: goals and objectives are outlined (understanding initial req.).
2. Elicitation: gathering information from stakeholders.
3. Elaboration: analyzing and creating comprehensive understandings.
4. Negotiation: reconciling conflicting requirements and priorities among stakeholders.
5. Specifications: formal documentation of detailed project req.
6. Validation: documented requirements are reviewed and verified to ensure they meet the stakeholders' expectations.
7. Waterfall Model:
   1. Classis
      1. Well understood problem, minimal or no changes in requirement.
      2. No way to cater change in req.
      3. Long wait for final product, lots of documentation, suitable for large
   2. with prototyping:
      1. Partially developed product.
      2. Design prototype: developer will know how it looks like.
      3. Interface prototype: user understands how it will look like.
   3. The V model:
      1. Uses unit testing to verify procedural design.
      2. Uses integration testing to verify architectural (system) design.
      3. Uses acceptance testing to validate the requirements.
8. Phased Development:
   1. Two systems functioning in parallel.
      1. the production system (release *n*): currently being used.
      2. the development system (release *n+1*): the next version.
   2. **Incremental development:** starts with small functional subsystem and adds functionality with each new release.
   3. **Iterative development:** delivers a full system at the very beginning and then changes the functionality of each subsystem with each new release.
9. Prototyping mode:
   1. Allow repeated investigations, reduces risk and uncertainties.
   2. May cause delay, work is lost in throwaway prototype.
10. Spiral Model:
    1. Pros:
       1. There is continuous customer involvement.
       2. Development risks are managed.
       3. It is suitable for large, complex projects.
       4. It works well for extensible products.
    2. Cons:
       1. Risk analysis failures can doom the project.
       2. The project may be hard to manage.
       3. It requires an expert development team.
11. UP:
    1. Project will be **use-case** driven and modeled using UML.
    2. Highly iterative life cycle.
    3. Inception, elaboration, construction, transition, production.
    4. Five UP phases do not occur in a sequence, but rather with staggered concurrency.
12. Rapid Application Development:
    1. Linear sequential software development.
    2. -Business, data, process-modelling, application generation, testing.
    3. RAD process enables a development team to create a fully functional system within a concise time period.
    4. When the system should need to create the project that modularizes in a short span time (2-3 months).
    5. When the requirements are well-known.
    6. When the technical risk is limited.
    7. Progress needs to be made visible.
13. Agile: iterative, early delivery, invest time in producing working software rather than in producing comprehensive documentation.
    1. XP:
       1. User stories, customer assign a priority value and agile team assigns a cost (in weeks).
       2. CRC cards.
       3. pair programming.
       4. Unit testing of user stories.
    2. Scrum:
       1. Fixed length iteration-sprints-(2-4weeks).
       2. shippable **product increment** with every iteration.
       3. Code increment: union of all product backlog items completed in previous sprints and all backlog items to be completed in the current sprints.
       4. Project velocity: average of all (stories completed in a sprint \* story points) previous sprints.
    3. Kanban:
       1. Columns (where cards are placed). WIP (card limit for a column).
       2. Lead time: new card arrival in workflow -> departure from workflow.
       3. Cycle time: new card arrival in working state -> when card is ready to release.
       4. Throughput: WIP/cycle time.

* Problem based Estimation: *S* = (*sopt* + 4*sm* + *spess*)/6
* aaa