ASPIRE
ACHIEVE
ADVANCE

Level 4

Title for this framework at level 4

Higher Apprenticeship for IT, Software, Web & Telecoms Professionals

Systems Development Level 4

(Unit 402)

www.3aaa.co.uk

Systems Development L4

Unit 402 - Aim

The purpose of this unit is to provide learners within an understanding of the systems development life cycle. Using a project approach, learners will explore the stages in detail, gathering and analysing customer requirements, designing an IT solution, and planning its testing and implementation.

Learners will identify the various stakeholder perspectives to ensure both that the solution meets requirements and that the wider implications are considered.

Learners will develop practical skills in the use of the various tools and techniques associated with the various methodologies for systems development.

Learning Outcomes

- 1. Know how to plan systems development activities against agreed quality standards
- 2. Be able to establish customer requirements

Systems Development L4

Unit 402 – Learning Outcomes

1. Know how to plan systems development activities against agreed quality standards

1.1 Outline ***stages*** of the systems development ***life cycle***

1.2 Explain the ***deliverables*** associated with systems development activities

1.3 Identify the ***sponsors*** and ***stakeholders*** involved in systems development and review

1.4 Explain the importance and role of systems integration to systems development process

Systems Development L4

1.1 Outline ***stages*** of the systems development ***life cycle***

Systems Development Life Cycle models (SDLC):

- Waterfall;
- Spiral;
- Agile;
- Rapid prototyping;
- Rapid Application Design;
- Dynamic Systems Design Methodology (DSDM).

Systems Development L4

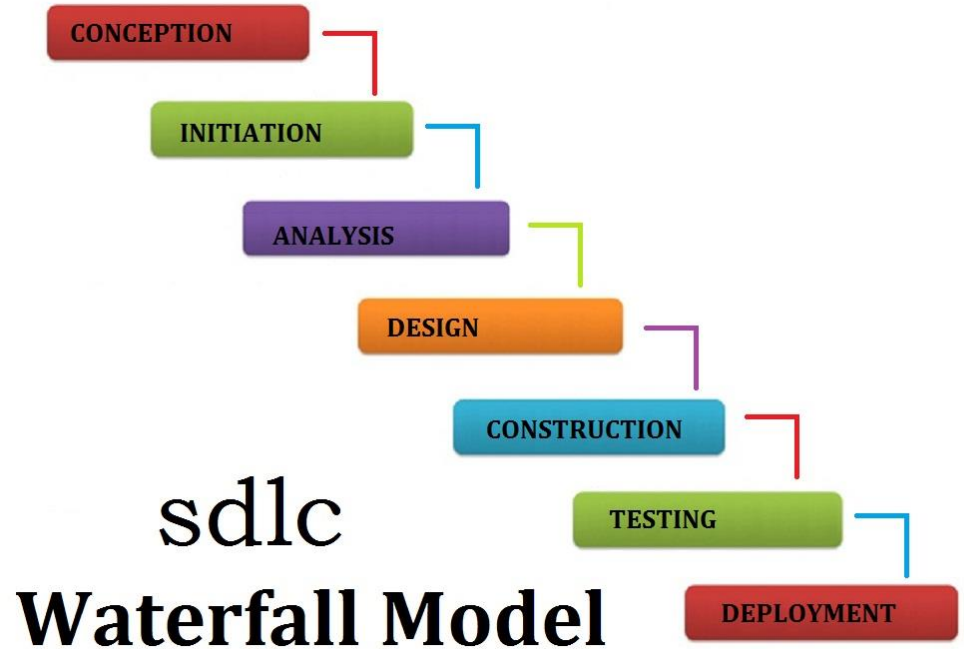
1.1 Outline ***stages*** of the systems development ***life cycle***

SDLC Models

Waterfall

The **waterfall model** is a sequential design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance.

(http://en.wikipedia.org/wiki/Waterfall_model)

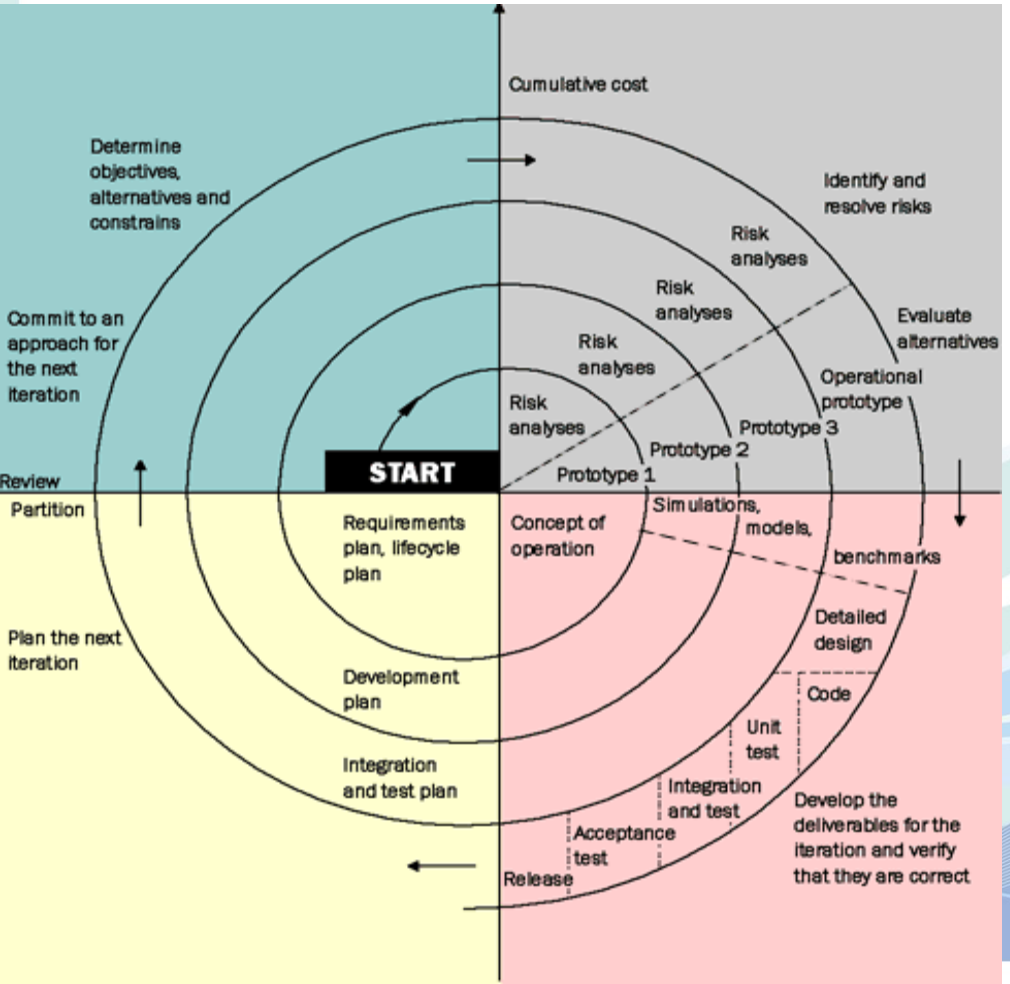


Systems Development L4

1.1 Outline ***stages*** of the systems development ***life cycle***

SDLC Models

Spiral



The **spiral model** is a risk-driven process model generator for software projects.

Based on the unique risk patterns of a given project, the spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall, or evolutionary prototyping.

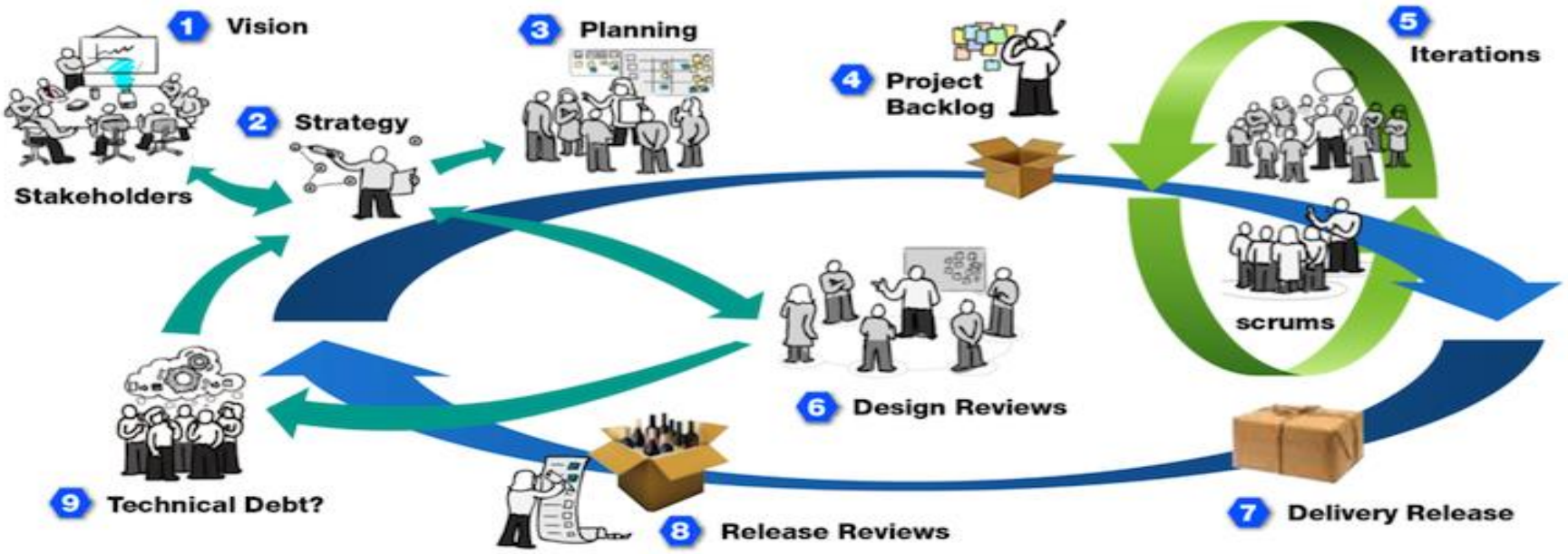
(http://en.wikipedia.org/wiki/Spiral_model)

SDLC Models

Agile

Agile software development is a group of software development methods in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, continuous improvement and encourages rapid and flexible response to change.

(http://en.wikipedia.org/wiki/Agile_software_development)



SDLC Models

Rapid prototyping

Software prototyping is the activity of creating prototypes of software applications, i.e., incomplete versions of the software program being developed. It is an activity that can occur in software development and is comparable to prototyping as known from other fields, such as mechanical engineering or manufacturing.

A prototype typically simulates only a few aspects of, and may be completely different from, the final product.

Prototyping has several benefits: The software designer and implementer can get valuable feedback from the users early in the project.

The client and the contractor can compare if the software made matches the software specification, according to which the software program is built. It also allows the software engineer some insight into the accuracy of initial project estimates and whether the deadlines and milestones proposed can be successfully met.

(http://en.wikipedia.org/wiki/Software_prototyping)

SDLC Models

Rapid Application Design

Rapid application development (**RAD**) is both a general term used to refer to alternatives to the conventional waterfall model of software development as well as the name for James Martin's approach to rapid development.

In general, RAD approaches to software development put less emphasis on planning tasks and more emphasis on development. In contrast to the waterfall model, which emphasizes rigorous specification and planning,

RAD approaches emphasize the necessity of adjusting requirements in reaction to knowledge gained as the project progresses. This causes RAD to use prototypes in addition to or even sometimes in place of design specifications.

(http://en.wikipedia.org/wiki/Rapid_application_development)

SDLC Models

Rapid Application Design [continued]

RAD approaches also emphasize a flexible process that can adapt as the project evolves rather than rigorously defining specifications and plans correctly from the start.

In addition to James Martin's RAD methodology, other approaches to rapid development include Agile methods and the spiral model.

RAD is especially well suited (although not limited to) developing software that is driven by user interface requirements.

Graphical user interface builders are often called rapid application development tools.

(http://en.wikipedia.org/wiki/Rapid_application_development)

SDLC Models

Dynamic Systems Design Methodology (DSDM)

Dynamic systems development method (**DSDM**) is an agile project delivery framework, primarily used as a software development method. First released in 1994, DSDM originally sought to provide some discipline to the rapid application development (RAD) method.

In 2007 DSDM became a generic approach to project management and solution delivery. DSDM is an iterative and incremental approach that embraces principles of Agile development, including continuous user/customer involvement.

DSDM fixes cost, quality and time at the outset and uses the **MoSCoW** prioritisation of scope into ***Musts***, ***Shoulds***, ***Coulds*** and ***Won't haves*** to adjust the project deliverable to meet the stated time constraint. DSDM is one of a number of Agile methods for developing software and non-IT solutions, and it forms a part of the Agile Alliance.

(http://en.wikipedia.org/wiki/Dynamic_systems_development_method)

Systems Development L4

1.1 Outline ***stages*** of the systems development ***life cycle***

Stages of the SDLC:

- Initiation;
- Requirements definition;
- Systems analysis;
- Systems design;
- Systems integration & testing;
- Implementation;
- Maintenance;
- Disposal.

The ISTQB use a six phase model related to these stages

Software life cycle models describe phases of the software cycle and the order in which those phases are executed.

Each phase produces deliverables required by the next phase in the life cycle.

Requirements are translated into design.

Code is produced according to the design which is called development phase.

After coding and development the testing verifies the deliverable of the implementation phase against requirements.

(http://istqbexamcertification.com/what-are-the-software-development-life-cycle-sdlc-phases/)

Stages of the SDLC

Initiation

The Initiation Phase begins when it is determined that a business process requires enhancement.

Enhancements or changes to business processes may be prompted by business process improvement activities, changes in business functions, IT advancements, and/or external sources, such as changes in public law or federal statutes.

Stages of the SDLC

Requirements definition and Systems analysis

Both of these stages are covered in one phase of the ISTQB model.

1) Requirement gathering and analysis:

Business requirements are gathered in this phase.
This phase is the main focus of the project managers and stake holders.
Meetings with managers, stake holders and users are held in order to determine the requirements like;

- **Who is going to use the system?**
- **How will they use the system?**
- **What data should be input into the system?**
- **What data should be output by the system?**

These are general questions that get answered during a requirements gathering phase. After requirement gathering these requirements are analyzed for their validity and the possibility of incorporating the requirements in the system to be development is also studied.

Finally, a Requirement Specification document is created which serves the purpose of guideline for the next phase of the model.

Stages of the SDLC

Systems design

Design:
In this phase the system and software design is prepared from the requirement specifications which were studied in the first phase.

System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

The system design specifications serve as input for the next phase of the model.

Stages of the SDLC

Systems integration and testing and Implementation

The ITSQB uses a slightly different model but the concepts are equivalent

Implementation / Coding: On receiving system design documents, the work is divided in modules/units and actual coding is started. Since, in this phase the code is produced it is the main focus for the developer. This is the longest phase of the software development life cycle.
Testing: After the code is developed it is tested against the requirements to make sure that the product is actually solving the needs addressed and gathered during the requirements phase.

During this phase unit testing, integration testing, system testing, acceptance testing are done.

Deployment: After successful testing the product is delivered / deployed to the customer for their use.

Stages of the SDLC

Maintenance

Maintenance:
Once when the customers start using the developed system unforeseen problems frequently arise that could not be predicted during prior stages of the SDLC model.

These issues are dealt with during the maintenance stage.

This stage also handles minor evolutions of the system that would not require a full revision of the system.

Stages of the SDLC

Disposal

This covers disposal of a system and closeout of any contracts in place. Information security issues associated with information and system disposal should be addressed explicitly. When information systems are transferred, become obsolete, or are no longer usable, it is important to ensure that government resources and assets are protected.

Usually, there is no definitive end to a system. Systems normally evolve or transition to the next generation because of changing requirements or improvements in technology. System security plans should continually evolve with the system.

Much of the environmental, management, and operational information should still be relevant and useful in developing the security plan for the follow-on system.

The disposal activities ensure the orderly termination of the system and preserve the vital information about the system so that some or all of the information may be reactivated in the future, if necessary. Particular emphasis is given to proper preservation of the data processed by the system so that the data is effectively migrated to another system or archived in accordance with applicable records management regulations and policies for potential future access.

Systems Development L4

LO1.2 Explain the ***deliverables*** associated with systems development activities

Deliverable is a term used in **project** management to describe a tangible or intangible object produced as a result of the **project** that is intended to be delivered to a customer (either internal or external).

(http://en.wikipedia.org/wiki/Deliverable)

Systems Development L4

LO1.2 Explain the ***deliverables*** associated with systems development activities

Deliverables:

- Output document from each stage of systems development life cycle

For example: From the Initiation stage:

- Terms of reference;
- Feasibility study;
- Cost-benefit analysis;
- Business case;

- Documentation according to methodology adopted

For example:

- Conceptual model;
- Relational data model;
- Use cases;

Systems Development L4

LO1.2 Explain the ***deliverables*** associated with systems development activities

Deliverables: [continued]

- System documentation
- Documentation within organisational guidelines or standards:
  - documentation content and layout,
  - naming conventions,
  - diagram requirements,
  - quality standards and control
- Organisational processes and requirements for project management

Systems Development L4

LO1.3 Identify the ***sponsors*** and ***stakeholders*** involved in systems development and review

The ***Project Sponsor*** is the business manager responsible for providing the overall business direction for the project; he or she acts as the senior spokesperson for the project.

The Project Sponsor ensures that the needs and accomplishments within the business area are widely known and understood. With review and approval of project documents and careful stage reviews, the Project Sponsor ensures that the design of the system meets functional and non-functional business goals.

The Project Sponsor is also responsible for ensuring that adequate financial and business process resources to address business area needs are made available in a timely manner.

The Project Sponsor is expected to actively assist project teams to address risks and resolve project issues throughout the project life cycle.

(MARYLAND, Roles and Responsibilities, p.19)

(http://doit.maryland.gov/SDLC/Documents/SDLC%20Roles%20and%20Responsibilities.pdf)

Systems Development L4

LO1.3 Identify the ***sponsors*** and ***stakeholders*** involved in systems development and review

Stakeholders are individuals who may be positively or negatively impacted by the execution or completion of a project or may exude influence on the project and its deliverables.

(MARYLAND, Roles and Responsibilities, p.21)
*(http://doit.maryland.gov/SDLC/Documents/SDLC%20Roles%20and%20Responsibilities.pdf**)***

Systems Development L4

LO1.4 Explain the importance and role of systems integration to systems development process

In information technology, **systems integration** is the process of linking together different computing systems and software applications physically or functionally, to act as a coordinated whole.

A system is an aggregation of subsystems cooperating so that the system is able to deliver the overarching functionality. System integration involves integrating existing often disparate systems.

System integration (SI) is also about adding value to the system, capabilities that are possible because of interactions between subsystems.

In today's connected world, the role of system integration engineers is becoming more and more important: more and more systems are designed to connect, both within the system under construction and to systems that are already deployed.

(http://en.wikipedia.org/wiki/System_integration)

2. Be able to establish customer requirements

2.1 Identify the ***requirements*** or business specification of proposed systems

2.2 Obtain information on existing and required inputs, outputs and processes of ICT systems

2.3 Explain any ***constraints*** new systems may encounter

2.4 Recommend ***solutions*** for customer consideration

2.5 Explain the impact and implications of any changes to customer requirements

LO2.1 Identify the ***requirements*** or business specification of proposed systems

Requirements

Intended audience:

- Customer;
- Business;
- Stakeholder;
- Supplier

The specific needs for a system depend on the intended audience

Some needs will be common

– A standard user interface for example.

Other needs may be very specific

– Suppliers will have some needs that are irrelevant to an organisation's customers for example.

LO2.1 Identify the ***requirements*** or business specification of proposed systems

Requirements

It is important to consider the scope and objectives of a system

The ***system scope*** should be documented in the project plan before the system size is estimated. The scope statement defines what the project will and will not include, in enough detail to clearly communicate to all participants.

(http://www.testablerequirements.com/testablerequirements/def_sys_scope_bound.htm)

An ***objective*** is a specific result that a person or system aims to achieve within a time frame and with available resources. In general, objectives are more specific and easier to measure than goals. Objectives are basic tools that underlie all planning and strategic activities. They serve as the basis for creating policy and evaluating performance.

(http://www.businessdictionary.com/definition/objective.html)

LO2.1 Identify the ***requirements*** or business specification of proposed systems

Requirements

Systems may require specific structures to be used or developed

– for example a part of a revised system may need to be able to inter-operate with an existing Oracle database.

Certain restrictions may be placed on the system ***interface***

There will almost certainly be ***deadlines*** for the development and implementation of the system

There may also be ***functional*** requirements for inputs, outputs and processes.

Planning and scheduling activity

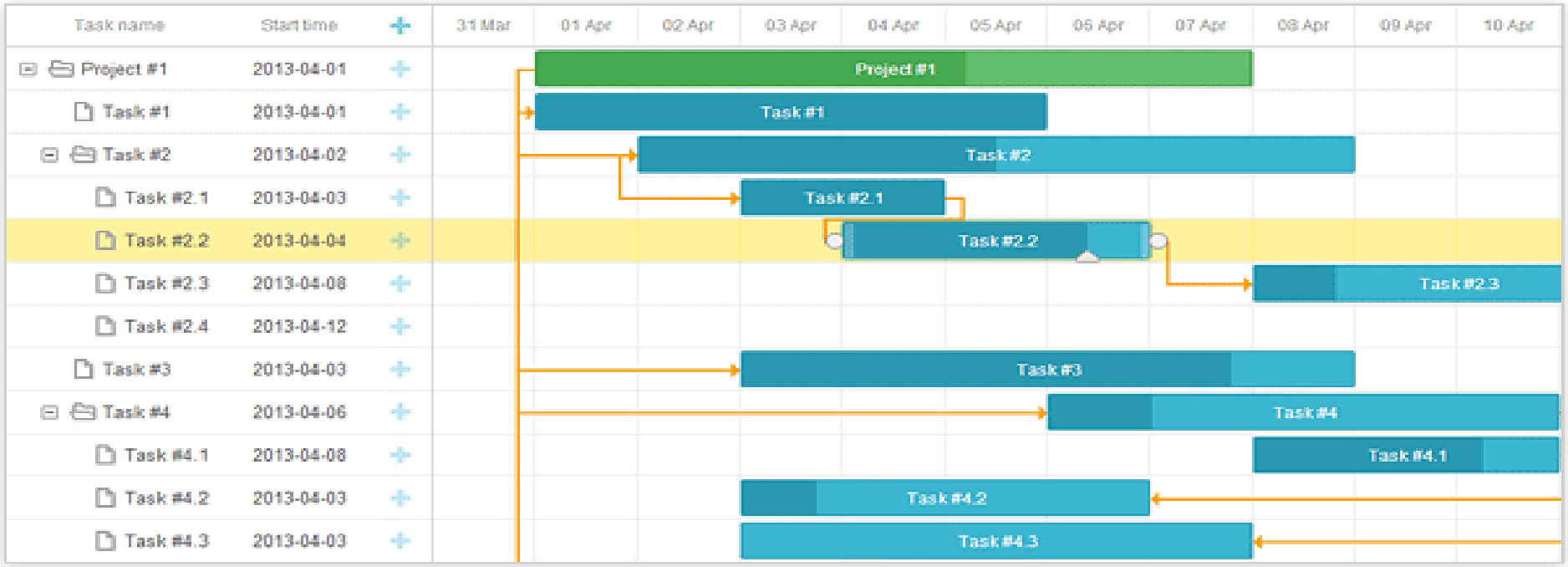
Analysis using ***PERT*** or ***GANTT*** charts can be of great assistance when working towards the requirements of specification of a system

Systems Development L4

LO2.1 Identify the ***requirements*** or business specification of proposed systems

GANTT Chart

A **Gantt chart** is a type of bar chart, developed by Henry Gantt in the 1910s, that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements comprise the work breakdown structure of the project. Modern Gantt charts also show the dependency (i.e., precedence network) relationships between activities. (http://en.wikipedia.org/wiki/Gantt_chart)

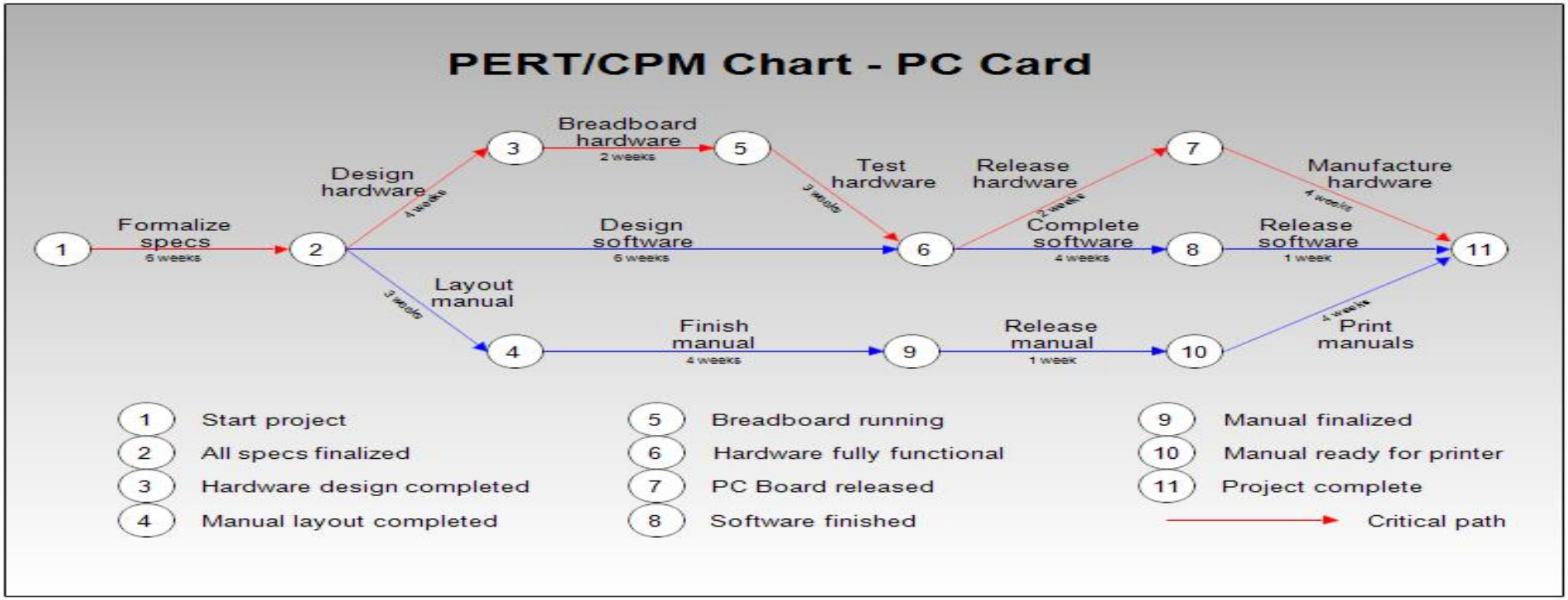


Systems Development L4

LO2.1 Identify the ***requirements*** or business specification of proposed systems

PERT Chart

A PERT chart is a project management tool used to schedule, organize, and coordinate tasks within a project. PERT stands for *Program Evaluation Review Technique*, a methodology developed by the U.S. Navy in the 1950s to manage the Polaris submarine missile program. A similar methodology, the *Critical Path Method* (CPM) was developed for project management in the private sector at about the same time. (http://searchsoftwarequality.techtarget.com/definition/PERT-chart)



Systems Development L4

LO2.2 Obtain information on existing and required inputs, outputs and processes of ICT systems

Information Gathering

- Inputs
- Outputs
- Processes to be carried out

Detailed investigation of these factors for both the existing system (if one exists) and the proposed system needs to be carried out

LO2.3 Explain any ***constraints*** new systems may encounter

Constraints

There may be constraints, restrictions or limitations on the new system

These could relate to:

- IT platform
- Language (programming or human)
- Cost
- Reliability
- Security
- Performance
- Accessibility

Compatibility with legacy systems

Regulatory or legal constraints; organisational policies or staffing factors

2.4 Recommend ***solutions*** for customer consideration

Solutions

There may be more than one potential way to achieve the desired results…

Alternative approaches such as:

- Upgrade of existing system;
- New development;
  - Bespoke
  - Off-the-shelf solution
  - A combination of bespoke and standard production items

There will be Advantages and Disadvantages of different types of systems in different contexts – these should always be described as fully as possible.

Systems Development L4

LO2.5 Explain the impact and implications of any changes to customer requirements

Customers can (and will!) change their requirements – This can potentially happen at any stage in the System Lifecycle

Such changes will have potentially far-reaching implications for the system.

These implications need to be handled to minimise any negative outcomes.