021A – Linden
JUL 03 ONLINE

Monitoring Progress in Software Development

Joop van der Linden
The Haagse Hogeschool

To manage software development projects, a suitable model of the specific type of project is required. This article presents key performance indicators, a procedure, tables, and some graphs as characteristics of such a model. From this, a metric is derived to meet managerial needs for a tool to measure and present project progress. It provides an easy-to-follow picture for both the client and the project manager of how to manage a development project. Percentages of technical completion can be reported periodically in scheduled and actual figures. The use of the metrics is described and illustrated with examples along with a discussion of the possible effects on the behaviour of a project-team are discussed.

The process of developing software shows some similarities with the process of designing industrial buildings. The approach to an engineering project to build a refinery only differs from a software development project in terms of scale. In both types of projects, using a metric is required to measure progress. It creates a feedback loop that enables pronouncements to be made on the project's development pace. In addition, unexpected changes in the scope of the project can be pointed out at an early stage.

Just like any feedback loop, a metric must also be suitable for reviewing the status of a project at regular intervals. When measuring progress in engineering projects, terms are used such as the percentage of technical completion, the scope of work, and exceeding the completion date. These terms can be used in the same way in a software development project. There are, however, remarkable differences. Usually, in projects for the design, procurement, and construction of a refinery, several hundreds of thousands of man-hours are involved, and the amount of required design drawings, specifications, and details drawings is enormous. Software development projects are considerably smaller.

The deliverables are hardly visible, and a lot of people with a wide range of disciplines are involved. Due to this, the need for a flexible and simple metric is important. Analyzing the work to be done in a software development project gives rise to the construction of a metric. From this it is possible to issue progress and status reports for managing the project without the need to draw up a fully integrated network plan.

Breaking Down the Work

To start, using a metric requires an estimate or quantitative analysis of the scope of a project. An empirical *a priori* estimate enables us to find things such as the number of database tables required, the number of functions to be developed, etc. This estimate is usually based on early project documents from the design phase or from sales documents like a business case.

In a scheduling process, the metric data from the estimate relates to the period between the start and finish of a project. The construction of a global bar chart plan of the project with up to 15 activities may be helpful for this. With one bar in this chart, the estimated start and finish of the realization of, for instance, the database tables can be indicated. Revising the estimate of the number of required tables at regular intervals during the course of the project creates a row of numbers. In addition, the number of tables that are completed by the project must be recorded with the same frequency. Table 1 shows this process in a project after four weeks. The initially

crude estimate of the number of database tables required will increasingly indicate the final number as the development process continues.

Time now

Software item	Week 1 (project start)	Week 2	Week 3	Week 4	Week 7
tables required.	12	13	13	14	
Number of tables ready.	0	5	9	13	
Number of tables in progress.	4	7	3	0	

Table 1: *The Principle of Breaking Down the Work*

The *Time Now* label in the table relates to the time when the status of the project is being reviewed. It acts as the cutoff point of a work period by measuring the progress made. On this date the metric is issued to the project client. The status is made known by showing both the newly estimated values and the historical figures together with the results actually achieved.

The intention is that the values printed before the Time Now cutoff point should remain unchanged. They will not change for the remainder of the time that the project has to run. The project that relates to the example in Table 1 has reached the fourth week of the project. Four measurements have been made and four progress reports have been drawn up. The top row of figures in the metric shows estimated values. The bottom two rows consist of values that have actually occurred. If necessary, the documents with the technical definition of the database tables that are ready can be submitted to the client for approval.

The interpretation of the metric can give rise to various contentions. It can be seen that the number of tables has risen from 12 to 14, which means that other software items in the project will probably also increase in number. Examples include an increase in the number of functions or the number of reports expected as output from the system to be developed. The sample metric also shows that there is one database table for which the work is not started. Members of the project now have to investigate why this is so and what the consequences will be for the project. There may be a lack of information, this database table may be part of an external system, or the table may only be relevant during the test phase of the system.

It is only after that database tables have been reported as completed that a start can usually be made on developing the appropriate screen designs, functions, and objects. The metric shows us the most suitable time for this, thereby avoiding a false start.

The final scope of the project in Table 1 in terms of the number of database tables will probably be established in the fifth week. A more accurate view on the project is achieved at that time. Changes that arise thereafter may indicate that the original scope of the project has changed, the possible consequence being additional work.

Sometimes the scope of work of a project indicates that tables with special features, e.g. binary large object (BLOB) elements are required (binary logical objects such as an image or a soundtrack). If this occurs, the metric can be easily expanded to include an additional software item, if necessary, halfway through the project.

The metric also applies to other aspects of software development. Apart from recording the status of the database tables, it can at the same time show how work on other items is progressing. Table 2 contains information relating to functions, screens, and printed reports.

Table 2 shows that an estimated 245 functions will be developed, of which 27 are actually finished at the end of week eight. If we reckon on a completion period of 24 weeks, this means that the remaining 218 functions with approximately 10 functions per week should become available. These 24 weeks were calculated by reducing the remaining 32 weeks available for the project by the time needed to develop the last database table and by reserving time for testing and other detailed work.

The project manager may now be wondering how feasible it is to develop the specified software with the number of staff available. The availability of manpower and expertise has a part to play in this. Simply dividing the number of functions by the number of developers produces interesting information.

To check the numbers stated in the metric, a client calls up the titles of the functions or checks which developers have worked on which items. Accounting for the man-hours spent on a specific part of the development work can provide enlightening background information on the further development of the project. A comparison with metrics from other projects provides more detailed information on capacity in terms of man-hours expressed as units of work.

Scheduled and Actual Progress

The figures noted in Table 2 after *functions in progress* can be compared with the figures under the heading *functions ready*. This shows us that the weekly increase in functions in progress is substantially higher than the corresponding figure for number of functions ready. We can conclude from this that it is possible to start developing a number of details on functions, whereas it is apparently not possible to finally complete work on those functions.

This makes project management more difficult and complicated as unfinished work piles up on developers' desks. We may wonder whether this is acceptable and what the reasons for it are. It is, for example, possible that the developers have insufficient in-house expertise, that there is a lack of information, that a start was made on developing the wrong functions, or that there has been a breakdown in communication somewhere. A delay may also have occurred because one table crucial to the project has not yet been completed.

The metric shows that work was done on one screen during five weeks of the project. If this happens with the other 23 screens, the project is sure to be completed late. In this case, it is worth analyzing the technical contents of the screens, the number of HTML pages of which a single screen is made up, and/or whether it is possible to re-use them. If the numbers recorded in the metric show substantial variations in one or more rows, this means that the project manager is probably not aware of the scope of his project. Possibly there is a lack of experience with the project manager.

Software item	Week (proje start)	Week	Week	Week	Week	Week	Week	Week	Week	Week 40 (project finish)
Total number of tables requi	12	13	13	14	14	14	14	14		
Number of tables ready.	0	5	9	13	13	13	13	13		
Number of tables in progres	4	7	3	0	0	0	0	0		
Total number of functions required.	230	230	241	241	241	241	241	245		
Number of functions ready.	0	0	3	12	18	25	25	27		
Number of functions in prog	0	1	14	18	13	31	44	59		
Total number of screen definitions.	23	23	23	23	23	24	24	24		
Number of screen definities ready.	0	0	0	0	0	0	0	0		
Number of screen definitions in progress.	0	0	0	1	1	1	1	1		
Total number of output repo	31	31	31	31	31	31	31	31		
Number of reports ready.	0	0	0	0	0	0	0	0		
Number of reports in progre	0	0	0	0	0	0	0	1		
Percentage of technical completion.	0%	1%	4%	8%	10%	12%	12%	13%		

Table 2: *A Complete Overview of a Project Contents*

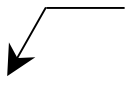
Technical Completion

The project, as shown above, has 245 functions of which 27 were reported as completed in the eighth week. In this case, the percentage of technical completion of the functions would be

27/245 x 100% = 10%.

The complete project, including all items, contains 314 elements. The number of these that have been completed is 40, which would make the percentage of completion for the project as a whole about 13 percent. It is useful to add these figures to the metric as an extra row. Be aware that each detail item of the remaining work might differ in complexity, in size, etc.

The version of the metric in Table 2 shows the actual performance achieved each week. Information in the columns after Time Now has not been entered. It is possible to expand the metric and use this space. The figure that indicates the total quantity of a specific software item can be split. This then produces portions of the item to be reported complete each week. If this split has been determined with any certainty, it will be reproduced in the metric. Each week, after Time Now, indicates the number of tables the project manager expects to report as completed. The last week, the week of the project finish, contains the final number of an item that is expected. In this way, the table acquires the appearance of Table 3.



Software item	Week 1 (project	Week	Week	Week	Week	Week	Week	Week	Week 40 (p finish)
Number of tables reac									
Scheduled	0	3	12	13	13	13	13	13	13
Actual	0	5	9						

Table 3: *The Table with Actual and Scheduled Information*

The section of the metric content after the Time Now indication may change every report period. All the figures in the *scheduled* row are entered at the start of the project. The project manager estimates the total number of tables required, in this case 13, and distributes them over the weeks of the project**.** This calls for experience that goes beyond estimating the totals only.

The project manager uses this experience to state the rate at which he estimates the tables will be developed. It is possible to apply an estimating tool for this**.** As shown in Table 3, the project delivers three tables in the second week, nine are added in the third week, and the last one is completed in the fourth week. The *actual* row shows the numbers that were actually completed during the weeks concerned.

Table 4 shows the situation after four weeks of the project. It appears that the total number of tables required had been underestimated and should be adjusted to 14. Work is also finished on another four tables.

Time now

Software item	Week 1 (project s	Week	Week	Week	Week	Week	Week		finish)
Scheduled	0	3	12	14	14	14	14	14	14
Actual	0	5	9	13					

Table 4: *Updating from Workweek Three to Four*

Working With S-Curves

The metric can be converted to a graphic representation. Figure 1 is an example of this. The project manager's forecasts are compared with the state of affairs that actually exists. The horizontal axis shows the duration of the project divided into one-week periods. The vertical axis indicates the numbers of database tables that have been reported as complete in the corresponding weeks or expected to be reported as complete. The metric data, therefore, form two curves. The curve showing the forecasts is used as the standard against which measurements are made. The curve showing the actual situation is opposite. The place where the curves do not cover each other indicates that the project is either behind schedule or ahead of schedule.

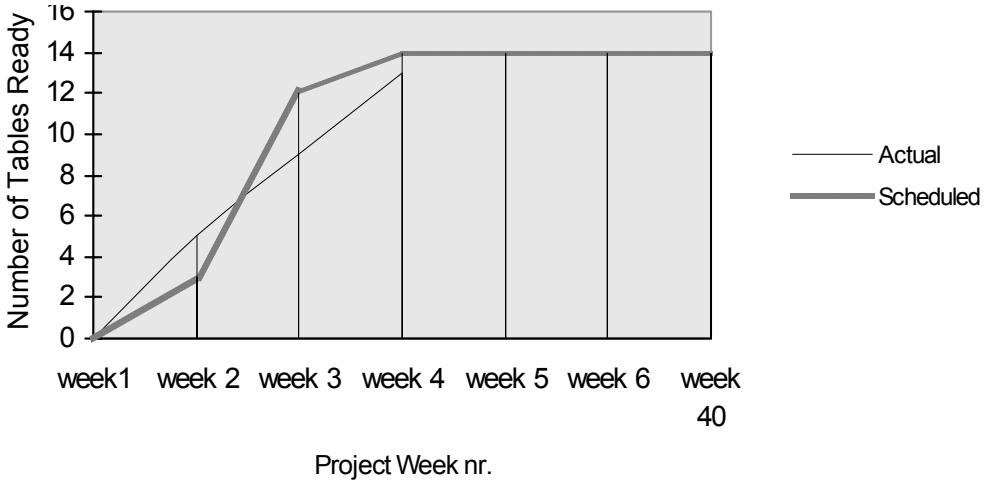


Figure 1: *The Work Expressed in a Graph Called S-Curv*

The Complete Schedule

The metric as discussed so far can be expanded with other items to be developed such as adding functions, screens, and reports involved. The metric will now appear as shown in Table 5. Also another time interval is chosen over which to make measurements. As shown in the metric, 25 weeks have passed at intervals of five weeks. The complete project has a duration of 40 weeks.

Any interpretation of the actual situation based on the metric may go in a number of directions. If the value at which the number of *functions completed* shows an ascending line after it has been constant for a number of periods, it is immediately noticeable. The why and wherefore form the basis for later discussion. If a row shows that figures are decreasing instead of increasing, that items are suddenly being added or removed or that there is an explosive increase somewhere, a more detailed analysis is required. It may be the case that the number of estimated functions is small in the experience of the person assessing the metric. It is not fanciful to draw the conclusion that the scope of the functions may be too wide and that, for this reason, complicated, high-maintenance software is being written. A meeting between the project manager and the client must be held to clarify matters. The metric justifies its existence by operating as a simple tool that gives rise to consultation based on facts.

Other items can be recorded in the metric besides the development items listed in Table 5, e.g., the number of processes that are expected to be designed before work can be started. The same applies in the case of software to be purchased externally or objects to be integrated. Even graphic display images such as logos and screen embellishments have a measurable effect on software development and can be included in the metric.

It is almost always sensible to state the hours worked during the specified intervals in the metric. The percentage of technical completion in Table 5 consists of two components: a predicted value and a value indicating what has been achieved in reality. Progress of

44 – 31 = 13%

behind schedule is measured. To translate this into the actual situation, it is useful to compare these percentages to the man-hours budget. If necessary, the number of man-hours worked can be included in a separate table row.

A comparison provides potent information on the current situation with the project. Imagine that, in the project shown in Table 5, a percentage of 67 of the man-hours budget have been used up. This is not inconceivable in view of the fact that 67.5 percent of the project time has elapsed. This means that 69 percent of the work is still to be done with 33 percent of the

budgeted man-hours. The question then arises whether budget overruns at the end of the project can still be avoided and how long a project extension will be. The S curve that can be drawn on the basis of the metric in Table 5 is shown in Figure 2.

Time now

Software item	Week (proje start)	Week	Week	Week			Week	Week	Week 40 (project fi
Number of tables ready:									
Scheduled	0	14	14	14	14	14	14	14	14
Actual	0	13	13	13	13	13			
Number of functions rea									
Scheduled	0	5	10	20	50	100	150	200	241
Actual	0	12	12	14	20	70			
Number of screen lay-out designs ready:									
Scheduled	0	0	0	0	5	10	20	23	23
Actual	0	0	0	0	0	3			
Number of reports ready									
Scheduled	0	0	0	0	0	15	20	25	31
Actual	0	0	0	0	5	10			
Percentage of technical completion:									
Scheduled	0	6%	8%	11%	22%	44%	66%	84%	100%
Actual	0	8%	8%	9%	11%	31%			

Table 5: *The Final Application of the Metric*

The S curve can be read in the following ways.

- The horizontal distance between the actual curve and the scheduled curve in week 25 shows that the project is two weeks behind schedule.
- There is a period of more than 10 weeks in which the project failed to register any measurable technical progress.
- The end of the scheduled curve in the last project periods is not sufficiently horizontal. This leads one to suspect that in reality a substantial overrun will follow. Because the actual curve runs parallel to the scheduled curve it may not be possible, without intervention, to catch up on the two-week delay.

A third curve can be drawn in to represent the hours worked. This is done by marking out the hours worked at the end of each period as percentages of the project budget.

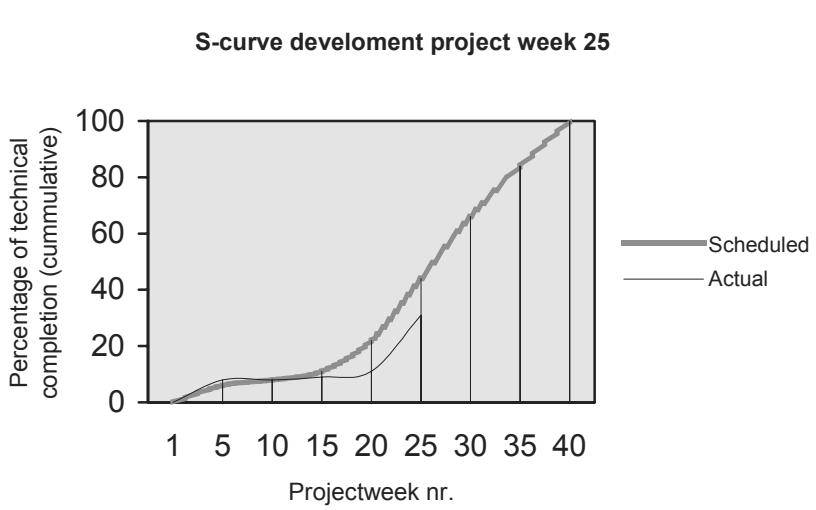


Figure 2. An Overall View of the Project in an S-Curve Showing Cumulative Totals Per Period

In Figure 3, the metric is transformed into a diagram that shows the *software development life cycle* in a different way. In Unified Modelling Language (UML) this curve is known as a Process Workflow. This model curve shows the periodic returns from the project. Comparing the *actual* curve with the *scheduled* curve can produce new information. For the present project the graph indicates insufficient performance during weeks 10 to 25. The tardy and peculiar start to the project during the first 10 weeks is clearly perceptible.

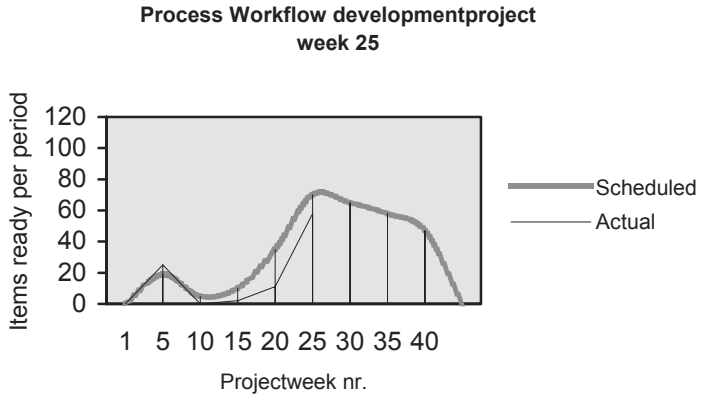


Figure 3. An Overall View of the Work in Items Ready Per Period

Conclusion

The important thing is that the values entered before the Time Now indication in the metric should remain unchanged. If the project manager does not adhere to this, each periodic issue will have to be kept. And this makes trends more difficult to detect.

Reporting forms, graphs, and tables to be printed on paper as completed will not be an arbitrary matter. The same applies to the composition and functionality of a screen, as it is not always clear when functions, procedures, subroutines, or objects have been completed. Many of these items are only complete when the project has reached its conclusion. To avoid any ambiguities, a technical completion of 95 percent is used as the time when completion is reported in the metric. However, it often appears that this is not accurate enough. The time an item is signed *approved for implementation* by the client can be used instead, to indicate that a component is 100 percent complete.

The establishment and periodic adjustment of the metric requires a certain amount of experience. In practical use, the first type of experience required is in project management. However, a table of the metric containing estimated values that are initially completely wrong could be used to good effect, provided that it is consistently updated and issued. It provides an easy-to-follow picture for both the client and the project manager of how to manage a development project. In addition, the costs are low.

About the Author

Joop van der Linden is a lecturer in information technology at the Haagse Hogeschool in Den Haag, the Netherlands. He worked with CE Lummus, Cincinnatti Milacron, and a Marine Consultant company where he managed numerous information technology projects in the field of offshore technology, office automation, enterprise application integration, and scientific engineering.

The Haagse Hogeschool
Fluiterlaan 246
2903 HS Capelle ad IJssel
The Netherlands
Phone: 010-4510733
E-mail: j.b.vanderlinden@hhs.nl