9) Project planning (Part 1)

Table of contents

General remarks

The project charter (covered in chapter 7) incorporates a summary project plan. Before moving on to the actual execution phase of a project, it is necessary to establish a more comprehensive and detailed **project management plan** or **project plan**, which is based on an **in-depth analysis of the requirements specification** and involves **identifying and evaluating the various components of the project**.

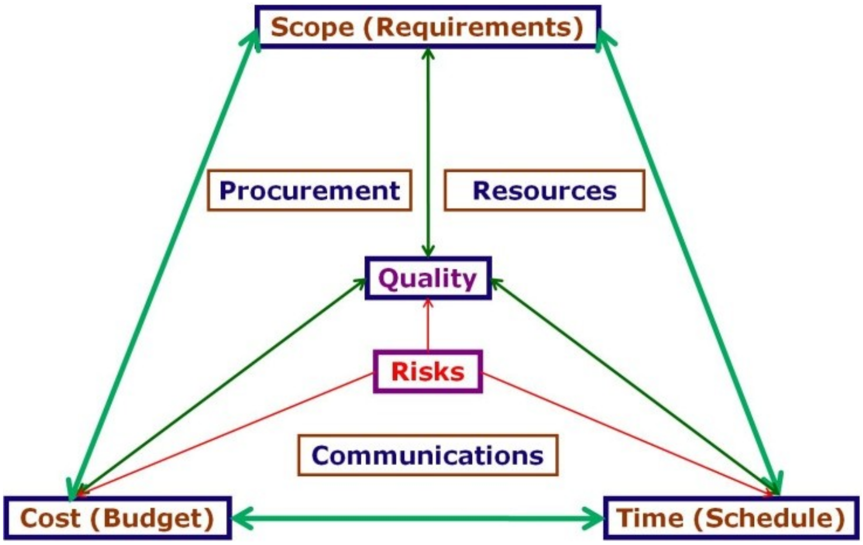
Note that although **the requirements specification is the foundation of the project plan**, it may need to be revised and completed as a result of its in-depth analysis by subject matter experts. That is why preliminary project plans may include the following set of tasks: reviewing, complementing, fine-tuning and agreeing on a revised version of the requirements specification, of the scope of the project and of the project plan itself.

The project plan, which can be considered as the "**road map**" for project execution, defines the **steps** to take and the **resources required** to successfully complete the project, **how long** it will take and **how much** it will cost. It should also include the **identification of risks**, and define **how the project will be executed**, **monitored and controlled, and finally closed**.

The project plan is a **Master plan** incorporating several **subsidiary plans** related to the following **areas of project management** (as listed in the Project Management Institute (PMI) methodology, but not necessarily in the same order).

- **Scope** management
- **Time** management
- **Cost** management
- **Resource** management
- **Procurement** management
- **Quality** management
- **Communications** management
- **Risk** management

Those areas of project management are interrelated and interact with one another, as partially depicted in the following diagram (which also appears at the end of chapter 1).

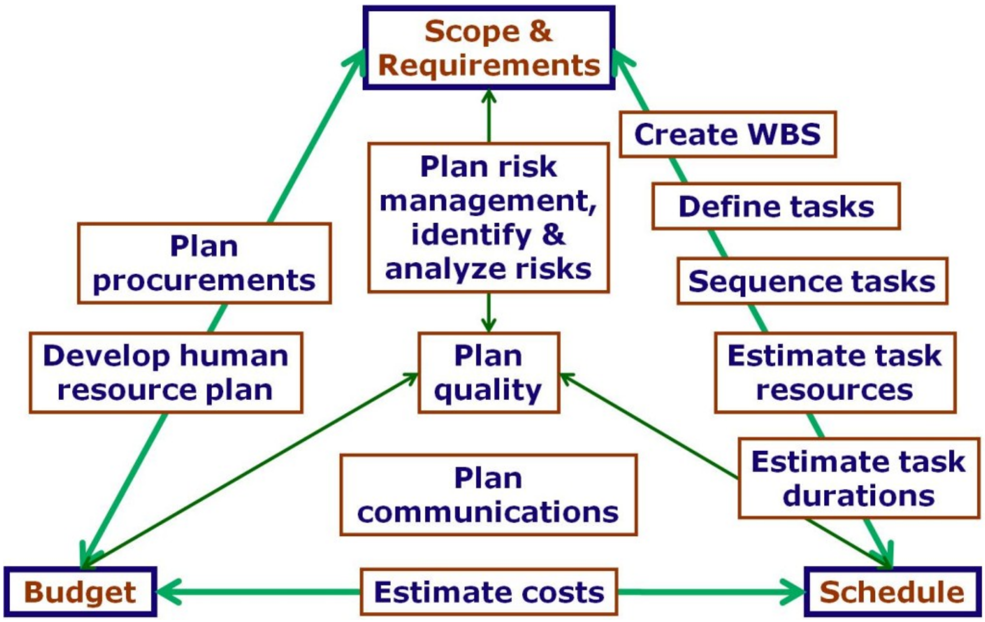


Planning processes: introduction

The **planning processes** related to the project management areas are the following.

- Define scope and Collect requirements.
- Create work breakdown structure (WBS).
- Define tasks.
- Sequence tasks.
- Estimate task resources.
- Estimate task durations.
- Develop schedule.
- Estimate costs.
- Determine budget.
- Develop human resource plan.
- Plan procurements.
- Plan quality.
- Plan communications.
- Plan risk management, identify and analyze risks.

Below is a version of the project management areas diagram which is somewhat rearranged and expanded in order to include some of the above-listed processes.



The first two processes (concerning scope and requirements) have been addressed at the beginning of this chapter as well as in previous chapters of this guide. The following sections of this chapter deal with the other processes. They are presented sequentially, in the above-listed order, but the **processes** and their results are actually very much **interdependent**. For example, the duration of a task depends on the resources applied to it, and if adequate resources are available, work may be divided into tasks that can be executed in parallel. Project planning is therefore necessarily an **iterative process**.

Create the work breakdown structure (WBS)

Creating the "**work breakdown structure** (**WBS**)" consists in **dividing the project** and related work **into "manageable" components**.

The WBS provides an **overview** of the total scope of the project and of its organization.
The **higher levels** of the structure, which is where to start when creating a WBS, may correspond to **phases** of the project, to **subprojects** or to project **work areas**.
The **lowest level** may consist of **tasks** (sometimes called activities) or **groups of tasks** called "**work packages (WPs)"** or a combination of tasks and WPs.
The level of detail of WPs that needs to be shown in a WBS varies with the size and complexity of a project.

It is important to ensure that the WBS covers 100% (no less, no more) of the work to be done to produce all of the deliverables, It should also take into account project management work, but not necessarily documented at the lowest level of detail.
This is the so-called "**100% rule**", which also applies to the set of tasks within any WP.

The **WBS** may be represented by a **text outline** (comparable to a table of contents) and/or by a **diagram** (generally a tree structure). The diagram may be less detailed than the text version of the WBS. As a starting point, it is useful to draw a **simple WBS diagram** (preferably on a **single page** for the sake of readability) to provide a **global view of the work to be done** in order to complete the project.

The WBS may be supplemented with a **WBS dictionary**, the purpose of which is to document details of the work packages (WPs). Each WP entry typically includes an identifier, a brief description of the work to be performed, and the organization, team or individual responsible for the work.

The WBS is a fundamental project management tool. It serves as a reference for many processes such as scheduling and costing, resource assignment and risk assessment. It provides a framework for project supervision and is useful for communication purposes.

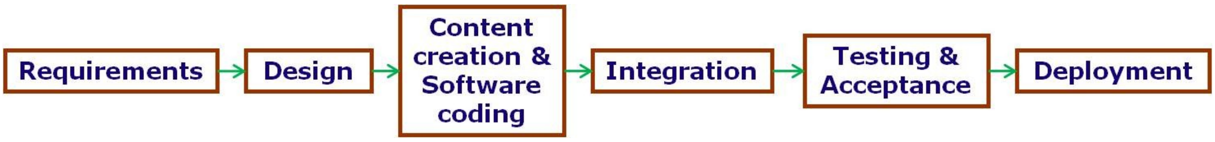
As project work advances, changes to the WBS may be required. The initial WBS should however be retained as a **baseline for change control**.

The **requirements specification** is the major **source of information** for the creation of the WBS. A summary WBS may actually have been created in the process of conducting a feasibility study, preparing the business case and the project charter.

In order to develop the WBS, the PM may need **help from experts** in the various project areas. Such experts may be people in the organization who have already been assigned to the project. Obviously, the PM and other people participating in the effort need to use their **imagination** to build an appropriate WBS for the project!
The WBS of **previous, similar projects** may be helpful as a starting point.

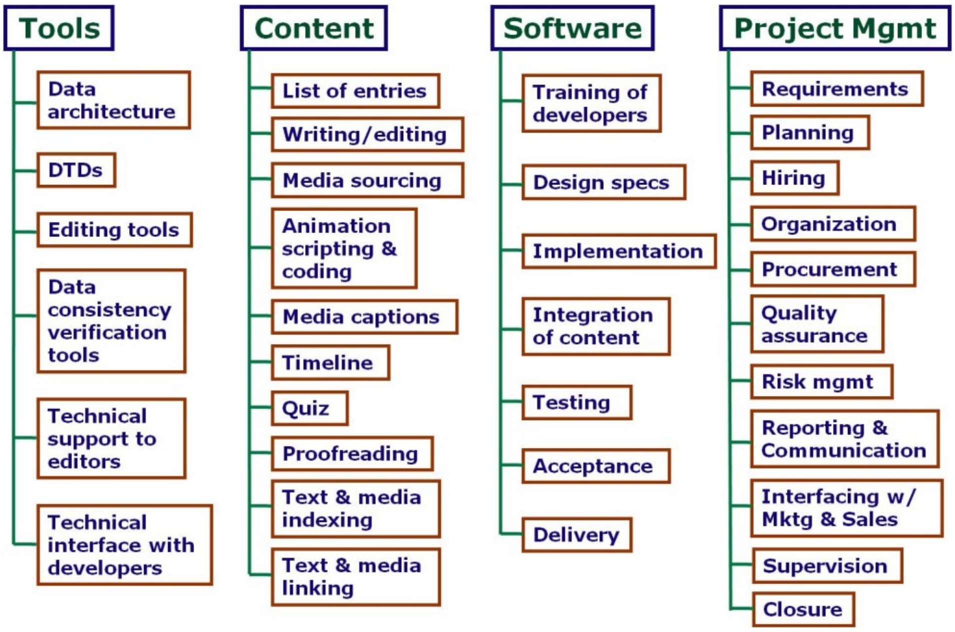
Note that projects resulting in an information system, a software application or a website generally involve the **major phases** shown in the diagram below. The corresponding subprojects or work packages should therefore appear in some form in the WBS of such projects. Furthermore, the particular **software development model or methodology** that may be chosen has a direct impact on a project's WBS (as presented in chapter 12).

Also note that **content creation and software coding** (implementation) can generally be executed **in parallel**.



The initial EHM project was broken down into subprojects and work packages as follows (text outline followed by a diagram version of the WBS):

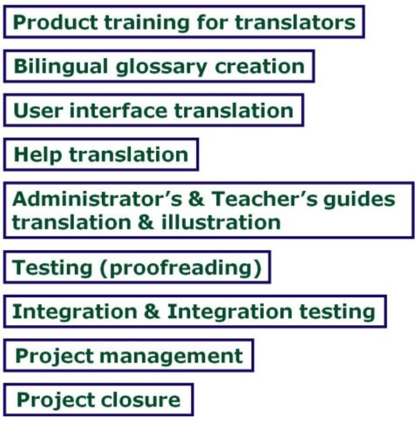
- *<u>"Tools" subproject</u> (person in charge: Director of Data Engineering):*
    - *Design of data architecture and structure*
    - *Development of DTDs (encyclopedia and dictionary)*
    - *Development of editing tools (for data and metadata)*
    - *Development of data consistency verification tools*
    - *Technical support to editorial team*
    - *Technical interface with developers*
- *<u>"Content" subproject</u> (person in charge: Editorial Director):*
    - *Creation of the list of entries*
    - *Text writing/editing*
    - *Sourcing of multimedia assets (photos, drawings, audiovisuals, etc.)*
    - *Scripting and development of animations*
    - *Writing media captions*
    - *Development of the Timeline*
    - *Development of the Quiz*
    - *Text proofreading*
    - *Indexing of texts and multimedia assets*
    - *Linking of media to texts, and of texts to texts*
- *<u>"Software" subproject</u> (person in charge: Project Director, with the assistance of the Director of Data Engineering and the Editorial Director):*
    - *Training of developers on data structure*
    - *Cooperation with developers for the writing of design specifications*
    - *Implementation of the software by the subcontractor*
    - *Integration of content*
    - *Testing, acceptance and delivery of the finished product*
- *<u>"Project Management" subproject</u> (person in charge: Project Director):*
    - *Requirements specification*
    - *Planning*
    - *Hiring*
    - *Organization*
    - *Procurement (incl. choice of subcontractor for software development)*
    - *Quality assurance*
    - *Risk management*
    - *Reporting and communication*
    - *Interfacing with Marketing & Sales*
    - *Supervision, monitoring and control*
    - *Closure*



*The "**Implementation**" work package that appears under "Software" in the previous example of the EHM project WBS was a **major subproject** in itself. This* *subproject obviously required a specific WBS, which was created by the development contractor for the purpose of detailed development planning and management.*

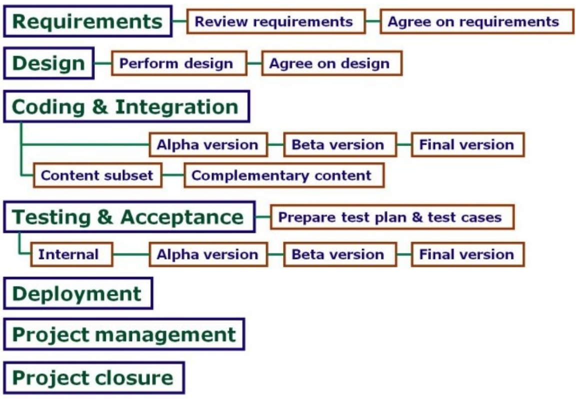
Note that the "**Project management**" work package **generally does not need to be described at a low level of detail** (the details in the previous example were given for purely pedagogical purposes as a reminder of what project management involves!). Indeed, apart from "Requirements" and "Project closure", which generally involve more resources than just the PM, the "Project management" tasks may be featured in the WBS as a single item (work package) that implicitly includes all types of activity and work required to manage the project.
It may also be split into "usual work" and "exceptional work", the latter concerning tasks that may need to be evaluated explicitly in the project plan (eg in terms of duration), because of their impact on the overall schedule (eg "Hiring" and "Procurement").
"Project management" may also be divided into phases, for example in the case where the PM is expected to work full-time on the project in some of its phases and only part-time in other phases.

As another example, here is the simple high-level WBS of a product localization project.

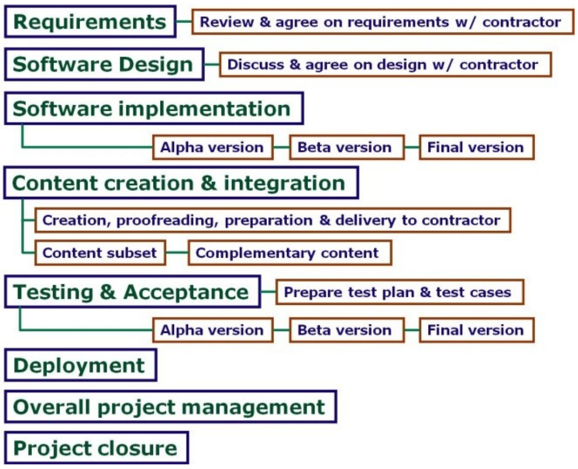

Product training for translators
Bilingual glossary creation
User interface translation
Help translation
Administrator's & Teacher's guides translation & illustration
Testing (proofreading)
Integration & Integration testing
Project management
Project closure

Finally, here are two WBS diagrams of a "hypothetical" project named **EXONE** (project EXONE is used as an example on several occasions in this guide). The WBS on the left covers the work to be done by a software development contractor for a client, the WBS on the right represents the client's view of the work to be done by both the client and the contractor. The client's WBS has been derived from the contractor's WBS, which was submitted to the client, along with a tentative project plan, after the contractor had analyzed the client's requirement specification.

Contractor's WBS



Client's WBS


Requirements
Review & agree on requirements w/ contractor
Software Design
Discuss & agree on design w/ contractor
Software implementation
Alpha version
Beta version
Final version
Content creation & integration
Creation, proofreading, preparation & delivery to contractor
Content subset
Complementary content
Testing & Acceptance
Prepare test plan & test cases
Alpha version
Beta version
Final version
Deployment
Overall project management
Project closure

Define and sequence tasks

General remarks

A complete work breakdown structure (WBS) is divided into **work packages (WPs)** which include more **detailed work components** called **tasks** (or activities), for which estimates can be made in terms of resources required, duration and, as a result, cost.

The process of defining tasks will eventually result in a **list** that covers the **whole set of WPs/tasks** that are **necessary to complete the project**. The WPs/tasks should be presented in a hierarchical fashion, eg a list of **WPs broken down into a specific list of subordinate tasks** whenever such detail is necessary for any given WP.
The term "**WBS**" is often used in project management software applications to refer to the **complete hierarchical list of tasks of a project**.

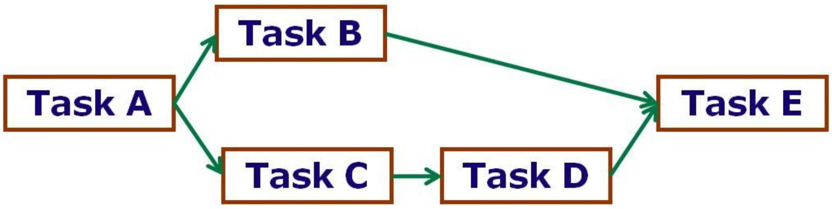
Tasks may be given **attributes**: identification code, description, resource requirements, duration, assumptions, dependencies...
Not all of the information relating to each task is necessarily available at the beginning of the task definition process, so the set of task attributes will become more complete and detailed as the planning effort moves forward.

The process of identifying the set of tasks required to complete the project is usually made easier by first **drawing** a so-called "**network diagram**", which is generally more detailed than the WBS diagram and shows how tasks should be sequenced, as explained below.

Network diagram

The sequence of tasks in a project may be represented by a **network diagram** where **nodes** represent **tasks** (or work packages), and **arrows** show the **logical relationships** between them.
(Such relationships are generally **chronological**.)

Here is a partial network diagram showing the relationships between five tasks in an imaginary project:



The above diagram provides the following information:

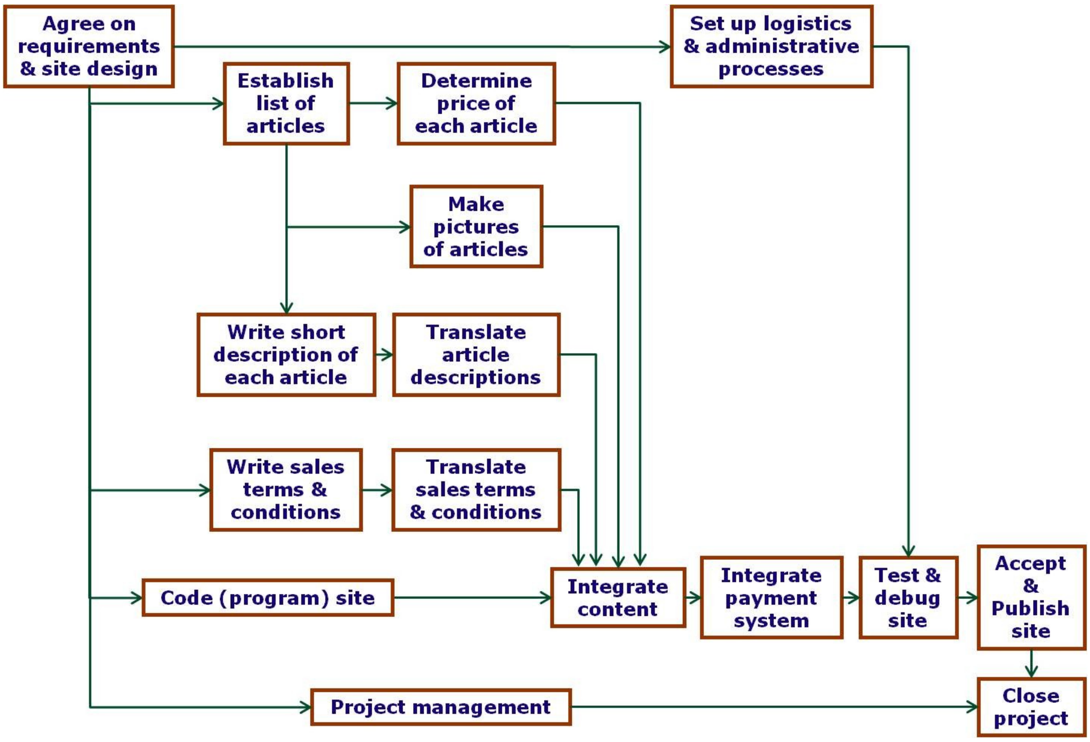
- tasks B and C, which are executed in parallel, cannot start until task A has been completed,
- task D cannot start until task C has been completed,
- task E cannot start until tasks B and D have been completed.

As mentioned above, it is useful to draw a **simple network diagram** (preferably on a **single page** for the sake of readability) to provide a **global view of the sequence of tasks** in the project.

Additional diagrams may be necessary to show the sequence of tasks within the various work packages (WPs) at a **more detailed level**.

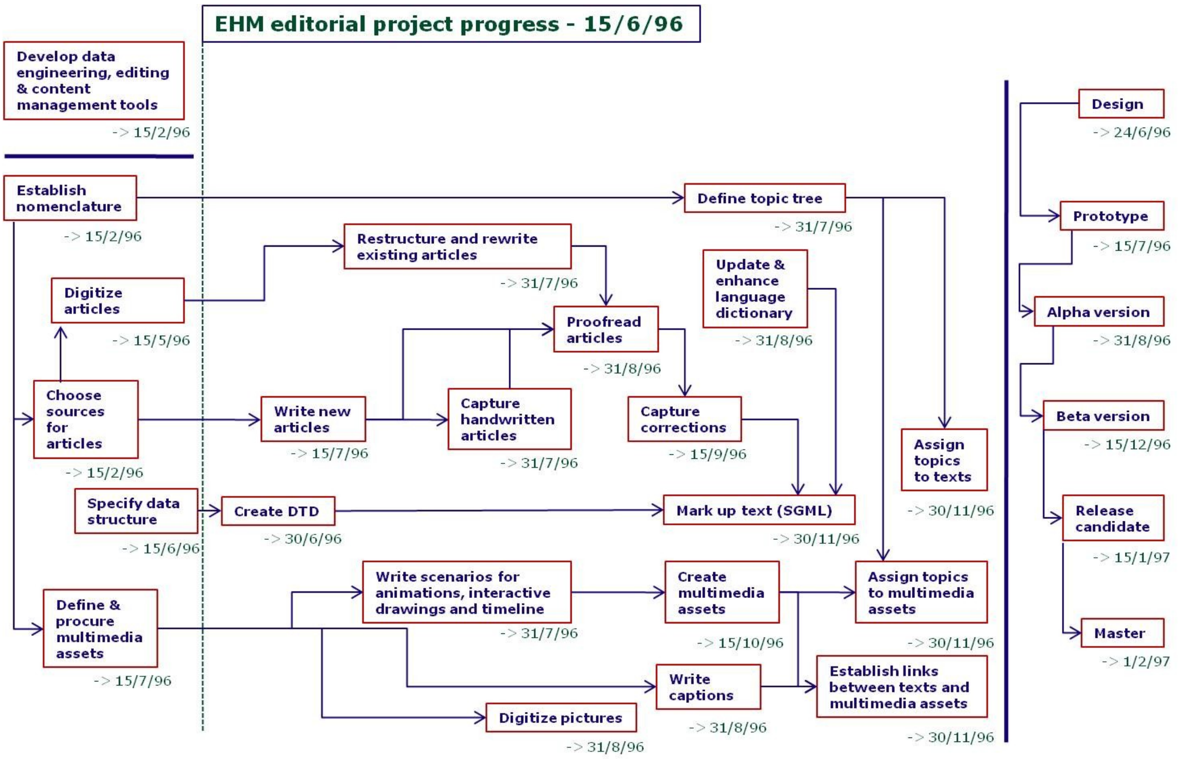
A few examples of network diagrams are given hereafter.

Here is a simplified diagram showing the sequence of WPs/tasks in a project for the creation of an online store.

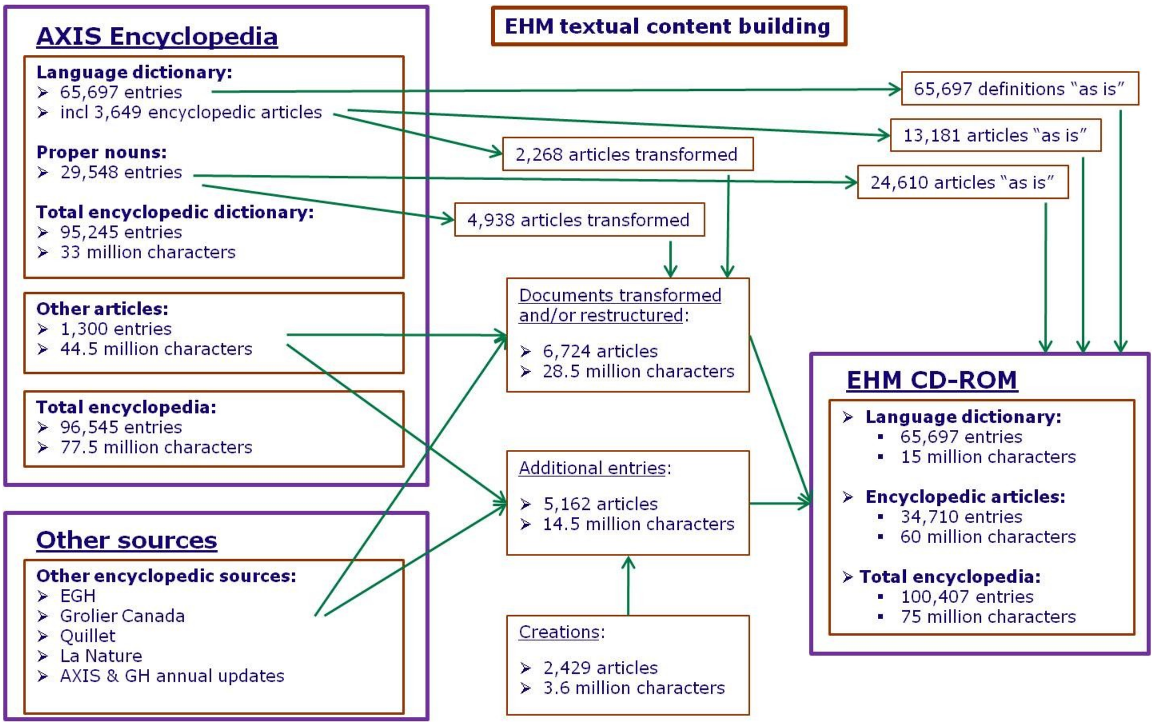

Agree on requirements & site design
Set up logistics & administrative processes
Establish list of articles
Determine price of each article
Make pictures of articles
Write short description of each article
Translate article descriptions
Write sales terms & conditions
Translate sales terms & conditions
Code (program) site
Integrate content
Integrate payment system
Test & debug site
Accept & Publish site
Project management
Close project

Even though the detailed scheduling process may not be complete at this stage of planning, schedule information (eg major milestones) or any other **available relevant information** (eg human resources) may be featured in a summary network diagram in order to make it more informative.

> *The following example is a one-page network diagram that provides a global view of the progress of the initial EHM project at a fairly advanced stage (some tasks had already been completed). The focus is on the "Content" (editorial) WPs. "Tools" and "Software" WPs are summarized respectively in the top left box and on the right.*

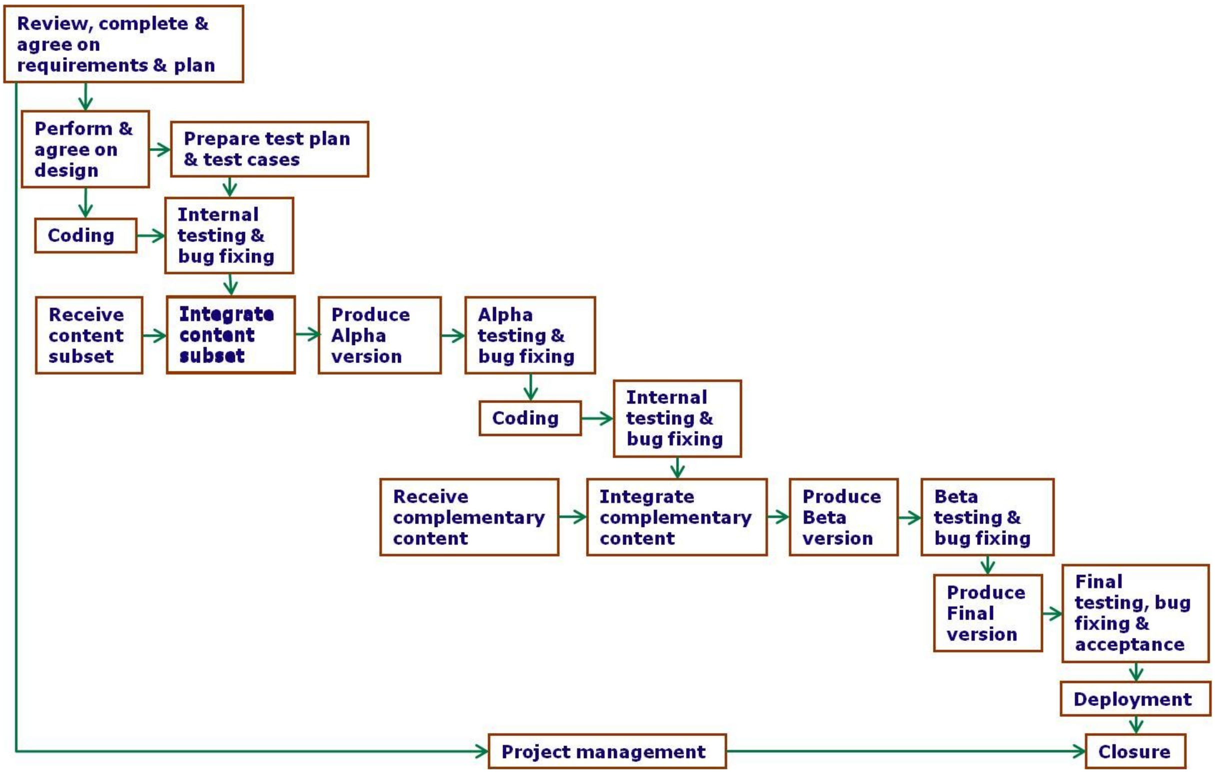


The following diagram shows how the textual content of the EHM was to be sourced or created, and finally assembled. It was used as a summary view of background information required for establishing the editorial part of the previous network diagram.

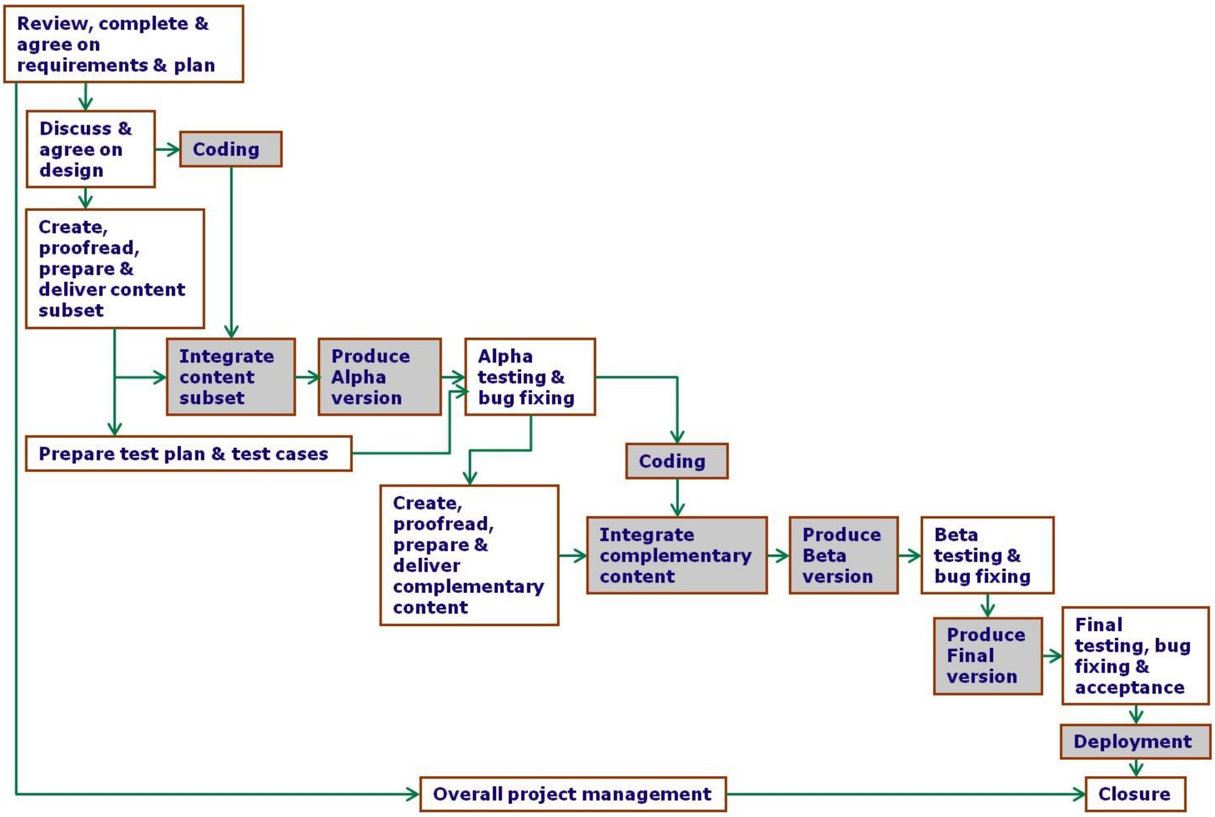


Other example: the following two summary diagrams show the sequence of WPs/tasks in project EXONE, first from the contractor's point of view then from the client's point of view (refer to this project's WBS diagrams on a previous page).

Contractor's network diagram


Review, complete & agree on requirements & plan
Perform & agree on design
Prepare test plan & test cases
Coding
Internal testing & bug fixing
Receive content subset
Integrate content subset
Produce Alpha version
Alpha testing & bug fixing
Coding
Internal testing & bug fixing
Receive complementary content
Integrate complementary content
Produce Beta version
Beta testing & bug fixing
Produce Final version
Final testing, bug fixing & acceptance
Deployment
Project management
Closure

Client's network diagram



Review, complete & agree on requirements & plan
Discuss & agree on design
Coding
Create, proofread, prepare & deliver content subset
Integrate content subset
Produce Alpha version
Alpha testing & bug fixing
Prepare test plan & test cases
Coding
Create, proofread, prepare & deliver complementary content
Integrate complementary content
Produce Beta version
Beta testing & bug fixing
Produce Final version
Final testing, bug fixing & acceptance
Deployment
Overall project management
Closure

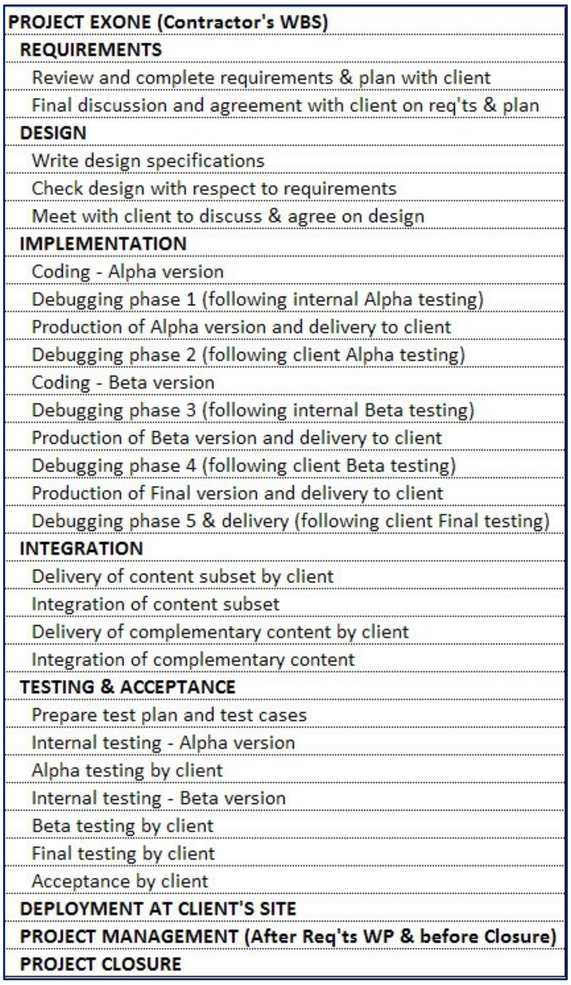
In the above diagram the boxes with a grey background represent WPs/tasks to be performed exclusively by the contractor; the other WPs/tasks concern either work to be done exclusively by the client or work involving both the client and the contractor. Including work to be done by the contractor in the client's network diagram provides the client with an **overall view of the project**.

Note that the expression "**Overall project management**" is used in order to distinguish project management on the client's side from project management on the contractor's side. The project manager working for the client (project owner) has **overall responsibility for the whole project**, whereas the project manager working for the contractor is only responsible for work done by the contractor.

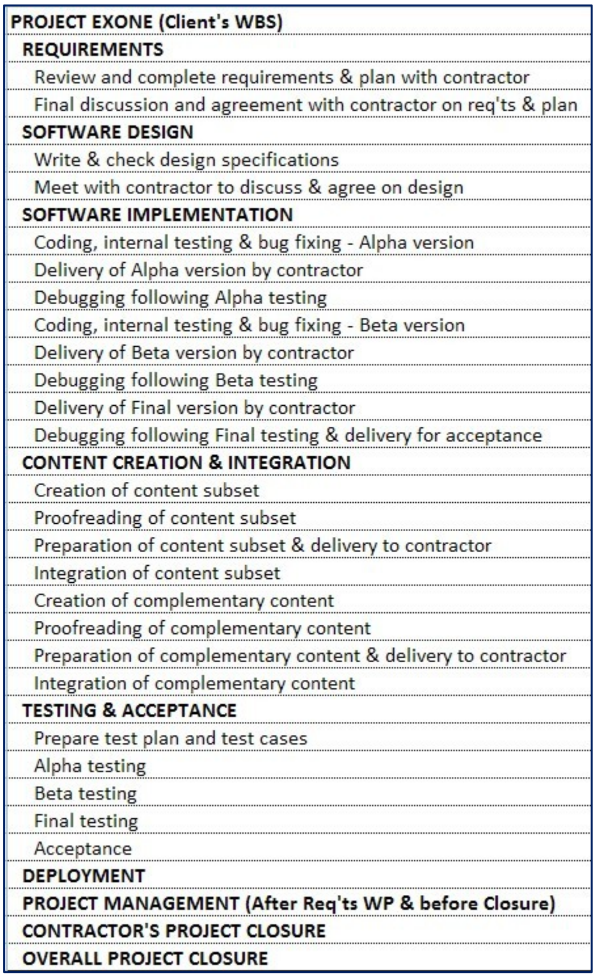
As demonstrated by the previous examples, a **network diagram** can be **created "by hand"**, possibly with a software application that incorporates drawing tools. It can also be produced automatically from a sequenced list of tasks by a project management application such as Microsoft Project or ProjectLibre, as explained further on, but such applications do not necessarily provide an **easy-to-read single-page overview**.

Task list

The two illustrations below are **structured task lists** for project EXONE, respectively from the contractor’s standpoint and from the client’s standpoint, derived from the network diagrams of the project, shown on previous pages.



PROJECT EXONE (Contractor's WBS)
REQUIREMENTS
Review and complete requirements & plan with client
Final discussion and agreement with client on req'ts & plan
DESIGN
Write design specifications
Check design with respect to requirements
Meet with client to discuss & agree on design
IMPLEMENTATION
Coding - Alpha version
Debugging phase 1 (following internal Alpha testing)
Production of Alpha version and delivery to client
Debugging phase 2 (following client Alpha testing)
Coding - Beta version
Debugging phase 3 (following internal Beta testing)
Production of Beta version and delivery to client
Debugging phase 4 (following client Beta testing)
Production of Final version and delivery to client
Debugging phase 5 & delivery (following client Final testing)
INTEGRATION
Delivery of content subset by client
Integration of content subset
Delivery of complementary content by client
Integration of complementary content
TESTING & ACCEPTANCE
Prepare test plan and test cases
Internal testing - Alpha version
Alpha testing by client
Internal testing - Beta version
Beta testing by client
Final testing by client
Acceptance by client
DEPLOYMENT AT CLIENT'S SITE
PROJECT MANAGEMENT (After Req'ts WP & before Closure)
PROJECT CLOSURE



PROJECT EXONE (Client's WBS)
REQUIREMENTS
Review and complete requirements & plan with contractor
Final discussion and agreement with contractor on req'ts & plan
SOFTWARE DESIGN
Write & check design specifications
Meet with contractor to discuss & agree on design
SOFTWARE IMPLEMENTATION
Coding, internal testing & bug fixing - Alpha version
Delivery of Alpha version by contractor
Debugging following Alpha testing
Coding, internal testing & bug fixing - Beta version
Delivery of Beta version by contractor
Debugging following Beta testing
Delivery of Final version by contractor
Debugging following Final testing & delivery for acceptance
CONTENT CREATION & INTEGRATION
Creation of content subset
Proofreading of content subset
Preparation of content subset & delivery to contractor
Integration of content subset
Creation of complementary content
Proofreading of complementary content
Preparation of complementary content & delivery to contractor
Integration of complementary content
TESTING & ACCEPTANCE
Prepare test plan and test cases
Alpha testing
Beta testing
Final testing
Acceptance
DEPLOYMENT
PROJECT MANAGEMENT (After Req'ts WP & before Closure)
CONTRACTOR'S PROJECT CLOSURE
OVERALL PROJECT CLOSURE

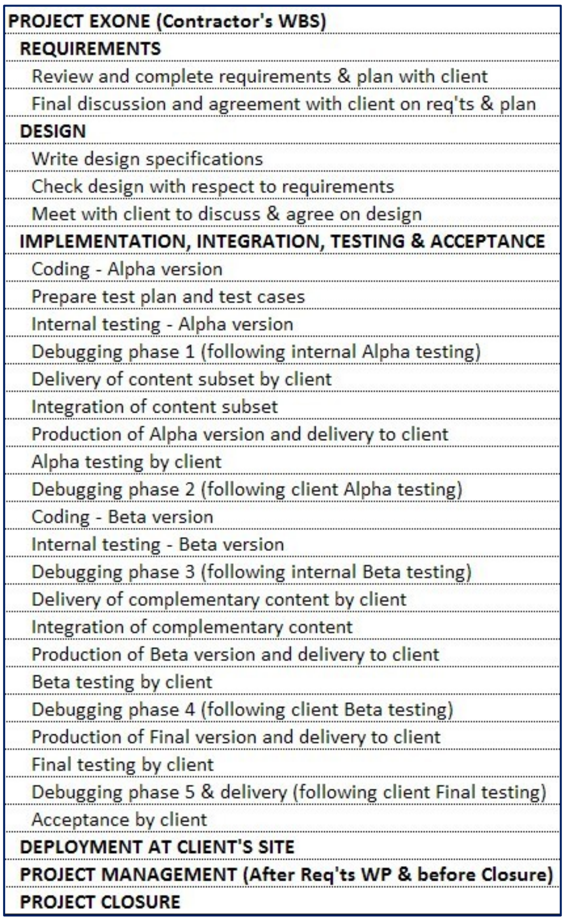
A project’s task list needs to be **exhaustive**, so it is generally more detailed than the diagram version of the WBS and may be more detailed than the “hand-drawn” network diagram.

Although it is not mandatory to do so, the name of the project, in this case “**PROJECT EXONE**”, is featured at the **highest level of the task list** on the first line of the above tables, so that **consolidated information at project level** can be viewed easily, for example the total duration, the overall schedule and the total cost of the project, which are automatically calculated by project management applications, as will be explained and illustrated further on in this chapter.

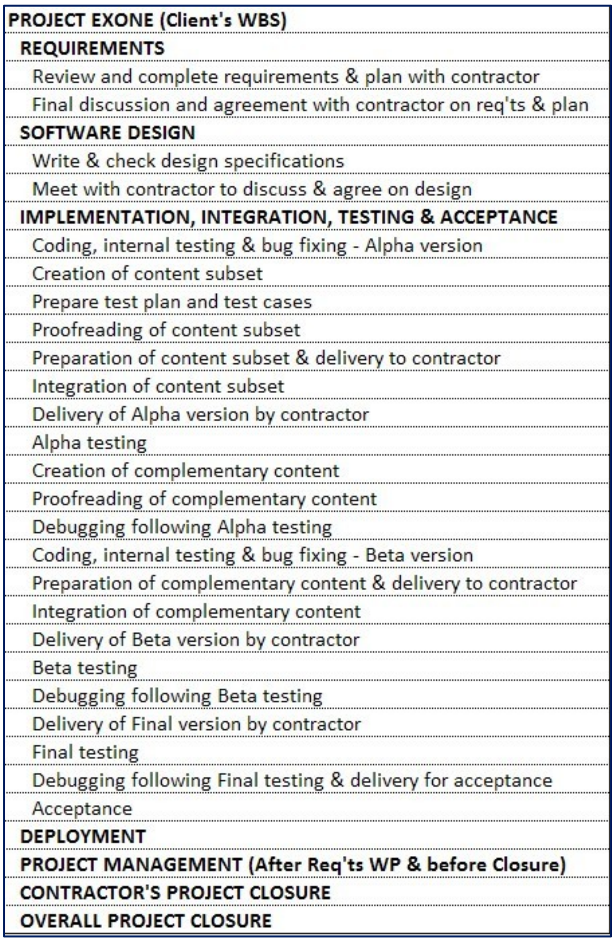
Note that the client's task list incorporates some of the tasks that appear in the contractor's task list, with an appropriate level of detail, in order to provide the client with an overall view of the project, including those contractor's WPs/tasks on which the client's WPs/tasks are dependent.

Also note that the client's list features two "Project closure" entries, the first one corresponding to the closure of the contractor's project, the second one concerning the closure of the project from the client's standpoint.

The set of tasks shown on the previous page may be presented with some of the work packages grouped together, and in a somewhat different order, as shown below.



PROJECT EXONE (Contractor's WBS)
REQUIREMENTS
Review and complete requirements & plan with client
Final discussion and agreement with client on req'ts & plan
DESIGN
Write design specifications
Check design with respect to requirements
Meet with client to discuss & agree on design
IMPLEMENTATION, INTEGRATION, TESTING & ACCEPTANCE
Coding - Alpha version
Prepare test plan and test cases
Internal testing - Alpha version
Debugging phase 1 (following internal Alpha testing)
Delivery of content subset by client
Integration of content subset
Production of Alpha version and delivery to client
Alpha testing by client
Debugging phase 2 (following client Alpha testing)
Coding - Beta version
Internal testing - Beta version
Debugging phase 3 (following internal Beta testing)
Delivery of complementary content by client
Integration of complementary content
Production of Beta version and delivery to client
Beta testing by client
Debugging phase 4 (following client Beta testing)
Production of Final version and delivery to client
Final testing by client
Debugging phase 5 & delivery (following client Final testing)
Acceptance by client
DEPLOYMENT AT CLIENT'S SITE
PROJECT MANAGEMENT (After Req'ts WP & before Closure)
PROJECT CLOSURE



PROJECT EXONE (Client's WBS)
REQUIREMENTS
Review and complete requirements & plan with contractor
Final discussion and agreement with contractor on req'ts & plan
SOFTWARE DESIGN
Write & check design specifications
Meet with contractor to discuss & agree on design
IMPLEMENTATION, INTEGRATION, TESTING & ACCEPTANCE
Coding, internal testing & bug fixing - Alpha version
Creation of content subset
Prepare test plan and test cases
Proofreading of content subset
Preparation of content subset & delivery to contractor
Integration of content subset
Delivery of Alpha version by contractor
Alpha testing
Creation of complementary content
Proofreading of complementary content
Debugging following Alpha testing
Coding, internal testing & bug fixing - Beta version
Preparation of complementary content & delivery to contractor
Integration of complementary content
Delivery of Beta version by contractor
Beta testing
Debugging following Beta testing
Delivery of Final version by contractor
Final testing
Debugging following Final testing & delivery for acceptance
Acceptance
DEPLOYMENT
PROJECT MANAGEMENT (After Req'ts WP & before Closure)
CONTRACTOR'S PROJECT CLOSURE
OVERALL PROJECT CLOSURE

That second example of task list shows **tasks in the order in which they should be executed**. This straightforward ordering of tasks is **recommended when using ProjectLibre** (an alternative to Microsoft Project), which in the case of more sophisticated ordering may flag certain task sequences as erroneous although they may actually be correct.

As compared to ProjectLibre, **Microsoft Project is a more powerful application** and allows more flexibility in ordering tasks.

Note that producing a project plan does not necessarily require the use of specialized project management software. A multi-purpose tool such as **a spreadsheet application** (eg Microsoft Excel) **can be used for projects that are not too complex** in terms of WBS and task sequencing.

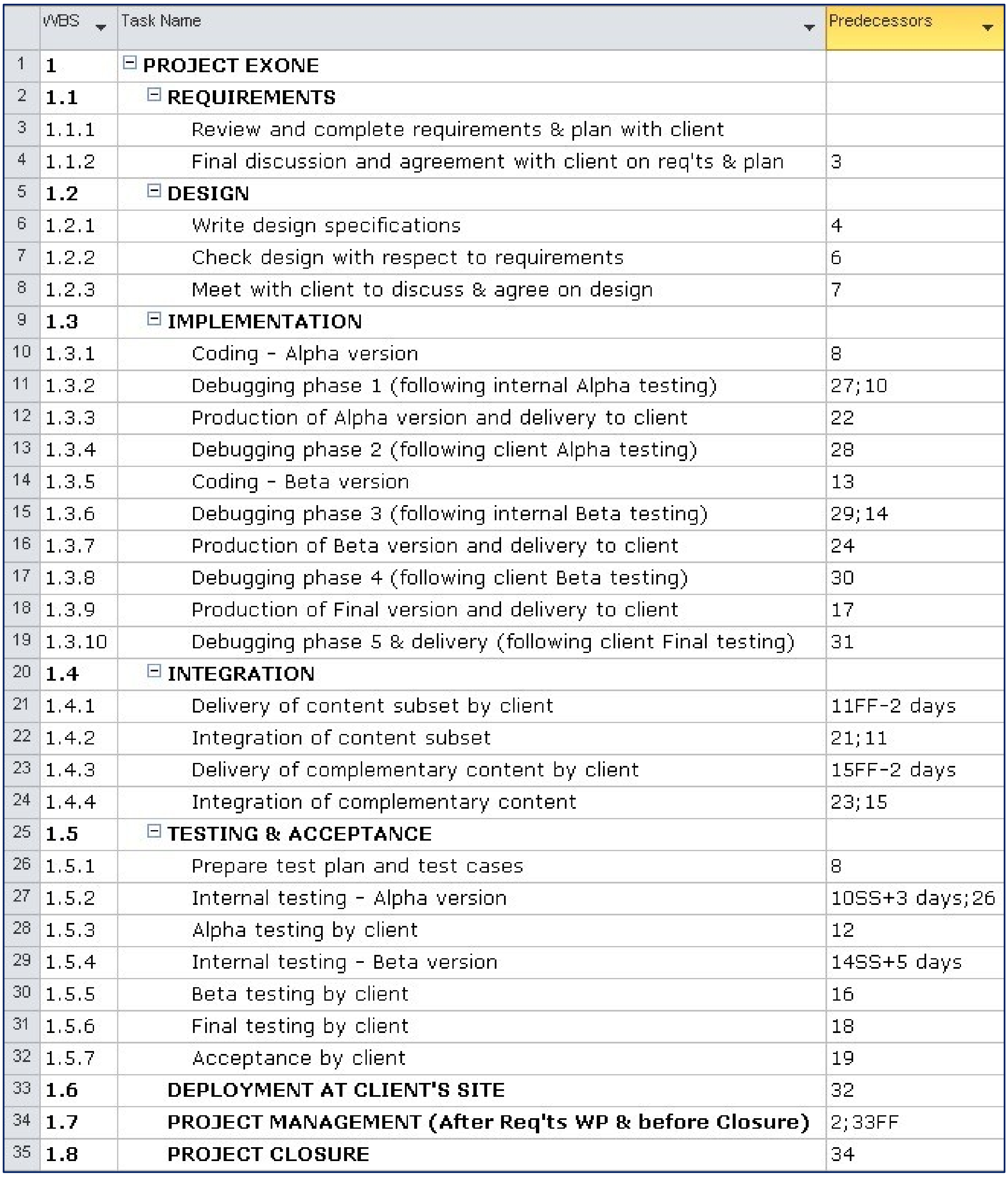
Specialized applications do however make project planning (and monitoring) easier thanks to **built-in tools** such as automatic scheduling and cost calculation, and to a **variety of "views"** which provide useful information such as resource usage and critical paths, as explained further on in this chapter.

Note that most of the screenshots concerning the EXONE project featured in subsequent sections of this guide are illustrations produced with Microsoft Project, not with ProjectLibre. Other screenshots represent documents produced with Microsoft Excel.

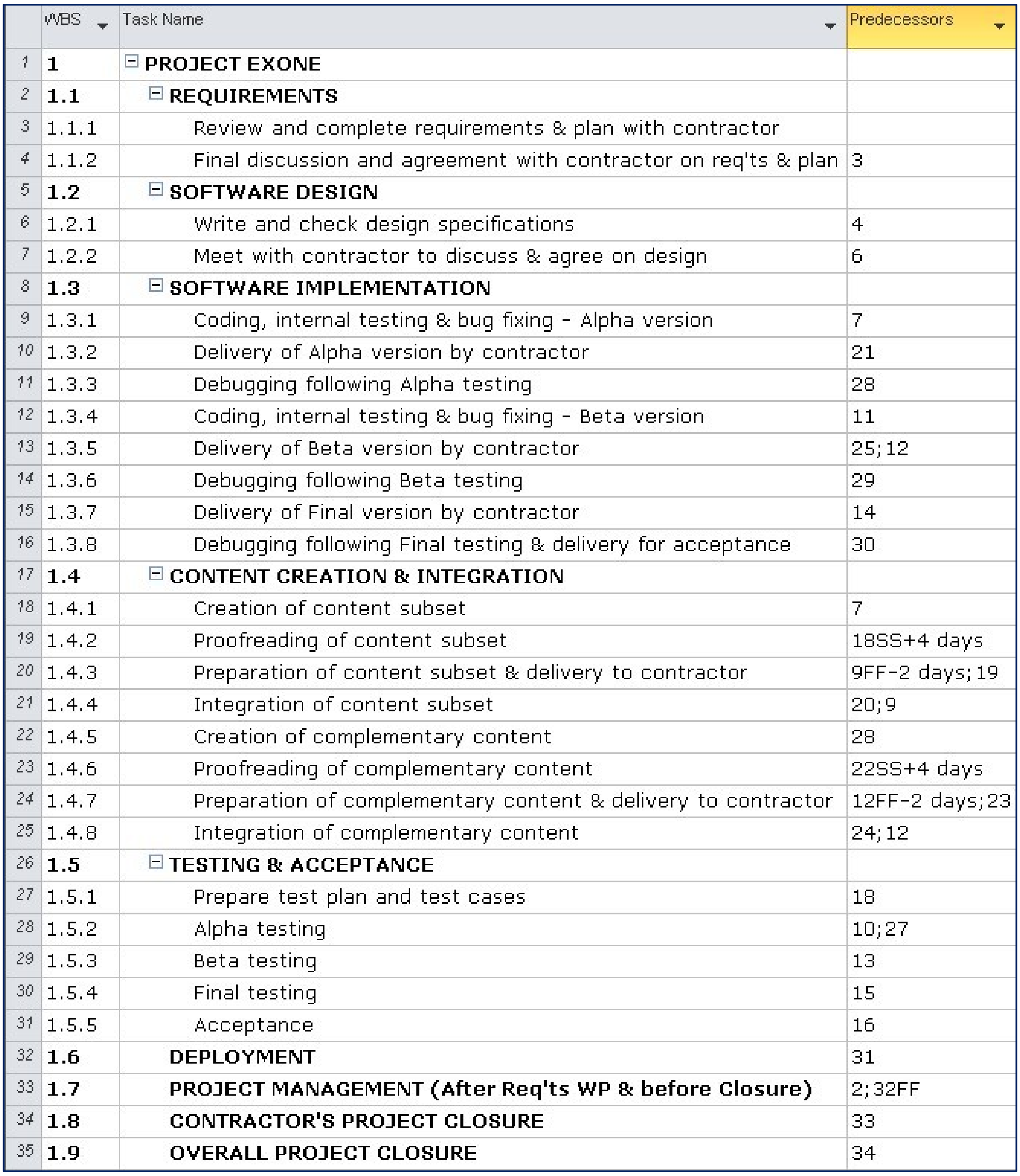
Task sequencing

The so-called “**Gantt view**” of a project plan built with MS Project (or ProjectLibre) consists of a table showing in the "Task name" column the comprehensive list of project tasks, hierarchically organized in work packages, and additional information in other columns.

One of those additional columns, labelled “**Predecessors**”, is where the **dependencies between tasks** are set, using the **line numbers** in the leftmost column of the table, as shown in the following illustrations for project EXONE, first for the contractor’s plan then for the client’s plan.



	WBS	Task Name	Predecessors
1	**1**	⊟ **PROJECT EXONE**	
2	**1.1**	⊟ **REQUIREMENTS**	
3	1.1.1	Review and complete requirements & plan with client	
4	1.1.2	Final discussion and agreement with client on req'ts & plan	3
5	**1.2**	⊟ **DESIGN**	
6	1.2.1	Write design specifications	4
7	1.2.2	Check design with respect to requirements	6
8	1.2.3	Meet with client to discuss & agree on design	7
9	**1.3**	⊟ **IMPLEMENTATION**	
10	1.3.1	Coding - Alpha version	8
11	1.3.2	Debugging phase 1 (following internal Alpha testing)	27;10
12	1.3.3	Production of Alpha version and delivery to client	22
13	1.3.4	Debugging phase 2 (following client Alpha testing)	28
14	1.3.5	Coding - Beta version	13
15	1.3.6	Debugging phase 3 (following internal Beta testing)	29;14
16	1.3.7	Production of Beta version and delivery to client	24
17	1.3.8	Debugging phase 4 (following client Beta testing)	30
18	1.3.9	Production of Final version and delivery to client	17
19	1.3.10	Debugging phase 5 & delivery (following client Final testing)	31
20	**1.4**	⊟ **INTEGRATION**	
21	1.4.1	Delivery of content subset by client	11FF-2 days
22	1.4.2	Integration of content subset	21;11
23	1.4.3	Delivery of complementary content by client	15FF-2 days
24	1.4.4	Integration of complementary content	23;15
25	**1.5**	⊟ **TESTING & ACCEPTANCE**	
26	1.5.1	Prepare test plan and test cases	8
27	1.5.2	Internal testing - Alpha version	10SS+3 days;26
28	1.5.3	Alpha testing by client	12
29	1.5.4	Internal testing - Beta version	14SS+5 days
30	1.5.5	Beta testing by client	16
31	1.5.6	Final testing by client	18
32	1.5.7	Acceptance by client	19
33	**1.6**	**DEPLOYMENT AT CLIENT'S SITE**	32
34	**1.7**	**PROJECT MANAGEMENT (After Req'ts WP & before Closure)**	2;33FF
35	**1.8**	**PROJECT CLOSURE**	34



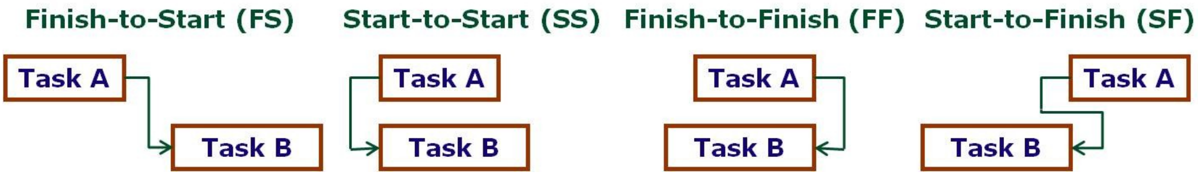
	WBS	Task Name	Predecessors
1	**1**	⊟ **PROJECT EXONE**	
2	**1.1**	⊟ **REQUIREMENTS**	
3	1.1.1	Review and complete requirements & plan with contractor	
4	1.1.2	Final discussion and agreement with contractor on req'ts & plan	3
5	**1.2**	⊟ **SOFTWARE DESIGN**	
6	1.2.1	Write and check design specifications	4
7	1.2.2	Meet with contractor to discuss & agree on design	6
8	**1.3**	⊟ **SOFTWARE IMPLEMENTATION**	
9	1.3.1	Coding, internal testing & bug fixing - Alpha version	7
10	1.3.2	Delivery of Alpha version by contractor	21
11	1.3.3	Debugging following Alpha testing	28
12	1.3.4	Coding, internal testing & bug fixing - Beta version	11
13	1.3.5	Delivery of Beta version by contractor	25;12
14	1.3.6	Debugging following Beta testing	29
15	1.3.7	Delivery of Final version by contractor	14
16	1.3.8	Debugging following Final testing & delivery for acceptance	30
17	**1.4**	⊟ **CONTENT CREATION & INTEGRATION**	
18	1.4.1	Creation of content subset	7
19	1.4.2	Proofreading of content subset	18SS+4 days
20	1.4.3	Preparation of content subset & delivery to contractor	9FF-2 days;19
21	1.4.4	Integration of content subset	20;9
22	1.4.5	Creation of complementary content	28
23	1.4.6	Proofreading of complementary content	22SS+4 days
24	1.4.7	Preparation of complementary content & delivery to contractor	12FF-2 days;23
25	1.4.8	Integration of complementary content	24;12
26	**1.5**	⊟ **TESTING & ACCEPTANCE**	
27	1.5.1	Prepare test plan and test cases	18
28	1.5.2	Alpha testing	10;27
29	1.5.3	Beta testing	13
30	1.5.4	Final testing	15
31	1.5.5	Acceptance	16
32	**1.6**	**DEPLOYMENT**	31
33	**1.7**	**PROJECT MANAGEMENT (After Req'ts WP & before Closure)**	2;32FF
34	**1.8**	**CONTRACTOR'S PROJECT CLOSURE**	33
35	**1.9**	**OVERALL PROJECT CLOSURE**	34

Information entered in the Predecessors column determines the sequencing of tasks (as shown in the "hand-drawn" network diagram but sometimes with a greater level of detail, and, possibly, with slight sequencing differences).

Note that **"Predecessor" information is always entered at the lowest level of the hierarchical list**, namely at task level, or WP level if the WP has no explicit subordinate tasks.

Precedence Diagramming Method

Task sequencing is performed by applying (sometimes unconsciously!) the so-called "Precedence Diagramming Method (**PDM**)", which includes four types of dependencies or (chrono)logical relationships between tasks, as illustrated below.



- **Finish-to-Start (FS)**: task B cannot start until task A has been completed.

FS is the **most commonly used type of relationship**. It is set by **default** with MS Project and ProjectLibre.
For example, in the contractor's tabular view of project EXONE, "Write design specifications" has "Final discussion and agreement with client on req'ts & plan" as a predecessor, with an implicit **FS** relationship, meaning that design cannot start before agreement on the requirements has been reached with the client.

- **Start-to-Start (SS)**: the start of task B is dependent on the start of task A.

For example, the translation of web pages from one language to another cannot start before writing pages in the original language has started. The translation task may however start some time after writing has begun but without waiting for all pages to be written and proofread. Likewise, as shown in the client's tabular view of project EXONE, proofreading of content is to start some time after the start of content creation. In such cases there is an "**SS**" relationship between the tasks.
Other example: debugging can be started as soon as a bug has been reported in the "bug tracking system" by the testers, without waiting for all bugs to be documented.
In both examples, task B will be completed some time after the end of task A.

- **Finish-to-Finish (FF)**: completion of task B is dependent on completion of task A.

For example, in project EXONE, the "Project management" WP has the "Deployment" WP as a predecessor with an "**FF**" relationship, which means that both WPs are scheduled to be completed at the same time, both before project closure. Note that this "FF" relationship was not shown in the EXONE network diagrams featured a few pages back, but was introduced for pedagogical purposes in the detailed tabular view of task sequencing.

- **Start-to-Finish (SF)**: task B cannot be completed until task A has started.

SF is the least commonly used relationship, the other three generally being sufficient to describe the relationships between tasks in a project, with the possible addition of so-called "lags" and "leads", which are explained below.

Lags and leads

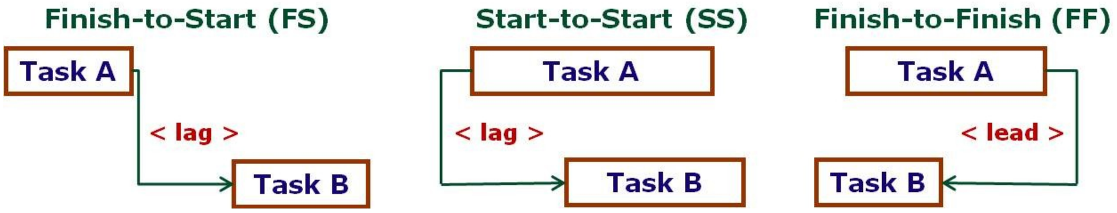
A **lag** is a situation where a **delay** is required between the start of a task and the start of its successor, or between the completion of a task and the completion of its successor.

Using the contractor's plan for project EXONE as an example, assuming that internal testing of the Alpha version (task 1.5.2 – line 27) is expected to begin three days after the start of coding (task 1.3.1 – line 10), then the relationship between these tasks should be defined as "10SS + 3 days".

A **lead** is a situation where a successor task can or must be started or completed sometime **before** its predecessor is started or completed.

For example, in the contractor's plan for project EXONE, it is assumed that the delivery of the content subset by the client (task 1.4.1 – line 21) is required by the contractor, as a safety margin, two days before completion of the first debugging phase (task 1.3.2 – line 11). The relationship between these tasks is therefore defined as "11FF - 2 days". Likewise, in the client's plan, the scheduling constraint imposed by the contractor (and accepted by the client) for the delivery of the content subset (task 1.4.3 – line 20) is featured as a "9FF - 2 days" relationship, where line 9 corresponds to "Coding, internal testing & bug fixing - Alpha version".

Lags are often applied to FS or SS relationships and leads are often used with FF relationships, as shown in the following illustration (SF relationship deliberately omitted, to keep things simple…).



The **lengths of lags and leads** set at an early stage of the planning process are often rough estimates which **need to be adjusted** after task durations have been established.

Estimate task resources

An in-depth analysis of the tasks to be performed should lead to the identification and evaluation of the **types and quantities of resources** required for each task. Resources include people, facilities, materials, equipment, supplies, services, etc.

Defining and estimating resources required for a project is a difficult exercise, and it is obviously easier for a seasoned PM than it is for a junior PM. The PM may use his **experience** with previous projects, information in the company's project **knowledge base** (if there is one…) and **advice** from his peers or from any other person liable to provide help, in particular people (editors, developers, etc.) who have already performed tasks similar to those for which required resources need to be estimated.

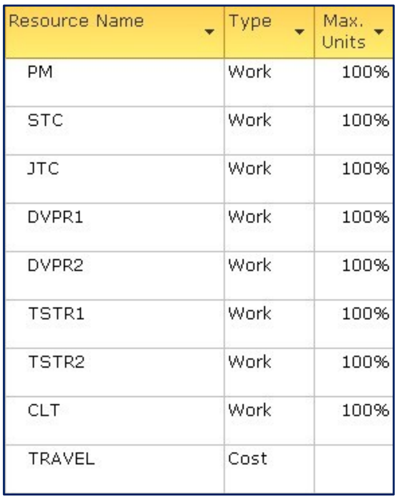
There may be several approaches for the execution of a given task, each with different resource requirements in terms of personnel, equipment, etc. For some tasks, a "**make or buy**" evaluation may be needed to choose between internal and external resources.

The **maximum lead time imposed by the overall schedule** of the project for any given task must be taken into account for the estimation of required resources, since there is a **close relationship between the amount of time required to complete a task and the resources assigned to it**.

Note that there is often a limit to the number of people it is reasonable to assign to any given task: assigning too many resources may be counterproductive due for example to increased communication and coordination complexity.

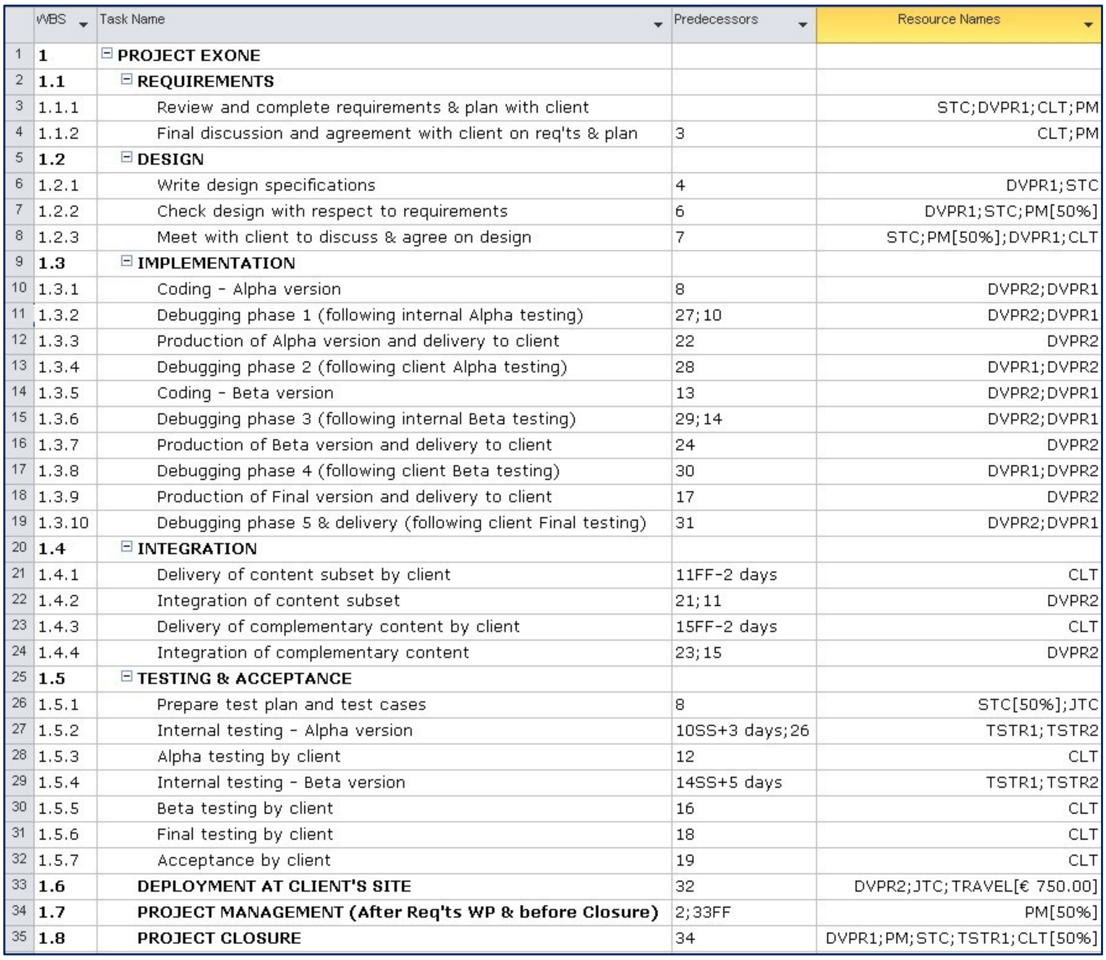
If the possibility of hiring and subcontracting is excluded, the assignment of human resources to tasks will be **constrained** and limited to the "in-house" people available for the project.

In the following Gantt views of the plan for project EXONE, first for the contractor then for the client, the "Resource names" column shows the resources assigned to the various WPs/tasks. The resources are chosen from the list prepared in the "**Resource sheet**", as shown below for the contractor and on the next page for the client.



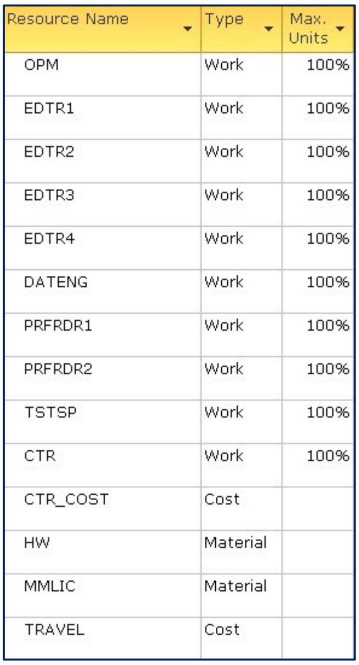
Resource Name	Type	Max. Units
PM	Work	100%
STC	Work	100%
JTC	Work	100%
DVPR1	Work	100%
DVPR2	Work	100%
TSTR1	Work	100%
TSTR2	Work	100%
CLT	Work	100%
TRAVEL	Cost	

Abbreviations are used here, except for TRAVEL, in order to **keep the "Resource Names" column in the Gantt view as narrow as possible** for presentation purposes (PM = project manager; STC = senior technical consultant; JTC = junior technical consultant; DVPR1 & 2 = developers 1 & 2; TSTR1 & 2 = testers 1 & 2; CLT = client).



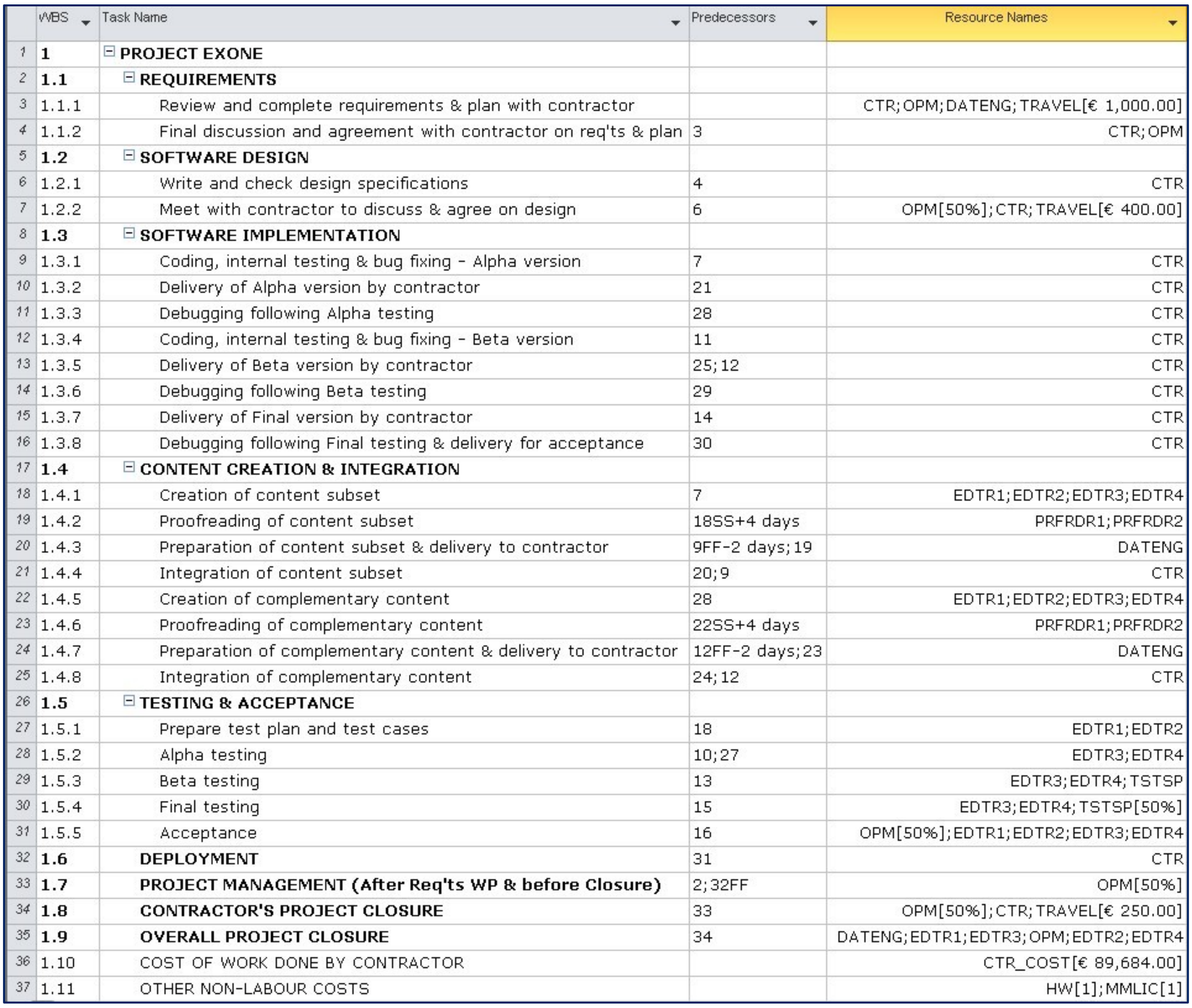
	WBS	Task Name	Predecessors	Resource Names
1	**1**	**PROJECT EXONE**		
2	**1.1**	**REQUIREMENTS**		
3	1.1.1	Review and complete requirements & plan with client		STC;DVPR1;CLT;PM
4	1.1.2	Final discussion and agreement with client on req'ts & plan	3	CLT;PM
5	**1.2**	**DESIGN**		
6	1.2.1	Write design specifications	4	DVPR1;STC
7	1.2.2	Check design with respect to requirements	6	DVPR1;STC;PM[50%]
8	1.2.3	Meet with client to discuss & agree on design	7	STC;PM[50%];DVPR1;CLT
9	**1.3**	**IMPLEMENTATION**		
10	1.3.1	Coding - Alpha version	8	DVPR2;DVPR1
11	1.3.2	Debugging phase 1 (following internal Alpha testing)	27;10	DVPR2;DVPR1
12	1.3.3	Production of Alpha version and delivery to client	22	DVPR2
13	1.3.4	Debugging phase 2 (following client Alpha testing)	28	DVPR1;DVPR2
14	1.3.5	Coding - Beta version	13	DVPR2;DVPR1
15	1.3.6	Debugging phase 3 (following internal Beta testing)	29;14	DVPR2;DVPR1
16	1.3.7	Production of Beta version and delivery to client	24	DVPR2
17	1.3.8	Debugging phase 4 (following client Beta testing)	30	DVPR1;DVPR2
18	1.3.9	Production of Final version and delivery to client	17	DVPR2
19	1.3.10	Debugging phase 5 & delivery (following client Final testing)	31	DVPR2;DVPR1
20	**1.4**	**INTEGRATION**		
21	1.4.1	Delivery of content subset by client	11FF-2 days	CLT
22	1.4.2	Integration of content subset	21;11	DVPR2
23	1.4.3	Delivery of complementary content by client	15FF-2 days	CLT
24	1.4.4	Integration of complementary content	23;15	DVPR2
25	**1.5**	**TESTING & ACCEPTANCE**		
26	1.5.1	Prepare test plan and test cases	8	STC[50%];JTC
27	1.5.2	Internal testing - Alpha version	10SS+3 days;26	TSTR1;TSTR2
28	1.5.3	Alpha testing by client	12	CLT
29	1.5.4	Internal testing - Beta version	14SS+5 days	TSTR1;TSTR2
30	1.5.5	Beta testing by client	16	CLT
31	1.5.6	Final testing by client	18	CLT
32	1.5.7	Acceptance by client	19	CLT
33	**1.6**	**DEPLOYMENT AT CLIENT'S SITE**	32	DVPR2;JTC;TRAVEL[€ 750.00]
34	**1.7**	**PROJECT MANAGEMENT (After Req'ts WP & before Closure)**	2;33FF	PM[50%]
35	**1.8**	**PROJECT CLOSURE**	34	DVPR1;PM;STC;TSTR1;CLT[50%]

Note that the detail of the client's resources does not need to be featured in the contractor's plan, so the use of "CLT" is sufficient. Likewise, in the client's plan, "CTR" is a sufficient representation of the contractor's resources, as featured in the following illustration.



Resource Name	Type	Max. Units
OPM	Work	100%
EDTR1	Work	100%
EDTR2	Work	100%
EDTR3	Work	100%
EDTR4	Work	100%
DATENG	Work	100%
PRFRDR1	Work	100%
PRFRDR2	Work	100%
TSTSP	Work	100%
CTR	Work	100%
CTR_COST	Cost	
HW	Material	
MMLIC	Material	
TRAVEL	Cost	

Like the contractor's resource names, the client's **resource names** are **abbreviated**, with the exception of TRAVEL (OPM = overall project manager; EDTR1, 2, 3 & 4 = editors 1, 2, 3 & 4; DATENG = data engineer; PRFRDR1 & 2 = proofreaders 1 & 2; TSTSP = testing service provider; CTR = contractor, CTR_COST = cost (price) of contractor's work; HW = hardware (& software); MMLIC = multimedia asset licences).



	WBS	Task Name	Predecessors	Resource Names
1	**1**	⊟ **PROJECT EXONE**		
2	**1.1**	⊟ **REQUIREMENTS**		
3	1.1.1	Review and complete requirements & plan with contractor		CTR;OPM;DATENG;TRAVEL[€ 1,000.00]
4	1.1.2	Final discussion and agreement with contractor on req'ts & plan	3	CTR;OPM
5	**1.2**	⊟ **SOFTWARE DESIGN**		
6	1.2.1	Write and check design specifications	4	CTR
7	1.2.2	Meet with contractor to discuss & agree on design	6	OPM[50%];CTR;TRAVEL[€ 400.00]
8	**1.3**	⊟ **SOFTWARE IMPLEMENTATION**		
9	1.3.1	Coding, internal testing & bug fixing - Alpha version	7	CTR
10	1.3.2	Delivery of Alpha version by contractor	21	CTR
11	1.3.3	Debugging following Alpha testing	28	CTR
12	1.3.4	Coding, internal testing & bug fixing - Beta version	11	CTR
13	1.3.5	Delivery of Beta version by contractor	25;12	CTR
14	1.3.6	Debugging following Beta testing	29	CTR
15	1.3.7	Delivery of Final version by contractor	14	CTR
16	1.3.8	Debugging following Final testing & delivery for acceptance	30	CTR
17	**1.4**	⊟ **CONTENT CREATION & INTEGRATION**		
18	1.4.1	Creation of content subset	7	EDTR1;EDTR2;EDTR3;EDTR4
19	1.4.2	Proofreading of content subset	18SS+4 days	PRFRDR1;PRFRDR2
20	1.4.3	Preparation of content subset & delivery to contractor	9FF-2 days;19	DATENG
21	1.4.4	Integration of content subset	20;9	CTR
22	1.4.5	Creation of complementary content	28	EDTR1;EDTR2;EDTR3;EDTR4
23	1.4.6	Proofreading of complementary content	22SS+4 days	PRFRDR1;PRFRDR2
24	1.4.7	Preparation of complementary content & delivery to contractor	12FF-2 days;23	DATENG
25	1.4.8	Integration of complementary content	24;12	CTR
26	**1.5**	⊟ **TESTING & ACCEPTANCE**		
27	1.5.1	Prepare test plan and test cases	18	EDTR1;EDTR2
28	1.5.2	Alpha testing	10;27	EDTR3;EDTR4
29	1.5.3	Beta testing	13	EDTR3;EDTR4;TSTSP
30	1.5.4	Final testing	15	EDTR3;EDTR4;TSTSP[50%]
31	1.5.5	Acceptance	16	OPM[50%];EDTR1;EDTR2;EDTR3;EDTR4
32	**1.6**	**DEPLOYMENT**	31	CTR
33	**1.7**	**PROJECT MANAGEMENT (After Req'ts WP & before Closure)**	2;32FF	OPM[50%]
34	**1.8**	**CONTRACTOR'S PROJECT CLOSURE**	33	OPM[50%];CTR;TRAVEL[€ 250.00]
35	**1.9**	**OVERALL PROJECT CLOSURE**	34	DATENG;EDTR1;EDTR3;OPM;EDTR2;EDTR4
36	1.10	COST OF WORK DONE BY CONTRACTOR		CTR_COST[€ 89,684.00]
37	1.11	OTHER NON-LABOUR COSTS		HW[1];MMLIC[1]

Like task predecessors, **resources are always assigned at the lowest level of the hierarchical list**, ie at task level, or WP level if the WP has no explicit subordinate tasks.

Note that **resources may not or need not be available 100% of their time**.
For example, in the contractor's plan for project EXONE the STC is expected to spend only 50% of his time (coded as "STC[50%]") on test planning and writing test cases, and the PM is assumed to dedicate 100% of his time to the project during its Requirements and Closure phases, but only 50% of his time (coded as "PM[50%]") during the other phases.

Also note that, in principle, **people should not work more than 100% of their time!** For example, in the client's plan for project EXONE "OPM[50%]" has been assigned to tasks 1.2.2 and 1.5.5 in order to avoid overloading the OPM, who is also busy 50% of his time with WP 1.7, namely "(OVERALL) PROJECT MANAGEMENT".

Resource overload is a frequent error made in project planning, which the PM should be careful to avoid (apart from justified exceptions…). The "**Resource usage**" function of MS Project or ProjectLibre can be very helpful in identifying such overload situations.

A **simulation**, in "real-life" conditions or on paper, may be needed in order to estimate resource requirements for certain tasks. Project management software applications can be used to make such simulations, but a spreadsheet is often quite sufficient.

> *One of the editorial WPs of the EHM project included a task which consisted in indexing each encyclopedic article by assigning one or several "triplets" of metadata to it, each triplet including a topic, a geographical location and a period. Because there was no previous experience of such a task, a group of editors performed it on a representative sample of articles, using editing tools developed for that purpose. The average amount of time required to index an article was measured. As a result, the number of persons required for the task was easily calculated, given the maximum duration of 6 months imposed by the overall schedule of the project.*
>
> - *Number of articles to index: 50,000*
> - *Number of minutes per article (in the sample): 5*
> - *Total number of minutes required: 250,000*
> - *Margin of error (as a precaution…): 15%*
> - *Number of minutes required after adjustment: 287,500*
> - *Number of hours: 4,792*
> - *Number of 7-hour days: 685*
> - *Number of 5-day weeks: 137*
> - *Number of 4-week months: 34 (A)*
> - *Maximum lead time in months: 6 (B)*
> - *Number of persons required: 6 (= A/B, rounded)*
>
> *This estimate led to the hiring of 5 "editors-indexers" with a 6-month temporary contract (the 6th person required was already on board).*
> *The progress of this activity was measured on a regular and frequent basis, in order to detect any possible deviation with respect to the schedule that had been fixed.*

Apart from **human resources** (in-house as well as external), **other types of resources** are generally required for the execution of a project, for example (non-exhaustive list):

- offices (building, floor space, furniture, utilities, etc.),
- workstations (computers and software licences),
- printers, scanners, audiovisual equipment,
- servers, hosting service, domain name, etc.,
- consumables,
- documentation (reference books, etc.),
- multimedia asset licences
- travel & accommodation (eg for certain meetings).

Contractors may have equipment of their own. In some cases however, the client will need to provide equipment which is specifically required for the subcontracted work.

> *Two servers required for the EHM "back office" were purchased by Hachette and installed at the contractor's facility for the duration of the development and debugging of the tools. They were then moved and put into operation at Hachette.*

The result of the "Estimate task resources" process is a set of additional attributes for tasks in the task list and a "**resource schedule**" providing the list of all resources required for the project. The list is generally organized by type of resource: people (with a definition of their skills, roles and responsibilities), equipment, materials, etc.

The "**Resource sheet**" of a plan built with MS Project or ProjectLibre identifies human resources by their names and/or abbreviations and other attributes (eg percentage of time spent on the project, cost), as shown further on in this chapter.

As shown on previous pages in the examples for project EXONE, resources other than people, which may be called "**non-labour resources**", can also be featured in the resource sheet of a project management application, and will need to be assigned to existing tasks, or to "dummy" tasks specially created for cost calculation purposes.
For example, in the contractor's plan for project EXONE, "TRAVEL" has been assigned to the "DEPLOYMENT AT CLIENT'S SITE" WP since DVPR2 and STC will need to travel to the client's site to do the corresponding work.
In the client's plan for project EXONE, "TRAVEL" has been assigned to those tasks/WPs corresponding to meetings for which OPM and DATENG need to travel to the contractor's place of work. The "CTR_COST" resource, namely the cost (price) of the work to be done by the contractor, and the "HW" and "MMLIC" resources have been assigned to dummy tasks/WPs (1.10 & 1.11) so that their costs are taken into account by the project management software application for the calculation of the total cost of the project, as illustrated in the "Estimate costs" section of this chapter.

Estimate task durations

This process consists in estimating the **duration** (ie **lead time) required to complete the work to be done** for each task, **with the resources assigned to it**.

"Estimate task durations" and "Estimate task resources" processes are of course closely related. Several **iterations** may be required in order to complete these processes.
For some work packages or tasks, **durations may be imposed** and therefore need to be accepted as **planning constraints**.
For example, the project owner may require a prototype to be available within 6 weeks from the start of project execution, regardless of the number and availability of resources for the corresponding work. As another example of scheduling constraint, resources required for a given task may be available to work on the task only within certain time frames.

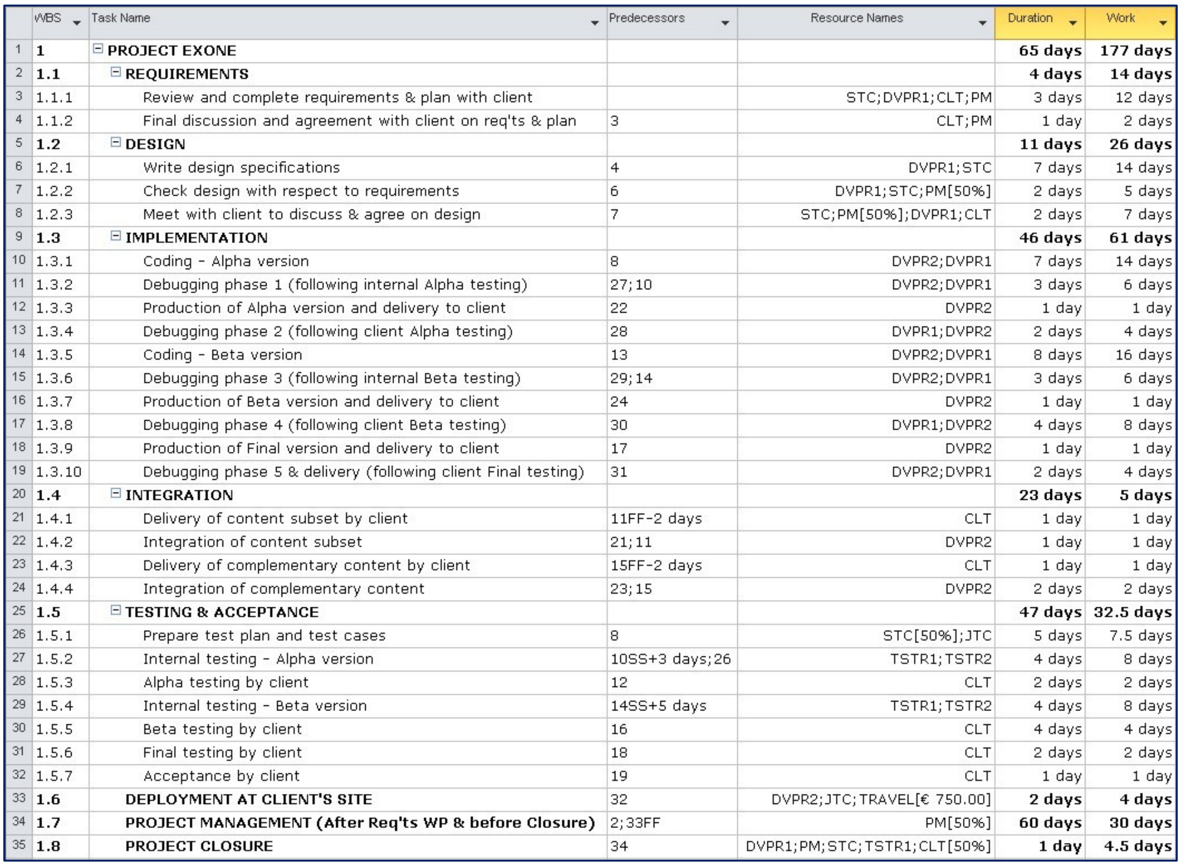
Estimating the **workload** (number of persons x duration) for a given task may sometimes be easier than estimating its duration. In this case, the task duration will be derived from the workload and the resources assigned to the task.

As mentioned previously, human resources may not or need not be available 100% of their time for a given task, which obviously has an impact on the duration of the task. For example, if a task for which a workload of 10 "person-days" has been estimated is assigned to a person working only the first 4 days of any week, then the actual lead time for the completion of this task will be 12 working days. The **duration** attributed to this task should therefore be 12 days, whereas the actual **workload** is 10 person-days.

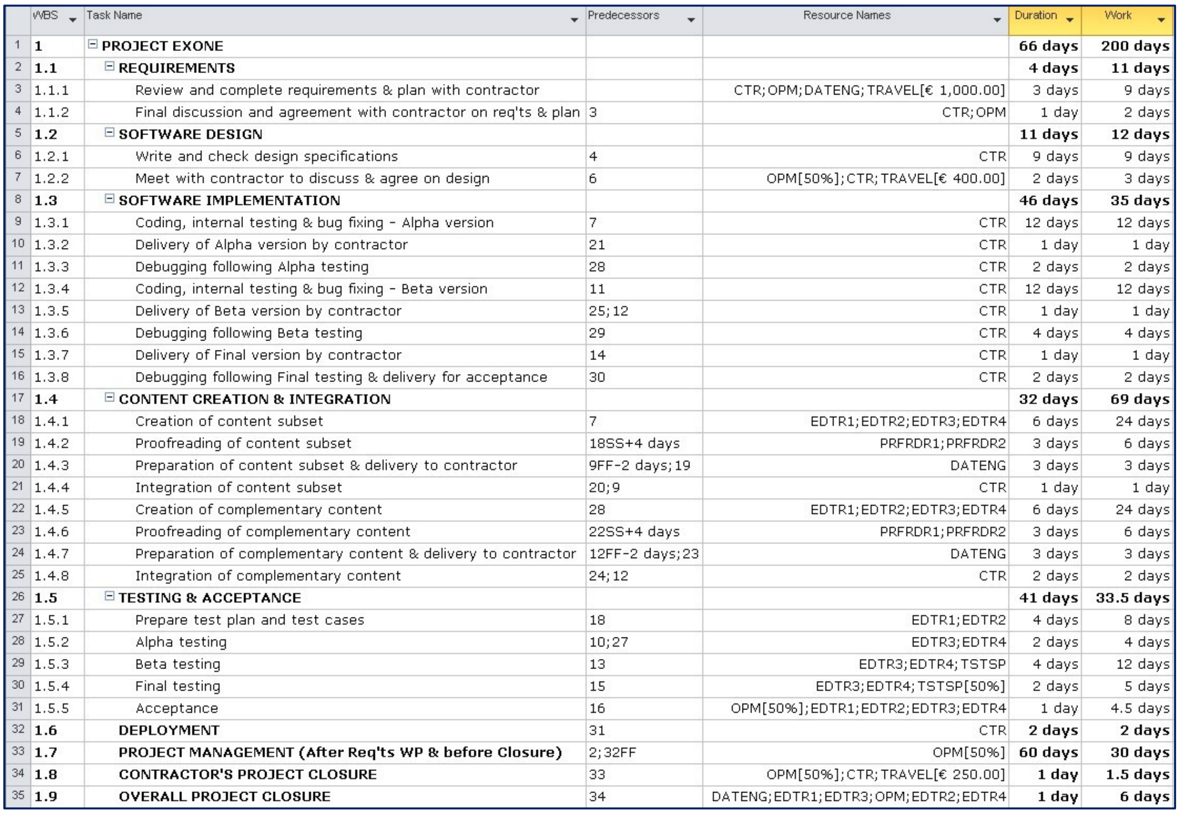
It is the **workload**, not the duration, which is taken into account for **cost calculation**.

Note that **durations** are always expressed in units of **work time** (workdays, workweeks, etc.), ie **excluding weekends and holidays** (except of course if the project requires working on some or all weekends and/or holidays…).

Project management applications automatically calculate workloads from duration and resource information (or calculate durations from workload and resource information), as shown in the following examples for project EXONE (contractor's view then client's view).



	WBS	Task Name	Predecessors	Resource Names	Duration	Work
1	**1**	**⊟ PROJECT EXONE**			**65 days**	**177 days**
2	**1.1**	**⊟ REQUIREMENTS**			**4 days**	**14 days**
3	1.1.1	Review and complete requirements & plan with client		STC;DVPR1;CLT;PM	3 days	12 days
4	1.1.2	Final discussion and agreement with client on req'ts & plan	3	CLT;PM	1 day	2 days
5	**1.2**	**⊟ DESIGN**			**11 days**	**26 days**
6	1.2.1	Write design specifications	4	DVPR1;STC	7 days	14 days
7	1.2.2	Check design with respect to requirements	6	DVPR1;STC;PM[50%]	2 days	5 days
8	1.2.3	Meet with client to discuss & agree on design	7	STC;PM[50%];DVPR1;CLT	2 days	7 days
9	**1.3**	**⊟ IMPLEMENTATION**			**46 days**	**61 days**
10	1.3.1	Coding - Alpha version	8	DVPR2;DVPR1	7 days	14 days
11	1.3.2	Debugging phase 1 (following internal Alpha testing)	27;10	DVPR2;DVPR1	3 days	6 days
12	1.3.3	Production of Alpha version and delivery to client	22	DVPR2	1 day	1 day
13	1.3.4	Debugging phase 2 (following client Alpha testing)	28	DVPR1;DVPR2	2 days	4 days
14	1.3.5	Coding - Beta version	13	DVPR2;DVPR1	8 days	16 days
15	1.3.6	Debugging phase 3 (following internal Beta testing)	29;14	DVPR2;DVPR1	3 days	6 days
16	1.3.7	Production of Beta version and delivery to client	24	DVPR2	1 day	1 day
17	1.3.8	Debugging phase 4 (following client Beta testing)	30	DVPR1;DVPR2	4 days	8 days
18	1.3.9	Production of Final version and delivery to client	17	DVPR2	1 day	1 day
19	1.3.10	Debugging phase 5 & delivery (following client Final testing)	31	DVPR2;DVPR1	2 days	4 days
20	**1.4**	**⊟ INTEGRATION**			**23 days**	**5 days**
21	1.4.1	Delivery of content subset by client	11FF-2 days	CLT	1 day	1 day
22	1.4.2	Integration of content subset	21;11	DVPR2	1 day	1 day
23	1.4.3	Delivery of complementary content by client	15FF-2 days	CLT	1 day	1 day
24	1.4.4	Integration of complementary content	23;15	DVPR2	2 days	2 days
25	**1.5**	**⊟ TESTING & ACCEPTANCE**			**47 days**	**32.5 days**
26	1.5.1	Prepare test plan and test cases	8	STC[50%];JTC	5 days	7.5 days
27	1.5.2	Internal testing - Alpha version	10SS+3 days;26	TSTR1;TSTR2	4 days	8 days
28	1.5.3	Alpha testing by client	12	CLT	2 days	2 days
29	1.5.4	Internal testing - Beta version	14SS+5 days	TSTR1;TSTR2	4 days	8 days
30	1.5.5	Beta testing by client	16	CLT	4 days	4 days
31	1.5.6	Final testing by client	18	CLT	2 days	2 days
32	1.5.7	Acceptance by client	19	CLT	1 day	1 day
33	**1.6**	**DEPLOYMENT AT CLIENT'S SITE**	32	DVPR2;JTC;TRAVEL[€ 750.00]	**2 days**	**4 days**
34	**1.7**	**PROJECT MANAGEMENT (After Req'ts WP & before Closure)**	2;33FF	PM[50%]	**60 days**	**30 days**
35	**1.8**	**PROJECT CLOSURE**	34	DVPR1;PM;STC;TSTR1;CLT[50%]	**1 day**	**4.5 days**



	WBS	Task Name	Predecessors	Resource Names	Duration	Work
1	**1**	**⊟ PROJECT EXONE**			**66 days**	**200 days**
2	**1.1**	**⊟ REQUIREMENTS**			**4 days**	**11 days**
3	1.1.1	Review and complete requirements & plan with contractor		CTR;OPM;DATENG;TRAVEL[€ 1,000.00]	3 days	9 days
4	1.1.2	Final discussion and agreement with contractor on req'ts & plan	3	CTR;OPM	1 day	2 days
5	**1.2**	**⊟ SOFTWARE DESIGN**			**11 days**	**12 days**
6	1.2.1	Write and check design specifications	4	CTR	9 days	9 days
7	1.2.2	Meet with contractor to discuss & agree on design	6	OPM[50%];CTR;TRAVEL[€ 400.00]	2 days	3 days
8	**1.3**	**⊟ SOFTWARE IMPLEMENTATION**			**46 days**	**35 days**
9	1.3.1	Coding, internal testing & bug fixing - Alpha version	7	CTR	12 days	12 days
10	1.3.2	Delivery of Alpha version by contractor	21	CTR	1 day	1 day
11	1.3.3	Debugging following Alpha testing	28	CTR	2 days	2 days
12	1.3.4	Coding, internal testing & bug fixing - Beta version	11	CTR	12 days	12 days
13	1.3.5	Delivery of Beta version by contractor	25;12	CTR	1 day	1 day
14	1.3.6	Debugging following Beta testing	29	CTR	4 days	4 days
15	1.3.7	Delivery of Final version by contractor	14	CTR	1 day	1 day
16	1.3.8	Debugging following Final testing & delivery for acceptance	30	CTR	2 days	2 days
17	**1.4**	**⊟ CONTENT CREATION & INTEGRATION**			**32 days**	**69 days**
18	1.4.1	Creation of content subset	7	EDTR1;EDTR2;EDTR3;EDTR4	6 days	24 days
19	1.4.2	Proofreading of content subset	18SS+4 days	PRFRDR1;PRFRDR2	3 days	6 days
20	1.4.3	Preparation of content subset & delivery to contractor	9FF-2 days;19	DATENG	3 days	3 days
21	1.4.4	Integration of content subset	20;9	CTR	1 day	1 day
22	1.4.5	Creation of complementary content	28	EDTR1;EDTR2;EDTR3;EDTR4	6 days	24 days
23	1.4.6	Proofreading of complementary content	22SS+4 days	PRFRDR1;PRFRDR2	3 days	6 days
24	1.4.7	Preparation of complementary content & delivery to contractor	12FF-2 days;23	DATENG	3 days	3 days
25	1.4.8	Integration of complementary content	24;12	CTR	2 days	2 days
26	**1.5**	**⊟ TESTING & ACCEPTANCE**			**41 days**	**33.5 days**
27	1.5.1	Prepare test plan and test cases	18	EDTR1;EDTR2	4 days	8 days
28	1.5.2	Alpha testing	10;27	EDTR3;EDTR4	2 days	4 days
29	1.5.3	Beta testing	13	EDTR3;EDTR4;TSTSP	4 days	12 days
30	1.5.4	Final testing	15	EDTR3;EDTR4;TSTSP[50%]	2 days	5 days
31	1.5.5	Acceptance	16	OPM[50%];EDTR1;EDTR2;EDTR3;EDTR4	1 day	4.5 days
32	**1.6**	**DEPLOYMENT**	31	CTR	**2 days**	**2 days**
33	**1.7**	**PROJECT MANAGEMENT (After Req'ts WP & before Closure)**	2;32FF	OPM[50%]	**60 days**	**30 days**
34	**1.8**	**CONTRACTOR'S PROJECT CLOSURE**	33	OPM[50%];CTR;TRAVEL[€ 250.00]	**1 day**	**1.5 days**
35	**1.9**	**OVERALL PROJECT CLOSURE**	34	DATENG;EDTR1;EDTR3;OPM;EDTR2;EDTR4	**1 day**	**6 days**

Note that, like task predecessors and resources, **durations are always assigned at the lowest level of the hierarchical list**, ie at task level, or WP level if the WP has no explicit subordinate tasks.

Also note that **the duration of a work package (WP) is not always the sum of the durations of its subordinate tasks**, since such tasks are not necessarily strictly consecutive or contiguous, ie there may be **parallelism or gaps** between them.
For example, with project EXONE the "(SOFTWARE) IMPLEMENTATION" WP has a total duration of 46 workdays, not 32 workdays, which is the sum of the durations of all tasks in that particular WP.

The "Work" column in the previous tables shows the **workload** (in person-days) calculated for each **task**, for each **work package** and for the **project** as a whole.
For example, WP 1.8 in the contractor's plan has a workload of 4.5 person-days because there are 4 people assigned to it full-time and one resource ("CLT") part-time (50%) for a duration of 1 workday.

Duration estimates need to take into account **time** that may be required **for learning**, since people usually need to familiarize themselves with new tools, techniques and procedures.

Among **tools and techniques for estimating task durations or workloads**, the following are widely used and generally combined:

- expert judgment,
- analogous estimating,
- parametric estimating,
- three-point estimates,
- reserve analysis.

Expert judgment is based on experience of previous projects and information from people (editors, developers, etc.) who have already performed tasks similar to those for which an estimate needs to be made.

Analogous estimating uses information on the duration or workload (and other parameters) of similar tasks in previous projects.

Parametric estimating is an extrapolation based on statistical data that establishes the amount of time needed for a "unit of work". Input data may be historical data or the result of a simulation (eg as described previously for the indexing task of the EHM project).

Three-point estimates: this technique, which is worth what it is worth (no less, no more!), consists in calculating the expected duration (Te) or workload of a task as the weighted average of the most likely duration (Tm) or workload estimate, which is given a weight of 4, and of the optimistic and pessimistic estimates (To and Tp), which are each given a weight of 1. The corresponding formula is:

- Te = (To + 4 x Tm + Tp) / 6

Reserve analysis relates to the natural uncertainty of task duration or workload estimates. Uncertainty may lead to the inclusion of "**contingency reserves**" (also called "**time reserves**" or "**buffers**") in the estimates. For example, a percentage of the estimated duration or a number of work periods (eg days or weeks) may be added to the estimate as a contingency reserve.

The result of the "Estimate task durations" process is a set of additional attributes for the tasks in the task list.

Develop the schedule

The project schedule is derived from the **sequence of tasks**, their **durations** and **resources**, as well as **constraints** such as **milestones** (start or finish dates) that have been **fixed** (eg imposed by the project owner) for certain tasks or work packages.
A typical example of constraint is the **product launch date**, which may be used as a "starting point", so to speak, for "**backward planning**". Another example is that of data delivery which cannot take place before a certain date imposed by the content provider.

The **initial schedule** serves as a **baseline** to track project progress. The schedule generally needs to be revised as the project moves forward, but the initial schedule should be kept as a **reference**.

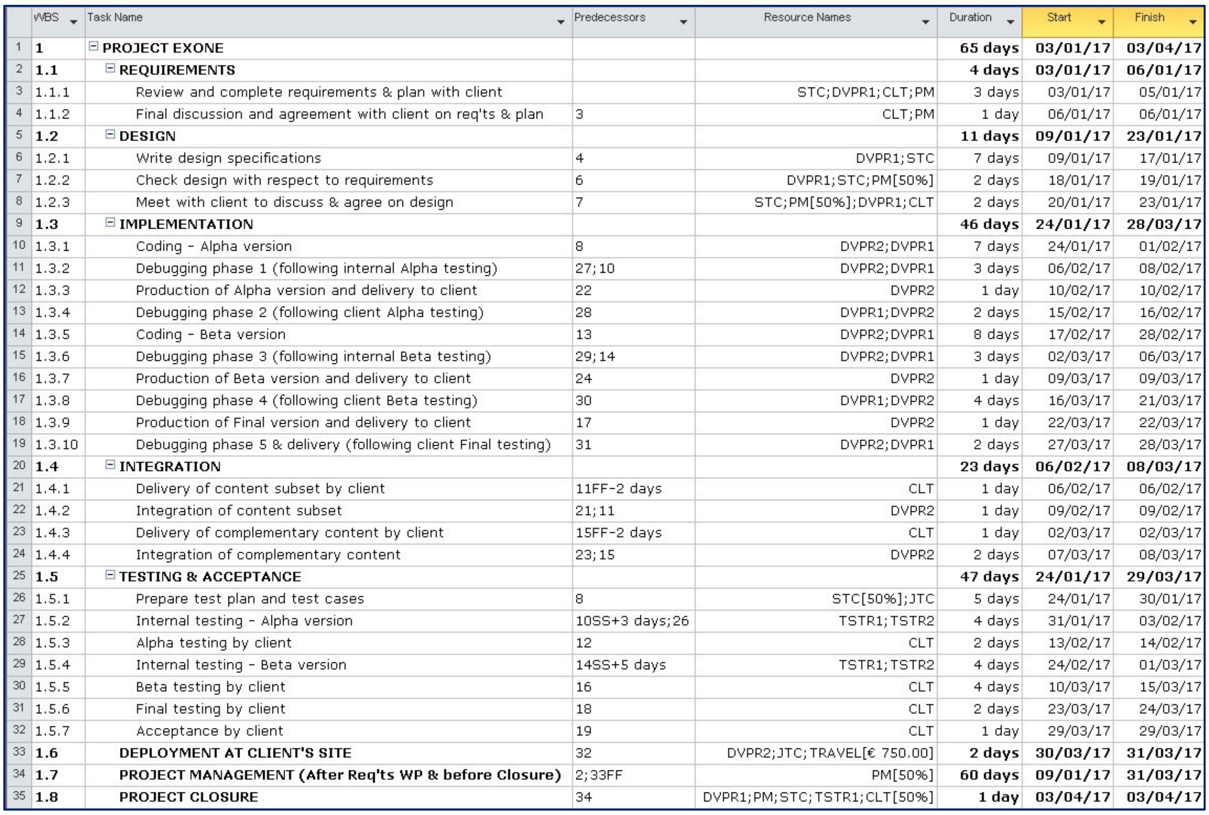
Major schedule revisions need to be approved by all stakeholders concerned.
One important notion in a project schedule is that of "**critical path**", ie sequences of tasks that determine the project's finish date. For critical-path tasks, there is no "**float**", ie no **margin of error**. Any delay in completing these tasks has a **direct impact** on the project's finish date.

Project management applications, such as MS Project and ProjectLibre, are very helpful for creating a project's schedule and for displaying critical paths.

Less specialized tools, such as a **spreadsheet**, may however be used for relatively simple project schedules or to provide a high-level view of the schedule of more complex projects, as illustrated further on.

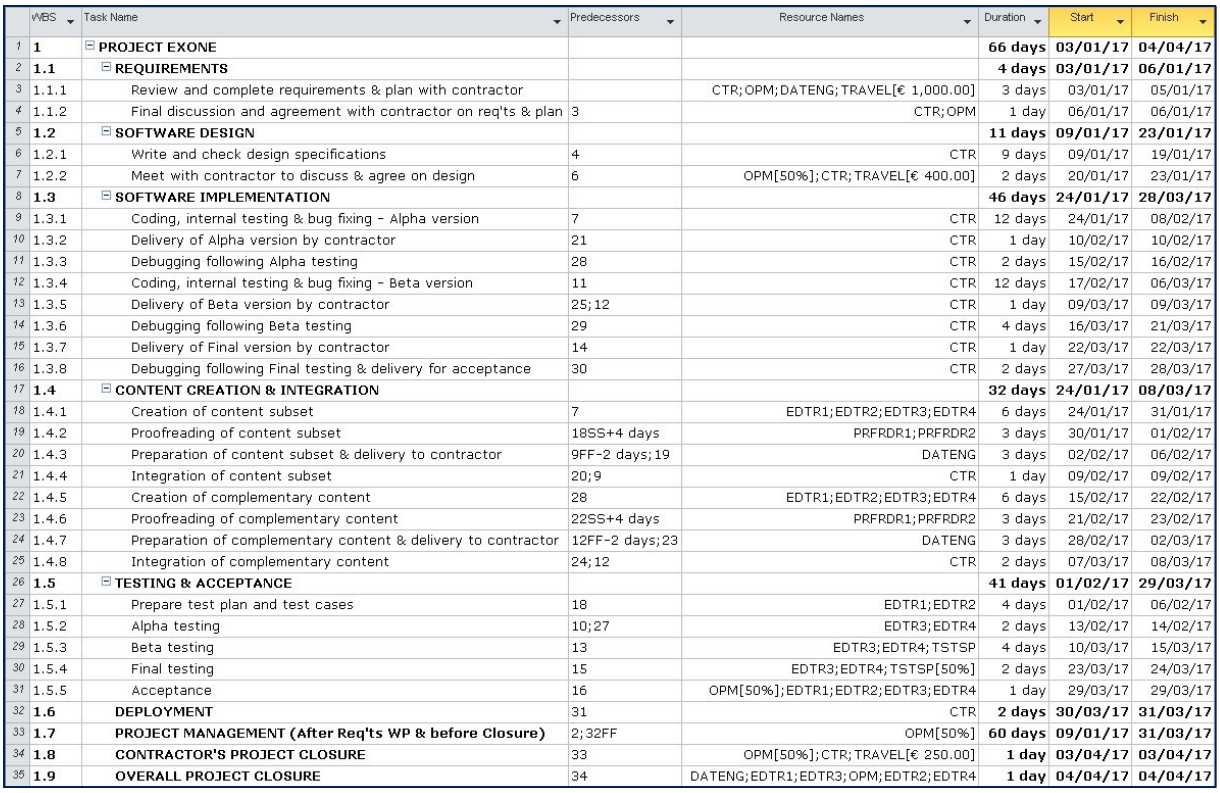
The following two illustrations show the Gantt tabular views for project EXONE, including "Start" and "Finish" dates.

- EXONE project – Contractor's view:



	WBS	Task Name	Predecessors	Resource Names	Duration	Start	Finish
1	**1**	⊟ **PROJECT EXONE**			**65 days**	**03/01/17**	**03/04/17**
2	**1.1**	⊟ **REQUIREMENTS**			**4 days**	**03/01/17**	**06/01/17**
3	1.1.1	Review and complete requirements & plan with client		STC;DVPR1;CLT;PM	3 days	03/01/17	05/01/17
4	1.1.2	Final discussion and agreement with client on req'ts & plan	3	CLT;PM	1 day	06/01/17	06/01/17
5	**1.2**	⊟ **DESIGN**			**11 days**	**09/01/17**	**23/01/17**
6	1.2.1	Write design specifications	4	DVPR1;STC	7 days	09/01/17	17/01/17
7	1.2.2	Check design with respect to requirements	6	DVPR1;STC;PM[50%]	2 days	18/01/17	19/01/17
8	1.2.3	Meet with client to discuss & agree on design	7	STC;PM[50%];DVPR1;CLT	2 days	20/01/17	23/01/17
9	**1.3**	⊟ **IMPLEMENTATION**			**46 days**	**24/01/17**	**28/03/17**
10	1.3.1	Coding - Alpha version	8	DVPR2;DVPR1	7 days	24/01/17	01/02/17
11	1.3.2	Debugging phase 1 (following internal Alpha testing)	27;10	DVPR2;DVPR1	3 days	06/02/17	08/02/17
12	1.3.3	Production of Alpha version and delivery to client	22	DVPR2	1 day	10/02/17	10/02/17
13	1.3.4	Debugging phase 2 (following client Alpha testing)	28	DVPR1;DVPR2	2 days	15/02/17	16/02/17
14	1.3.5	Coding - Beta version	13	DVPR2;DVPR1	8 days	17/02/17	28/02/17
15	1.3.6	Debugging phase 3 (following internal Beta testing)	29;14	DVPR2;DVPR1	3 days	02/03/17	06/03/17
16	1.3.7	Production of Beta version and delivery to client	24	DVPR2	1 day	09/03/17	09/03/17
17	1.3.8	Debugging phase 4 (following client Beta testing)	30	DVPR1;DVPR2	4 days	16/03/17	21/03/17
18	1.3.9	Production of Final version and delivery to client	17	DVPR2	1 day	22/03/17	22/03/17
19	1.3.10	Debugging phase 5 & delivery (following client Final testing)	31	DVPR2;DVPR1	2 days	27/03/17	28/03/17
20	**1.4**	⊟ **INTEGRATION**			**23 days**	**06/02/17**	**08/03/17**
21	1.4.1	Delivery of content subset by client	11FF-2 days	CLT	1 day	06/02/17	06/02/17
22	1.4.2	Integration of content subset	21;11	DVPR2	1 day	09/02/17	09/02/17
23	1.4.3	Delivery of complementary content by client	15FF-2 days	CLT	1 day	02/03/17	02/03/17
24	1.4.4	Integration of complementary content	23;15	DVPR2	2 days	07/03/17	08/03/17
25	**1.5**	⊟ **TESTING & ACCEPTANCE**			**47 days**	**24/01/17**	**29/03/17**
26	1.5.1	Prepare test plan and test cases	8	STC[50%];JTC	5 days	24/01/17	30/01/17
27	1.5.2	Internal testing - Alpha version	10SS+3 days;26	TSTR1;TSTR2	4 days	31/01/17	03/02/17
28	1.5.3	Alpha testing by client	12	CLT	2 days	13/02/17	14/02/17
29	1.5.4	Internal testing - Beta version	14SS+5 days	TSTR1;TSTR2	4 days	24/02/17	01/03/17
30	1.5.5	Beta testing by client	16	CLT	4 days	10/03/17	15/03/17
31	1.5.6	Final testing by client	18	CLT	2 days	23/03/17	24/03/17
32	1.5.7	Acceptance by client	19	CLT	1 day	29/03/17	29/03/17
33	**1.6**	**DEPLOYMENT AT CLIENT'S SITE**	32	DVPR2;JTC;TRAVEL[€ 750.00]	**2 days**	**30/03/17**	**31/03/17**
34	**1.7**	**PROJECT MANAGEMENT (After Req'ts WP & before Closure)**	2;33FF	PM[50%]	**60 days**	**09/01/17**	**31/03/17**
35	**1.8**	**PROJECT CLOSURE**	34	DVPR1;PM;STC;TSTR1;CLT[50%]	**1 day**	**03/04/17**	**03/04/17**

- EXONE project – Client's view:

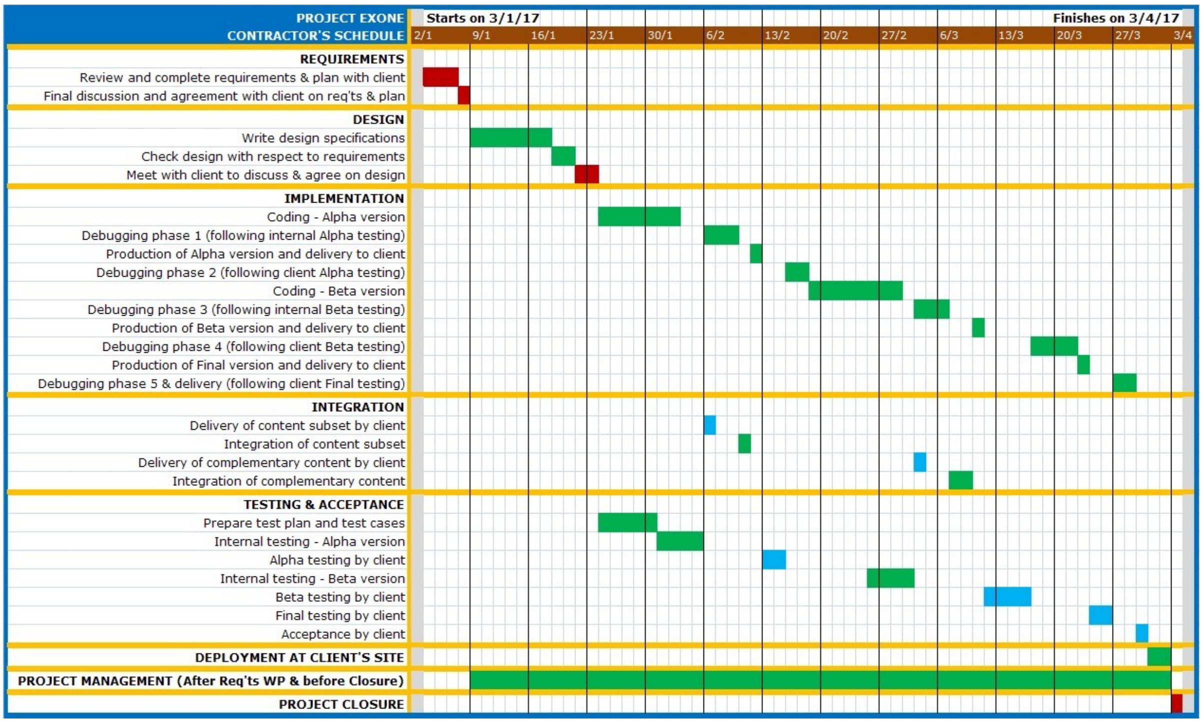


	WBS	Task Name	Predecessors	Resource Names	Duration	Start	Finish
1	**1**	⊟ **PROJECT EXONE**			**66 days**	**03/01/17**	**04/04/17**
2	**1.1**	⊟ **REQUIREMENTS**			**4 days**	**03/01/17**	**06/01/17**
3	1.1.1	Review and complete requirements & plan with contractor		CTR;OPM;DATENG;TRAVEL[€ 1,000.00]	3 days	03/01/17	05/01/17
4	1.1.2	Final discussion and agreement with contractor on req'ts & plan	3	CTR;OPM	1 day	06/01/17	06/01/17
5	**1.2**	⊟ **SOFTWARE DESIGN**			**11 days**	**09/01/17**	**23/01/17**
6	1.2.1	Write and check design specifications	4	CTR	9 days	09/01/17	19/01/17
7	1.2.2	Meet with contractor to discuss & agree on design	6	OPM[50%];CTR;TRAVEL[€ 400.00]	2 days	20/01/17	23/01/17
8	**1.3**	⊟ **SOFTWARE IMPLEMENTATION**			**46 days**	**24/01/17**	**28/03/17**
9	1.3.1	Coding, internal testing & bug fixing - Alpha version	7	CTR	12 days	24/01/17	08/02/17
10	1.3.2	Delivery of Alpha version by contractor	21	CTR	1 day	10/02/17	10/02/17
11	1.3.3	Debugging following Alpha testing	28	CTR	2 days	15/02/17	16/02/17
12	1.3.4	Coding, internal testing & bug fixing - Beta version	11	CTR	12 days	17/02/17	06/03/17
13	1.3.5	Delivery of Beta version by contractor	25;12	CTR	1 day	09/03/17	09/03/17
14	1.3.6	Debugging following Beta testing	29	CTR	4 days	16/03/17	21/03/17
15	1.3.7	Delivery of Final version by contractor	14	CTR	1 day	22/03/17	22/03/17
16	1.3.8	Debugging following Final testing & delivery for acceptance	30	CTR	2 days	27/03/17	28/03/17
17	**1.4**	⊟ **CONTENT CREATION & INTEGRATION**			**32 days**	**24/01/17**	**08/03/17**
18	1.4.1	Creation of content subset	7	EDTR1;EDTR2;EDTR3;EDTR4	6 days	24/01/17	31/01/17
19	1.4.2	Proofreading of content subset	18SS+4 days	PRFRDR1;PRFRDR2	3 days	30/01/17	01/02/17
20	1.4.3	Preparation of content subset & delivery to contractor	9FF-2 days;19	DATENG	3 days	02/02/17	06/02/17
21	1.4.4	Integration of content subset	20;9	CTR	1 day	09/02/17	09/02/17
22	1.4.5	Creation of complementary content	28	EDTR1;EDTR2;EDTR3;EDTR4	6 days	15/02/17	22/02/17
23	1.4.6	Proofreading of complementary content	22SS+4 days	PRFRDR1;PRFRDR2	3 days	21/02/17	23/02/17
24	1.4.7	Preparation of complementary content & delivery to contractor	12FF-2 days;23	DATENG	3 days	28/02/17	02/03/17
25	1.4.8	Integration of complementary content	24;12	CTR	2 days	07/03/17	08/03/17
26	**1.5**	⊟ **TESTING & ACCEPTANCE**			**41 days**	**01/02/17**	**29/03/17**
27	1.5.1	Prepare test plan and test cases	18	EDTR1;EDTR2	4 days	01/02/17	06/02/17
28	1.5.2	Alpha testing	10;27	EDTR3;EDTR4	2 days	13/02/17	14/02/17
29	1.5.3	Beta testing	13	EDTR3;EDTR4;TSTSP	4 days	10/03/17	15/03/17
30	1.5.4	Final testing	15	EDTR3;EDTR4;TSTSP[50%]	2 days	23/03/17	24/03/17
31	1.5.5	Acceptance	16	OPM[50%];EDTR1;EDTR2;EDTR3;EDTR4	1 day	29/03/17	29/03/17
32	**1.6**	**DEPLOYMENT**	31	CTR	**2 days**	**30/03/17**	**31/03/17**
33	**1.7**	**PROJECT MANAGEMENT (After Req'ts WP & before Closure)**	2;32FF	OPM[50%]	**60 days**	**09/01/17**	**31/03/17**
34	**1.8**	**CONTRACTOR'S PROJECT CLOSURE**	33	OPM[50%];CTR;TRAVEL[€ 250.00]	**1 day**	**03/04/17**	**03/04/17**
35	**1.9**	**OVERALL PROJECT CLOSURE**	34	DATENG;EDTR1;EDTR3;OPM;EDTR2;EDTR4	**1 day**	**04/04/17**	**04/04/17**

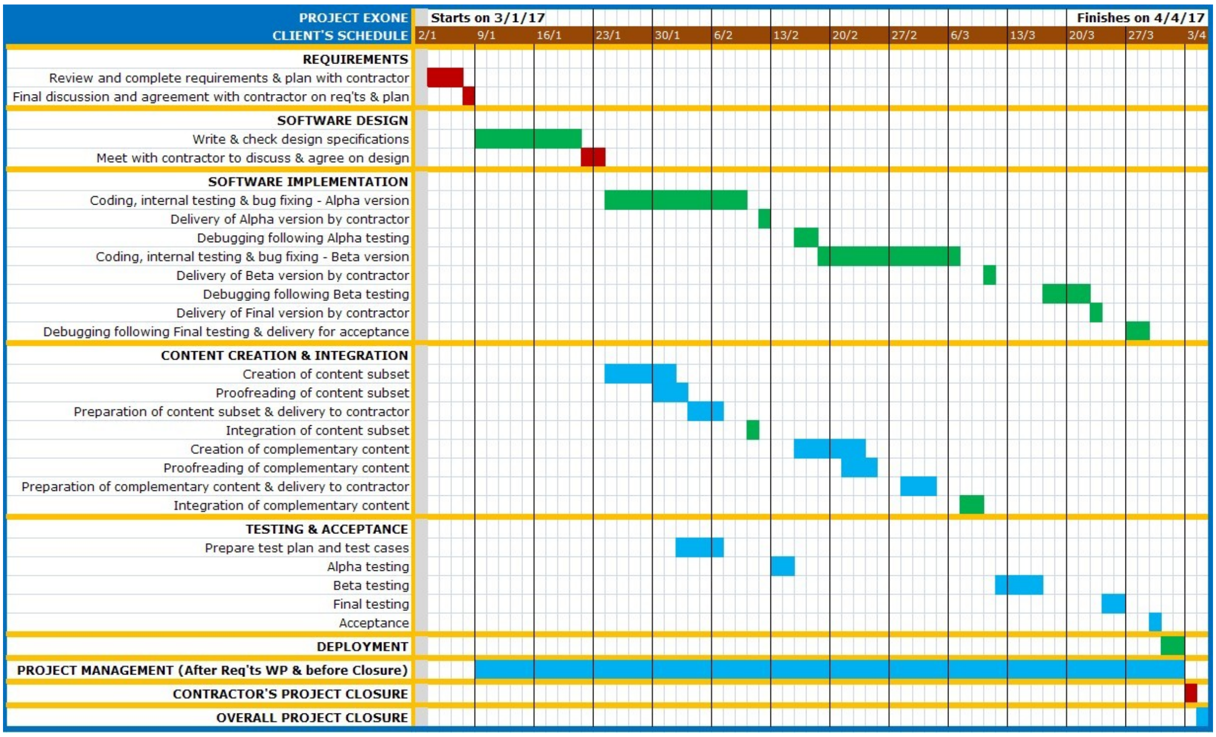
Tools such as MS Project (or ProjectLibre) can display schedules in various forms, but they are not always suited to presentations or inclusion in reports because they may extend over several pages.

I strongly recommend creating a **single-page overview** of the schedule for project supervision and reporting purposes, in addition to the schedule produced with a project management application, which may be difficult to read and understand at a glance.

- Example: Contractor's schedule for project EXONE created with a spreadsheet tool:



- Example: Client's schedule for project EXONE created with a spreadsheet tool:



PROJECT EXONE
CLIENT'S SCHEDULE

Starts on 3/1/17 — Finishes on 4/4/17

2/1 | 9/1 | 16/1 | 23/1 | 30/1 | 6/2 | 13/2 | 20/2 | 27/2 | 6/3 | 13/3 | 20/3 | 27/3 | 3/4

REQUIREMENTS
Review and complete requirements & plan with contractor
Final discussion and agreement with contractor on req'ts & plan

SOFTWARE DESIGN
Write & check design specifications
Meet with contractor to discuss & agree on design

SOFTWARE IMPLEMENTATION
Coding, internal testing & bug fixing - Alpha version
Delivery of Alpha version by contractor
Debugging following Alpha testing
Coding, internal testing & bug fixing - Beta version
Delivery of Beta version by contractor
Debugging following Beta testing
Delivery of Final version by contractor
Debugging following Final testing & delivery for acceptance

CONTENT CREATION & INTEGRATION
Creation of content subset
Proofreading of content subset
Preparation of content subset & delivery to contractor
Integration of content subset
Creation of complementary content
Proofreading of complementary content
Preparation of complementary content & delivery to contractor
Integration of complementary content

TESTING & ACCEPTANCE
Prepare test plan and test cases
Alpha testing
Beta testing
Final testing
Acceptance

DEPLOYMENT

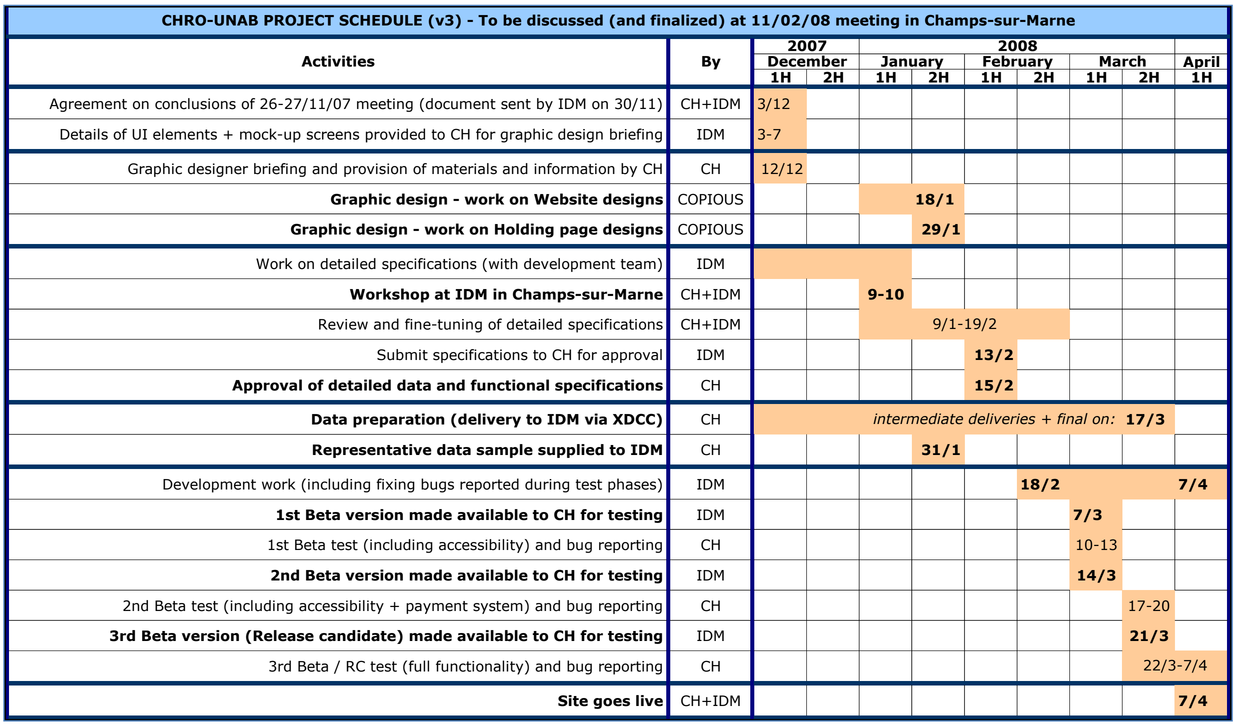
PROJECT MANAGEMENT (After Req'ts WP & before Closure)

CONTRACTOR'S PROJECT CLOSURE

OVERALL PROJECT CLOSURE

In these two EXONE schedules, colour codes are used to show whose resources are involved in the various tasks/WPs: one colour for tasks/WPs performed by contractor's resources only, one colour for tasks/WPs performed by client's resources only, and a third colour for tasks/WPs involving both contractor's resources and client's resources.

- Other example of a summary schedule in spreadsheet format:



CHRO-UNAB PROJECT SCHEDULE (v3) - To be discussed (and finalized) at 11/02/08 meeting in Champs-sur-Marne

Activities	By	2007		2008						
		December		January		February		March		April
		1H	2H	1H	2H	1H	2H	1H	2H	1H
Agreement on conclusions of 26-27/11/07 meeting (document sent by IDM on 30/11)	CH+IDM	3/12								
Details of UI elements + mock-up screens provided to CH for graphic design briefing	IDM	3-7								
Graphic designer briefing and provision of materials and information by CH	CH	12/12								
Graphic design - work on Website designs	COPIOUS				**18/1**					
Graphic design - work on Holding page designs	COPIOUS				**29/1**					
Work on detailed specifications (with development team)	IDM									
Workshop at IDM in Champs-sur-Marne	CH+IDM			**9-10**						
Review and fine-tuning of detailed specifications	CH+IDM			9/1-19/2						
Submit specifications to CH for approval	IDM					**13/2**				
Approval of detailed data and functional specifications	CH					**15/2**				
Data preparation (delivery to IDM via XDCC)	CH			*intermediate deliveries + final on:*					**17/3**	
Representative data sample supplied to IDM	CH				**31/1**					
Development work (including fixing bugs reported during test phases)	IDM						**18/2**			**7/4**
1st Beta version made available to CH for testing	IDM							**7/3**		
1st Beta test (including accessibility) and bug reporting	CH							10-13		
2nd Beta version made available to CH for testing	IDM							**14/3**		
2nd Beta test (including accessibility + payment system) and bug reporting	CH								17-20	
3rd Beta version (Release candidate) made available to CH for testing	IDM								**21/3**	
3rd Beta / RC test (full functionality) and bug reporting	CH								22/3-7/4	
Site goes live	CH+IDM									**7/4**

The above schedule (for a "real-life" online dictionaries project, which I was involved in as a freelance PM) features milestone dates and a Resources ("By") column, where "CH" refers to the client (Chambers Harrap, the publisher), "IDM" is the software development contractor and "COPIOUS" is the digital agency used by CH for graphic design work.