AGE

3

8

16

35

WITH THE AGE OF OBJECT ORIENTED PROGRAMMING (OOP), HAVE WE PASSED McCARTHY'S THEORY?

Dr. W.A.K. AL- Hamdani
Dept. of Computer Science
Univ. of Technology
Baghdad . Iraq

ABSTRACT

There are some recent arguments in the community of computer enlightenment respecting the Oject Oriented Programing (OOP). This discussions foucs on one of the major questions that is :

" are we starting the duration of new theory of computer science ? How ? "

In this short paper we try to answer the most significant question of *" With the use of OOP, are we passed Mccarthy's theory ? Are we starting new theory of computer science ? "* we will first look at McCarthy thoery, the principles of OOP , then finally we will attempt to answer the question and show our claims.

1. INTRODUCTION

The classical formalism for the theory of computer science was entirely based on Turing Machine (TM), or its equivalent the General Recursive Function (GRF) [Br77,Co86, AU72a, AU72b, Kr83, Ma74, Rs83].

This was stated in [Br77] as

> " *Any formalism for the theory of computer science must be capable of representing all PROGRAMS and ALGORITHMS* "

That is to announce, by TURING thesis, " *any appropriate formalism must have the expression power of TM or GRF* " . Although it is pointed out that FSMs are suitable for a particular category of problems that ideally involve a smale number of states .

However the challenge of this decade is to bring computer personality as simple advantageous and beneficial to people without certain training in programming. Visual Programming (so called) represents revolutionary and refractory approach to meet this challenge [Sh89, HM86, Le86, Go83,Ko89, Te89, WH89].

Visual programming (VP) has gained momentum in recent years principally because the collapse cost of graphics, related hardware and software has made it achievable to use pictures as a means of communication with computers and to use computer graphics as a medium to teach programming. Nevertheless, " *even though work on visual-oriented computing mushrooming, there is no agreement on what visual programming is* " [Sh88] !

There are some argument in the computer science community respecting the Object Oriented Programming (OOP) -so called- Visual programming and graphics techniques. Some clear quesitions ariesed here , such as :

> " *Have we start new computer science theory ? How? Can we formalize a computer science theory in parallel to Turning theases based theory? On which bases ? on which primatives ?* "

In this short paper we will try to constrain on the most considerable qustion of

> " *In this period of the advance techniques , have we passed McCarthy's work ?* "

2. McCARTHY's CONTRIBUTION.

McCarthy indicate formalism based on similar idea to that of GRF. His contributions were to many sections of computer science, and suggests rich inter-relationships between them. His major contributions where the formulation of a list of problems with which the emergent theory should concern itself, and the development of a set of methods and formalisms in his initial analysis of attack those problems.

Conceivably we can classify McCarthy's work into three categories:

a: The development of a mathematical formalism to the theory of computer Science [Mc59a, Mc59b, Mc60].
b: A preliminary theory of a class of (non-numerical) symbolic expressions called S-expression [Mc63] .
c: Application of the formulation : The equivalence of programs, semantic of programming languages and proof checking [Mc62a, Mc62b, Mc63, Mc66] .

McCarthy's major suggestion formalism, is based on an abstraction similar to that underlying GRF :

> *" There is a set F of primitive functions, and a set C of strategies for building new functions out of old ones , such that C(F) comprises all computable functions "*

McCarthy suggested a prefixed C of strategies, leaving the user to define a base set F appropriate to his data structures. F might be primitive function of GRF.

This work is classified as a list construction which could be summarized as:

> *If (l) is a list of sequence of integers $< X_1, X_2 ..X_n >$ & if (l) is a list of no integer '< >'called NIL. There were three functions of l; CONSTRUCT , (H)EAD and (T)AIL*

The whole idea is to handle not just lists of integers, but lists of lists, list of list of lists, etc. This could be handle through the use of the functions H(X) and T(X) because they come out to be more symmetrical. The notationof " DOT " were introduced with the following primitives: ATOM , EQUAL and SYMBOLIC EXPRESSION (S-expression) which generate SYMBOLIC FUNCTION

3. THEOREM .

> *OOP dos not have Object Grammar (Graphics Grammer) but string grammer*

To prove this claim , we will work with example from PASCAL OOP to study it's grammer

```
< object type >        ::= OBJECT [ <heritage>  | <field list> | ";"
                           <method list > ] END
<method list>          ::= <method heading> | <method heading> ";"
                           VIRTUAL ";" |  <method heading>
<method heading>       ::= <procedure heading>     | <function heading> |
                           <construction heading>  | <destructuor heading>
<procedure heading>::= PROCEDURE <identifier>
```

Identifier is defined as in PASCAL .

To generalize the grammar of this example suppose the assumption of the terminals to be: <object type> = S, <method list> = M, <method heading> = H ; and the non terminals <procedure heading> = P; <identifier> = I, <letter> = t , <digit> = d, <procedure> = r <object> = b <end> = e .

A programming statement of this grammer could be:

```
base = object
    procedure dump; virtual ;
    end
```

This could be driven as:

S —> bMe | be
M —> H
H —> P | F | C | D
P —> rI
I —> t | td

(i) according to Chomisky definition [AU72a, AU72b, Ba78,Ch56, Co86, GU77, Rs83] We have successfully interpret an Object Oriented Code (OOC) into production rules (P) i.e. OOC —> P

(ii) According to Kleene's theory [Kl36, Kl56], it would be very clear to map P to TG i.e. P —> TG and TG —> FA [Ba83,Br77, Co86].

(iii) Therefore, our productions P could be represented as FA

3. CLAIMS.

As we show that an OOP has a string representation (theorem 1) , this logically means that the OOP is within the limitation of the classical theory of computabon. This could be interpreted as :

> *" OOC is significantly part of McCarthy contributionand it is behind it , and OOC is within the limit of classical theory of computation FSM and GRF "*

Inconsequent we could claim that :

> *" We are not even past McCarthy establishment to generate new thoory of computer science "*

4. CONCLUSION

In this short paper we attempted to answer one of the most significant questions concernning the relation ships between the OOP (Object Visual programming) and the classical theory of computer i.e. McCarthy's work.The paper shows that it is quite feasible to deliver that

> *" There is no Object Oriented PROGRAMMING but rather Object Oriented METHOD or STYLE or MANNER for administration two dimension matrices of pixels within the screen matrix.The Object Oriented languages are a methodology or techniques for handling string code . "*

REFERENCES

[Al72a] Aho, A. and Ullman, J.D.
"*The Theory of Parsing, Translation and Compiling*";
vol. 1 Prince-Hall 1972

[Al72b] Aho, A. and Ullman, J.D.
"*The Theory of Parsing, Translation and Compiling*";
Vol.2, Prince-Hall 1972

[Ba78] Backhouse, R.C.
"*Syntax of Programming Languages: Theory and Practice*";
Prentice-Hall Inc. 1978

[Br77] Braddy, J.M.
"*The Theory of Computer Science : A Programming Approach*";
Chapman and Hall 1977

[Ch56] Chomsky, N.
"*Three models for the description of language*";
IRE Trans. Inform. Theory, IT2,113-124

[Co88] Cohen, D.I.A.
"*Introduction to Computer Theory*";
John Wiley and Son, Inc. 1986

[Cl07] Cleaveland, J.C. and Uzgalis, R.U.
"*Grammars for Programming Languages*";
Elsevier North-Holland, Inc. 1977

[Go83] Goldberg, A.
"*A Smalltalk - 80: The Interactive Programming Environment*";
Addison-Wesley 1983

[HM86] Hughes, C.D. and MOdhell J.M.
"*Visible Pascal: A graphic-Based Learning Environment*";
Proceeding of Computer Graphics 86,
National Computer Graphics Association,
May 1986.

[Ke89] Keene, S.E.
"*Object Oriented Programming in COMMON LISP: A Programmer's Guide to COLS*"
Addison-Wesley Pub. Com. 1989

[Kl36] Kleene, S.C.
"*General Recursive Functions of Natural Numbers*";
Math. Annal. 112,340-353, 1936

[Kl56] Kleene, S.C.
"*Representation of Event in nerve net and Finite automata*";
Automata Studies, Princeton 1956

[Kr83] Krishnamurthy, E.V.
"*Introductory Theory of Computer Science*";
Macmillan press 1983

[Le86] Levien, R.
"*Visual Programming*";
Byte Vol 11 No 2 1986