Customization of Function Point Analysis at Nokia

Petri Vesterinen and Kari Käärä (ed.)

Nokia Research Center
P.O.Box 45, FIN-00211 Helsinki, FINLAND
tel: +358 0 43761, fax: +358 0 4376 6855
email: petri.vesterinen@research.nokia.com

Abstract:

This paper describes a project to enhance SW project effort estimation at a SW production unit (SPU) within Nokia Corporation. The SPU produces embedded SW for Nokia’s DX200 telecom switches.

Within that project the standard function types (EI, EO, EQ, ILF, EJF) of the original function point analysis (FPA) were replaced by selected elements of the Nokia’s DX200 switching SW architecture. Data from 13 projects (out of over 40 available projects) was collected to be used for weight calibration, and statistical methods (e.g. regression analysis) were used to get the basic weights.

Complexity determination is not used in the customized method, but functions are categorized into new, modified and removed classes.

Six general characteristics, having clear impact to the effort, are used as productivity factors. These characteristics are specific to the SPU.

The early results are very promising. The customized method has been taken into daily use in all new projects at the SPU. There is no extensive statistical data yet to compare actual data to estimates by the customized method. It seems, however, as if the SPU will achieve a significant improvement in estimation accuracy.
INTRODUCTION

Importance of accurate estimates in SW project planning and control is never emphasized too much. Also at Nokia there is a permanent need to improve accuracy of effort and duration estimates of SW projects systematically.

This paper describes a project to enhance SW project effort estimation at a SW production unit (SPU) within Nokia Corporation. The SPU produces embedded SW for Nokia's DX200 telecom switches. The SPU had not applied function point analysis (FPA) earlier, but wanted to take a formal estimation technique based on FPA approach into use in new SW projects.

PROBLEMS WITH THE TRADITIONAL FPA

The original FPA by Albrecht was selected as the basis. Very soon some problems were discovered, as follows:

1. The FPA count depends on how to define the system boundary. There are several alternatives how to define the boundary in the SPU's SW architecture. It is not clear which abstraction level would be the most suitable to be considered as the 'system'.

2. The identification of standard FPA function types (EI, EO, EQ, ILF, EIF) is not a trivial task, because the DX200 SW architecture is so visible in design and specification phases. The basic FPA counting is considered to cause too much variation to be reliable, being too dependable on the person who counts.

3. There are some suspicions that standard FPA function types do not depict correctly all aspects of a real-time switching system as implemented.

4. It is also suspected that standard FPA does not result as accurate estimates in SW enhancement projects as in new SW development. FPA does not make any difference whether a function is new or modified. Collected experience suggests that there is a clear difference.

5. A preliminary study indicated that some SPU-specific systems characteristics (i.e. productivity factors) has to be added to the estimation model. Most IFPUG characteristics seemed to have no impact at all. On the other hand, according to our observations characteristics do not have an equal impact range, but clearly some characteristics at the SPU had larger impact than others.

6. The FPA is also quite a laborious method when defining SW size, the main element in this sense being the complexity determination. It is not reasonable to add so much cost determining complexity when the function types are not self-evident.

All these problems could certainly have been resolved to some extent, but an obvious question was put: why should the SPU adapt to FPA instead of adapting FPA to current concepts of the SPU's SW specification and design processes, and SW architecture?
PROS AND CONS OF THE CUSTOMIZED FPA

By customizing FPA we could easily overcome all problems mentioned above.

A drawback is that we can not define SW size so early within the SW life-cycle than using traditional FPA, because of more concrete function types (as will be shown later). We assume that selecting more concrete function types results to more accurate estimates.

Another potential drawback is the missed opportunity for benchmarking. Standard FPA is quite popular, and using a customized method we lose the ability to compare the SPU’s productivity rates to public-domain FPA data. We can, however, make comparisons within Nokia easily. Actually, a couple of other SPU’s have taken the similar method already into use. It is also possible to define conversion tables between the customized FPA and the original FPA.

FUNCTION TYPES OF THE CUSTOMIZED FPA

First we had to define the system boundary. It was defined to the (embedded SW) process level, being normally one SW source code file.

We wanted to improve definitions of the basic countable items (function types) so that they are more obvious in the SPU’s SW architecture. We investigated for example what is the external input and output in the DX200 SW architecture. We found that the communication over the system boundary is normally done with messages between processes. Internal logical files we replaced by physical files, etc.

We examined similarly all FPA function types and we came up with eight basic types in the SPU’s SW architecture. They are: master processes, hand processes, other programs, messages, fields, interface routines, MML commands, and MML command parameters.

Initially we considered more than these eight function types, but in the end we selected those who had the most evident correlation and impact to effort. We were able to define nearly all function types using existing SPU terminology and definitions. Thus there was no need for extensive training in this sense.

Originally we selected data from 13 SW projects to be used for weight calibration (currently we have a database consisting of dozens of SW projects). We then gathered data and used statistical methods (e.g. regression analysis) to get the basic weights.

COMPLEXITY DETERMINATION OF THE FUNCTION TYPES

We did not want to increase estimation costs by determining complexity of function types. On the other hand, the ‘complexity’ is strongly correlated to function types. We have, however, categorized each function type into three classes, depending on whether a new item is created, or an existing item is modified, or an existing item is deleted.
EARLY RESULTS

The early results are very promising. The customized method has been taken into use in all new projects at the SPU. Also a couple of other SPUs is using the customized method. Currently there are hundreds of separate users of the customized model.

There is no extensive statistical data yet to compare actual data to estimates by the customized method. Early results indicate significant improvement in estimation accuracy compared to earlier estimation practices.
CLASSIFICATION OF FUNCTION TYPES

We found that all occurrences of a function type are not equal. The effort seems to be different depending on whether the same SW functionality is created from scratch or from an existing SW block. We categorized each function type further into new, modified and removed ones. All classes inside a function type have accordingly their own weights.

This distinction seems to be very important, because most SW projects at the SPU are enhancements, or they add new functionality to the existing SW platform.

PRODUCTIVITY FACTORS

We also considered some general system characteristics (i.e. productivity factors). We came up with 6 key factors that have clear impact to the effort. At the SPU most important characteristics were associated with architectural and communicational complexity.

For example, there is a composite factor that we call 'communicational complexity'. It takes into account number of tasks, persons and working groups in a SW project. This factor reflects communicational overhead needed. The needed effort is clearly higher if the projects are poorly divided into tasks, and these few tasks are assigned to many persons.

These characteristics are specific to the SPU. Data derived from other SPUs suggest that each SPU has a specific set of productivity factors, and the impact of a common characteristic varies a lot.

CONVERTING SIZE TO EFFORT

The FPA size needs to be converted to an effort estimate. Of course we can simply multiply the size by the SPU's standard productivity. An interesting question still remains: do we have economy-of-scale or diseconomy-of-scale?

The architectural level of the 'system' was defined to be quite low (embedded SW process level). Thus also work items of SW projects to be estimated are relatively small. We noticed that we have three separate economy areas:

- for the smallest items we have diseconomy-of-scale because of the 'get-started' overhead,
- for the typical-size items we have obvious economy-of-scale
- after a certain SW size limit it turns out to be diseconomy-of-scale because of the increasing (human and SW) communication

We have given such WBS planning instructions that only the economy-of-scale area is utilized.

The economy-of-scale impact is incorporated into our estimation model so that we can take this into account together with productivity.