Early & Quick COSMIC-FFP Analysis using Analytic Hierarchy Process

Luca Santillo, DPO, Italy
luca.santillo@iol.it

Abstract

COSMIC-FFP is a rigorous measurement method that makes possible to measure the functional size of the software, based on identifiable functional user requirements allocated onto different layers, corresponding to different levels of abstraction. The key concepts of COSMIC-FFP are software layers, functional processes and four types of data movement (sub-processes). A precise COSMIC-FFP measure can then be obtained only after the functional specification phase, while for forecasting reasons the Early & Quick COSMIC-FFP technique has been subsequently provided, for using just after the feasibility study phase.

This paper shows how the Analytic Hierarchy Process, a quantification technique of subjective judgements, can be applied to this estimation technique in order to improve significantly its self-consistency and robustness. The AHP technique, based on pair-wise comparisons of all (or some of) the items of the functional hierarchical structure of the software provided by E&Q COSMIC-FFP, provides the determination of a ratio scale of relative values between the items, through a mathematical normalization. Consequently, it is not necessary either to evaluate the numerical value of each item, or to use statistical calibration values, since the true values of only one or few components are propagated in the ratio scale of relative values, providing the consistent values for the rest of the hierarchy.

This merging of E&Q COSMIC-FFP with AHP results in a more precise estimation method which is robust to errors in the pair-wise comparisons, and self-consistent because of the redundancy and the normalization process of the comparisons.

1. COSMIC Full Function Point Overview

The COSMIC-FFP measurement method consists of the application of a set of rules and procedures to a given piece of software in order to measure its functional size. Two distinct and related phases are necessary to perform the measurement: mapping the functional user requirements (FURs) for the software to be measured onto the *COSMIC-FFP software model* and then measuring the specific elements of this software model (Figure 1).

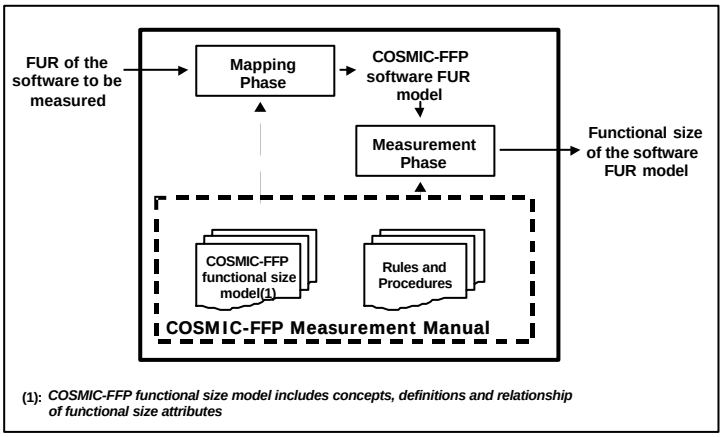


Figure 1. *COSMIC-FFP measurement process model [1].*

The COSMIC-FFP software model captures the concepts, definitions and relationships (functional structure) required for a functional size measurement exercise. Depending on how the

FURs are allocated, the resulting software might be implemented in a number of pieces. While all the pieces exchange data, they will not necessarily operate at the same level of abstraction. The COSMIC-FFP method introduces the concept of the software layer to help differentiate levels of abstraction of the FURs.

The functionality of each layer may be composed of a number of functional processes. A functional process is defined as a "unique and ordered set of data movements (Entry, eXit, Read, Write) implementing a cohesive set of FURs." The COSMIC-FFP software model distinguishes four types of data movement sub-process: in the "front end" direction, two types of movement (Entry and eXit) allow the exchange of data attributes with the users (or other layers); in the "back end" direction, two types of movement (Read and Write) allow the exchange of data attributes with the storage hardware (Figure 2). These data movements are also referred to as BFC's (Base Functional Components).

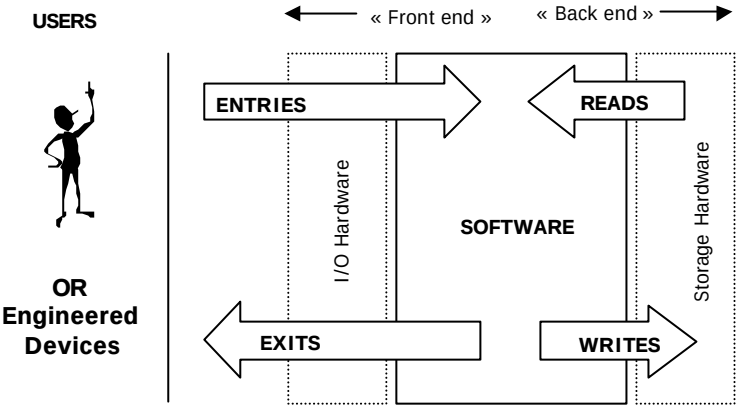


Figure 2. *COSMIC-FFP software model and data movement types [1].*

The COSMIC-FFP measurement rules and procedures are then applied to the software model in order to produce a numerical figure representing the functional size of the software, layer by layer. The unit of measurement is 1 data movement, referred to as 1 COSMIC Functional Size Unit, e.g. 1 C_{FSU}. Theoretically, functional processes can be assigned any size expressed in C_{FSU} (from 1 to no theoretical limit - they are not bounded, but in practice they're expected to have some sort of "natural" upper boundary, or *cut-off*).

Conceptually, the mapping phase of the COSMIC-FFP method can be considered as a process of "viewing" a software from different levels of functional detail. First, the software is viewed at the highest level as composed of software layers, if applicable. Then, each software layer is viewed at a lower level of detail, i.e. functional processes. Finally, each functional process is in turn viewed at the lowest level of detail of interest for measurement with COSMIC-FFP, that is, sub-processes (data movement types, or BFC's).

2. Early & Quick COSMIC-FFP Overview

Functional size of software to be developed can be measured precisely after functional specification stage. However, functional specification is often completed relatively late in the development process and a significant portion of the budget has already been spent. If we need the functional size earlier, we must accept a lower level of precision since it can only be obtained from less precise information.

The Early & Quick COSMIC-FFP method (E&QCFFP, [2]) has been designed to provide practitioners with an *early* and *quick* forecast of the functional size, based on the hierarchical system representation cited in the previous section, which can be used for preliminary technical and managerial decisions at early stages of the development cycle. Of course, a precise standard measure must always be carried out in the later phases to confirm the validity of decisions already

taken. Here, "Early" means that we may obtain this value before having committed a significant amount of resources to a project; "Quick" means that we may also use the technique when the software is an existing asset and some constraints (such as costs and time) prevent a precise measurement.

The starting point for an E&QCFFP estimation is the acknowledgement of the hierarchical structure in the functional requirements for the software to be estimated: when we document a software structure, we usually name the root as the application level and then we go down to defining single nodes, each one with a name that is logically correlated to the functions included; we reach the leaf level when we don't think it is useful to proceed to a further decomposition. In the COSMIC-FFP model, the leaves are the functional processes.

On the one hand, in the early stages it is not possible to distinguish the single data movements, or BFC's, because the information is not available at this level of detail. On the other hand, however, the preliminary hierarchical structure of the software shows as leaves what are actually nodes in the detailed version. What is required early on in the life cycle is, then, to assign forecasts of average process size, in C_{FSU}, at the intermediate and top levels in such a way that the final result will be obtained by the aggregation of the intermediate results.

The E&QCFFP technique is based on the capability of the estimator to "recognize" a software item as belonging to a particular functional class; an appropriate table, then, allows the estimator to assign a C_{FSU} average value for that item (this is applied for each identified layer separately). Each functions can be classified, in order of increasing magnitude, as Functional Process, General Process, or Macro-Process (Figure 3):

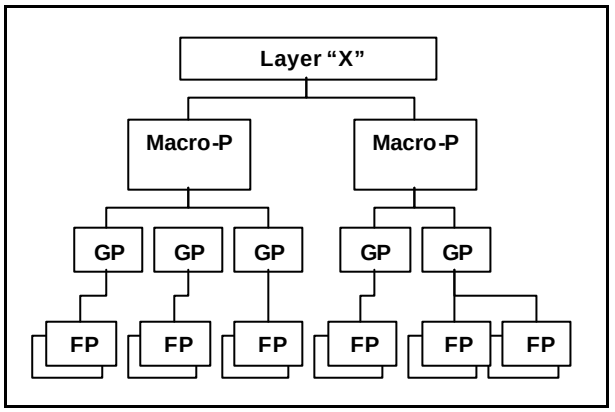


Figure 3. *Hierarchical process decomposition in E&QCFFP.*

a) A Functional Process (FP) is the smallest process, performed with the aid of the software system, with autonomy and significance characteristics. It allows the user to attain a unitary business or logical objective at the operational level. It is not possible, from the user's point of view, to proceed to further useful decomposition of a Functional Process without violating the principles of significance, autonomy and consistency of the system. A Functional Process can be Small, Medium or Large, depending on its estimated number of BFC's (E,X,R,W).

b) A General Process (GP) is a set of medium Functional Processes and may be likened to an operational sub-system of the application. A GP can be Small, Medium or Large, based on its estimated number of Functional Processes.

c) A Macro-Process (MP) is a set of medium General Processes and may be likened to a relevant sub-system of the overall Information System of the user's organisation. A MP can be Small, Medium or Large, based on its estimated number of General Processes.

Note that each level is built up on the basis of the previous one. There is a 4^{th} type of process, the Typical Process (TP), which is off-line from the hierarchical structure outlined: it's just the set of

the four frequently used Functional Processes, which are: Create, Retrieve, Update and Delete (CRUD) information in a relevant data group.

Each E&QCFFP element is associated with three values in terms of C_{FSU} (minimum, most likely and maximum). These numerical assignments are not reported, since they are currently subject to definition and trial on the basis of the data collection activity and statistical analysis for actual projects in the Field Trial Phase of the COSMIC-FFP method. Next Table 1 reports the ranges to help in classify the items of the estimation (the quantities n_1, n_2, n_3 are to be found out empirically during the Field Trial Phase).

Table 1. *Scale ranges and numerical EFP assignments.*

Small Functional Process	n_1 (C_{FSU})
Medium Functional Process	n_2 (C_{FSU})
Large Functional Process	n_3 (C_{FSU})
Small General Process	6-12 FP's
Medium General Process	13-19 FP's
Large General Process	20-25 FP's
Small Macro-Process	2-3 GP's
Medium Macro-Process	4-7 GP's
Large Macro-Process	8-12 GP's

One advantage of this technique is that estimates can be based on different and non homogeneous levels of detail in the knowledge of the software structure. If a part of the software is known at a detail level, this knowledge may be used to estimate it at the Functional Process level, and, if another part is only superficially known, then a higher level of classification may be used. The overall global uncertainty in the estimate will then be the weighted sum of the individual components' uncertainties. This property is better known as multi-level estimation.

Another characteristic of the E&QCFFP technique is that it mixes both an analytical approach (use of the composition table, Table 1) and an analogy-based approach (the analogy can be used with respect to an abstract model or to a concrete set of software objects actually collected and classified, helping to classify the unknown items).

3. The Analytic Hierarchy Process (AHP)

The Analytic Hierarchy Process ([4]) provides a means of making decisions or choices among alternatives, particularly where a number of objectives have to be satisfied (*multiple criteria* or *multi-attribute* decision making) (Figure 4).

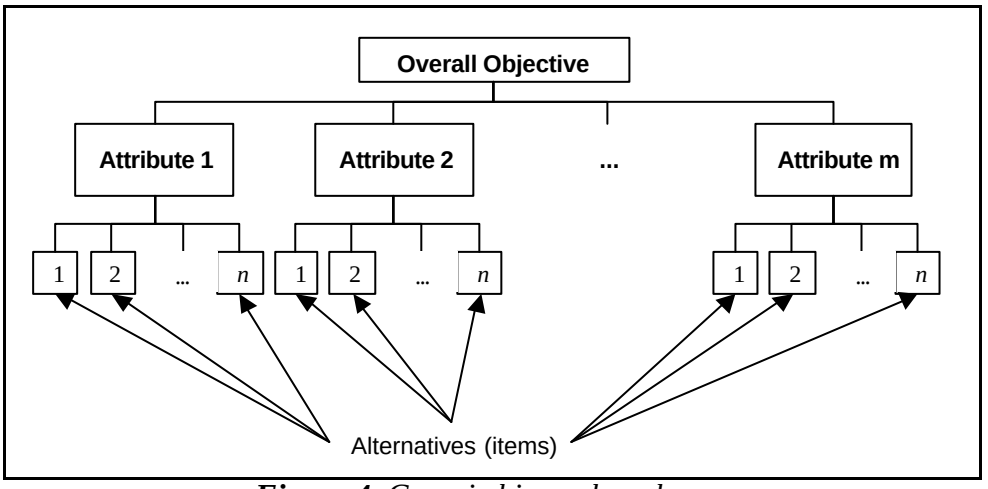


Figure 4. *Generic hierarchy scheme.*

Let's assume that n items are being considered with the goal of providing and quantifying *judgements* on the *relative weight* (*importance)* of each item with respect to all the other items. The first step (*design phase*) set the problem as a hierarchy, where the topmost node is the overall objective of the decision, while subsequent nodes at lower levels consist of the criteria used in arriving at this decision. The bottom level of the hierarchy consists of the *alternatives* from which the choice is to be made, i.e., the n items we wish to compare.

The second step (*evaluation phase*) requires pair-wise comparisons to be made between each two items (of the given level of the hierarchy), with respect to their contribution towards the factor from the level immediately above them. The comparisons are made by posing the question 'Of two elements i and j, which is *more important (larger)* with respect to the given factor and how much more?'. The strength of preference is usually expressed on a ratio scale of 1-9. A preference of 1 indicates *equality* between two items while a preference of 9 (*absolute importance*) indicates that one item is 9 times larger or more important than the one to which is being compared. This scale was originally chosen, because in this way comparisons are being made within a limited range where perception is sensitive enough to make a distinction [4].

These pair-wise comparisons result in a *reciprocal n-by-n matrix A*, where $a_{ii} = 1$ (i.e., on the diagonal) and $a_{ji} = 1/ a_{ij}$ (*reciprocity property*, i.e., assuming that if element i is "x-times" more important than item j, then necessarily item j is "1/x-times" more important, or equally "x-times" less important than item i).

Suppose firstly that we provide only the first column of the matrix A, i.e., the relative importance of items 2, 3, .., n, with respect to item 1. If our judgements were completely *consistent*, the remaining columns of the matrix would then be completely determined, because of the transitivity of the relative importance of the items. However we do not assume consistency other than by setting $a_{ji} = 1/ a_{ij}$. Therefore we repeat the process of comparison for each column of the matrix, making independent judgements over each pair. Suppose that at the end of the comparisons, we have filled the matrix A with the exact relative weights; if we multiply the matrix with the vector of weights $w = (w_1, w_2, \ldots, w_n)$, we obtain:

$$Aw = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \begin{bmatrix} w_1/w_1 & w_1/w_2 & \cdots & w_1/w_n \\ w_2/w_1 & w_2/w_2 & \cdots & w_2/w_n \\ \vdots & \vdots & & \vdots \\ w_n/w_1 & w_n/w_2 & \cdots & w_n/w_n \end{bmatrix} \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = n \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix}$$

So, to recover the (overall) scale from the matrix of ratios, we must solve the problem:

$$Aw = nw, \text{ or } (A-nI)w = 0,$$

that is a system of homogenous linear equations (I is the unitary matrix). This system has a nontrivial solution if and only if the determinant of $(A-nI)$ vanishes, i.e., n is an *eigenvalue* of A. Notice that A has *unit rank* since every row is a constant multiple of the first row and thus all its eigenvalues except one are zero. The sum of the eigenvalues of a matrix is equal to its *trace* and in this case, the trace of A is equal to n. Thus n is an eigenvalue of A and we have a nontrivial solution, unique to within a multiplicative constant, with all positive entries. Usually the normalized vector is taken, obtained by dividing all the entries w_i by their sum.

Thus given the comparison matrix we can recover the scale. In this *exact* case the solution is any column of A normalized. Note also that in the *exact* case A is consistent, i.e., its entries satisfy the condition $a_{jk} = a_{ji}/a_{ki}$ (*transitivity property*). However in real cases we cannot give the precise values of w_i/w_j but estimates of them, the *judgements*, which in general are different from the actual weights' ratios. From matrix theory we know that small perturbation of the coefficients implies

small perturbation of the eigenvalues. Therefore, we still expect to find an eigenvalue, with value near to n: this will be the *largest eigenvalue* (λ_{max}), since due to the (small) errors in the judgement, also other eigenvalues are different from zero, but still the trace of matrix (n) is equal to the sum of eigenvalues (some of which can be complex).

The solution of the largest eigenvalue problem, i.e., the weight eigenvector w corresponding to λ_{max}, when normalized, gives a unique estimate of the underlying ratio scale between the elements of the studied case. Moreover, the matrix whose entries are w_i/w_j is still a consistent matrix, and is a consistent estimate of the "actual" matrix A. A itself need not be consistent (for example the judgements could have stated that item 1 is more important than item 2, 2 is more important than 3, but 3 is more important than 1!). It turns out that A is consistent if and only if $\lambda_{max} = n$ and that we always have $\lambda_{max} \geq n$. That's why we take as a "consistency index" (CI) the (negative) average of the remaining eigenvalues, which is exactly the difference between λ_{max} and n, divided by the normalizing factor (n-1):

$$CI \equiv \frac{-\sum_{i=2}^{n} l_i}{n-1} = \frac{l_{max} - n}{n-1}, \quad l_{max} = l_1$$

To measure the error due to inconsistency, we can compare the CI of the studied case with the average CI obtained from corresponding random matrices with order n and maximum ratio scale r. Table 2 shows the random average consistency indexes $Ci_{n,r}$ for various n and r. Revisions in the pair-wise comparisons are recommended if the consistency ratio (CR) between the studied CI and the corresponding $CI_{n,r}$ is considerably higher than 10%.

Table 2. *Consistency indexes ($Ci_{n,r}$).*

	R								
n	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
5	0,07	0,13	0,20	0,26	0,31	0,37	0,41	0,48	0,51
6	0,07	0,14	0,21	0,27	0,34	0,39	0,46	0,50	0,57
7	0,07	0,15	0,22	0,29	0,35	0,42	0,48	0,53	0,60
8	0,07	0,15	0,23	0,30	0,37	0,43	0,49	0,57	0,62
9	0,08	0,15	0,23	0,31	0,38	0,44	0,50	0,57	0,64
10	0,08	0,16	0,23	0,31	0,38	0,45	0,52	0,59	0,65
11	0,08	0,16	0,24	0,31	0,39	0,46	0,53	0,60	0,66
12	0,08	0,16	0,24	0,32	0,39	0,47	0,54	0,61	0,67

This consistency ratio CR simply reflects the consistency of the pair-wise judgements and shows the degree to which various sets of importance relativities can be reconciled into a single set of weights. In the above example, (1 larger than 2, 2 larger than 3, and 3 larger than 1) the consistency score would be poor, and would be considered a violation of the axiom of transitivity. AHP tolerates inconsistency through the amount of redundancy of judgements. For a matrix of dimension n only (n-1) comparisons are required to establish weights for the n items. The actual number of comparisons that can be performed in AHP is $n(n-1)/2$. This redundancy is conceptually analogous to estimating a number by calculating the average of repeated observations: the resulting set of weights is less sensitive to errors of judgement.

A quick way to find the weight eigenvector, if one cannot solve exactly the largest eigenvalue problem, is that of normalizing each column in A, and then average the values across the rows: this "average column" is the normalized vector of weights (or priorities) w. We then obtain an estimate of λ_{max} dividing each component of Aw (= $\lambda_{max}w$) by the corresponding component of w, and averaging. Finally, we can compute CI (and the corresponding CR) from this estimate of λ_{max} in order to verify the goodness of the judgements.

So far, we have illustrated the process for only one level in the hierarchy: when the model consists of more than one level then hierarchical composition is used to weight the eigenvectors by the weights of the criteria. The sum is taken over all weighted eigenvector entries corresponding to those in the lower level, and so on, resulting in a global priority vector for the lowest level of the hierarchy. The global priorities are essentially the result of distributing, or propagating, the weights of the hierarchy from one level to the next level below it. For the purpose of applying AHP to E&QCFFP estimation, this multi-level weighting is not required, as shown in the following section.

4. Merging E&QCFFP and AHP

The analogy between the hierarchical functional decomposition of E&QCFFP and the intrinsic hierarchy of AHP can be quite confusing; we must recall that the nodes in different levels in a AHP hierarchy carry very different meaning (going from the objective level, to the attribute level, to the alternative level), while in the E&QCFFP approach the decomposition is made only in order to separate different ranges (or groups) of functions. This means that the elements of a E&QCFFP hierarchy are indeed all homogenous with respect to the attribute to be estimated, i.e., the functional size. So there is no strict correspondence between the hierarchical structures in the two techniques, but still a strong tie can be found. Although AHP was developed as a mathematical method for prioritizing the alternatives, we can recognize that what we called *importance* is just an extensive property as many others, as software functional *size* is expected to be, too.

When estimating the software functional size (number of C_{FSU}), the only criteria is the size itself. Consequently, we can consider a simple AHP hierarchy, with only one level (and the objective "estimated size" above it); the nodes of this level are the *n* items listed by the estimator, eventually prior to the functional decomposition (this list could even include all from possible functional processes to macro-processes).

In order to review the possible ways to merge E&QCFFP and AHP, let's recall the intrinsic characteristics of both: AHP makes the subjective comparisons consistent through a mathematical step (the largest eigenvalue solution) and provides the *CR* to evaluate the self-consistency of the estimation, while the E&QCFFP alone provides a estimation together with an uncertainty range (minimum, most likely, and maximum values), permitting to assign a class to each item based on analogy (eventually with respect to known cases); note that the uncertainty range in the E&QCFFP can be quite large when using mostly the macro-process level.

We could gain better forecasts by combining the two techniques; the possible ways to do the join are basically the following:

a) AHP technique first applied to prioritize the items on a numerical scale, then automatic assignation of the E&QCFFP class from the scale.

b) E&QCFFP technique first applied to allocate the set of items in functional classes, then AHP applied to refine the first estimation.

The a) case can be considered as a "re-allocation" of a pure AHP estimation on the E&QCFFP classes; here some not-yet-solved problems may rise, as for example how to decide which AHP resulting numerical range should be assigned to a given E&QCFFP class. If we manage to solve this and similar problems, we can obtain a hierarchical representation of the estimated system as in a pure E&QCFFP technique, but with more robustness in the input (nonetheless, this could not result always in a more robust output forecast, due to the fact that E&QCFFP categories necessarily "blur" the exact ratios given by AHP).

The b) case is to be considered more significantly, since it requires firstly an analogical approach, which is usually easier at the beginning for the human estimator, and after that a robust refinement of the estimation in a mathematical way.

Depending on the desired precision or the time at our disposal in doing the estimation, we should decide on which variant to apply to estimate the COSMIC-FFP number: only by E&QCFFP, only

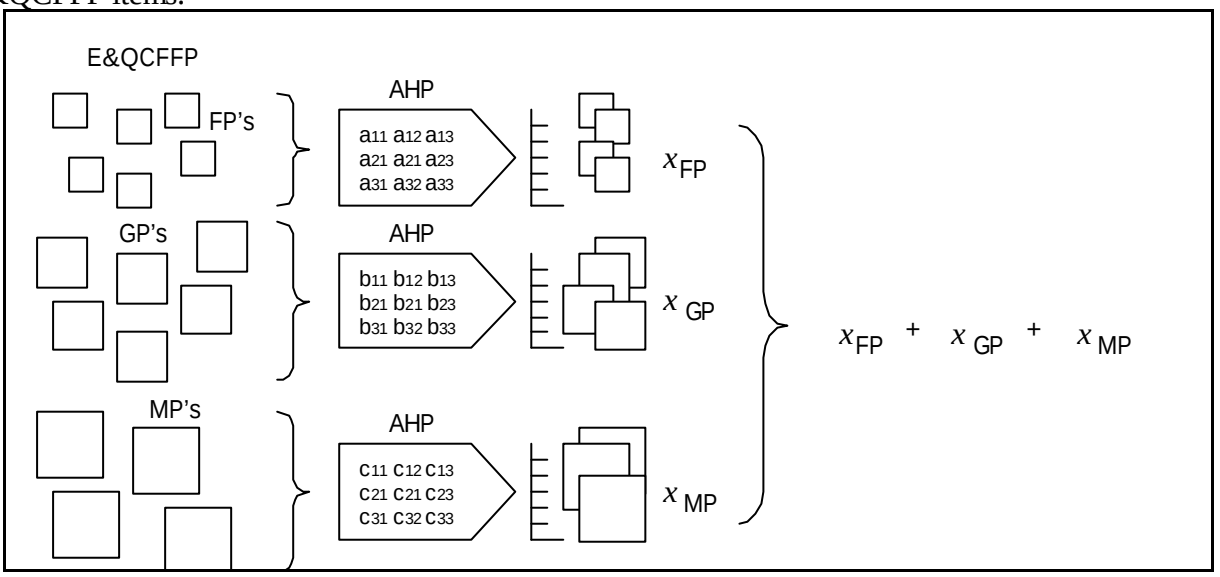
by AHP, with the a) case or with the b) case. The last approach should result in the most accurate forecast, still saving us from applying an exact (and more time-consuming) COSMIC-FFP counting procedure. Next section deals more deeply with the b) case.

5. The "b) case": E&QCFFP + AHP

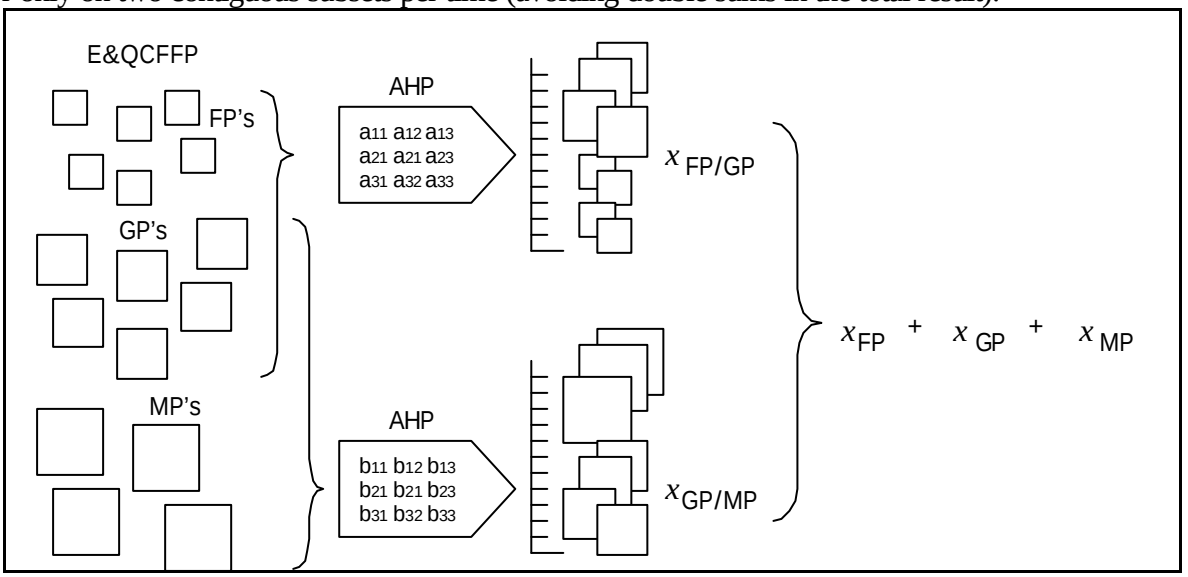
The case is:

1. E&QCFFP to allocate the items in subsets;
2. AHP to revise/refine the estimation.

Note that the first step already provides a first estimation, but its uncertainty could be quite wide, if the estimator dealt with one or more high-level class (e.g. general processes or macro-processes). The second step could be an AHP application on the global set of items from the first step, but since the pairwise ratios involved in such a global application would be of magnitude 10^2 and higher, it would be obviously very hard for a human estimator to provide such estimated ratios in the comparisons. An enhancement is to apply AHP separately on homogeneous subsets of the E&QCFFP items:



or only on two contiguous subsets per time (avoiding double sums in the total result):



The second variant, mixing and comparing Functional Processes with General Processes, and General Processes with Macro-Processes, would be the more self-consistent and coherent one.

In any case, this approach would maintain the hierarchical representation of the system as firstly posed by the E&QCFFP estimator, but with a more consistent and robust numerical evaluation of each item compared to the others; the final estimated value is a revision of the first, pure E&QCFFP forecast, but with a lower uncertainty range (the original uncertainty range should be reduced, based on the value of the resulting *CR*). Eventually, some items could be re-allocated in terms of their E&QCFFP category, if the AHP step shows some significant change with respect to their original E&QCFFP allocation.

We should not be too scared of the quantity of different comparisons to perform in every AHP step, since we know from section 3 that not all the comparisons have to be effectively performed, unless the *CR* is not low enough. So, monitoring the value of the *CR* after several incremental iterations of the AHP step, we could decide to stop them when the *CR* satisfies a predefined accuracy level.

When deriving the final estimation result, two approaches are possible: one or more items should be fixed in their C_{FSU} value, as "landmarks", to propagate the number of assigned C_{FSU} to the whole set, or the whole set can be mapped in a "fuzzy" way onto an ordered scale of items, as the E&QCFFP classes, with assigned quantities of C_{FSU}. Future field trials should show which approach is preferable.

The landmarks could be put among the original unknown items to help in both the E&QCFFP and the subsequent AHP step. These landmarks could be taken from a so-called Experience Data Base (or Catalogue of Typical Elements). This catalogue could contain for example all the typical processes or functions that can be identified in a generic project, and their average quantities of C_{FSU}. Once some of these typical elements are identified among the list of items, the comparison matrix (or matrices) would greatly benefit of the relative ratios between them. A generic useful case of "typical element" would be the already cited Typical Process, or CRUD, which is usually very easy to identify and to use as a comparison landmark. In case of more than one landmark, further research is necessary to establish the exact mathematical procedure to fix their values, while propagating the quantities of CFSU through the unknown items.

A special case of application would be when the E&QCFFP step provides a list of items, all classified at the Functional Process level. In this case, the whole set would be taken into account for a unique AHP step, in order to compare directly the quantities of data movements contained in each process; this means that it could be significant to compare directly estimated quantities of CFSU (but still without exactly counting them).

6. Numerical examples

Several AHP cases have been studied, as depicted in the following tables. In every case, we have $n = 10$ items, and we assume that the comparisons made between the 1st item and each of the remaining items (i.e. the first column/row of the matrix A) are the "best" estimates; eventual inconsistency is put in the remaining comparisons (i.e. between 2nd, 3rd, …, and 10th item). What differentiates each case is the expected ratio between each of the 10 items. Since the field trials are still to provide actual numbers of C_{FSU} for E&QCFFP, for sake of clarity in the examples we consider the first item always with *unitary size*.

For each case, different inconsistency errors were introduced separately on each pairwise comparison (except for comparisons between the 1st item and the others, assumed as correct) to simulate the human pairwise comparisons: uniformly random ±10%, ±25%, ±50%, ±75%, ±90%, and ±100% errors. For example, the 100% error means that, while the estimator should evaluate "item i is p-times item j" the simulation could put "item i is $2p$-times item j" (doubling the expected ratio, i.e. with a 100% error). For each case and each error range, 1000-samples statistics have been

generated; all values are approximated at one decimal. The first column of each table denotes the maximum error for single pair comparison.

Case A: (1,1,1, 1,1,1, 1,1,1,1), *Total = 10, $CI_{(n=10,\ max\ ratio=10)}$=0.65.*

Error	λ_{max}	CR	Estimates (average)	Total	D %
10%	10.0	0.2%	(1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0)	10.3	3%
25%	10.1	1.6%	(1.1,1.1, 1.1,1.1, 1.1,1.1, 1.1,1.1, 1.1,1.1)	10.7	7%
50%	10.3	4.8%	(1.1, 1.2, 1.2, 1.2, 1.2, 1.2,1.1,1.1,1.1,1.1)	11.6	16%
75%	10.8	**14.2%**	(1.2,1.5,1.4,1.4,1.4,1.3,1.3,1.2,1.2,1.2)	13.1	**31%**
90%	11.7	**28.7%**	(1.3,1.8,1.7,1.6,1.6,1.5,1.4,1.3,1.3,1.2)	14.8	**48%**
100%	15.3	**90.3%**	(1.5,3.4,3.6,3.1,2.8,2.3,2.2,1.8,1.6,1.4)	23.6	**136%**

Case B: (1,1,1, 1,1,1, 1,1,1,10), *Total = 19, $CI_{(n=10,\ max\ ratio=10)}$=0.65.*

Error	λ_{max}	CR	Estimates (average)	Total	D %
10%	10.0	0.2%	(1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0,10.2)	19.5	3%
25%	10.1	1.2%	(1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1, 1.1,10.6)	20.2	6%
50%	10.3	4.8%	(1.1, 1.2, 1.2, 1.2, 1.2,1.1,1.1,1.1,1.1,11.1)	21.4	13%
75%	10.8	**14.2%**	(1.2,1.4,1.4,1.4,1.3,1.3,1.2,1.2,1.2,11.5)	23.0	**21%**
90%	11.7	**29.1%**	(1.2,1.8,1.7,1.6,1.5,1.4,1.4,1.3,1.2,11.8)	25.0	**32%**
100%	15.3	**90.1%**	(1.4,4.0,4.2,3.1,2.7,2.2,1.8,1.8,1.4,13.2)	35.8	**88%**

Case C: (1,2,3,4,5,6,7,8,9,10), *Total = 55, $CI_{(n=10,\ max\ ratio=10)}$=0.65.*

Error	λ_{max}	CR	Estimates (average)	Total	D %
10%	10.0	0.2%	(1.0,2.0,3.0,4.0,5.0,6.0,7.0,8.0,9.0,10.0)	55.0	0.0%
25%	10.1	1.2%	(1.0,2.0,3.0,4.0,5.1,6.0,7.0,8.0,9.0,10.0)	55.2	0.4%
50%	10.3	4.8%	(1.0,2.1,3.2,4.2,5.2,6.2,7.1,8.1,9.0,9.9)	56.1	2.0%
75%	10.8	**14.2%**	(1.0,2.4,3.5,4.6,5.6,6.5,7.4,8.3,9.1,9.9)	58.3	**6%**
90%	11.7	**29.6%**	(1.0,2.9,4.1,5.2,6.2,7.1,7.9,8.6,9.1,9.6)	61.8	**12%**
100%	15.3	**95.4%**	(1.0,4.6,6.4,8.2,8.5,10.1,9.8,10.0,10.0,9.4)	78.0	**42%**

Case D: (1,1,1, 1,1,10, 10,10,10,10), *Total = 55, $CI_{(n=10,\ max\ ratio=10)}$=0.65.*

Error	λ_{max}	CR	Estimates (average)	Total	D %
10%	10.0	0.2%	(1.0,1.0,1.0,1.0,1.0,10.2,10.2,10.2,10.2,10.2)	55.9	1.6%
25%	10.1	1.2%	(1.0,1.1,1.1,1.0,1.0,10.5,10.4,10.4,10.4,10.3)	57.3	4.2%
50%	10.3	4.8%	(1.1,1.1,1.1,1.1,1.1,10.9,10.9,10.8,10.6,10.6)	59.3	8%
75%	10.8	**14.2%**	(1.1,1.3,1.3,1.2,1.2,11.7,11.3,11.1,10.8,10.5)	61.4	**12%**
90%	11.7	**29.3%**	(1.1,1.5,1.5,1.4,1.3,12.9,12.2,11.7,10.9,10.5)	65.0	**18%**
100%	15.3	**90.1%**	(1.1,2.8,2.5,2.0,1.9,16.5,15.6,14.0,12.3,10.6)	79.5	**45%**

Case E: (1,5,10, 15,20,25, 30,35,40,45), *Total = 226, $CI_{(n=10,\ max\ ratio=50)}$=2.36.*

Error	λ_{max}	CR	Estimates (average)	Total	D %
10%	10.0	0.1%	(1.0,5.0,10.0,15.0,20.0,25.0,30.0,35.1,40.0,44.9)	226.0	0.0%
25%	10.1	0.3%	(1.0,5.1,10.1,15.2,20.2,25.2,30.1,35.2,40.0,44.9)	227.0	0.4%
50%	10.3	1.3%	(1.0,5.4,10.6,15.8,20.7,25.7,30.6,35.6,40.1,44.5)	230.0	1.8%
75%	10.8	3.9%	(1.0,6.1,11.8,17.2,22.4,27.2,32.2,35.9,40.0,44.4)	238.2	5%
90%	11.7	8.0%	(1.0,7.1,13.7,19.5,24.6,29.3,33.9,37.6,40.9,44.0)	251.6	11%
100%	15.4	**25.6%**	(1.0,12.3,21.6,28.7,32.3,41.4,41.2,43.5,42.5,42.6)	307.1	**36%**

Note that for uniformly random errors from 10% to 50% we always get acceptable *CR* values, and the final per cent deviation between expected and estimated values ($\Delta_{\%}$) is always no more than 3-times the *CR* value.

As we stated in section 3, the largest eigenvalue λ_{max} is always > *n*, and increases as the average error in the comparisons increases. Moreover, almost everywhere each item is overestimated with respect to its expected value; exception are cases *C* and *E* (those with the most widespread values), where the 10^{th} item is underestimated and counterbalances the overestimation of the remaining 9 items. However, every estimation is globally *over* the total expected value: this should be taken as a general property, i.e. the AHP estimation is to be taken as an *upper threshold*.

Relevant cases are:

Case A. In this case (all items expected as identical), the more error we put in the simulation, the most error we get in the estimated total. This could be explained as follows: if the set is strongly homogeneous (all items identical) we should not be too "easy" in estimating wrong ratios between the items.

Case E. This case involves a wide range of items, putting together the first item (unitary size) with a 45-times larger item (the 10^{th}). In this case even a strong (random) error up to 90% on some comparisons is "blurred" by AHP to give a 11% deviation for the total estimation.

7. Further discussion and conclusion

The examples above are very encouraging, but much investigation has still to be made. For example, very large cases (very high *n*) introduce difficulties in managing the items. From this perspective, is noticeable that the original AHP deals only with small *n*; a suggestion is to try to use homogenous clusters of items, and to make comparisons between these clusters. Of course, further research in realistic, field trials is strongly encouraged to test the proposed approach in different situations.

As cited above, the fact that only a single value is to be provided, besides of the relative weight estimates, does not mean that more than one true value cannot be used: e.g., if we know the values of items 1, 2 and 3, this means that we have more confidence in fixing several weights in the comparison matrix; *de facto*, in this way we do use the richer information. A further research theme should be on how to make some landmarks to "weight" more than others, if their value is far more accurate.

AHP is a powerful means for several tasks in the estimation and decision making field. The proposed combination with the E&QCFFP technique can solve those situation in which the only E&QCFFP does not provide good results, especially due to atypical or new situations, not collected in the historical statistics, or when it is used identifying few, high level items, providing too wide ranges of uncertainty.

8. References

[1] Abran, Desharnais, Oligny, St-Pierre, Symons, "COSMIC-FFP Measurement Manual, version 2.0", Ed. S. Oligny, Software Engineering Management Research Lab., Université du Québec à Montréal (Canada), October, 1999.

[2] Meli, R., Abran A., Ho V.T., and Oligny S., "On the applicability of COSMIC-FFP for measuring software throughout its life cycle", ESCOM-SCOPE 2000, Munich, April 18-20, 2000.

[3] Meli, R. and Santillo, L., "Function Point Estimation Methods - A Comparative Overview", FESMA 99, Amsterdam, October 6-8, 1999.

[4] Saaty, T.L. and Alexander, J.M., "Thinking with Models", Pergamon Press, 1981.

[5] Santillo, L., "Early FP Estimation and the Analytic Hierarchy Process", ESCOM-SCOPE 2000, Munich, April 18-20, 2000.