FAQ: TASK DEPENDENCIES – WHY USE FINISH-START IS PREFERRED?

This document addresses the question why using Finish-Start relationships is strongly recommended, if not mandatory from a best practice and CCPM perspective.

Why using Finish-Start is important?

The main reasons for using the F-S relationship are:

- it enforces that plans become "cause and effect" driven, which:
- ensures that the focus of the planning process moves to the key-deliverables to be delivered
- ability to focus on finishing one thing first, before starting something else, resulting in:
- the ability to keep the level of Work in Progress low → low WIP
- establish a clear **full-kit** rule: tasks can only start, if its predecessors have been completed successfully

Another important factor is that with FS relationships, the automatic scheduling process is able to deliver predictable and stable results. This is never the case, in case of FF,SS or SF, also because these type of relationships are often "ambiguous", leading to:

1. No automatic scheduling results
2. Generation of high WIP
3. An unclear logic with ambiguous relationships - defocusing

Note: Some experts even say that using any other dependency than Finish-Start is a symptom of "*lack of planning discipline and mental laziness*". Also many planning tools, like MS Project are quite forgiving on bad practices.

Transforming existing (MS Project) plans to a CCPM environment

The overall experience (and also our experience) is that almost all "as is" plans typically are not ready for CCPM:

- No clear cause and effect logic (see above)
- Use of all types of relationships
- Many constraints and "must" deadlines
- Many tasks without any relationship
- ASAP and early start behavior

How to make existing plans CCPM Ready?

From a (technical) CCPM perspective the steps are as follows:

- Replace any other relationships by a Finish – Start relationship. Any relationship can be replaced by a Finish – Start combintation (ask A-dato for examples)
- Remove any constraint (must start, must finish, etc.), except for the so-called "contractual milestones"
- Ask yourself if a milestone really meets the criteria for making it "contractual milestone"

Another task is of course to review the logic of existing plans and to validate if the structure from a cause and effect perspective (especially in case of projects that just started and certainly for new projects still to be started).

Transforming existing plans into LYNX

Existing project plans (MS Project) can easily be imported in LYNX. To get an existing plan work properly in the CCPM mode, do the following:

- Activate CCPM engine and set all scheduling parameters to ASAP (both for project as feeding chain)
- Remove / Replace any other type of relationship with Finish-Start
- Remove as many constraints as possible
- Check the project statistics
- Apply CCPM behavior (e.g. insert buffers, etc. / switch to JIT mode)

What if (existing) plans for current projects are not perfect (yet)? – "CCPM Light" Settings

In a "getting started" situation, LYNX provides the following flexibility:

- Setting the project and feeding chains scheduling behavior both to ASAP
- Application of "Finish-Start" relationships between summary tasks (= not best practice, but possible)
    - This handles also (child) tasks without dependencies within the CCPM logic
- If ASAP is activated for project and feeding chains:
    - Feeding buffers do not be inserted (makes plans easier)
    - Possibility to use certain constraints to position tasks on certain dates, without impacting the CCPM logic

CCPM light is often easier for users (especially Project Managers) to get started with CCPM. The plans look more similar to the previous structure, while at the same time Buffer Management and the Priority Mechanism can be introduced.

What about new projects?
While CCPM light is a good way to get started quickly, the CCPM best practices is include off course keeping WIP low, by planning and scheduling in Just in Time mode (from right to left) and having plans with a sound cause and effect logic. We recommend to define good templates and logic for your new projects and apply the best practice CCPM behavior as described above.

REFERENCES AND DOCUMENATION

The Defense Contract Management Agency (DCMA) 14-Points Assessment, and, in particular, its relationships assessment prefers finish-to-start (FS) relationships. The relationships assessment says that that 90% of all schedule relationships should be FS. One reason for this stipulation is that FS relationships improve schedule transparency, and, therefore, overall schedule quality.

Other relationships, such as SS and FF, are not forbidden; they are limited. Scheduler's should restrict SS and FF relationship types to not more than 10% of all schedule tasks. Although start-to-finish (SF) relationships are not forbidden they are considered counter-intuitive and should be limited to extremely rare situations. Assuming that SF relationships are completely avoided, the scheduler will want to review each instance of SS and FF relationships in the schedule. The scheduler must inspect each SS and FF relationship occurrence and consider replacing it with a FS relationship.

From https://tensix.com/2018/01/primavera-p6-and-highlighting-ss-and-ff-relationships/

Finish-to-start is considered a "natural dependency". The Practice Standard for Scheduling recommends, that "Typically, each predecessor activity would finish prior to the start of its successor activity (or activities)(known as finish-to-start (FS) relationship). Sometimes it is necessarily to overlap activities; an option may be selected to use start-to-start (SS), finish-to-finish (FF) or start-to-finish (SF) relationships....Whenever possible, the FS logical relationship should be used. If other types of relationships are used, they shall be used sparingly and with full understanding of how the relationships have been implemented in the scheduling software being used. Ideally, the sequence of all activities will be defined in such a way that the start of every activity has a logical relationship from a predecessor and the finish of every activity has a logical relationship to a successor".[2]

SF is rarely used, and should generally be avoided. Microsoft recommends to use SF dependency for just-in-time scheduling.[3] It can be easily shown however, that this would only work if resource levelling is not used, because resource levelling can delay a successor activity (an activity, which shall be finished just-in-time) in such a way, that it will finish later than the start of its logical predecessor activity, thus not fulfilling the just-in-time requirement.

From https://en.wikipedia.org/wiki/Dependency_(project_management)