*Military Technical College,
Kobry El-Kobbah,
Cairo, Egypt*

*9th International Conference
On Aerospace Sciences &
Aviation Technology*

A MATHEMATICAL MODEL FOR GOAL ORIENTED SYSTEM

EL- SHARAWY* G. A.

ABSTRACT

This work presents a mathematical model that incorporates goal oriented computer system problem with predicate calculus PC. In this model, the client - server paradigm is exploited so that the object, which provides a service, is a server while the object, which requests it, is regarded as a client. This work introduces a technique for modeling goal oriented computer system using mathematical notations. The model defines the system states and operations needed for system transition from state to another using first order language.

KEY WORDS

System Modeling, Predicate Calculus, and First Order language.

INTRODUCTION

Real world systems may be defined as an assembly of interconnected and/or interdependent objects. Techniques used for the simulation of such systems fall into two distinct categories; continuous and discrete. Continuous systems are generally modeled using differential equations. Such systems may also involve the use of analog computers. Modeling discrete systems, on the other hand, involves the use of a different set of techniques. Simulating the behavior of such systems usually involves the definition of operations which, providing their preconditions are satisfied, may be applied to a given state resulting in a transition to a new pre-defined state. The client-server model provides a means of solving many real world problems such as car service, shopping, and machine assembly. This research introduces a technique for solving problems in a client – server framework.

* Lecturer, Dpt., of Mathematics, Faculty of Science, Al-Azhar University, Cairo, Egypt

REAL TIME SYSTEM

A real time computer system is a goal oriented software system whose functioning depends on the results produced and the time at which those results are produced.
A "soft" real time system whose operation is degraded if results are not produced according to the specified timing requirements. A "hard " real time system is a system whose operation is incorrect if results are not produced according to the timing specification[1].

The real-time system problem may be described as goal oriented problem. Each problem presented to the model is a "goal " to be achieved in a specified " world ". The solution is a sequence of operations (operators) or a " plan " of action which when applied, the model achieves the given goal[2].

FIRST ORDER LANGUAGES

The formalism used in the description of the world and the effect of applying an operator to a given world (alternatively - an operation in a given world) is that of first order predicate calculus PC. In general a first order language L will have as its alphabet of symbols:

Variables	$x_1, x_2, \dots,$
Constants	$a_1, a_2, \dots,$
Predicates	${}^{n}A_i$
Functions	${}^{n}f_i$
Punctuation symbols	(,) , " , "
Connectives	$\neg, \wedge, \vee, \Rightarrow \quad \Leftrightarrow$
Quantifiers	$\exists, \forall$

A " term " of language L is defined as follows: -

(i) A variable is a term.

(ii) A constant is a term.

(iii) If ${}^{n}f_i$ is a function in L and $t_1, \dots, t_n$ are terms in L then ${}^{n}f_i(t_1, \dots, t_n)$ is a term .

(iv) Noting else is a term.

A " well formed formula" (wff) is defined as follows: -

(i) If ${}^{n}A_i$ is a predicate in L and $t_1, \dots, t_n$ are terms in L then ${}^{n}A_i(t_1, \dots, t_n)$ is a wff in L .

(ii) if A and B are wffs in L then $\neg A$, $A \Rightarrow B$, $A \wedge B$, $A \vee B$, $\exists\, x_i A$ and $\forall\, x_i A$ are wffs in L (where x_i is any variable[3]) .

The interpretation of the symbols of a first order language may alter depending on the model in mind. For example, in a first order language used to make statements about natural numbers, the following interpretation may be employed: -

a_1	to stand for zero
${}^{2}A_1$	to stand for =
${}^{1}f_1$	to stand for the successor function
${}^{2}f_1$	to stand for +
${}^{2}f_2$	to stand for *

In such a language ${}^2A_1$ (${}^2f_1$ (x_1 , x_1), ${}^2f_2$ (x_1 , x_1)) would be interpreted as: $x_1 + x_1 = x_1 * x_1$. Given the stated interpretation there is only one number which satisfies this wff , namely 2 .

Formally, an interpretation I of a first order language L is a non–empty set ${}^I D$ (the Domain of interpretation) together with a collection of distinguished elements (a_1, a_2, ...), a collection of function on ${}^I D$ (${}^n f_i$, i > o , n > o) (${}^n f_i : D^1 \times \ldots \times D^n \Rightarrow D$) and a collection of relations on ${}^I D$ (${}^n D_i$, i >, n > o) (${}^n D_i \subseteq D^1 \times \ldots \times D^n$).

In order to determine the truth of a specific statement, several additional concepts must be defined. The interpretations in I of a_i, ${}^n f_i$ and ${}^n A_i$ will be denoted a_i , ${}^n f_i$ and ${}^n A_I$. A "valuation" in I is a function v from the set of terms of L to the set ${}^I D$ with the properties:-

(i) v (a_i) = a_i for every constant symbol in L .

(ii) v (${}^n f_i$ (t_1 , ... , t_n)) = ${}^n f_i$ (v (t_1) , ... , v (t_n)) where ${}^n f_i$ is any function symbol of L and $t_1 \ldots t_n$ are any terms of L .

Tow valuation v and v' are deemed i_equivalent if v (x_{ij}) = v' (x_{ij}) for every i ≠ j . These valuations have the same values on each of the variable except possibly on x_{ij} [4,5]. A valuation v in I is thus said to satisfy wff A if it can be shown inductively to do so under the following conditions:-

(i) v satisfies ${}^n A_i$ (t_1, ... , t_n) if ${}^n A_i$ (v (t_1) , ... , v (t_n)) is true in ${}^I D$.

(ii) v satisfies $\neg$ B if it does not satisfy B .

(iii) v satisfies $B \wedge C$ if v satisfies B and v satisfies C .

(iv) v satisfies $B \vee C$ if v satisfies B or v satisfies C .

(v) v satisfies $B \Rightarrow C$ if v satisfies $\neg$ B or v satisfies C .

(vj) v satisfies $\forall x_i B$ if every valuation v' which is i _ equivalent to v satisfies B .

(vii) v satisfies $\exists x_i B$ if v satisfies $\neg \forall x_i \neg B$.

THE WORLD MODEL

The model employs PC in the description of the current states of the world. The description of the world may be divided into two distinct sets of wff's:-

(i) **General knowledge** which describe static facts. These facts are considered true in all world model conditions. They are not likely to alter unless something catastrophic happens, such as the entire set of world model components being blown up.

(ii) **Dynamic knowledge** which describe temporary facts. Whereas static components are not allowed to alter their positions during the model "lifetime", other objects such as movable components. are very likely to do so . For example suppose a world model which consists of a robot and boxes. The robot could carry boxes and move from position to position. If box B which is at position P_1 is moved to position P_2. Assuming the predicate At(Box , Position) is used to record these facts , then in the current world AT(B, P_1) is true . With the robot having moved the box to position P_2. At (B, P_2) become true . Unless at (B , P_1) is removed from the current description of the world it would have to be believed that the box in question can be in two places (positions) P_1 and P_2 at the same time (assuming P_1 and P_2 are distinct positions , i . e . $P_1 \neq P_2$).

Temporary facts are thus added or removed (deleted) from the description of the world to reflect the current state and maintain consistency. Manipulation of facts in this manner is analogous to Prolog's use of second order predicates Assert and Retract [6].

MODEL OPERATORS

The model operators have a dual nature. Viewed algebraically, like any operators, which manipulate some objects. They may also be viewed as the tools available for carrying out a given task. An analogy may be drawn, between a carpenter's tools and the world model operators. This view is based on the usage of operators. Within the models framework an operator's definition takes the following form: -

$< wff_1, {}^{n}op_i (x_1, \ldots, x_n), wff_2, wff_3 >$, where :-

wff_1 set of wff's, defining the operator's preconditions.

${}^{n}op_i$ operator(s) that may be applied to the world model according to wff_1.

x_i the i th parameter

wff_2 wff's defining the set of facts to be added to the world's description following the application of the operator (the Add list) .

wff_3 wff's defining the set of facts to be removed from the world's description following the application of the operator(the Delete list)

The meaning of an operator's definition may be given using a variation of axiomatic semantics. In this formalism assertions are made regarding the pre and post conditions of a given operator. For example, the square root (sqrt) function in a Pascal program may be specified as follows:-

Pre : $\{ x \in R \wedge x \geqslant 0 \}$
y := sqrt (x)
post : $\{ y * y = x \}$

The specification states that, if the square root function (sqrt) is applied in any state satisfying the given precondition, specifically, a state where x is a real number grater than zero. The execution of assignment statement is guaranteed to terminate in a state where y * y = x . This guarantee of termination in finite time, in a specific state is established through the use of axioms defining how the various operations such as assignment, sequencing, and control structures such as if statements and while loops effect the state [7,8].

The aspect that serves the current discussion is the use of assertions. An operator may be seen as stating its own assertions through it's add and delete lists. Consider the following specification of the operator INC x which add 1 to x

<{ x = a }, INC x, {x = a + 1}, { x = a} >, where:

{x = a }	(precondition)
INC x	(operator)
{x = a + 1 }	(add list)
{ x = a }	(delete list)

The specification states that operator INC x may be applied in any state where the value of x is equal to some constant a. Intuitively, once the operator is applied, the value of x is expected to be equal to a + 1. This fact is stated in the add list, which asserts that x = a + 1 must be added to the set of PC statements defining the new

state. In addition, to maintain consistency, the delete list states that the assertion x=a must be removed. If the assertion x =a not removed the resultant world would be one where $x = a \wedge x = a + 1$, allowing the perfectly valid derivation of the statement a=a +1 (reflexivity and transitivity of equality). Given this fact a robot may be justified in assuming that zero steps in the direction of an object are just as good as one or more generally an infinity of steps are just as good as zero. Formally, a state Õ is defined as a function from variables to elements of ^{l}D, Õ: var. $\Rightarrow$ D, a state may be viewed as a valuation function defined earlier. Given operator $^{n}op_{l}$ defined as follows: -
$< wff_1 \, ^{n}op_{l} \, (x1, \ldots , x_{n}) \, wff_{2,} \, wff_3 >$
The operator acts correctly if it is applied in a state where its preconditions are satisfied. In other words in a state Õ where Õ satisfies wff_1. Alternatively, let $wff_{\tilde{o}}$ be a set of predicates describing state Õ (Õ satisfies $wff_{\tilde{o}}$). The operator's preconditions will be satisfied iff $wff_{\tilde{o}} \Rightarrow wff1$.

The effect of applying an operator is defined using its add list (wff_2) and delete list (wff_3). In essence, operator application causes a transition from state Õ where $wff_{\tilde{o}}$ is satisfied to a successor state where wff_2 is satisfied along with all statements in $wff_{\tilde{o}}$ apart from those mentioned in wff_3. Assuming the statements are in conjunctive from: $wff_p = (wff_{\tilde{o}} - wff_3) \cup wff_2$
The effect of applying INC x defined in the previous example is illustrated in figure (1). The operator is applied in a state Õ where $wff_{\tilde{o}} = \{x = a\}$. Once the operator is applied the resultant state þ is one which satisfied wff þ , where :-
$wff\ þ = (wff_{\tilde{o}} - wff_3) \cup wff_2 = (\{ x = a \} - \{ x = a \}) \cup \{ x = a + 1) = \{ x = a + 1 \}$

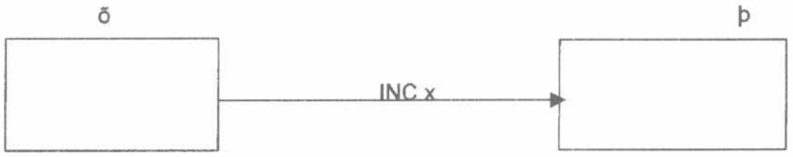

Fig. 1 Effect of operator application

INFERENCING AND OPERATOR SEQUENCE DETERMINATION

The purpose of the rather unconventional definition of operators in the model is to aid mechanized deduction. In the world of robot and boxes, the robot is given some task or goal to be achieved starting from a particular state. In such a setting, operators may be viewed as the robot's " tools " . It is up to the robot to decide on the sequence of operator applications to use in order to accomplish the given task.
Initially the model is presented with a pair $< M_0 , < G_0 >>$ where M_0 , is a set of wffs describing the initial world and G_n is a set of wffs specifying the goals to be achieved. Given a pair $< M_l , < G_l , \ldots , G_n >>$ the system first determines whether or not the goals are immediately achievable. If the former is the case then the task is achievable using the sequence of selected operations. If the letter is the case, the system switches to " means – ends " analysis with the aim of ascertaining the difference between the current state M_l and the goal G_n. The difference consists of any formulae that remain unresolved once the proof attempt is abandoned. The difference provides a guideline for operator choice, the criteria being to select an operator whose add list reduces part or all of the difference. If the operator's pre-

conditions are satisfied, operator will be applied, otherwise its preconditions become a new sub goal to be achieved [8].

As an example consider the case where the robot is asked to read a book . Let O be the set of operations available to the robot, where $O \subseteq < wff_1, {}^n op_i (x_1 , \ldots , x_n), wff_2 , wff_3 >$. An element of O has the form :-
⟨ preconditions , operator name , Add list (set) , Delete list (set) ⟩ .
The world's state is described using four propositions: -

lo – Light on
sd – setting down
dt – drinking tea
r – reading book

The robot may perform any of the following operations:-

st – sit down
rb – read book
gt – get a cup of tea
tlo – turn light on

The set O of operations available to the robot may therefore be fully defined as follows:-

$<\{ lo , sd , dt \} , rb, \{ r \}, \{ \} >$

A book may not be read unless the robot is sitting down, have a cup of tea and the light is on .

$< \{\neg lo \}, tlo, \{ lo \}, \{\neg lo \} >$

The light may not be turned on unless it is already off. Similarly for the subsequent operations.

$< \{\neg dt \}, gt , \{ dt \}, \{\neg dt \} >$,
$<\{\neg sd \}, st , \{ sd \}, \{\neg sd \}>$

The specification of the read book (rb) operation states that a book may not be read unless the robot is sitting down, have a drink and the light is on. Also, once the operation is started the robot carries on reading. Turning the light on is specified using the tlo operation. In this case the operation may not be applied unless the light is already off. In addition, once the operation is applied the light remains on (add list) and is no longer off (delete list).

Given the goal of reading a book the obvious sequence of operations is to turn the light on , get drink , sit down and then start reading as illustrated in the following derivation ([ai] refers to the axiom number) :-

$\langle$ world, goal , operation $\rangle \in$ seq	
$\langle \{lo , dt , st \} , nil , nil \rangle \in seq$	(1) [a1]
$\langle \{lo , dt , st \} , [\{ lo , dt , st \}] , nil \rangle \in seq$	(2) [a2 – from 1]
$\langle \{lo , dt , \neg st \} , [\{ lo , dt , st \}] , [st] \rangle \in seq$	(3) [a3 – from 2]
$\langle \{lo , \neg dt , \neg st \} , [\{ lo , dt , st \}] , [gt , st] \rangle \in seq$	(4) [a3 – from 3]
$\langle\{\neg lo , \neg dt , \neg st \} , [\{ lo , dt , st \}] , [tlo , gt , st] \rangle \in seq$	(5) [a3 – from 4]
$\langle \{lo , dt , st , r \} , nil , nil \rangle \in seq$	(6) [a1]
$\langle \{lo , dt , st , r \} , [\{r \}] , nil \rangle \in seq$	(7) [a2 – from 6]

$\langle \{lo , dt , st, \neg r\} , [\{r \}] , [r b] \rangle \in seq$ (8) [a3 – from 7]

$\langle\{\neg lo , \neg dt , \neg st, \neg r \}, [\{r \}] , [tlo , gt , st , rb] \rangle \in seq$ (9) [a4 – from 5 , 8]

The definition of seq allows for the derivation of other operation sequence which according to their definitions can achieve the goal, these include [st , gt , tlo , r]. Although such a sequence of operations may be impractical in the real world its exclusion from the set of possible solutions can only be accomplished through the strengthening of the operator's preconditions.

CONCLUSION:

This work has presented a mathematical model that could contribute to the enhancement of modeling the real time client–server computer system. The presented model incorporates software-engineering methodologies for modeling real time system with mathematical techniques for solving problems. The world model is defined using predicate calculus in the description of the current system. The description of the world model may be divided into two sets of wffs to define both static knowledge and dynamic knowledge. The static knowledge define the realm environment. The dynamic knowledge define the system states, starting from initial state to final state (goal state) and the operators needed for system transformation from state to another. The model defines preconditions needed when activating an operator from specified state and post conditions reached after operator activation. This way of modeling system may be helpful when designing on expert system or when implementing real time computer system using a predicate language like Prolog.

REFERENCES

[1] Somerville, I., Software Engineering, Fifth Edition, Addison-Wesley. Publishing Company, (New York) 1995.

[2] Robert J. Muller, Database Design for Smarties , Morgan Kaufmanns publishers INC , San francisco , (California) , 1999 .

[3] Richard C. Jeffrey, Formal Logic: its scope and limits, McGraw Hill Publication company , (New York) , 1990.

[4] Angelo Mergers, First Order Mathematical logic , Dover publications (Dover), 1990.

[5] Raymond M. smullyan, First Order Logic , Dover publication , (Dover) , 1995 .

[6] Qing, L., and McLeod, D., Conceptual Database Evolution through Learning in an Object Oriented Database System, Expert Database System, Proceeding from the second International conference. L., kerschberg, Ed., Benjamin / Cummings, Menlo Park , California , (1989) , 469 – 499

[7] Fikes, R., Hart, P., and Nilsson, N., Learning and Executing Generalized Robot Plans. Artificial Intelligence, vol 3, (1972), 251-288

[8] Zaki, M., El-Sharawy, G., A Plane Learning Model for Object Oriented Database, MS2000, International Conference on Modelling and Simulation, Las Palmas de Gran Canaria, (Spain), 2000.