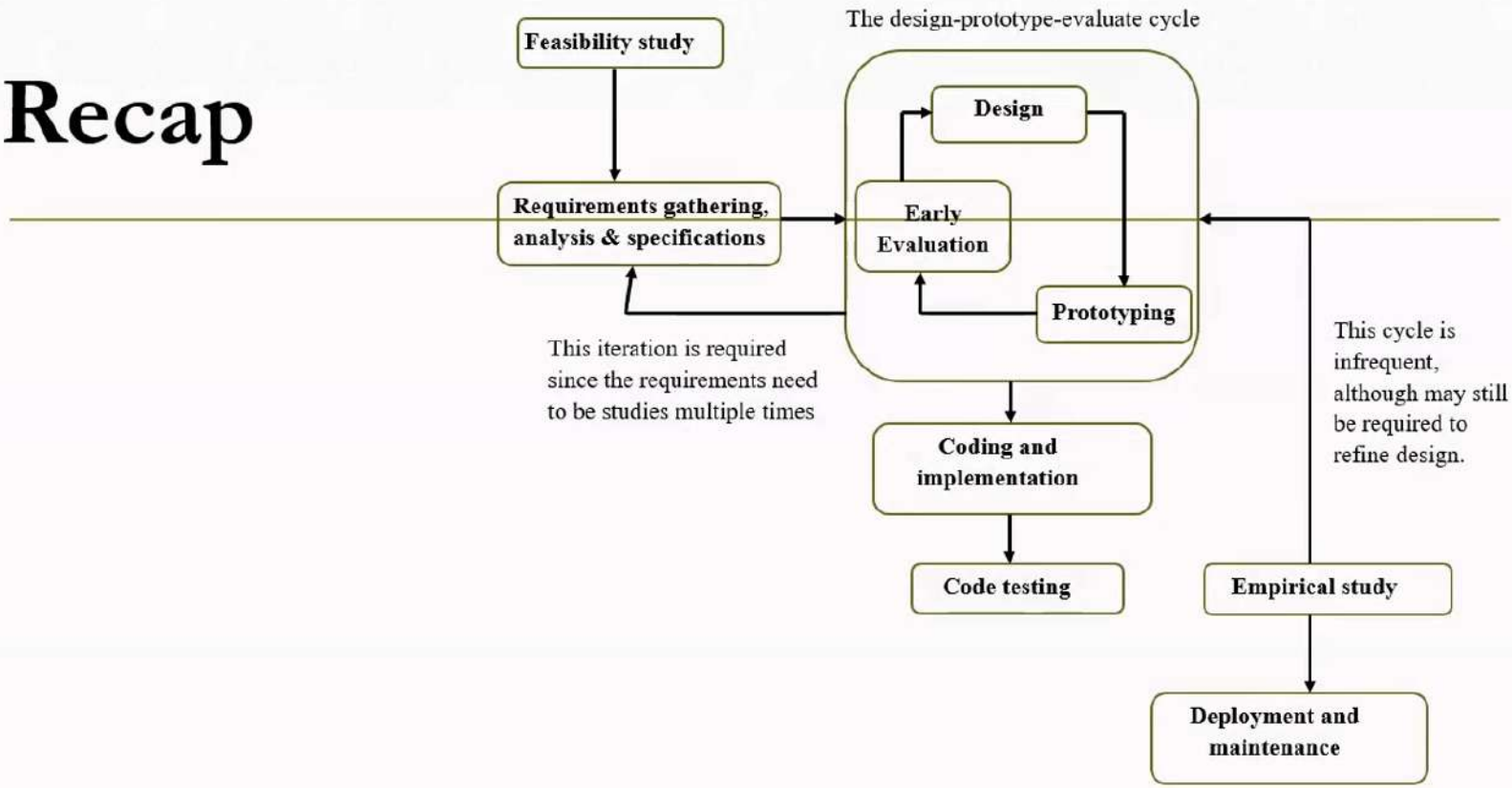
Recap



Recap

- Requirement stage – functional and usability (CI)
- Design– design guidelines (usability), DFD+ER, OOD+UML
- Prototype (including evaluation) – CW, HE
- Coding and code testing – black-box, white-box, integration + system testing
- Usability testing

Deployment & Maintenance

- Important (not much to discuss)
- Maintenance – includes updates, bug-fixes etc

Project Management

Why

- Important topic – particularly for large projects
- Prior estimation of manpower, resources and cost preferable

Project Planning

- Involves estimation of
  - Project size
  - Cost
  - Duration
  - Effort

Project Planning

- Based on such estimations, a project manager needs to
  - Create staff organization and staffing plan
  - Plan/arrange for other resources
  - Identify risks and plan for mitigation
  - Make plan for quality assurance (QA)
  - …

Project Size Estimation

- Lines of Code (LoC) – total number of lines of instructions (excluding comments)
  - Difficult to estimate at the beginning
  - Modular hierarchy can help – coupled with prior experience

Project Size Estimation

- Function Point (FP) metric
  - Project size related to the number of functions supported by the system (along with some other things)

COCOMO

- Stands for Constructive Cost Estimation Model
- Proposed by Boehm [1981]
- Can be used for cost, duration and effort estimation

COCOMO

- According to Boehm, software cost estimation should be done through three stages
  - Basic COCOMO,
  - Intermediate COCOMO
  - Complete COCOMO

Basic COCOMO

- Gives estimate with following equations
  - Effort = a x $(KLoC)^{a1}$ PM [PM = person months, KLoC = Kilo LoC]
  - Tdev = b x $(Effort)^{b1}$ Months
- Constants: a, a1, b, b1
- Takes different values for different types of projects

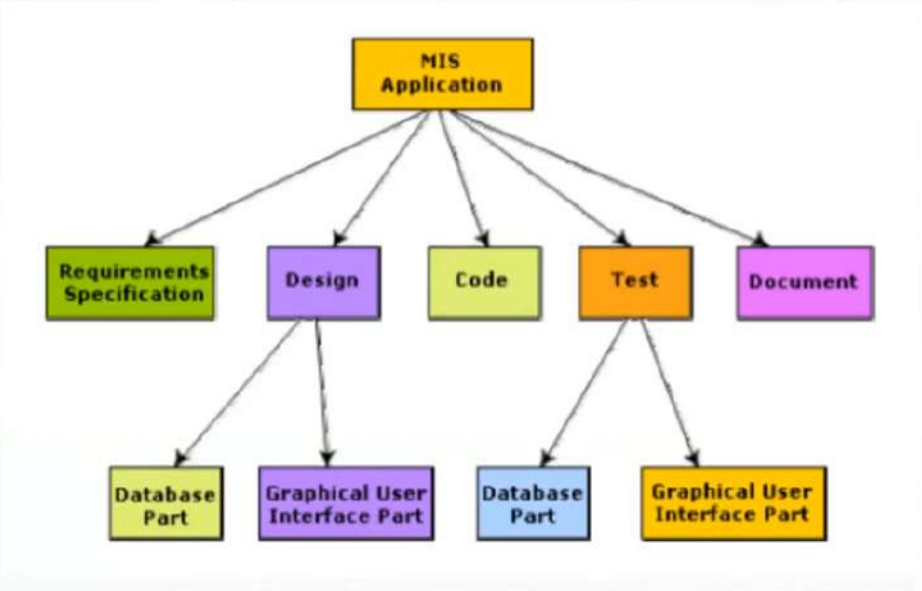
Project Types & Constant Values

- **Organic:** if project deals with well understood application program, team size is reasonably small with experienced team members [a=2.4, a1=1.05, b=2.5, b1=0.38]
- **Semidetached:** team consists of a mixture of experienced and inexperienced staff [a=3.0, a1=1.12, b=2.5, b1=0.35]
- **Embedded:** if software strongly coupled to complex hardware, or if stringent regulations on operational procedures exist [a=3.6, a1=1.2, b=2.5, b1=0.32]

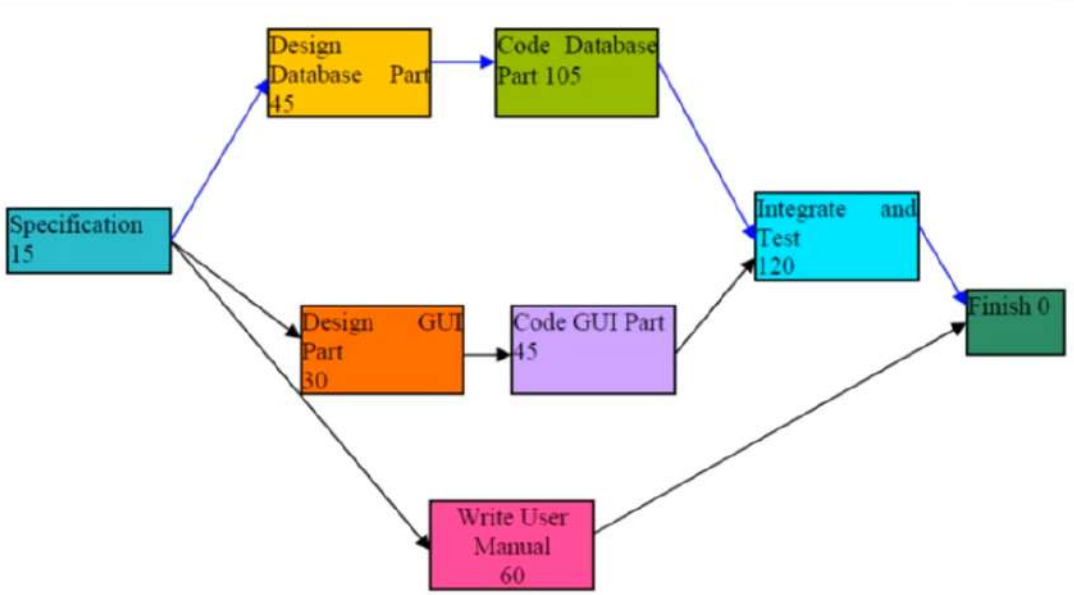
Scheduling

- Identify tasks and subtasks
- Determine dependencies (activity network)
- Allocate resources (Gantt chart)
- Time schedule (PERT charts)

Task Division (Hierarchy)



Activity Dependency (Activity Network)

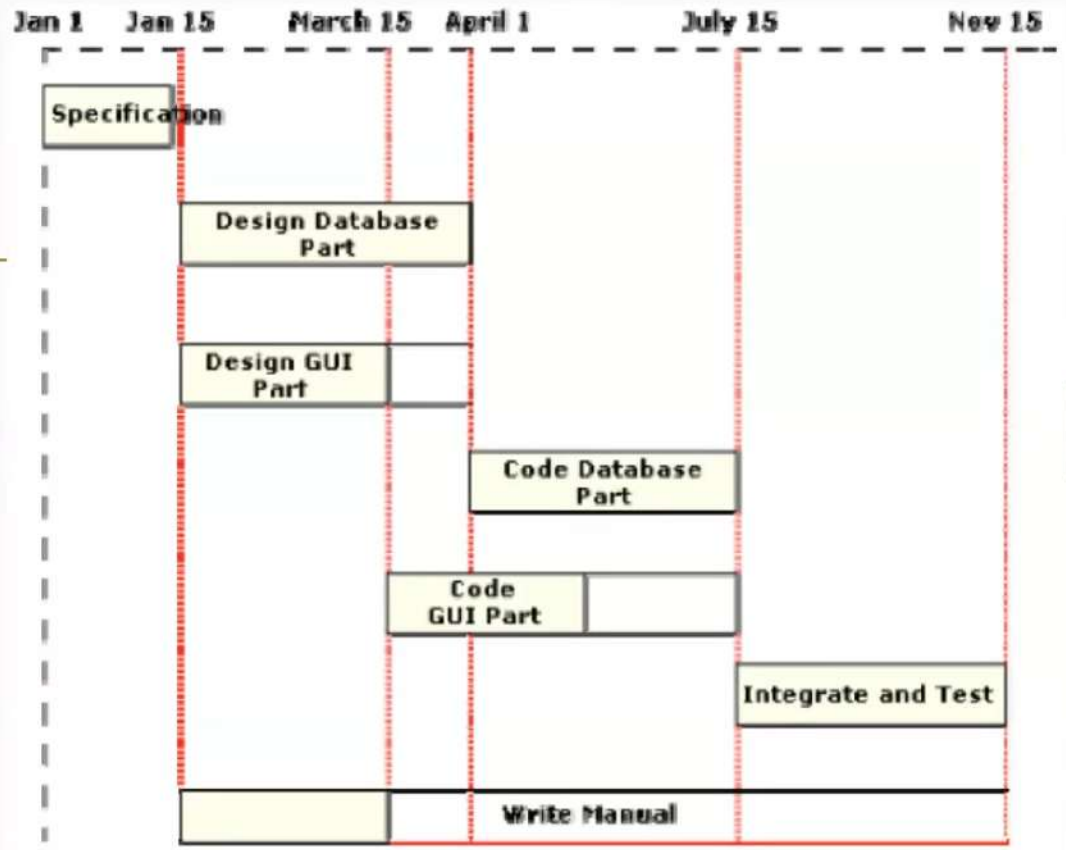


Critical Path Analysis

- Minimum time (MT) to complete project is the maximum of all paths from start to finish
  - Can be estimated using CPM (Critical Path Method)

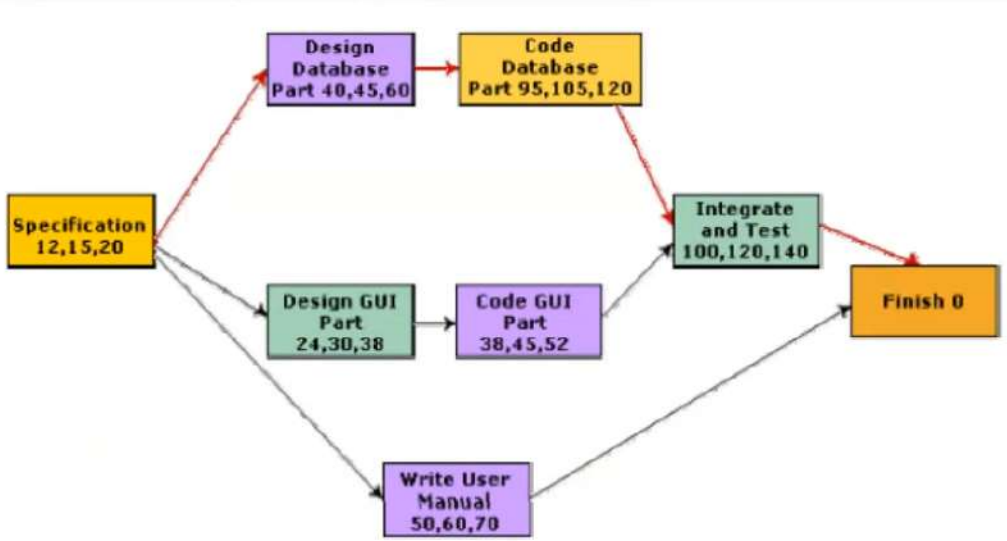
Gantt Chart

- A special type of bar chart (named after its developer Henry Gantt) where each bar represents an activity
  - Bar length proportional to duration planned for activity



PERT (Project Evaluation and Review Technique) Chart

- Consists of a network of boxes and arrows
- Boxes represent activities and arrows represent task dependencies
- Boxes usually annotated with pessimistic, likely, and optimistic estimates for every task



Agile Development

SDLCs

- We have seen two (water-fall, ISLC)
- There are others (spiral, iterative, evolutionary …)
- These are all "traditional" models
  - Stages are mostly the same with variations in execution sequence
  - Equally rigorous with strict requirements at each stage

Problem with Traditional SDLCs

- Rigidity – may lead to time and cost overrun (and even non-acceptance by customer)

Why Problematic

- Due to strict requirements at each stage

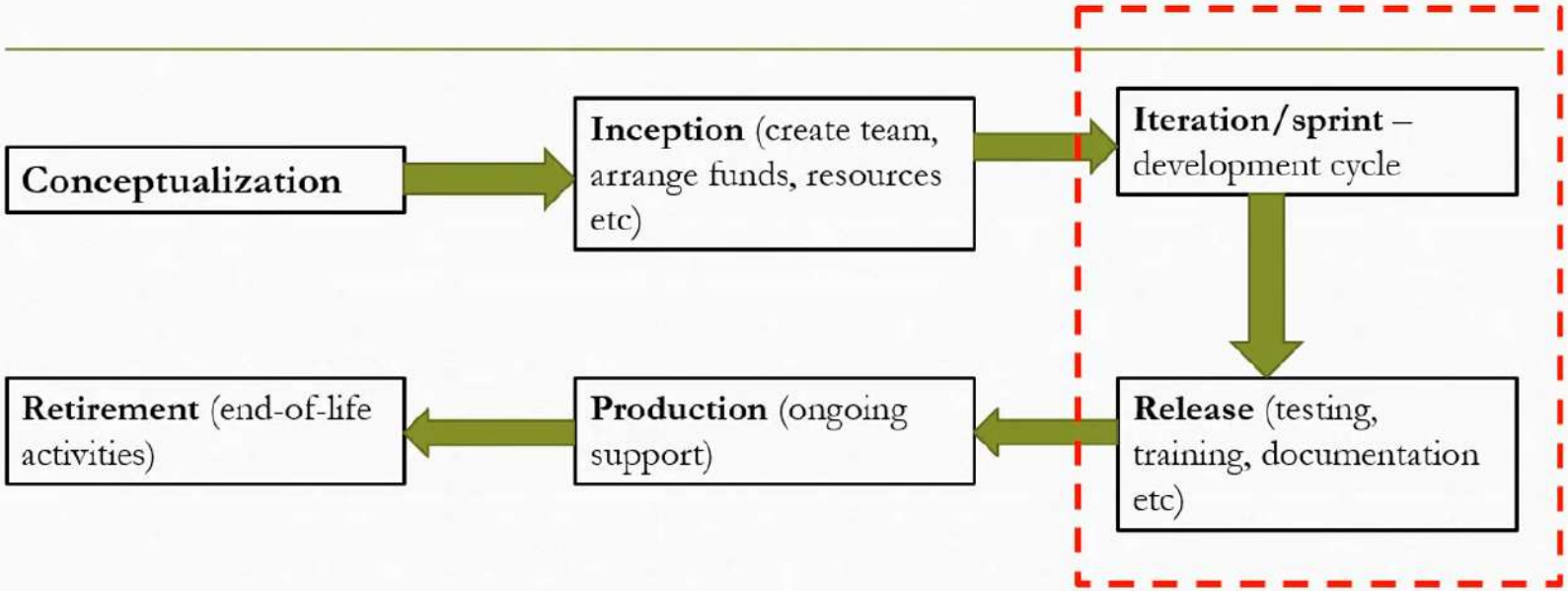
Why Problematic

- Due to strict requirements at each stage
  - All requirements to be identified first
  - Every stage detailed documentation (lots of documentations!)
  - Necessary to implements ALL requirements BEFORE deployments …

Agile Development - Idea

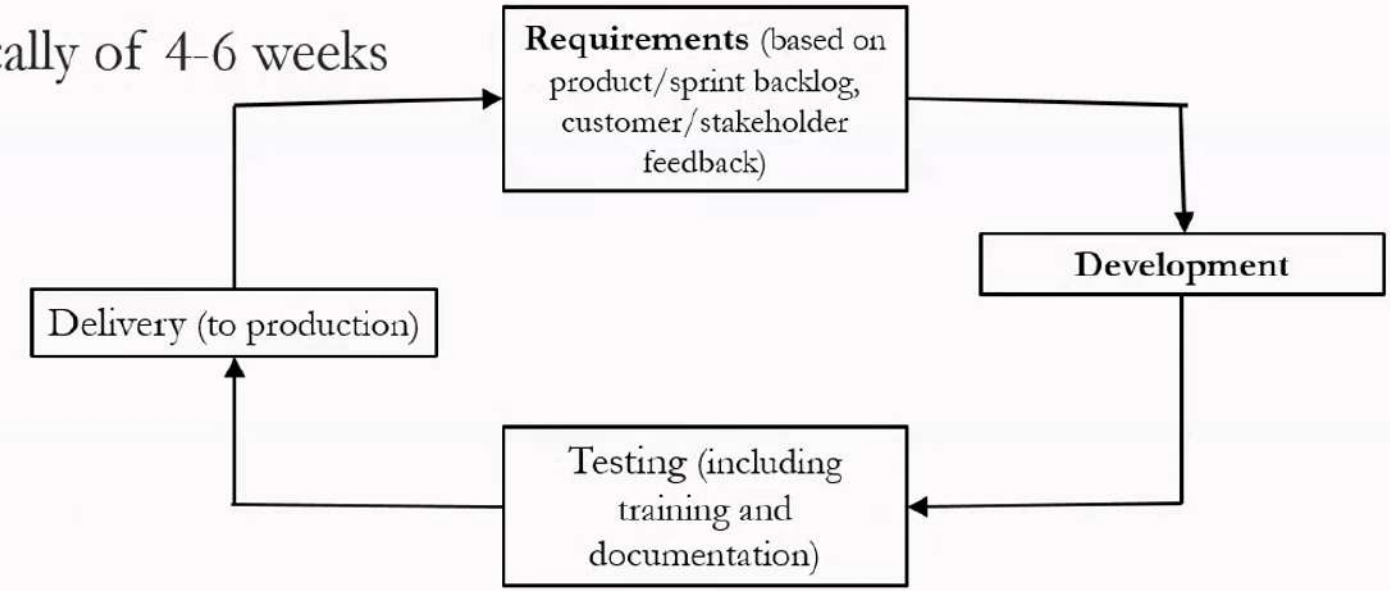
- Agile (fast) development – ensure quick turn-around of projects
  - Eliminate unnecessary things and documentations
  - Start small (don't try to do everything from start)

Agile Process Flow (A-SDLC)



Agile Iteration Workflow

- Typically of 4-6 weeks



How to Make it Work

- Daily meetings (of the development team)
- Live demo – after each iteration
- Feedback – regularly collect from stakeholders/customers and share with team before next sprint
- Remain agile – be open to make changes based on feedback

Some Agile Methods

- Scrum
- Extreme programming (XP)
- Disciplined agile delivery (DAD)
- Dynamic systems development method (DSDM)
- …

Further Reading

- Agile manifesto (https://agilemanifesto.org/)
- 12 agile development principles (https://www.agilealliance.org/agile101/12-principles-behind-the-agile-manifesto/)

The End!!