

Practical ways to use the ISBSG data

Introduction

The ISBSG has established and now grows and maintains a database of software project data that can be used by software project managers, IT managers, CIOs and IT customer business managers.

The ISBSG aims to help improve software engineering practices and the business management of IT and provides its data and resources to do this. By using the ISBSG Repository, publications and tools, organisations now have capabilities to:

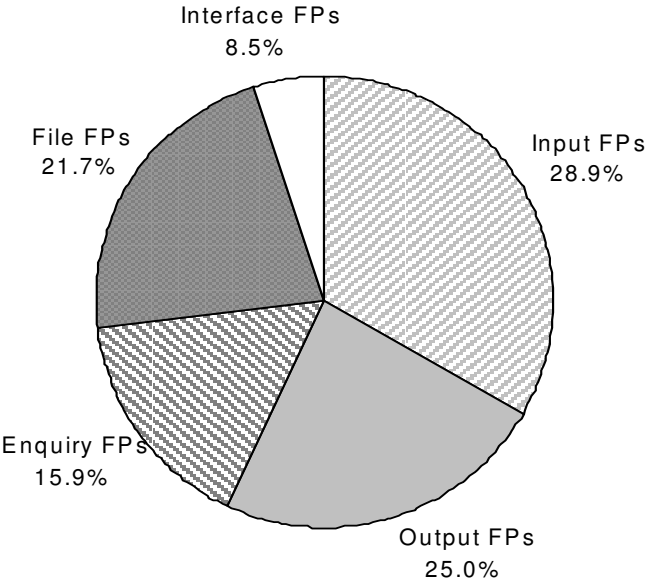
- Estimate software project size, effort, cost, duration
- Verify completeness of software requirements
- Be aware of project risk – by checking the reality of estimates
- Acquire custom-built software on a price per functional unit basis
- Determine an appropriate project team size
- Manage the progress of projects
- Benchmark project performance
- Plan software development infrastructure

Software Estimation

The ISBSG data and generated tables and equations can be used to help estimate the likely size, effort, duration, speed of delivery, productivity and cost of proposed software

Estimating software size

Software size can be estimated using the known relationships between the functional size components of software systems. The following pie chart shows the function point mix for the new development projects in the ISBSG repository:



Knowledge of these relationships has a number of valuable uses for the practitioner, one of which is the ability to predict the functional size of a new development project when the number of Internal Logical Files or External Inputs is known with any degree of certainty

For Example: If on developing a logical data model, there are found to be 40 logical tables, it may be reasonably assumed that these relate to approximately 40 Internal *Logical Files*. Analysis of the ISBSG Repository also shows that most Internal *Logical Files* in applications are rated as being 'low' to 'medium' in complexity. The mean score attributed to them across all projects is 8.6 function points.

Based upon the above, it can be assumed that the total score for the Internal *Logical Files* component of the function point count will be:

40 (ILFs) x 8.6 (mean score for Internal *Logical Files*)= 344 FPs

From the above pie chart it can be seen that the Internal *Logical Files* component of the function point count is typically around 22%. On this basis the total functional size of the required application is predicted to be around:

344 FPs x 100/22 = 1,564 FPs

This would be best relayed to the customer as ~1,570 FPs plus or minus 400 FPs (it is wise to add a contingency of 20 to 30% to early life cycle estimates to allow for undiscovered, required functionality).

This early life cycle sizing method should not be used as a substitute for a detailed measure produced by a formal count. It simply provides a rough, indicative estimate of the likely size of the project.

Estimating effort, duration, speed of delivery, productivity and cost

The ISBSG has provided data, tables and tools that allow IT projects to be estimated. Three macro estimating techniques are provided by the ISBSG: Regression Equation based estimation, comparative estimation and estimation by Analogy. All three techniques are detailed in the *Practical Project Estimation 2nd edition* book. The Comparative Estimating tool is available on the *Estimating Benchmarking & Research CD*.

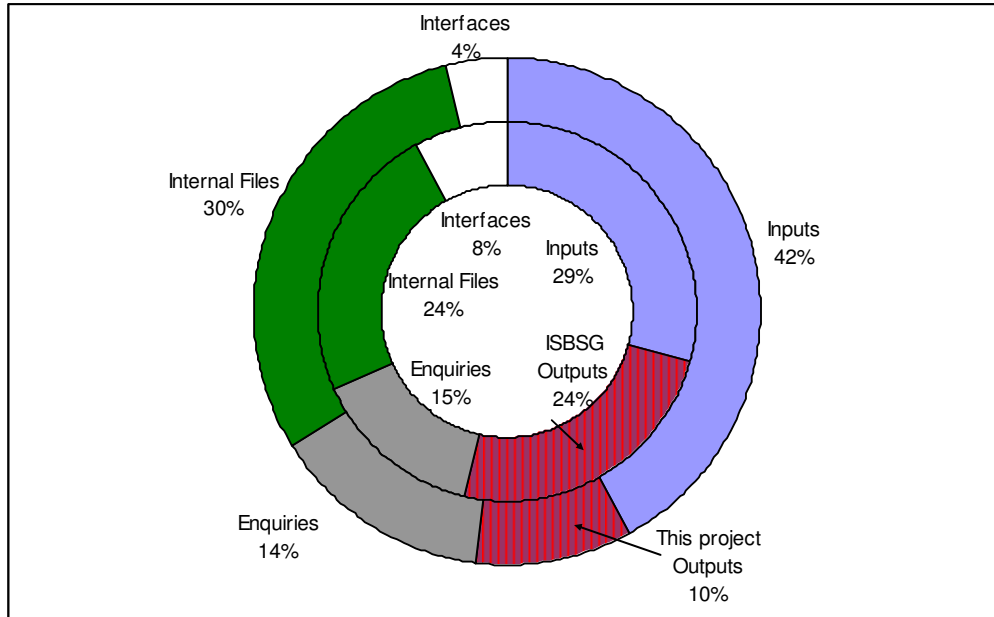
Reducing project risk

Two of the ways that the ISBSG data can be used to reduce software project risk are to verify the completeness of a specification and to assess the reality of a software project estimate.

Verifying the completeness of specifications

Analysis of the ISBSG history data over many years has revealed a consistent relationship between the phases of a project. These ratios can be used to verify the completeness of a project specification.

The chart below provides an example that shows the comparison between the ratio of the phases of a project specification and the ISBSG ratios. In this example the project manager should question whether all the likely required outputs are included in the specification.



Assessing the reality of software project estimates

The ISBSG has developed a tool called the Reality Check. An ISBSG Reality Check is a quick and low cost method which Project Boards can use to check that, prior to project approval, the development effort, cost and duration expectations of the project plan are realistic.

One of the major reasons that software projects fail is because the original cost and duration targets are not realistic. Typically they have been set by business imperatives which have directed that the project will deliver: a particular outcome, to a set budget, by a specified date. In the absence of any objective assessment of these outcomes the project is forced to proceed, whereas a simple reality check will indicate whether the expectations are realistic.

The ISBSG body-of-knowledge now makes the development of realistic estimates possible, timely and affordable.

Acquire software on a price per functional unit basis

The Victorian State Government in Australia has developed a new approach to purchasing software development. This approach is the methodology called "southernSCOPE". southernSCOPE allows businesses to purchase software development on a dollar per function point basis. This can be compared to the construction industry that often prices on a cost per square metre basis.

These are the eight steps to the southernSCOPE method:

1. The customer identifies that a need exists to acquire application software and engages a 'scope manager'. The scope manager is an independent person who specialises in software measurement.
2. The scope manager performs a preliminary function point count and provides an early but sound cost estimate and a realistic development timeframe for the project. (The scope manager uses the ISBSG tables as a guide to the likely hours per function point and duration that will be required.)
3. The customer prepares a short document outlining the need for the software, the characteristics and the constraints of the project and invites proposals to develop the software.
4. The customer selects the best and most realistic proposal then engages the successful developer with payment for the software based on the dollars per function point of the software delivered. (The ISBSG tables are used to ensure that the quote is realistic as well as competitive).
5. The development begins with a phase of analysis that produces a Requirements Specification.
6. The scope manager conducts a function point count on the Requirements Specification. Using this count, the customer decides exactly what functionality will be needed to produce the project outcomes while meeting the budget and the delivery date.

7. During the project, the scope manager ensures changes to the scope are understood and that the customer and the developer agree upon their price impact. (Additional functionality is sized and then multiplied by the \$ per FP price – a decision can then be made about its desirability)

8. At the conclusion of the project the customer makes payment to the developer on the basis of software delivered plus any agreed changes.

This methodology has proven to be very successful, reducing project over-runs to an average of only 4%. It delivers the core required functionality on time and budget. southernSCOPE is available in the public domain at: <http://www.egov.vic.gov.au/index.php?env=-2570:m1816-1-1-7:l0-0-1:-n0-0-0&reset=1>

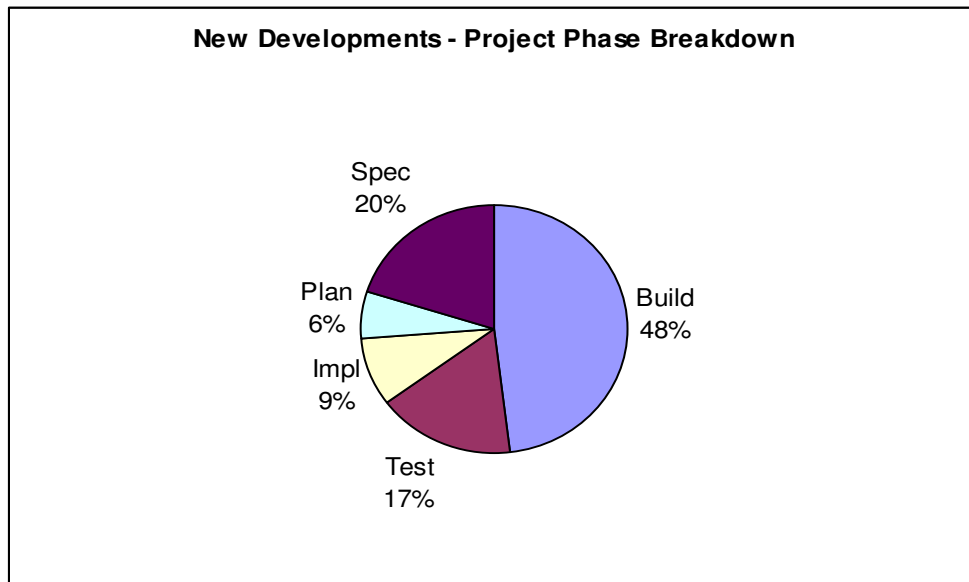
Team size planning

The ISBSG data shows that team size has a significant impact on productivity, after platform and language are taken into account. Team size impacts the productivity rate of a project because as the size of the team increases, communication becomes more difficult and more management, support & administration are needed.

The ISBSG Special Report on team size reveals that teams of nine or more are significantly less productive than smaller teams. Project managers faced with larger teams should adjust their project estimates to reflect this lower productivity expectation.

Manage the progress of projects

The ISBSG Project Phase Ratios can be used to manage the progress of a project and realign the project estimate based on the known effort taken for the early life-cycle phases. During the life of a project a project manager can measure how long each of the project phases are taking. If, in the early lifecycle phases, the project is experiencing over or under run, ie if there is a significant difference between the actual effort taken and the estimate, then the appropriate ISBSG pie chart of the ratios of the project phases can be used to estimate the likely overall impact on the project. These charts show the project phase ratios for new development, re-development and enhancement projects calculated from the projects in the ISBSG repository.



This chart shows the project phase ratios for the new development projects in the ISBSG repository.

So, for example, if the planning phase of a project has taken 120 hours instead of an estimated 80, then it will be worthwhile to use this actual figure to check the likely effort required for the remaining phases. Using the pie chart; if planning has taken 120 hours and represents 6% of the total project then the following figures can be used as a guide for the effort that might be required for the other phases:

Specification (20%)	400 hours
Build (48%)	960 hours
Testing (17%)	340 hours

Implementation (9%) 180 hours

The project manager can use these as a check against the estimated effort hours for each phase and adjust the figures if necessary.

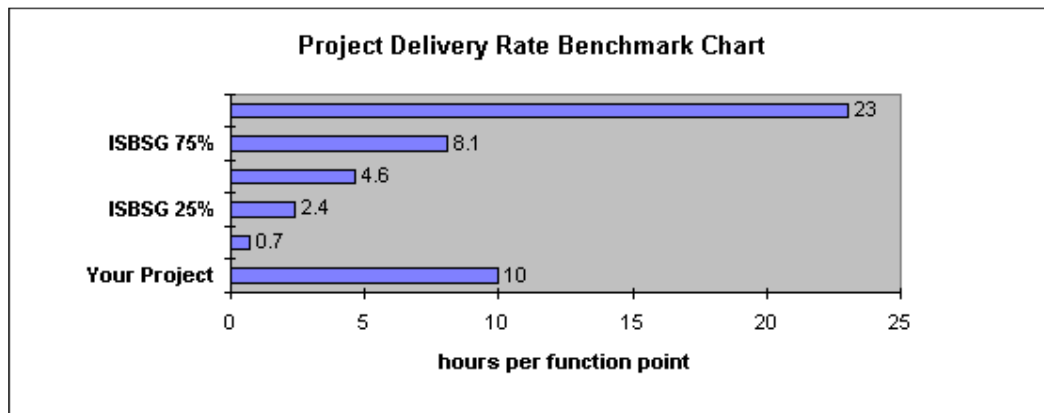
A full set of Phase ratio charts are available in *The Software Metrics Compendium*:

Benchmarking software performance

Many organisations want to understand how their software development performance compares to other similar projects, organisations or their industry sector as a whole. The ISBSG data allows benchmarking at all these levels.

When an organisation submits a project to the ISBSG repository a confidential, two page **project** benchmark report is provided. The following is a brief extract from that report:

For the two factors with the most significant impact on productivity, Development Platform and Language Type, the chart below shows how your Project Delivery Rate compares to projects with the same Development Platform and Language Type.



As the ISBSG project history data is open and publicly available there are many levels of benchmarking which are now possible, allowing organisations to understand their current level of performance and set goals for the future.

Plan software development infrastructure

The ISBSG regularly analyses its growing project history database and produces tables that show the productivity that can be expected from the many language and platform combinations. These same analyses also provide information on the impact, positive or negative, of various tools and techniques. This information can provide valuable input to consideration of new development environments.

Detailed productivity tables are available in the *Software Metrics Compendium*:

Summary

Now that a significant body-of-knowledge on IT projects is publicly available, IT Customers, IT Providers, Software Metrics Practitioners and Academia have data and information available to them that allows better planning and management of IT projects and research to improve the IT industry's performance in the future.