



To backfire or not to backfire...remains a question.

The conversion from Standard Lines Of Code to Function Points is known as 'backfiring'. It can seem an attractive option because it is much, much faster and cheaper than counting Function Points.

But how accurate is it? In a benchmarking context, jobs can depend on the answer but the precision of backfiring has never been more than a matter of opinion. In 2005, the UK Software Metrics Association (UKSMA) attempted to collect data to establish evidence for the effectiveness or otherwise of backfiring. David G. Rogers of the management committee of the UK Software Metrics Association, explains what happened.

Function Points were invented around 30 years ago as a way of measuring the size of a piece of software, so that you could rationally assess a software development department's productivity. It was no use counting how much code the programmers wrote, because computer languages differ so much: for example you might need 50 lines of Assembler to accomplish what one line of Cobol could do. But Function Points (FPs) were and are a way of putting a size on what a piece of software actually does.

Now suppose your Applications Maintenance organization is going to be assessed using a price or cost benchmark. The key commercial finding will be expressed as **£ / Maintained FP**. This is the price for maintaining the software portfolio divided by the size of that portfolio in Function Points.

But how do you measure the size of the portfolio? Either:

- Examine the software and count FPs, or
- Count Source Lines of Code (SLOC) and calculate the corresponding FPs using some long-standing ratios (for example, the average from all kinds of Cobol everywhere is around 100 lines of Cobol per FP).

The conversion from SLOC to FPs is known as 'backfiring'. It can seem an attractive option because it is much, much faster and cheaper than counting Function Points.

The precision of backfiring is a commercially important issue because of its use in benchmarking: remember, a benchmark is usually done because someone is asking the critical question: "How much SHOULD I be paying for the support of my software?" Jobs can depend on the answer.

Yet the precision of backfiring has never been more than a matter of opinion. Some software measurement experts (for example, David Consulting Group) assert that backfiring is wildly imprecise, with typical variability between 100% and 400%. Others claim that it can be as good as 20% or even 10%.

In 2005 UKSMA launched an initiative to gather enough data to settle the question once and for all. We asked people to report on their software:

- How big is it as measured in FPs?
- How big is it as measured in SLOC?
- What's the computer language?

There were a few additional questions for quality control purposes.

The idea was to accumulate enough data to know how variable those standard ratios are for various languages. If (for example) most of the Cobol programs reported were reasonably near the average number of SLOC per Function Point, that particular backfiring ratio would be vindicated; if most of them were not close, it would be discredited.

But ... almost no one answered the call for data, and the initiative has therefore failed. So if you want to benchmark an Applications Maintenance function, and someone suggests using backfiring to size the software ... UKSMA knows all the different opinions, and who holds them and why: but we still cannot say on the basis of evidence who is right!

David G. Rogers of EDS is a member of the management committee of the UK Software Metrics Association.