Université de Paris

Génie Logiciel
Elements of a software project

Sylvain Lobry

25/09/2023

Elements of a software project

Before we start…

Elements of a software project

Wooclap

https://www.wooclap.com/L3GL233

Elements of a software project

What is a software project?

Definition:

A software project is the complete procedure and activities to achieve an intended software product.

Elements of a software project

What is a software project?

Definition:

A software project is the complete procedure and activities to achieve an intended software product.

- Functional objectives
- Technical specifications

Elements of a software project

What is a software project?

Definition:

A software project is the complete procedure and activities to achieve an intended software product.

- Functional objectives
- Technical specifications
- Definition of the scope
- Planning
- Development
- Risk analysis
- Management
- Monitoring

Elements of a software project

Some people of a software project

- Maître d'ouvrage (= project owner or client): stakeholder that benefits from the project's results
  - Identifies the needs
  - Defines the goals
  - Finances the project
  - Oversees the project's planning and realization
  - Take general decisions if needed
- Maître d'oeuvre (= contractor): proposes and implements a solution to realize the project

Elements of a software project

Definition of the scope

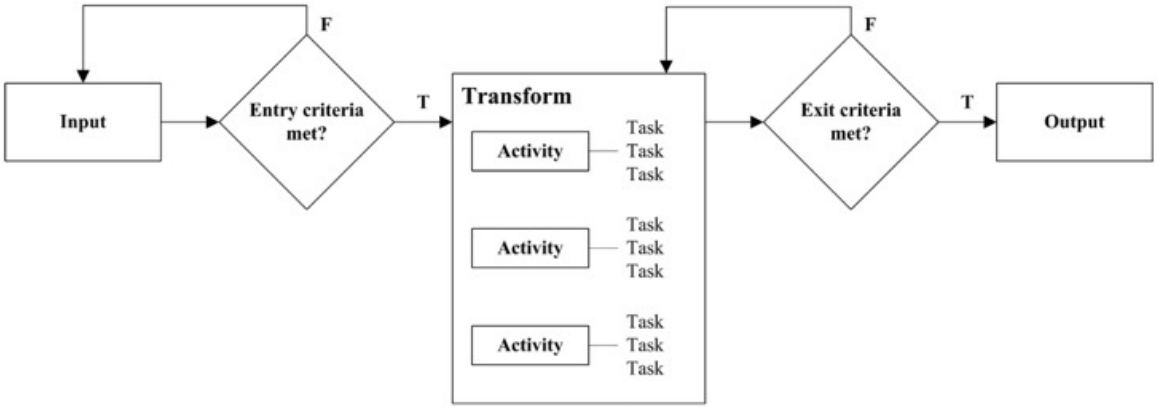
- Scope must be a balance of:
  - Time
  - Cost
  - Quality

Elements of a software project

Software processes

Defintion:

a software process is a set of interrelated activities and tasks that transform input work products into output work products, SWEBOK v3



Elements of a software project

Main software activities

- List of activities to be conducted during a software project
- Each can be seen as a process

Elements of a software project

Main software activities

- Objectives definition
- Requirement analysis
- Feasibility analysis
- Requirements specifications
- Design
- Implementation
- Unit testing
- Integration
- Validation
- Deployment
- Maintenance

Elements of a software project

Main software activities

- Objectives defintion: understanding what will be the usage (in its context) of the software

Elements of a software project

Main software activities

- Requirement analysis: determining the needs of the stakeholders

Elements of a software project

Main software activities

- Feasability analysis: determining which outcomes can be achieved in the specific context of the project

Elements of a software project

Main software activities

- Requirements specifications: formalization of the requirements in the form of a document that can be systematically reviewed, evaluated and approved

Elements of a software project

Main software activities

- Design: precise definition of the components of the software based on the requirements

Elements of a software project

Main software activities

- Implementation: building-up the program following the design and instructions.

Elements of a software project

Main software activities

- Unit testing: verifying individually that each component of your software answer its specification.

Elements of a software project

Main software activities

- Integration: connection of the different sub components of the program.

Elements of a software project

Main software activities

- Validation: validation that the software, as a whole, is answering the initial objectives and expectations from the customer.

- Not to be confused with verification!
- Verification = analysis (often without executing code) during development period to check whether a specific requirement is met

Elements of a software project

Main software activities

- Deployment: activities to make the software available for use.

Elements of a software project

Main software activities

- Maintenance: to modify the application after its deployment to fix bugs, improve performance or improve functionalities

Elements of a software project

Main software activities

- Objectives definition
- Requirement analysis
- Feasibility analysis
- Requirements specifications
- Design
- Implementation
- Unit testing
- Integration
- Validation
- Deployment
- Maintenance

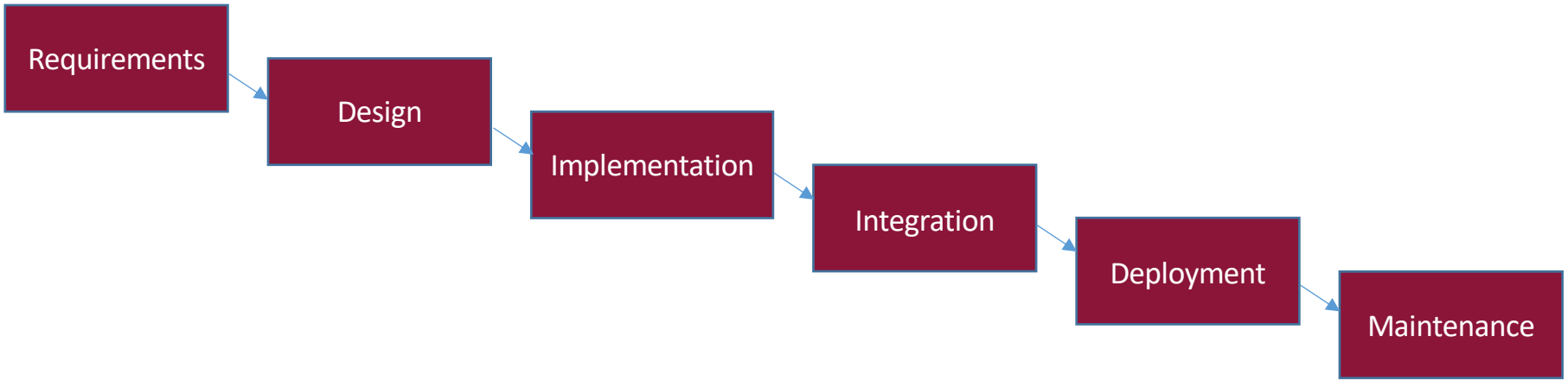
Software Development Life Cycle

- A software Development Life Cycle (SDLC) puts the different processes in order
- Also know as Software process model
- Chosen at the start of the project
- Brings discipline to software development
- 4 SDLC models today

Software Development Life Cycle

Waterfall model

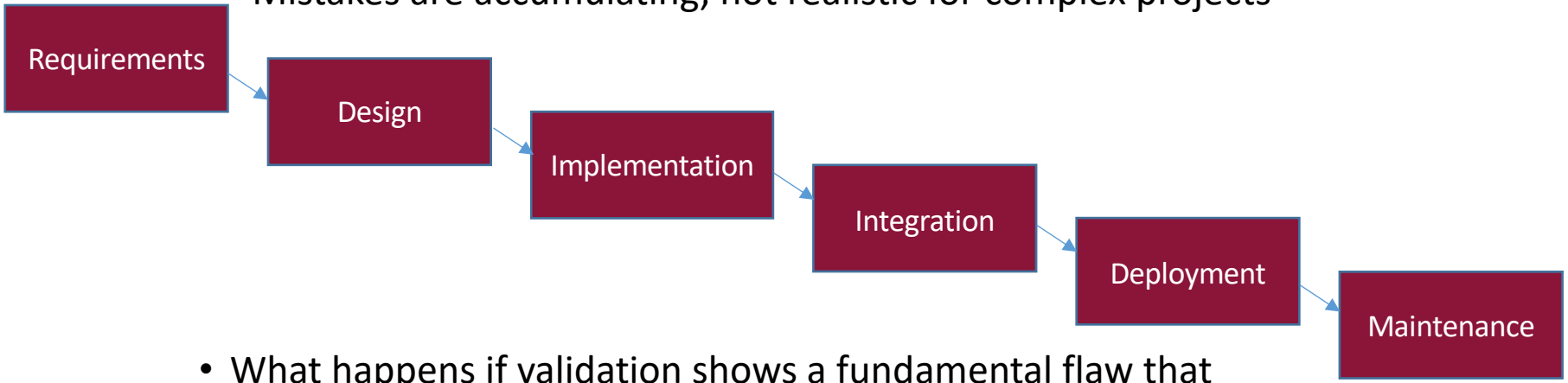
- Simplest SDLC model
- Proposed by Royce in 1970



Software Development Life Cycle

Waterfall model

- + Easy to plan and to follow
- +- Requirements cannot change
- - Mistakes are accumulating, not realistic for complex projects

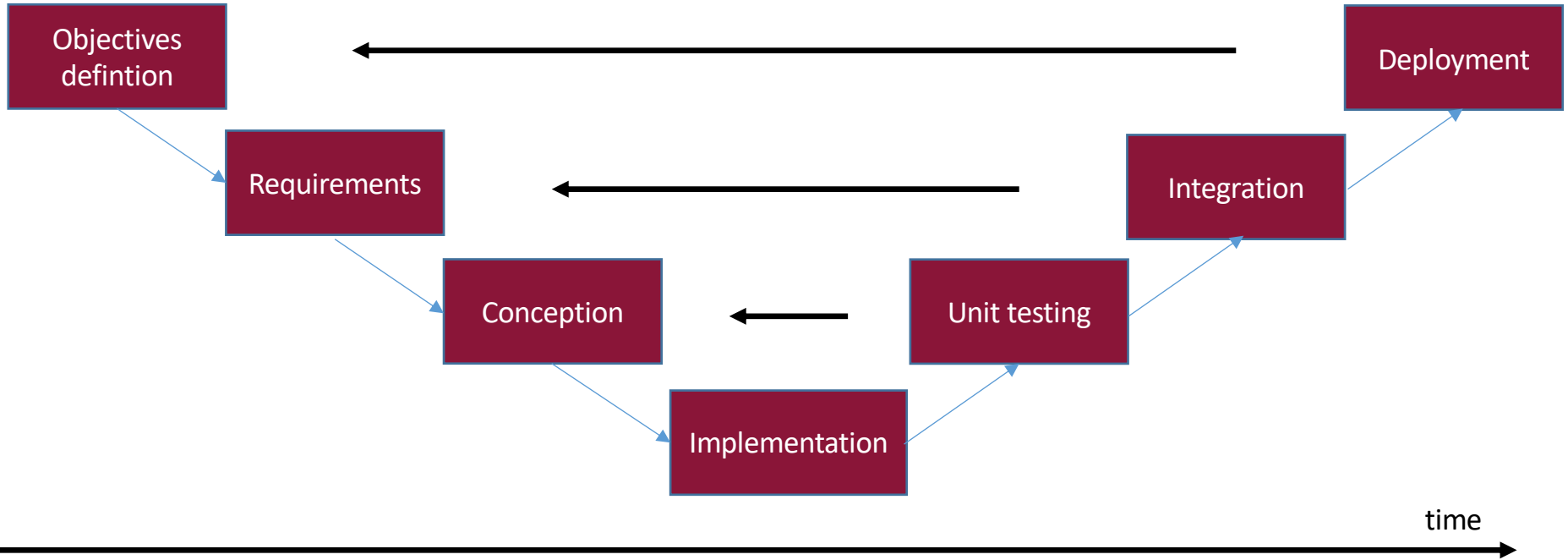


- What happens if validation shows a fundamental flaw that requires a design change ?

Software Development Life Cycle

V-model

- Extension of waterfall



Software Development Life Cycle

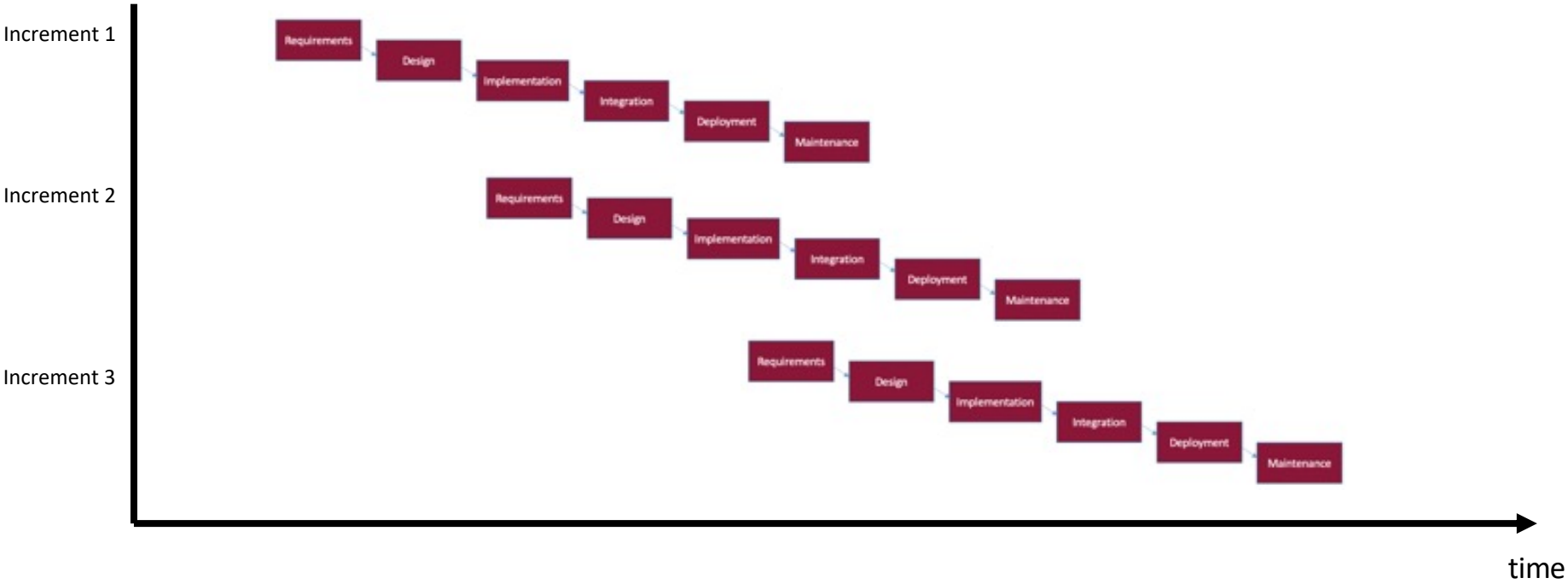
V-model

- Extension of waterfall
- Still simple
- With each components, verification (downward phase) or validation (upward) procedures are defined
- Still not flexible enough for complex projects

- In general, what you have been doing

Software Development Life Cycle

Incremental model



Software Development Life Cycle

Incremental model

- Software broken down in sub-components
- First increment: core functionality
- Subsequent increments take into account feedback

- + Aligns better with the customer needs
- + Fast delivery
- - Fundamental flaws can exist

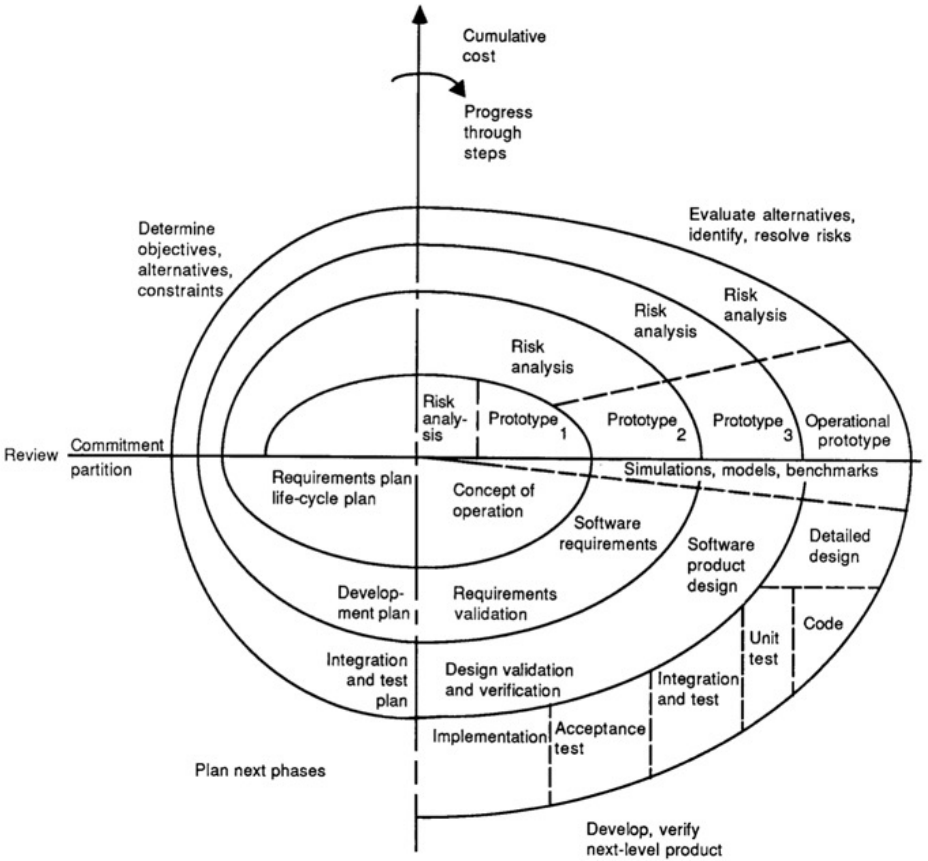
Software Development Life Cycle

Spiral model

Boehm, 1988

- Based on 4 quadrants
- Can be seen as a generalization of previous models
- Risk driven model

- + Suitable for complex projects
- - Requires experience, costly



Software Development Life Cycle

SDLC models

- 4 models
- Different levels of complexity
- Relatively rigid
- Solution since 2000s : Agile (coming up later in this class)

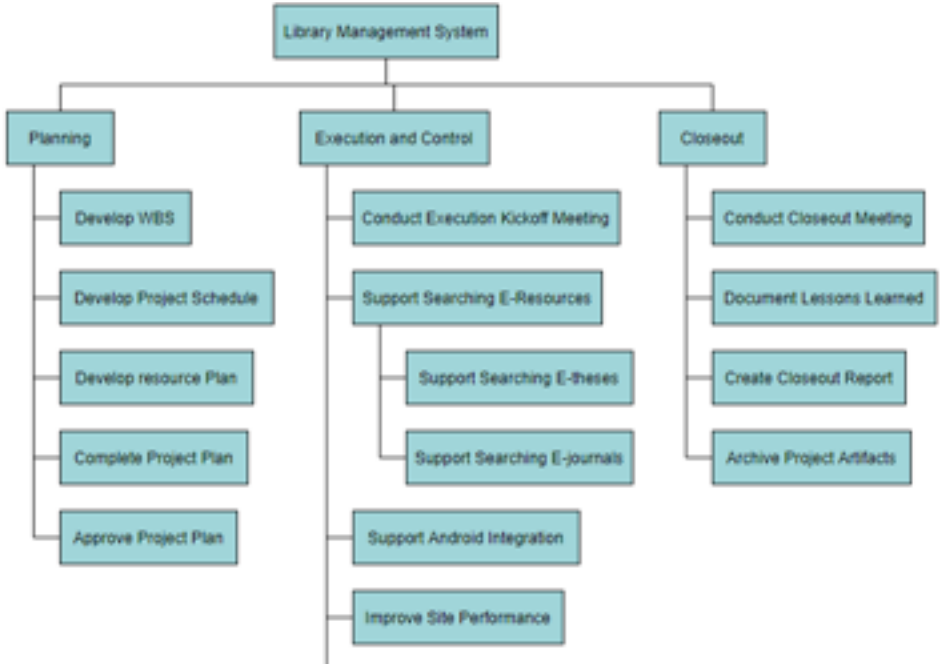
Planning a project

Planification

- Planification is an essential component of a successful project
- Remember: scope of a project depends on Quality, Time, Cost
- We know the main tasks from our SDLC model. We need to decompose them in smaller, achievable tasks
- Work Breakdown Structure (WBS) or Organigramme des Tâches (OT)
- Tree structure with at least 3 levels:
  - Level 1: name of the project
  - Level 2: main activities seen before
  - Level 3 and more: sub tasks

Planning a project

Example of a WBS



Source

Planning a project

WBS rules

5 rules to follow:

1) It has to be a tree structure
2) Each task should be clearly defined, including potential deliverables
3) Each task should have a clear finsihing action
4) Each deliverable should be associated to a task
5) Achievement of every sub-task implies the achievement of the parent task

Planning a project

PERT

- Program Evaluation and Review Technique (PERT) is a method of analyzing the different tasks in the project.
- In particular, allows to analyze:
  - Dependencies between tasks
  - Duration of the tasks
  - Duration of the project (through critical path)
- Often represented as a diagram

Planning a project

PERT

Task name	Time allocated	Predecessor(s)
A	8	
B	5	
C	6	B
D	7	A, B
E	5	C, D
F	4	E
G	3	E
H	7	G

Elements of a software project

Wooclap

https://www.wooclap.com/L3GL233

Planning a project

Estimating time

- Comes with experience
- In practice, we tend to underestimate the time necessary
- in PERT, we can compute the expected time as a weighted average of
  - o, the optimistic time (everything goes perfectly) weight = 1
  - p, the pessimistic time (everything goes wrong) weight = 1
  - m, the most likely time, weight = 4

- expected time = $\frac{o+p+4m}{6}$
- Derivated from Beta distribution

Planning a project

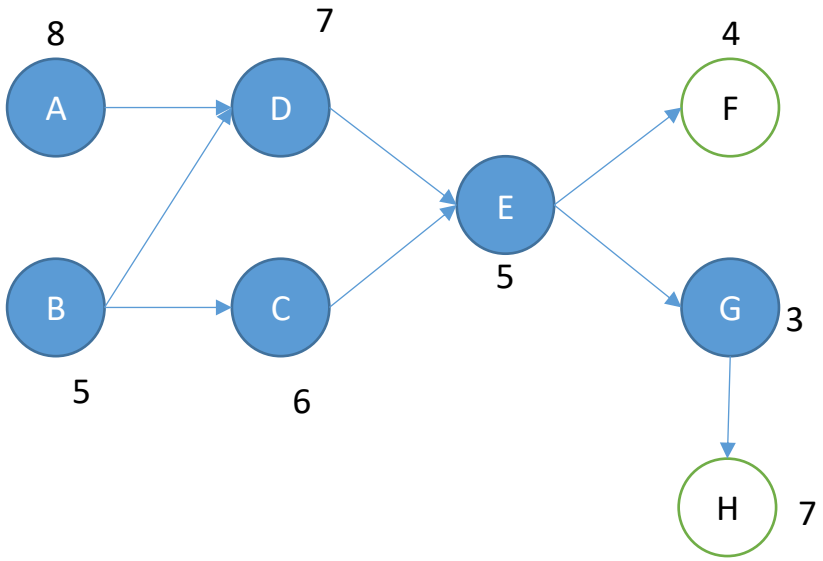
PERT

Task name	Time allocated	Predecessor(s)
A	8	
B	5	
C	6	B
D	7	A, B
E	5	C, D
F	4	E
G	3	E
H	7	G

Planning a project

PERT

Task name	Time allocated	Predecessor(s)
A	8	
B	5	
C	6	B
D	7	A, B
E	5	C, D
F	4	E
G	3	E
H	7	G



Planning a project

PERT

Task name	Time allocated	Predecessor(s)
A	8	
B	5	
C	6	B
D	7	A, B
E	5	C, D
F	4	E
G	3	E
H	7	G

Task name	Start date	End date
A		
B		
C		
D		
E		
F		
G		
H		

Planning a project

PERT

Task name	Time allocated	Predecessor(s)
A	8	
B	5	
C	6	B
D	7	A, B
E	5	C, D
F	4	E
G	3	E
H	7	G

Task name	Start date	End date
A	0	8
B	0	5
C	5	11
D	8	15
E	15	20
F	20	24
G	20	23
H	23	30

Planning a project

Critical Path

- Critical path: the set of tasks that allow to obtain the shortest time to finish the project
- Consequence: if one of the tasks from the critical path takes longer to be performed, the project will take longer to finish

- Algorithm for critical path:
  1) Select the taks with the latest finish date
  2) Put the selected task in the critical path
  3) Select the predecessor(s) of the selected task with the latest finishing date
  4) Repeat 2-3 until reaching starting node(s)

Planning a project

PERT

1) Select the taks with the latest finish date
2) Put the selected task in the critical path
3) Select the predecessor(s) of the selected task with the latest finishing date
4) Repeat 2-3 until reaching starting node(s)

Task name	Time allocated	Predecessor(s)
A	8	
B	5	
C	6	B
D	7	A, B
E	5	C, D
F	4	E
G	3	E
H	7	G

Task name	Start date	End date	Critical path
A	0	8	
B	0	5	
C	5	11	
D	8	15	
E	15	20	
F	20	24	
G	20	23	
H	23	30	

Planning a project

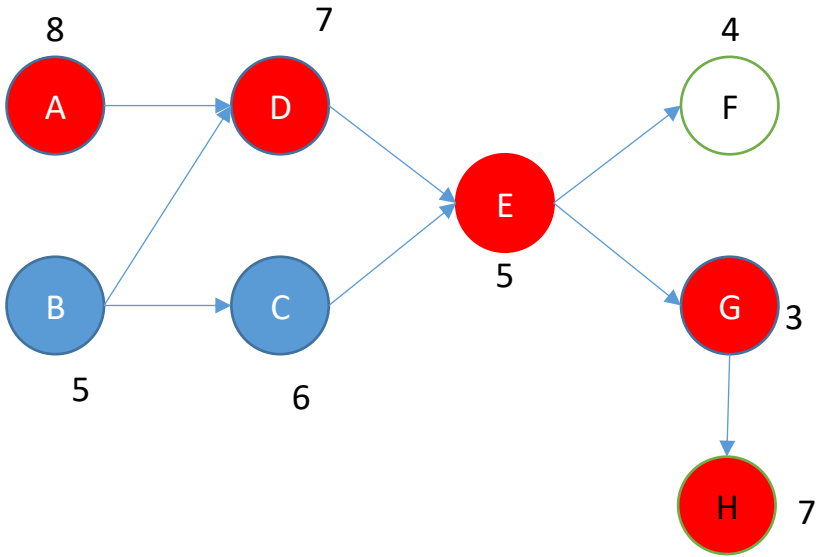
PERT

Task name	Time allocated	Predecessor(s)
A	8	
B	5	
C	6	B
D	7	A, B
E	5	C, D
F	4	E
G	3	E
H	7	G

Task name	Start date	End date	Critical path
A	0	8	X
B	0	5	
C	5	11	
D	8	15	X
E	15	20	X
F	20	24	
G	20	23	X
H	23	30	X

Planning a project

PERT



Task name	Start date	End date	Critical path
A	0	8	X
B	0	5	
C	5	11	
D	8	15	X
E	15	20	X
F	20	24	
G	20	23	X
H	23	30	X