The Next Frontier: Measuring and Evaluating the Non-Functional Productivity

LUIGI BUGLIONE

ETS / Engineering.IT SpA – Rome (Italy)
Email: luigi.buglione@eng.it

RECENTLY IFPUG RELEASED THE NEW SNAP (SOFTWARE NON-FUNCTIONAL ASSESSMENT PROCESS) METHOD, AIMED TO SIZE THE NON-FUNCTIONAL SIDE OF A SOFTWARE APPLICATION. FROM FPA CREATION ON, NFRS (NON-FUNCTIONAL REQUIREMENTS) WERE TYPICALLY DEALT AS "REQUIREMENTS OF A LESSER GOD", WHILE THEY REPRESENT SIMPLY A DIFFERENT AND COMPLEMENTAR CONTRIBUTION TO A PROJECT THAN FURS (FUNCTIONAL USER REQUIREMENTS). THIS PAPER WILL TRY TO INTRODUCE THE RATIONALE AND TIPS FOR MEASURING THE NON-FUNCTIONAL PRODUCTIVITY AND USE IT JOINTLY WITH THE 'FUNCTIONAL' ONE IN ORDER TO OBTAIN MORE RELIABLE ESTIMATES FOR NEXT PROJECTS.

Part 1 – "How to think about it"

What is Productivity?

Webster-Merriam dictionary defines productivity as "*the rate at which goods are produced or work is completed*". This general definition applies to any domain. Applied to FPA for sizing software projects, this is calculated as the ratio between the number of Function Points (goods produced) and its project effort (FP/Effort).

But analyzing such formula, a simple question arises: which is the related entity for any part of such formula? Applying the **EAM** (Entity-Attribute-Measure) analysis [15], 'FP' are a measure (M) of the functionality (A) of a software product (E), while 'man-days (or man-hours)' represents a measure (M) of the effort (A) of a software project (E). The two parts refer to different entities and are therefore not directly comparable - the formula needs refinements. Just a short example: let's suppose to have deployed 100 FP in 250 man-days. Productivity – as typically calculated – should be 100FP/250 man-days (0.4FP per man-day). But if we add 15 more man-days for new stress and performance tests or for a further quality audit for guaranteeing a proper software quality level before its release, no new FP would be created, but a higher number of man-days would be computed in the productivity formula, with an updated value of (still) 100FP/265 man-days, with a lower 0.38 FP/man-day productivity value.

Again, the introduction of an 'adjustment factor' as a multiplier of the product functional size from an arithmetic point of view is not proportional because we could run VAF-related activities for several man-days but with a final TDI lower than 35 points for obtaining a final 'adjusted' FP (AFP) value lower than the initial one. Observing numbers, it would be paradoxical in both cases: doing more (non-functional activities, anyway more man-days) for being (apparently) less productive (taking care of the current version of this formula)… thus it's possible to call this as a 'nominal' productivity [1]. From a logical viewpoint, the two parts – functional and non-functional - should have been added, not multiplied. They are simply different in nature (it is possible to run a change request only containing the requirement to improve system performance without any modification to FURs – Functional User Requirements).

Thus we need to look at the whole picture and apply the 'divide-et-impera' concept, as shown in Figure 1, for a better understanding of how to improve the whole size & estimate process flow.

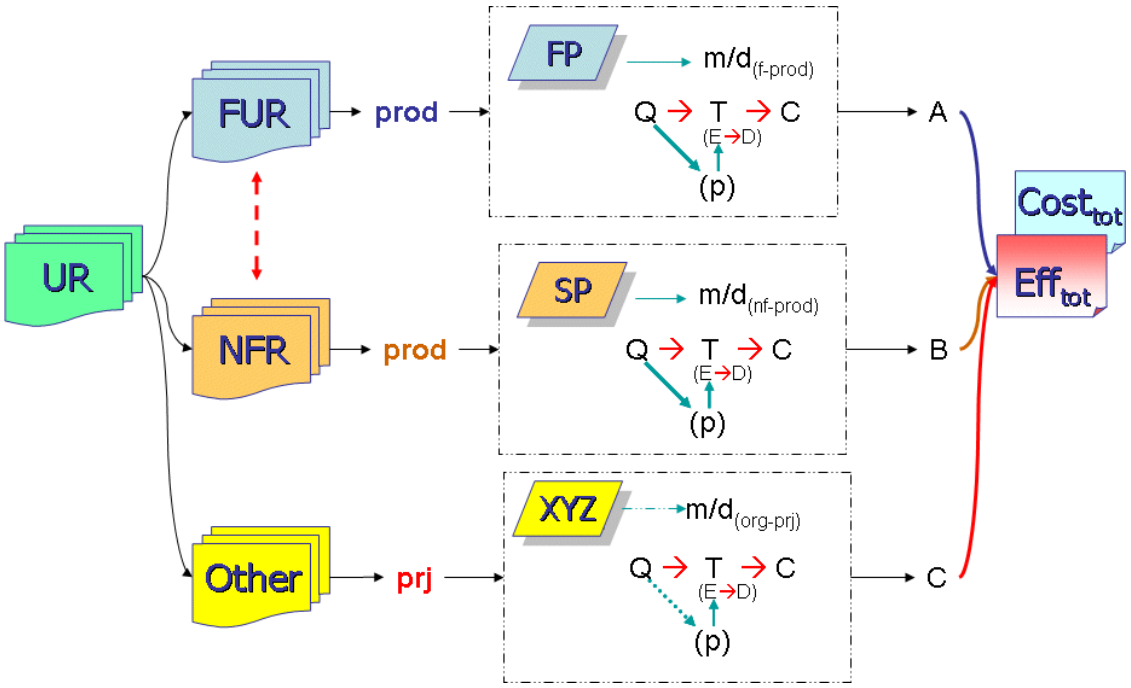

Fig. 1: *From User Requirements to the final overall project effort and costs*

Original User Requirements (UR) can be classified at two levels: product-related and project-related. The first one can contain FUR (Functional User Requirements) and NFR (Non-Functional Requirements), where the first ones can refer only to the 'product' entity while the latter refer both to the 'product' (e.g. make the system compliant with usability/accessibility guidelines)[1]. The second level contains requirements about the 'project' itself (e.g. a weekly project status review must be run), sometimes expressed as constraints, but producing additional effort to be considered within the project scope[2].

The 'Q→ T→ C' chain in Figure 1 expresses '**Quantity** → **T**ime → **C**ost'. That is a generic, logical chain from the beginning to the end for any size & estimation activity: you need to determine the amount of activities to run, measured by one or more unit(s) of measure (Q). According to this value, it's possible to estimate how much time (T) – that must be meant as 'E' (effort) and 'D' (duration) we need to run such work, whether by experience, analogy or statistically derived. In any case, we use – implicitly or explicitly – a reference productivity (p) value, as usual, returning the effort needed for working on that amount of requirements. Once the time needed (both in terms of efforts and schedules) is estimated, it is possible to calculate the final cost, summing up fixed and variable costs for any chunk of activities. In fact, the risk is to pay too much attention to the 'sizing' issue more than to the real goal, which is to achieve superior project estimation, reducing as much as possible the so-called 'cone of uncertainty' [16] from earlier phases till the project closure. Sizing – even fundamental and really valuable – is therefore only an intermediate step for the more comprehensive Estimation process.

Thus, looking back to Figure 1 and referring to the 'product' entity, the product size unit for FUR can be IFPUG FP or any other *fsu* (functional size unit), while for NFR there are several possible approaches. Recently IFPUG published an experimental measure, called **SNAP** (Software **N**on-functional **A**ssessment **P**rocess), whose unit of measure is called **SP** (**S**NAP **p**oints). Other requirements related to project activities (e.g. measurement, quality assurance, project management, etc.) will be discussed later.

As summarized in the table below, the 'productivity' formula could evolve from the current (a) definition to the (c) one (where 'XYZ' stands for a possible size measure for expressing the effort for *project*-level activities), but achieving as earlier as possible at least the (b) one, according to a higher maturity and capability levels of an organization in sizing & estimating projects.

$\frac{FP_{FUR-prod}}{Effort_{prj}}$	$\frac{FP_{FUR-prod}}{Effort_{FUR-prod}} + \frac{SP_{NFR-prod}}{Effort_{NFR-prod/Org-Prj}}$	$\frac{FP_{FUR-prod}}{Effort_{FUR-prod}} + \frac{SP_{NFR-prod}}{Effort_{NFR-prod}} + \frac{XYZ_{Org-Prj}}{Effort_{Org-Prj}}$
(a) Nominal	*(b) Functional and Non-Functional*	*(c) Functional, Non-Functional and Org-Project*

1 The dotted line between FUR and NFR is because a NFR can be 'functionalized' (see [21])

2 All ISO/IEEE/PMI standards definitions are available in SEVOCAB: www.computer.org/**sevocab**

productivity	*Productivities (Level-1)*	*Productivities (Level-2)*

Therefore, the 'next frontier' is to stimulate organizations into putting in action the (b) scenario: starting to measure NFRs and calculate a distinct productivity value, taking into account different efforts from different requirements types within the project scope for achieving better estimates. Anything can be refined (see the 'c' scenario), but one step at a time, in an evolutionary manner is usually the best approach.

Of course, a further scenario could be to have all sizes gathered (functional, non-functional, org-project) and use them in a multiple regression analysis as independent variables to be related to the whole project effort. It could be useful when a project will measure correctly all the three pieces, but with some attention points, because at least two situations can arise:

- SP can be referred to a large number of combinations in terms of categories part (or not) of the final size value (impacting therefore on the final effort) → in a benchmarking we'd compare only similar non-functional profiles for reducing the probability of lower R^2 in a correlation analysis.
- Some maintenance projects could only require to derive the effort for product NFRs and Org-Project levels (e.g. populate a yet existing database; updating the user interface of a system with accessibility mechanisms). In that case, a project manager should derive the whole project effort as the summation of the (b) and (c) flows from Figure 1. But since it's difficult at this time to define which project attributes could be sized and which productivity levels use for deriving the related effort, it seems more viable to sum the two efforts, independently calculated.

That's why the proposal is to look at in an evolutionary manner (maturity) to the way organizations can gather their size and effort data at a certain level of granularity for improving their estimates. But before the size data, the effort data can represent the 'here and now' for a project, and they can be refined and split according several criteria, revealing many valuable information about the way a project has been managed.

FURs and NFRs: two pieces of the same puzzle

The latest IFPUG CPM v4.3.1 [11] starts Part 2 with this sentence: "*IFPUG's method for function point analysis is an ISO standard and must be conformant to ISO/IEC 14143-1:2007. The method can measure "functional size" only and not "non-functional size". This does not mean that the non-functional size cannot, or should not, be measured, only that it must be clearly stated as a separate measure*". Later in the same page, it introduced ISO/IEC 14143 standard [7] that defines the Functional Size as "*a size of the software derived by quantifying the Functional User Requirements (FUR)*", where FURs are related to the software 'product' entity and not to the whole, more comprehensive 'project' scope, as discussed before.

It was 'embedded' in the FPA concept and idea since Albrecht's first 1979 paper [12], simply because the 'F' letter stands for 'functional', explicitly excluding its opposite, the 'non-functional' side. The inclusion of the VAF (Value Adjustment Factor) – representing the product and project non-functional side (and increasing the number of GSCs (General System Characteristics) from 10 to 14 [13]) must be analyzed from an historical point of view, coming from a period with a lower level of knowledge on Software Engineering than now. An 'application' or 'system' was interpreted as 'the project', and with a lower NFR contribution than in modern projects [14]. That's probably why often in many presentations and papers FPs are still presented as a 'project size' and not (as they really are by definition) a 'size of the product FURs', with the consequence of not including size related to NFRs.

Another possible reason: since "*you cannot control what you cannot measure*" but "*you cannot measure what you cannot define*" and "*you cannot define what you don't know*", the adoption of an ordinal scale for GSCs as well as in COCOMO [16][18] for scale and cost drivers is the objective evidence that anything can be measured, but at different levels of granularity, due to the inner knowledge of a certain entity of interest (EoI). The more you know about your EoI, the easier you can define it; the easier

you define it, the easier the way to determine its quantitative view by a formula. And now the time to start to quantify NFRs has come.

IFPUG SNAP: a new method for sizing product NFRs

This clarification during the years, pushed by the ISO/IEC 14143 family of standards as well as a deeper knowledge than 30 years ago, represented the trigger for creating a new method for sizing NFRs, the other 'side of the story'. IFPUG recently released one concept, called **SNAP** (Software Non-functional Assessment Process) [5], in September 2010. SNAP proposes a 6-step process to be run in parallel to the 6-step CPM process, as depicted in Figure 2.

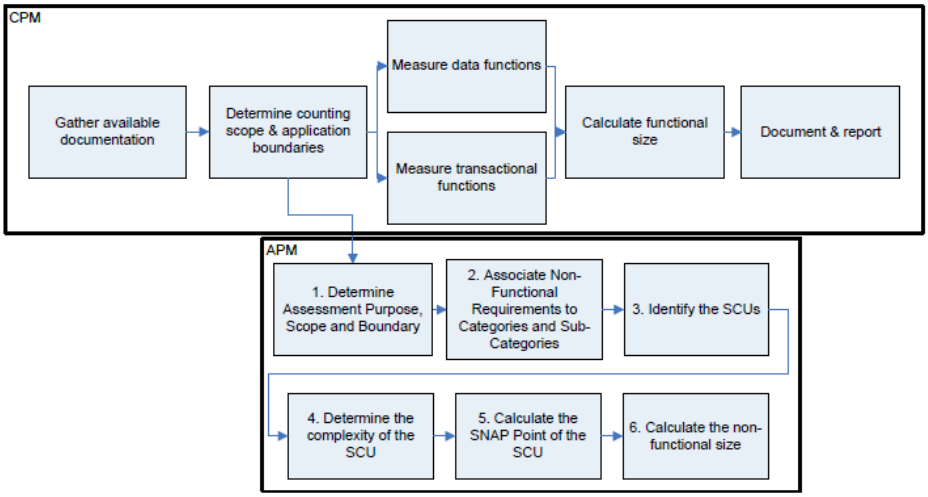

Fig. 2: *The link between FPA and SNAP processes [5]*

From the analysis of product NFR, they can be classified into a series of categories (4), sub-categories (16) and associated to SCUs (SNAP Counting Unit). The summation of these parts will return the final number of SP (SNAP Points), in a similar manner to traditional Function Points (FP) when dealing with FURs. The APM (Assessment Practice Manual) provides all the definitions and complexity tables for calculating SP.

Non-functional productivity: what I need to calculate it?

As previously discussed, our target must be to implement scenario (b) and progress to scenario (c). The upper parts of those formulas are about sizing, but the lower ones are about effort. Thus, for obtaining proper productivity values, we need to provide criteria and strategies for gathering NFR-related effort, based on common-sense and – of course – moving from the official and shared definitions contained in the IFPUG and ISO guides. Figure 3 proposes the ISO definition for a product NFR:

> **nonfunctional requirement. (1)** a software requirement that describes not what the software will do but how the software will do it *(ISO/IEC/IEEE 24765:2010 Systems and software engineering--Vocabulary) Example:* software performance requirements, software external interface requirements, software design constraints, and software quality attributes.
>
> *Note:* Nonfunctional requirements are sometimes difficult to test, so they are usually evaluated subjectively. *Syn:* design constraints, non-functional requirement *See Also:* functional requirement, quality requirement

Fig. 3: *The 'non-functional requirement' ISO definition (www.computer.org/sevocab)*

Of course, for achieving the scenario (b) and (c) productivity definitions we need also to derive the other NFR-project related activities and effort. The best source of information for deriving such information comes from the typical planning & estimation work product many people use, the project **Gantt chart** and its list of planned activities.

ID	Task Name	Work
1	**'Splitting Effort' Project**	**278 hrs**
2	**Project Management**	**19 hrs**
8	**Quality Assurance (QA)**	**9 hrs**
11	**Analysis**	**38 hrs**
17	**Design**	**72 hrs**
23	**Construction**	**80 hrs**
27	**V&V**	**56 hrs**
30	**Release**	**4 hrs**

(a) – compacted view

ID	Task Name	CMMI-DEV PA	CMMI-DEV Process Group	Req. Type	Work	FUR-related Effort	NFR-Prj related Effort
1	**'Splitting Effort' Project**				**278 hrs**	**0**	**0**
2	**Project Management**				**19 hrs**	**0**	**0**
3	Planning	PP	Prj Mgmt	Org-Prj	16 hrs	0	16
4	**Monitoring & Control**				**3 hrs**	**0**	**0**
5	Meeting #01	PMC	Prj Mgmt	Org-Prj	1 hr	0	1
6	Meeting #02	PMC	Prj Mgmt	Org-Prj	1 hr	0	1
7	Meeting #...	PMC	Prj Mgmt	Org-Prj	1 hr	0	1
8	**Quality Assurance (QA)**				**9 hrs**	**0**	**0**
9	Product QA	PPQA	Support	NFR	6 hrs	0	6
10	Process QA	PPQA	Support	Org-Prj	3 hrs	0	3
11	**Analysis**				**38 hrs**	**0**	**0**
12	Req. Elicitation	RD	ngineering	FUR	20 hrs	20	0
13	**User Requirements**				**18 hrs**	**0**	**0**
14	UR - functional	RD	ngineering	FUR	8 hrs	8	0
15	UR - nonfunctional	RD	ngineering	NFR	6 hrs	0	6
16	FP-sizing	MA	Support	NFR	4 hrs	0	4
17	**Design**				**72 hrs**	**0**	**0**
18	Functional Specification	RD	ngineering	FUR	28 hrs	28	0
19	Architectural Specification	RD	ngineering	NFR	14 hrs	0	14
20	**Test Plan**				**30 hrs**	**0**	**0**
21	TP - Functional part	VER	ngineering	FUR	12 hrs	12	0
22	TP - Non-functional par	VER	ngineering	NFR	18 hrs	0	18
23	**Construction**				**80 hrs**	**0**	**0**
24	Construction	TS	ngineering	FUR	52 hrs	52	0
25	Customization	TS	ngineering	NFR	12 hrs	12	0
26	Fixing bugs	TS	ngineering	FUR	16 hrs	16	0
27	**V&V**				**56 hrs**	**0**	**0**
28	Black-box	VER	ngineering	FUR	20 hrs	20	0
29	White box	VER	ngineering	NFR	36 hrs	0	36
30	**Release**				**4 hrs**	**0**	**0**
31	Release F-xyz	VAL	ngineering	NFR	4 hrs	0	4

(b) by requirement type

Fig. 4: *Activities on a Gantt chart*

Some tips/point of attentions could be as follows:

- Analyzing a Gantt chart, any activity can potentially contain both FUR-related and NFR-related activities: e.g. a 'Test' activity can be split at least into two sub-activities, Functional Testing (black box tests from FURs) and Non-Functional Testing (white box tests on source code as well as black box tests from NFR, e.g. performance, stress tests). Again, Analysis can be split into Functional Analysis (generating FPs) and Non-Functional (or Technical) analysis (generating SPs). Also, this split aligns well with a project management viewpoint, as these activities are potentially run by different team members with different daily costs, impacting on the final overall schedule and economic planning.
- Analyzing User Requirements (UR), any FUR has its non-functional side: a FUR represents a 'what' to do, not the 'how' to do it (the non-functional side e.g. usability or maintainability as well as all the "ilties" proposed by ISO/IEC 9126-1:2001 standard [19] (now evolved as ISO/IEC 25010:2011 [20])). Since that macro task will be completed only after the verification and validation of both items (the rough functionality and its compliance to usability parameters), it needs to be split in two from the beginning, and each one sized according to the relevant size unit. Otherwise the risk is to continuously undersize (and therefore underestimate) the overall project effort from early phases and establish contracts on understated project scope.
- Reach the higher granularity level as possible when analyzing requirements: the target would be to describe as early as possible the EPs – Elementary Processes. The higher the granularity level, the more affordable the estimate of such activity from early phases. A short example: let's suppose to plan a generic 'Analysis' task for 40 man-days. When refining the single bar into at least two ones (functional vs technical analysis), typically the sum of the two single effort values can be higher than the old one, simply because the more the knowledge we have on what we have to do, the more the sub-activities to be run we'll insert in the chart. And the same happens also when estimating FPs from FURs: initially we could e.g. estimate a generic CRUD (4 EPs), but refining the analysis according to the 'how' we typically realize such functionality, possibly there will be more EPs (e.g. further EQs for combo boxed realizing partial queries in a window) and LFs than at a higher analysis level.
- Classify tasks by their related processes and process groups: it can be possible to not be able to classify as FUR or NFR-related a certain activity. A practical way to achieve it can be to link activities to their related processes and understand which kind of process we are dealing with. The

choice and/or reference to some models can help a lot. E.g. looking to CMMI-DEV [2][3] measuring Function points is an activity related to MA (Measurement & Analysis, SG 2), that's a Support process area, thus a NFR-related activity. In fact, *a rule of thumb* to be applied to a well-known model as CMMI is that all activities referable to process areas from the Support, Project Management and Process Management process groups are NFR-related[4]. The ones about 'Engineering' processes are typically FUR-related but not always - there is a sort of 'grey zone'. E.g. as said before, the testing process (VER – Verification) can contain both functional as well as non-functional tests. Thus, any single task from Engineering processes should be analyzed and classified looking at the specific entity it refers to.

- Classify tasks - the 'Deployment' concept: another viewpoint for validating the previous 'rule of thumb' is to observe if a task directly creates or contributes to create a FP (or a SP). The common-sense rule is that we have to take into account only those activities needed for *deploying* a single (functional or non-functional) requirement. If the answer is 'yes', it can be classified accordingly as FUR-related in the first case, otherwise as NFR-related. Some examples: in the case of FPA, writing a user requirement (linked to the RD process– Requirement Development) is a mandatory input for determining FURs which determine the number of FPs - thus it is a FUR-related activity. On the opposite side, a quality audit is an activity related to the PPQA (Process/Product Quality Assurance) CMMI-DEV process area (PA). That is a Support PA that does not contribute directly to create any further FP and it is thus classified as NFR-related. On the opposite side, creating a Test Plan with test cases can be classified as FUR-related for the part (and effort) depicting functional tests while as NFR-related for the part (and effort) depicting performance and stress test. These tasks are usually performed by different team members with different productivities and daily costs, as said before. And so on. In conclusion, the activities coming from the Analysis-Design-Construction-Test lifecycle phases are potentially contributors for determining the FUR (or NFR)-based effort at the product level, while the remaining ones (Planning, Monitoring & Control, Release, Post-Release support) are *organizational-project* -related[5].

Once one has refined the initial Gantt chart and classified each bottom level leaf as FUR or NFR-related, it is possible to have two 'Effort' columns, whose sum is of course the final project estimated effort. Of course, this is an oversimplification but it can be useful and pratical, arising from the observation that many organizations are currently able to size product FURs with FP but are not able to do the same for the other two parts (product NFRs and org-project requirements). One can describe a sort of 'project profile' from a simple proportion (the two effort %), which can help to classify and better filter historical data in right clusters. E.g. a MIS project typically can have a 70-30% (FUR vs NFR) related effort balancing, a web-based one c.a. a 60-40% FUR vs NFR proportion, a DWH one c.a. 50-50% or 45-55%. Thus, when an organization does not have such data, they risk creating project clusters with projects really different in nature. 'Nominal' productivity levels are quite variable, and, as a consequence, lead to poor estimates, with high MRE (Mean Relative Errors) and low R^2 (determination coefficient) than expected.

What is missing today and represents the next challenge to face? Just to start measuring both the non-functional size and at the same time the related effort needed. Coming back to the "Q→ T→C" chain in Figure 1, with SP we can have a new 'quantity' for product NFR, but we need to track and gather also that kind of effort separately from the FUR-related. And it'll be represent the starting point for determining non-functional productivities.

[3] Of course, the same rationale can be adopted for any other process models such as ISO/IEC 15504 (aka SPICE) [10] and ISO/IEC 12207 [9].

[4] CMMI-DEV v1.3 classifies its processes into four (4) process categories: *Engineering* (covering the development and maintenance activities that are shared across engineering disciplines), *Project Management* (covering the project management activities related to planning, monitoring, and controlling the project), *Support* (covering the activities that support product development and maintenance), *Process Management* (covering cross-project activities related to defining, planning, deploying, implementing, monitoring, controlling, appraising, measuring, and improving processes)..

[5] This distinction has an impact in the case you'd like to implement the (b) or the (c) scenario formulas (see Figure 3).

Part 2 – "How to apply it"

Benchmarking: start internally, compare externally

After the proposal about the 'how' the non-functional effort could be gathered within an organization allowing to realize at least the (b) scenario, now it's time to discuss how to use the obtained productivities values and main improvement points to put in action in the short term. Differently from FSM methods, well known and applied from many decades, SNAP as well as other NFR-sizing methods is young and there is not yet an external benchmark available for comparing values. Therefore some strategies need to be established for making quicker as much as possible the determination and sharing of the new non-functional productivity values, as currently done with the 'nominal' productivities.

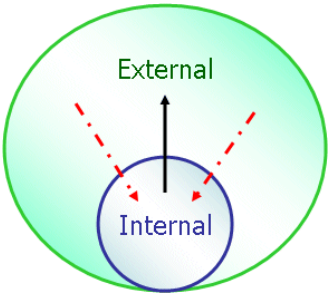

Fig. 5: *Internal and External benchmarks: a two-sided information flow*

Figure 5 proposes a simple information flow: (a) organizations with a good measurement capability gather internally their data and after share them for external benchmarks. ISBSG (www.isbsg.org) is the most valuable publicly source of information with its repositories both for D&E (Development & Enhancement) and M&S (Maintenance &Support); (b) organizations with a medium-low measurement capability typically find productivity figures from external benchmarks, trying to import data and possibly idea about how to realize internally the same data gathering process. Thus, from a holistic and more comprehensive viewpoint, it's a two-sided information flow, where both stakeholders (corporations and benchmarking organizations) take mutual advantages from the knowledge of others' practices, data and data management.

Since productivity is the ratio between a size and its related effort, in many benchmarks a lot of attention was paid to size figures but not so much to effort data. The aim is to allow organizations to determine project clusters in the more precise way, trying to do not mix 'apples with oranges': two 1000 man/days projects with different distributions of effort by SLC phase and requirement type will not have of course the same management issues, schedule, and costs. Thus, some tips/point of attentions could be as follows for obtaining valuable and affordable non-functional productivity data:

- Determine the affordability level for effort data: since the information from past project could be incomplete or not completely affordable (e.g. missing objective evidences, team members not anymore in the organization, etc…), as ISBSG does for classifying projects by the completeness of project data, a similar field could be inserted with the same labelling, from Level A (best) to Level D (worst).
- Establish the granularity level for effort data: because organizations has effort data in their historical databases, the above approach moving from Gantt charts could be easily adopted also analyzing past projects, in order to store more data in few time. Several level of granularities can be established:
    - Level 0 (lowest) – only the total project effort value (no split between FUR-NFR efforts)
    - Level 1 - approximated percentage for FUR and NFR efforts (assigned)
    - Level 2 – absolute values and percentages for FUR and NFR efforts (gathered)
    - Level 3 (highest) – abs value/percentages for FUR-NFR effort by SLC phase (matrix view – complete)

The Project Continuum: using FP and SP together

Last but not least, some final thoughts about the next joint usage of FPA and SNAP (or anyway of a FSM method and a NFR sizing method/approach), as stressed in the SNAP APM v1.0 Appendix B. Classifications and taxonomies help people in creating homogeneous clusters for any entity of interest, but their limit could be to do not offer a global view. For instance, looking at process improvement models, the above cited CMMI has three constellations (DEV – Development, SVC – Service, ACQ –

Acquisition): typically ICT organizations use the DEV one, but it doesn't include many maintenance issues contained in the SVC one. Thus, nonetheless the entity to be managed is the *project* often the risk is to apply only for one of the two models, returning a partial assessment of what really the project does and should do after implementing some improvement actions. The *project* must be seen as a *continuum* of activities during time and we need to cross partial views – that could be useful for some partial goals – into a broaden view for a global evaluation.

For instance, facing the business viewpoint, many ICT contracts are currently managed using the solely FP even if they cannot size the red-dotted line, that are the 'corrective maintenance' sub-type activities (Figure 6) [4][8] as well as some 'adaptive' maintenance activities as e.g. populating databases or deriving functionality by data configuration (that does not contribute to FPA, and is one of the new SNAP sub-categories), because no FUR is modified. It doesn't make sense to run NFR-based activities for (not) recognize the effort (and costs) for those activities using the wrong unit of measure, but it's what typically is done, maybe for the above mentioned misunderstanding that FP are NOT a *project* size, but a (software) *product* functional size, to be summed with other (partial) sizes.

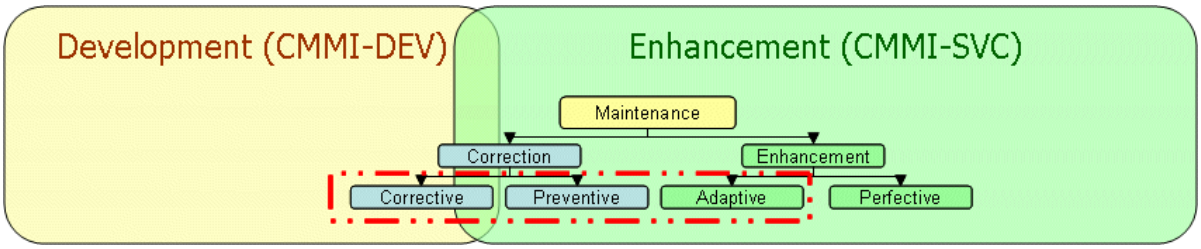

Fig. 6: *The Project Continuum - Development (DEV) and Maintenance (SVC – Service) (Elaboration from [8])*

Thus, more than approximating the size of those maintenance activities calculating proportions between FPs and LOCs and still paying by FP-related values, it'd be better to start introducing other non-functional measures (e.g. from ISO/IEC 9126 parts 2-3-4 standards or SNAP Points) for a better sizing & estimating and start give to those new measures an economic value. The sum of the two parts will be the final, total cost/price for that activity. Another objective evidence for supporting this thesis: the daily costs for a database administrator, a project manager, a CFPS or a CSMS expert, a QA assistant, etc. working on NFRs is typically higher than an analyst/programmer – typically working on FURs.

Therefore in order to properly determine/evaluate the cost for a project, whatever the distribution of effort (FUR vs NFR related, according to the development and project types), we need at least those two parts (and related elements, such costs), moving from their sizes and efforts.

Conclusions & Next Steps

Function Point Analysis (FPA) as well as all the other FSM methods is a very effective way that has helped organizations to improve the way they manage their requirements, with a focus on the functional ones. Since the less we know about one entity of interest, the less we are able to measure it, after some years we know enough to start measuring also NFRs. There are several approaches and techniques and IFPUG recently proposed its own one, called SNAP (Software Non-functional Assessment Process), whose unit of measure is called SP (SNAP Points). The introduction of a complementary measure to FP (or another fsu – functional size unit) brings to a win-win situation for many stakeholders:

- project managers and organizations, no matter if customers or providers, because they'll able to refine their estimates, reducing project/organizational estimation errors (e.g. measured by MRE/MMRE) leading to an optimized use of project resources and assets;
- the measurement community, because it will stimulate a way to use the SNAP categories and/or similar approaches and taxonomies (e.g. ISO/IEC 9126-1:2001, FURPS, etc...) for sizing NFR both at the product and project level.

So, the next frontier is simply to start and face this new exiting challenge!

"*Computers are nonfunctional.*"
(Dr. Spok to Captain Kirk, *Star Trek IV – The Voyage Home*, 1986)

5. References

[1] Buglione L.., *Some Thoughts on Productivity in ICT projects,* WP-2008-02, White Paper, version 1.3, August 2010, URL:

http://www.semq.eu/pdf/fsm-prod.pdf
[2] CMMI Product Team, *CMMI for Development,* Version 1.3, CMMI-DEV v1.3, CMU/SEI-2010-TR-033, Technical Report, Software Engineering Institute, November 2010, URL: http://goo.gl/sCC9i
[3] CMMI Product Team, *CMMI for Services,* Version 1.3, CMMI-DEV v1.3, CMU/SEI-2010-TR-034, Technical Report, Software Engineering Institute, November 2010, URL: http://goo.gl/C36ls
[4] IEEE Std 1291-1998, *IEEE Standard for Software Maintenance*, IEEE Computer Society, 1998
[5] IFPUG, *Software Non-functional Assessment Process (SNAP) Assessment Practice Manual (APM)*, version 1.0, September 2011, URL: www.ifpug.org
[6] ISO, 9001:2008 - Quality Management Systems: Requirements, December 2008
[7] ISO/IEC, *International Standard 14143-1 - Information Technology - Software Measurement - Functional Size Measurement - Part 1: definition of concepts*, February 2007
[8] ISO/IEC, *International Standard 14764:2006 – Software Engineering – Software Life Cycle Processes - Maintenance*, 2006
[9] ISO/IEC, IS 12207:2008, *Systems and Software Engineering -- Software Life Cycle Processes*, International Organization for Standardization, 2008
[10] ISO/IEC, IS 15504-5:2006, *Information technology -- Process Assessment -- Part 5: An exemplar Process Assessment Model*, International Organization for Standardization, 2006
[11] IFPUG, *Function Point Counting Practice Manual (CPM)*, release 4.3.1, January 2010, URL: www.ifpug.org
[12] Albrecht A. J., "Measuring application development productivity" in Proc. Joint SHARE, GUIDE, and IBM Application Development Symp., 1979, pp. 83-92.
[13] Albrecht A. J. & Gaffney J. E., "Software function, source lines of code, and development effort prediction: A software science validation," IEEE Trans. Software Eng., vol. 9, no. 6, November 1983, pp. 639-647.
[14] Buglione L., *Tutto quello che avreste voluto sapere sui Function Points (e non avete mai osato chiedere)*, Gruppo Utenti Function Point Italia - Italian Software Metrics Association (GUFPI-ISMA), Rome (Italy), 6 May 2008, URL: www.gufpi-isma.org
[15] Buglione L., Ebert C., *Estimation*, Encyclopedia of Software Engineering), Taylor & Francis Publisher, April 2012, ISBN: 978-1-4200-5977-9
[16] Boehm B., Software Engineering Economics, Englewood Cliffs N.J., Prentice-Hall Inc., 1981, ISBN 0138221227
[17] ISBSG, D&E (Development & Enhancement) repository r11, June 2009, URL: www.isbsg.org
[18] Boehm, B.W., Horowitz, E., Madachy, R., Reifer, D., Bradford K.C., Steece, B., Brown, A.W., Chulani, S., Abts, C., "Software Cost Estimation with COCOMO II", Prentice Hall, New Jersey, 2000.
[19] ISO/IEC, IS 9126-1:2001 - Software Engineering – Product Quality – Part 1: Quality Model, International Organization for Standardization, 2001
[20] ISO/IEC, IS 25010:2011 – Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models, International Organization for Standardization, March 2011
[21] Kassab M. Ormandjieva O., Daneva M., Abran A.., Non-Functional Requirements Size Measurement Method (NFSM) with COSMIC-FFP, Proceedings of IWSM/Mensura 2007, pp. 168-182, URL: http://goo.gl/8heVZ

About the Author

Luigi Buglione is a Measurement & Process Improvement Specialist at Engineering.IT SpA (www.eng.it - formerly Atos Origin Italy and SchlumbergerSema) in Rome, Italy and Associate Professor at the École de Technologie Supérieure (ETS) – Université du Québec, Canada. Previously, he worked as a Software Process Engineer at the European Software Institute (ESI) in Bilbao, Spain. Dr. Buglione is a regular speaker at international Conferences on Software/Service Measurement, Process Improvement and Quality, and is actively part of several International (ISO WG10-25, COSMIC, ISBSG, MAIN) and National (GUFPI-ISMA, AICQ, itSMF Italy, AutomotiveSPIN Italy) technical associations on such issues and contributes in IFPUG as member of MRC and Educational committees. He developed and was part of ESPRIT and of Basque Government projects on metric programs, EFQM models, the Balanced IT Scorecard and QFD for software and is a reviewer of the SWEBOK project. He received a Ph.D in Management Information Systems from LUISS Guido Carli University (Rome, Italy), a degree cum laude in Economics from the University of Rome "La Sapienza", Italy and is one of the few IFPUG CSMS (Certified Software Measurement Specialist) still certified from 2006. Further information on SPIMQ (www.eng-it.it/spimq) and SEMQ websites (www.semq.eu). He can be reached at luigi.buglione@eng.it or luigi.buglione@computer.org.