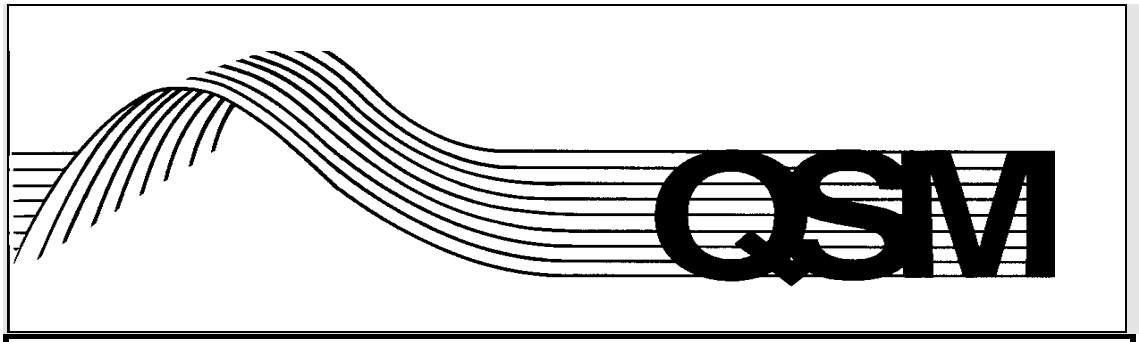

MEASURES FOR EXCELLENCE

MANAGING

MAJOR DISTRIBUTED

SOFTWARE DEVELOPMENT

QUANTITATIVE SOFTWARE MANAGEMENT LTD

7 Rue Fenoux		93 Blythe Road
75015 Paris	Internet: qsm.europe@pobox.com	London W14 0HP
Tel: 33-140-431210	CompuServe: 100113,3364	Tel: 44-171-603-9009
Fax : 33-148-286249	www.qsm.com	Fax : 44-171-602-6008

INTRODUCTION

"*Effective software process improvement will not start until management insists that product development work be planned and properly managed (Ref. 1).*" This becomes even more challenging in an increasing number of major system developments made up from distributed sub-system software projects. These sub-systems are integrated and validated to provide the final system and product release. The need is growing to estimate, risk assess, plan and manage the development of these distributed sub-systems and the final full system release.

Large scale distributed system developments are now common in telecommunications, air traffic control, defence and space. They are recognisable because of their sheer scale. In these systems a release will often be developed over 2 – 4 years and cost in excess of 200-300 person years of development effort. Features define the content of a release. The features are realised by mapping the full set of system feature requirements across the various sub-systems. Normally this involves building on a large existing software base made up of the existing sub-systems that are then modified and extended to engineer the new release.

Figure 1 illustrates in outline the specification and development of a release that consists of distributed sub-systems.

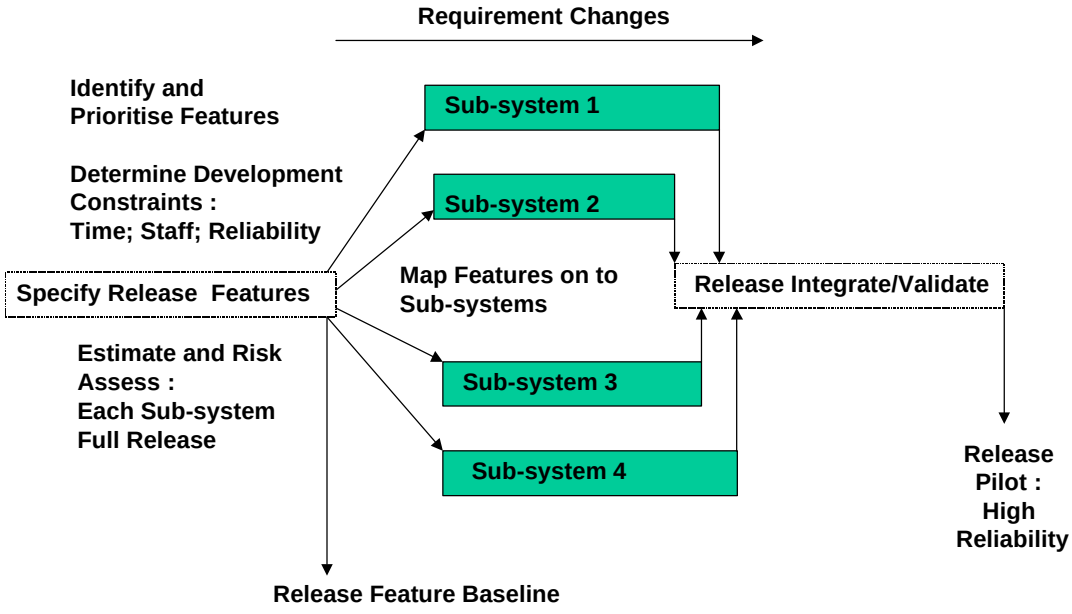

Figure 1: Release - Sub-systems

Requirement changes are common in these developments because of the long lead times involved. This adds to the complexity of planning and controlling the in-progress development both at the individual sub-system level and the release. A further complexity factor is the integration and validation of the release once the individual sub-systems are delivered. These large-scale systems must achieve high reliability because of their nature. In the final release validation software defects arise not only from the

new software but also due to latent defects in the existing code. Substantial regression tests must be run to checkout the existing software base that may amount to millions of lines of code.

In summary these distributed developments represent major management challenges to both developers and equally to purchasers. Purchasers are concerned to check that supplier's development proposals for these large systems are realistic and provide value for money (Ref. 2). Frequently the purchaser's competitive position depends on delivery of all the new features by the scheduled date, within budget and with high reliability.

Developers and purchasers are faced with a number of key issues:
1. Identifying what features are practical within commercial constraints
2. Ensuring realistic development plans are established that take in to account uncertainties and quantify risk
3. Agreeing and controlling the software baseline
4. Making progress continuously visible
5. Ensuring delivery takes place with high reliability

BACKGROUND TO THE CASE STUDY

This telecommunications division develops sub-systems at a number of different locations. These sub-systems are then integrated and the full release validated. Management is concerned at the history of significant cost overrun and slippage in these strategic product developments. Pressure is increasing to reduce the time to market in order to meet the needs of customers while at the same time improving quality and becoming more competitive by improving process productivity (Ref. 4.).

Product features change continuously in response to customer demands. An essential need is to assess the potential risk in developing a set of features within the time scales demanded and available staff limits. This motivated management to introduce the techniques described.

ANALYSIS METHOD and THE BASIC DATA

The first step in applying the analysis method is to map the division development lifecycle to four high level phases. These four phases comprise 1) Feasibility and Architecture, 2) Specification and High Level Design, 3) Main Software Development, 4) Factory Test and Pilot operation. Major milestones are defined for each sub-system and the entire release. Documentation estimates form part of the data. Integration tests at the sub-system level are quantified. The full release integration tests as well as the final validation tests and load tests are quantified to enable progress tracking.

Feature mapping is carried out across each sub-system. Modules are identified that are to be changed or added. Sizing is based on estimating the size range expressed in terms of the minimum, most likely and maximum number of source statements. These size range estimates are made for each module to give the baseline size and uncertainty for each sub-system and the full release. This data is used to generate high level estimates that are risk

protected by taking in to account the size uncertainty. The alternatives of developing sets of features are evaluated to meet specific development goals of time, staffing and reliability.

The project managers complete data collection templates. To provide the data for the entire release these templates are summed and extended to include the final integration tests for the sub-systems and the full release validation and load tests.

A key part of the estimation and risk assessment process uses measures of the process productivity achieved in past projects. This data (at its most basic for the main software development) comprises the size, time and effort for phase 3 of the completed sub-systems and releases (Ref. 6).

Estimates are then made taking in to account the management development constraints such as time, staffing and reliability to determine the risk in achieving these constraints. Comparison of the new estimates against the company's historic data ensures that unrealistic estimates are avoided.

BUILDING THE SUB-SYSTEMS AND RELEASE

Below we show two examples of the baseline plan determined first for a sub-system (Figure 2) and then for the entire release (Figure 3). In this instance the release consists of a number of sub-systems, each on of which is independently estimated and tracked. On the charts the bottom horizontal axis is time in months and numbers at the top represent major milestones.

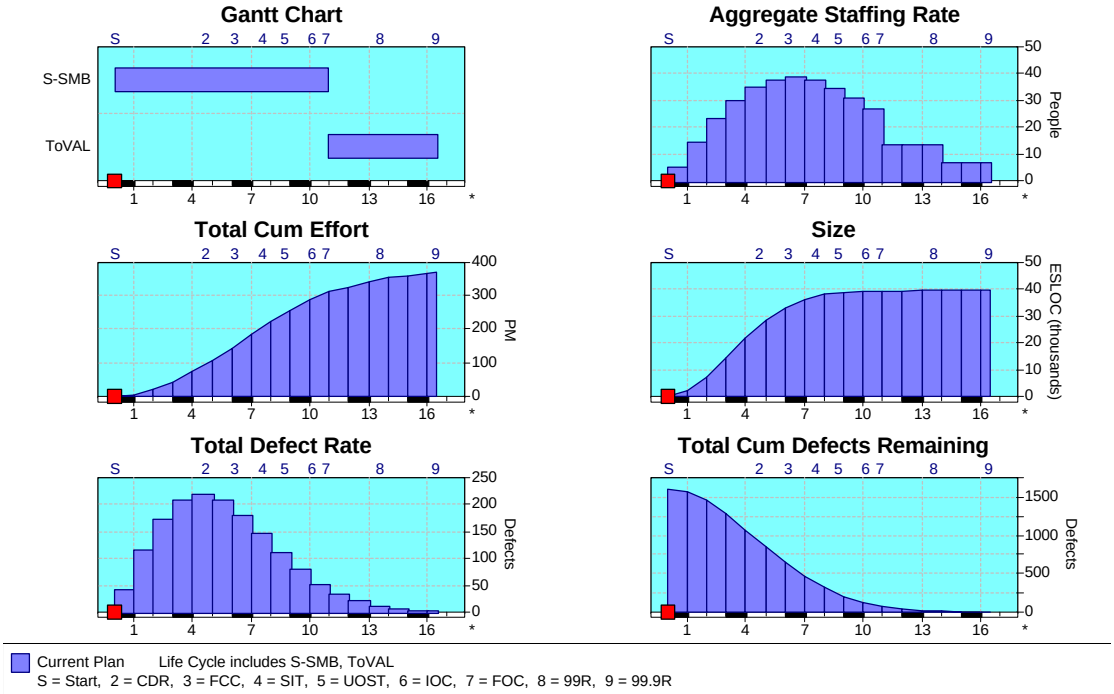

Figure 2: Example of Sub-System Baseline Plan

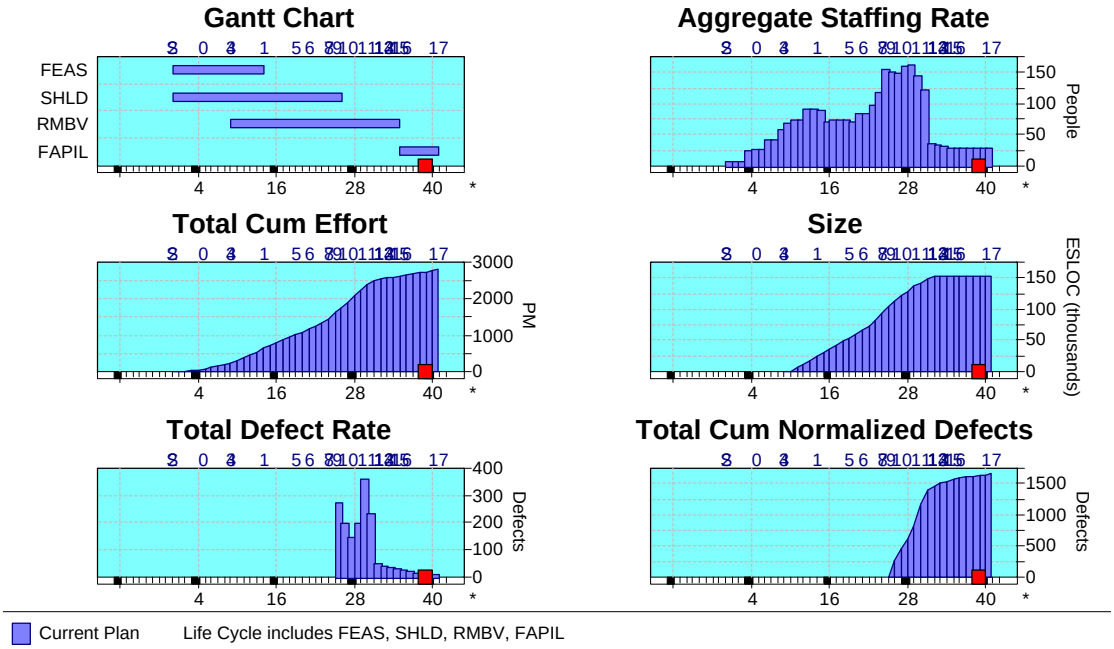

Figure 3: Example of the Release Baseline Plan

TRACKING PROGRESS AND FORECASTING COMPLETION

Progress is tracked using the baseline plan for each sub-system and the full release. Every two weeks, or each month, the project manager responsible for each sub-system supplies high-level progress data. This is used to track progress for each sub-system. Variance outside the established limits leads to forecasting the outstanding development plan for the sub-system. The forecasting makes use of sophisticated algorithms that determine the coefficient of variance for each progress metric. These coefficients enable weightings to be entered that reflect the best progress indicators at a given point in time and the corresponding completion forecast.

The data for all sub-systems is consolidated and used to track the full release. This continues until all sub-systems are completed and their integration begins. At this point the progress weightings are set to track the progress in carrying out the final release integration, validation and load tests. Equally important are the total defects, the different defect categories and their behaviour. These inputs and their variance analysis permits forecasting the completion date for the release to ensure high reliability when the final factory acceptance phase is entered (in this example named FAPIL).

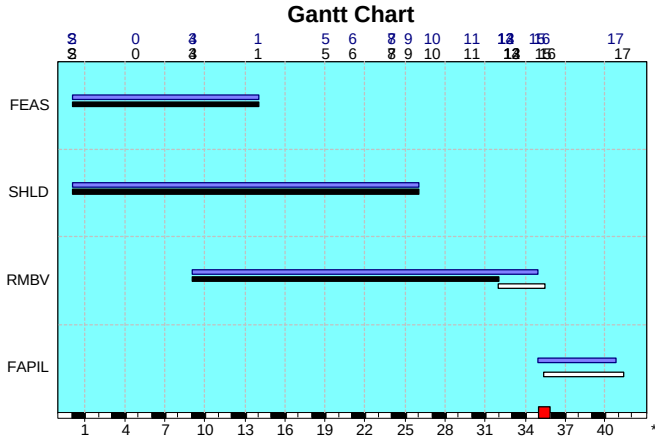

Figure 4: Release Progress Tracking and Release Forecast
(Note in the diagram the upper line is the plan, below are shown the progress to date in black and the forecast in white.)

Finally we show an output from tracking the full release defects and integration tests against their plan and the control bounds used to detect variance.

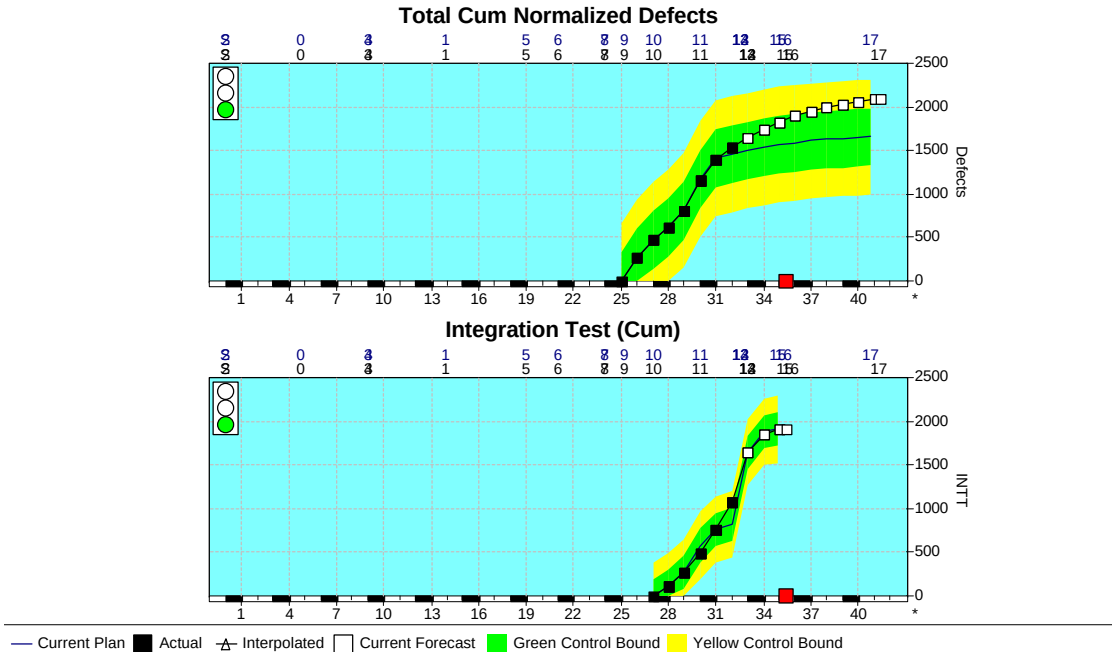

Figure 5: Defects and Integration Tests
(Note: Actual progress data points are black, forecast data are white)

OBSERVATIONS AND CONCLUSIONS

The results illustrated here from analysing a large scale distributed development show that it is practical to treat the individual sub-systems as projects in their own right. The full release behaves as a large development. Here the sub-systems are consolidated and their final integration and validation are included as part of the full release development.

High reliability determines when release delivery should take place. Reliability depends on the defects originating from the sub-systems. Tracking the sub-system defect behaviour reveals the contribution that factors such as time pressure and software size make to the defects found in each sub-system and later during the release integration and validation tests. These insights offer ways to improve reliability and avoid the premature delivery of the sub-systems and the release (Ref. 9).

The initial feature list is a key input to evaluate what can be developed within a given time and with available staff. Mapping the features on to the sub-systems and estimating the size range of the modified and new modules permits the rapid evaluation of the feature content that it is possible to develop within these management constraints. A risk-protected baseline estimate is produced that is viable within quantified management constraints and consistent with past projects.

Once development starts the quantified feature size baseline allows the impact of change requests to be determined. The potential impact of change

requests on individual sub-systems as well as the full release is used to decide whether to include each change request or defer to the next release.

As noted earlier (Ref. 1) software process improvement depends on product development being planned and properly managed. In practice the management techniques outlined here form part of a software control office. This office continuously evaluates and reports on all current developments and ensures that future release plans are realistic (Ref.7). We find the control office safeguards against runaway projects as well as satisfying many SEI-CMM Key Process Areas (Ref 4) and brings significant commercial benefit (Ref. 5). Runaway projects are common and disastrous in these large-scale complex developments (Ref. 8).

Disasters are avoided by sizing the features to determine what can be developed within specific constraints and their level of risk. In this way a realistic baseline-planning estimate is produced. Progress visibility during development is then ensured using basic data. This visibility results in the early detection of any variance against the baseline plan and initiates immediate corrective action.

These two aspects, a realistic estimate and continuous visibility during development, are fundamental to the successful management of software projects by development and purchasing organisations. While true for all software developments this is even more important where large scale distributed sub-systems are involved.

Jim Greene is Managing Director of Quantitative Software Management Europe in Paris, France: telephone 33-140431210; fax 33-148286249. He has over 30 years experience in software engineering, with a particular interest in management methods used by development and purchasing organisations based on the quantification of software development.

Ref.1. Watts S. Humphrey “Three Dimensions of Process Improvement - Part 1: Process Maturity ” *CROSSTALK* The Journal of Defense Software Engineering February 1998.
Ref. 2. Geerhard W. Kempff “Managing Software Acquisition” *Managing System Development* July 1998 Applied Computer Research Inc. P.O. Box 82266, Phoenix, AZ, USA.
Ref. 3. J. W. E. Greene “Sizing and Controlling Incremental Development” *Managing System Development* November 1996 Applied Computer Research Inc. P.O. Box 82266, Phoenix, AZ, USA.
Ref. 4. J. W. E. Greene “Software Process Improvement- Management Commitment, Measures and Motivation” *Managing System Development* February 1998 Applied Computer Research Inc. P.O. Box 82266, Phoenix, AZ, USA.
Ref. 5. Lawrence H. Putnam “The Economic Value of Moving up the SEI Scale *Managing System Development* July 1994 Applied Computer Research Inc. P.O. Box 82266, Phoenix, AZ, USA
Ref. 6. : L.H. Putnam “Measures For Excellence: Reliable Software, On Time, Within Budget:” Prentice Hall New York 1992
Ref.7 J.W.E. Greene, The Software Control Office EC2 Software Engineering Conference Toulouse 1991
Ref. 8: J.W.E. Greene Getting a Runaway Software Development under Control EC2 Software Engineering Conference Toulouse 1990
Ref. 9: J.W.E. Greene “Avoiding the Premature Delivery of Software” QSM paper see www.qsm.com 1996
Ref. 10: For further information on QSM’s practices, refer to Lawrence H. Putnam and Ware Myers, Industrial Strength Software: Effective Management Using Measurement, IEEE Computer Society Press, Los Alamitos, CA, 1997, 309 pp.