ISSN No:-2456-2165

Speculative Approach of Analyzing Complexity of a Software Using Different Programming Languages

Aniket Paul
School of Computing Science and Engineering,
Sandip University, Nashik
Maharashtra, India

Srishti Vashisht
Department of Computing Science and Engineering,
Sandip University, Nashik,
Maharashtra, India

Abstract :- No single language can fit all the purposes. In a rigidly static type system, a compiler is capable of great number of memory management decisions, turning types into a fixed memory layout, optimized for the target processor. The price the user has to pay is in the power of expression, as dynamic behavior must be explicitly written into the program if you want it. The price user pays are speed impact, and a bigger run-time system which may be unsuitable for resource-constrained environments. This creates confusion for choosing the essential language for a project without creating a bottleneck. Therefore, there is a necessity of building a model which creates a base for standard set of quality attributes that avoids limitations of existing models. Estimation of programming quality characteristics using AHP is the objective of this paper.

Keywords :- *Memory Management, Speed Impact, Bigger run-time, Constrained Environments, Bottleneck, Analytic Hierarchy Process.*

I. INTRODUCTION

Which is the best programming language? Questions about programming language and the properties of their programs are asked often but well-founded answers are not easily available. From an engineering viewpoint, the design of a programming language is the result of multiple trade-offs that achieve certain desirable properties (such as speed, reliability etc) at the expense of others (such as Learnability and pedagogical value). If we want reliably answer questions about properties of programming languages, we have to analyse, empirically, the artefacts programmers write in those languages. Answers grounded in empirical evidence can be valuable in helping language users and designers make informed choices.

To control factors that may affect the properties of the outcome, we have performed a controlled survey in which the Professors where shown programs written in two different languages and where made to select different properties and choose the best between them accordingly. Such controlled surveys provide the most reliable data about the impact of certain programming language features such as reliability, speed, accuracy etc., but they are also necessarily limited in scope and generalised by the type and number of tasks involved.

The study presented in this paper explores ground where highly controlled but small programming assignments are taken in consideration. Our study analyses result of 10 highly qualified Professors who analyse programming in C++ and Python.

II. METHODOLOGY

A. *The basic principle of Analytic Hierarchy Process*

The Analytic Hierarchy Process (AHP) is a multi-objective decision analysis method for quantitative and qualitative. Central to this approach is that policy makers will experience judgment given quantization to provide a quantitative basis for decision makers in the form of more practical goals in the complex structure and the lack of necessary data. Its basic principle is to a variety of factors related to the evaluation of alternatives to the system is divided into several levels, and in various elements of the same level on the layer elements according to the criteria, pair wise comparison judgment and calculate the weight of each element of weight, according to comprehensive weight by a maximum weight principle to determine the optimal solution

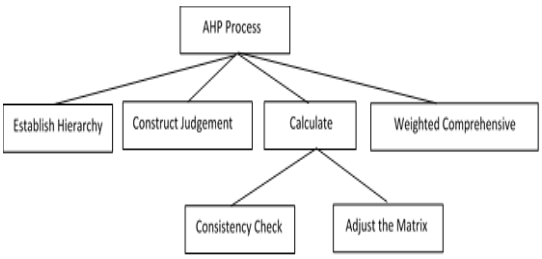

Fig 1:- The steps of Analytical Hierarchy Process

In this model, the part of the complex problem is broken into elements. These elements and forming several levels according to their attributes and relations. A hierarchy of elements as a criterion to the next level related elements reigns. The top is the target layer, it said institute to achieve goals, such as mobile Internet product availability level, or ultimately the decision to be made. Summarize its basic steps roughly divided into four steps, as shown in Figure 1.

B. *Case Study*

In this paper we have done a survey of 10 Professors. It is a classroom-based survey. In this survey we have taken two simple tic-tac-toe games as the components for the selection problem. These 2 games are implemented on Python and C++. The Professors have to go through the code and judge these games on the basis of the characteristics given below:

- Pedagogical Value
- Reliability
- Portability

ISSN No:-2456-2165

- Efficiency
- Learnability

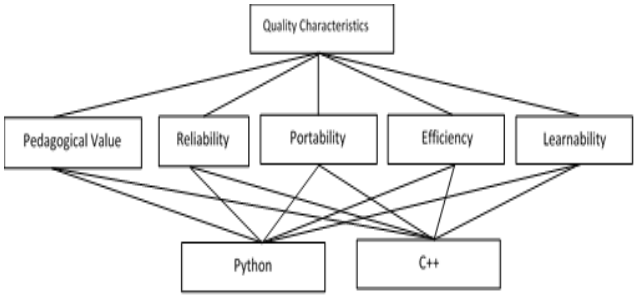

Fig 2 :- Proposed assessment model with static metrics

Now we discuss the components individually along with their judgement matrix and derive local priorities (preferences):

- *Pedagogical Value*

This component deals with the capability and scope of the language to support and enforce the concepts a Professor wants to teach. A judgement matrix (C_1) is defined on the pair wise comparison process, the matrix is based on the available information as:

	C++	PYTHON
C++	1	3
PYTHON	0.33	1

Table 1 :- Decision table for Pedagogical Value (C_1).

- *Reliability*

Reliability of a component refers that the game is reliable enough to sustain in any condition and should give consistently correct results. Product reliability is measured in terms of working of project under different working environment and conditions. A judgement matrix (C_2) is defined on the pair wise comparison process, the matrix is based on the information as:

	C++	PYTHON
C++	1	0.2
PYTHON	5	1

Table 2 :- Decision table for Reliability (C_2).

- *Portability*

Portability criteria concern the ability of program to be transferred from one environment to another. It is used to address that can user still use the software product when environment has been changed. A judgment matrix (C_3) is defined on the pair wise comparison process, the matrix is based on the available information as:

	C++	PYTHON
C++	1	0.25
PYTHON	4	1

Table 3:- Decision table for Portability (C3).

- Efficiency

The efficiency criteria concern the characteristics of a project that gives best results with the use of minimum resources. Factors such as Time Behaviour, Resource Behaviour usually need to be considered. A judgement matrix (C_4) is defined on the pair wise comparison process, the matrix is based on the available information as:

	C++	PYTHON
C++	1	0.2
PYTHON	5	1

Table 4 :- Decision table for Efficiency (C_4).

- *Learnability*

Learnability criteria is inclined on whether the language is easy to learn. Factors such as time plays a huge role as training is very expensive. A judgement matrix (C_5) is defined on the pair wise comparison process, the matrix is based on the available information as:

	C++	PYTHON
C++	1	0.2
PYTHON	5	1

Table 5 :- Decision table for Learnability (C_5).

C. *Testability Study*

In order to conduct testability study based on above AHP technique. The hierarchical model with factors-Pedagogical(F1), Reliability(F2), Portability (F3), Efficiency (F4) and Learnability (F5). A common scale is created and then individual matrix is sent out to 10 Professors to fill as discussed above.

	F1	F2	F3	F4	F5
F1	1	0.28	0.20	0.14	0.90
F2	3.57	1	2.00	1.10	6.00
F3	5.00	0.50	1	0.20	6.00
F4	7.14	1.91	5.00	1	9.00
F5	1.11	0.17	0.17	0.11	1

Table 6 :- Preferred over table

D. *Analysing Collected Data*

Now going Back to Table 1. We have used spreadsheet based approximate calculations for local priorities giving us Eigen Vector $\lambda_{max} = 5.29$ which is >= 5 (total no. of factors), which is consistent. Using this we calculate the CI and CR values as follows:

$$CI = (\lambda_{max} - 1) / (n - 1) \quad (1)$$
$$CI = (5.2902 - 1) / (5 - 1) \quad (2)$$
$$CI = 0.0725 \quad (3)$$
$$CR = CI / RI \quad (4)$$
$$CR = 0.0725 / 1.12 \quad (5)$$
$$CR = 0.0648$$

We found the calculated value of CR < 0.1 in all the samples of matrices, which indicates that the estimate is consistent and acceptable.

Now we will generate a Normalized weighted Table from the values of Table 1 to calculate the weight of the characteristics.

ISSN No:-2456-2165

	F1	F2	F3	F4	F5	Weight
F1	1	0.10	0.02	0.05	0.04	0.051
F2	0.20	0.35	0.24	0.43	0.26	0.295
F3	0.28	0.18	0.12	0.08	0.26	0.173
F4	0.40	0.32	0.60	0.39	0.39	0.438
F5	0.06	0.06	0.02	0.04	0.04	0.043

Table 7 :- Normalized Weighted Score.

E. *Deriving overall priorities (Model Synthesis)*

Up to this point we have obtained local priorities which indicate the preferred alternative with respect to each criterion. In this fourth step, we need to calculate the overall priority (also called final priority) for each characteristic (criteria); that is, priorities that consider not only our preference of alternatives for each criterion but also the fact that each criterion has a different weight. Given that we are using all the values provided in the model, this step is called model synthesis.

We start the calculation of the overall priority using the local priority of each alternative as the starting point (Table 8). Next, we need to take into consideration the weights of each criteria and for this purpose they are inserted in the table as shown in Table 9. For example, the Pedological Value (F1) criterion has a priority (or weight) of 0.051 and C++ has a local priority (or preference) of 0.75 relative to Pedological Value (F1); therefore, the weighted priority, with respect to Pedagogical Value, of C++ is: 0.051 * 0.75 = 0.3794. A similar calculation is necessary to obtain the C++ weighted priorities with respect to F2, F3, F4 and F5.

	F1	F2	F3	F4	F5
C++	0.75	0.166	0.2	0.166	0.310
Python	0.25	0.83	0.8	0.83	0.689

Table 8 :- Local Priorities table.

	F1	F2	F3	F4	F5
Criteria Weight	0.051	0.295	0.173	0.438	0.043
C++	0.75	0.166	0.2	0.166	0.310
Python	0.25	0.83	0.8	0.83	0.689

Table 9 :- Preparation for weighing of priorities.

Finally, the overall priority of C++ is obtained by adding these results along the row. This procedure is repeated for each of the alternatives being evaluated. The overall priorities of the alternatives are shown in the rightmost column of Table 10.

	F1	F2	F3	F4	F5	Overall Priority
Criteria Weight	0.051	0.295	0.173	0.438	0.043	
C++	0.75	0.166	0.2	0.166	0.310	0.2080
Python	0.25	0.83	0.8	0.83	0.689	0.8434

Table10 :- Overall Priority table.

The calculations for each alternative are shown below and the results are presented in Table 11 following the convention of showing the local priorities (cells) and the weights for each criterion (at the top of each column). This process is called the model synthesis.

Overall Priority of C++
= 0.51*0.75 + 0.295*0.166 + 0.173*0.2 + 0.438*0.166 + 0.043*0.310 (1)
= 0.2080 (2)

Overall Priority of Python
= 0.51*0.75 + 0.295*0.83 + 0.173*0.8 + 0.438*0.83 + 0.043*0.689 (1)
= 0.8434 (2) (2)

Now we can list the alternatives ordered by their overall priority or preference as follows:

Language	Overall Priority	Rank
C++	0.208086237	2
Python	0.4341624	1

Table11 :- Synthesis of the Model.

In other words, given the importance (or weight) of each characteristic (Pedological Value, Reliability, Portability, Efficiency and Learnability), Python is preferable (overall priority = 0.8434) compared to C++ (overall priority = 0.2080).

III. CONCLUSIONS AND FUTURE WORK

We conclude that to capture the decision-making process by AHP have been organized which is used to provide reliable and efficient decision. For handling MA decision problems in actual situations, AHP method is widely used. Despite its ease in concept and efficiency in calculation, it suffers from a few drawbacks. For DM, to better recognize the problem and their decision activities, this approach provides flexibility and toughness.

In future work, our method can be implemented using ANP and fuzzy ANP method, which is very efficient. When the relation of higher-level elements with lower level elements and their dependency should be considered, as many decision problems cannot be structured hierarchically. So, ANP provides a solution for such types of problem. Therefore, using a network, many good problems can be modelled.

REFERENCES

[1]. IEEE Standard Glossary of Software Engineering Terminology, IEEE Std 610.12-1990.
[2]. Chapter One Software Quality Models and Philosophies. A. Sharma, R. Kumar, P. S. Grover "Estimation of Quality for Software Components– an Empirical Approach" November 2008 Volume 33 Number.
[3]. Khashayar Khosravi, "A Quality Model for Design Patterns".
[4]. R. Al-Qutaish "Quality Models in Software Engineering Literature: An Analytical and Comparative Study" [Journal of American Science 2010; 6(3):166-175]. (ISSN: 1545-1003).

[5]. Dromey, R. G., "A model for software product quality". IEEE Transactions on Software Engineering 21, 1995, 146-162.
[6]. P. M. Shanthi, K. Duraiswamy, An Empirical Validation of Software Quality Metric Suites on Open Source Software for Fault-Proneness Prediction in Object Oriented Systems. European Journal of Scientific Research ISSN 1450-216x Vol.51 No.2 (2011), Pp.168-181 © Euro journals Publishing, Inc. 201.
[7]. Sanjay Kumar Dubey, Prof. (Dr.) Ajay Rana. A Comprehensive Assessment of Object-Oriented Software Systems Using Metrics Approach. (IJCSE) International Journal on Computer Science and Engineering Vol. 02, No. 08, 2010, 2726-2730.
[8]. Amjan Shaik, C. R. K. Reddy, Bala Manda, Prakashini. C, Deepthi. K. An Empirical Validation of Object Oriented Design Metrics in Object Oriented Systems, Journal of Emerging Trends in Engineering and Applied Sciences (JETEAS) 1 (2): 216-224(2010).
[9]. Ardhendu Mandal, S. C. Pal. Emergence of Component Based Software Engineering, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 3, March 2012.
[10]. M. Bertoa, A.Vallecillo, "Quality Attributes for COTS Components", In the Proceedings of the 6th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE), Spain, 2002.
[11]. Dandan HE, Can WANG "Usability Evaluation of Software Testing Based on Analytic Hierarchy Process", 4th International Conference on Machinery, Materials and Computing Technology (ICMMCT 2016).