ABSTRACT:
A number of proofs of the correctness of implementations for the "Block Concept" have been given. These proofs have been based on a definition using an abstract machine. This note attempts to repeat the exercise with an alternative definition. The relative merits of the approaches are reviewed.

LN 25.3.075
24 August 1970
INTRODUCTION

This note is an attempt to indicate how proofs of correctness of "block implementations" might be simplified in contrast to /2/, /1/ etc. The approach is to base the proof on a different style of definition: instead of having a base abstract interpreter machine, an attempt is made to give properties required of a model in terms of an equivalence relation over occurrences of identifiers in blocks.

The current form of the note relies heavily on /1/ with which the reader is assumed to be familiar. The two methods which are proved to be correct models of the definition are basically the defining model and mechanism 1 of /1/ except that call by reference has been included. This choice of mechanisms will facilitate the more complete discussion which is reserved for a later section, which will also review to what extent the given definition can be considered to be "machine-free".
NOTATION

The notation of /1/ will be used throughout the sequel. Certain formulas should be compared or contrasted to those of /1/ and in such cases the appropriate reference is given on the right.

Equivalence relations are used below and the relevant facts are now presented:

$\mathcal{R}^S$ is said to be an equivalence relation on a set $S$ if it is symmetric, reflexive and transitive over that set. (The name of the set is omitted when there is no danger of confusion.)

$\mathcal{R}^S$ partitions $S$ into disjoint subsets or cosets such that

$$\alpha \in \mathcal{R}^S \overset{\text{Df}}{\iff} \{\beta \mid \alpha \in S \land \beta \in S\}$$

for $\alpha \in S$

$\mathcal{R}^{ST}$ is an extension of $\mathcal{R}^S$ providing

$$\alpha, \beta \in S \implies (\alpha \mathcal{R}^{ST} \beta \iff \alpha \mathcal{R}^S \beta)$$

The following ways of relating elements, $\gamma$, of $T$ are examples of how the extension can be defined to satisfy the above

i) for exactly one $\alpha, \alpha \in S$, specify $\gamma \mathcal{R}^{ST} \alpha$

ii) specify $\gamma \mathcal{R}^{ST} = \{\gamma\}$ (no element except $\gamma$ is in the coset of $\gamma$)
DEFINITION

A "reference" can occur in an active block/procedure and is represented by an identifier, activation pair.

References are related using:

a) a set of all possible references
b) an equivalence relation over the set a)

for each active (dynamic) occurrence of a block/procedure. The effect of blocks/procedures on relations between identifiers can be expressed by showing how new sets and relations are formed.

basic sets:

\[ D1 \quad \text{ID} \quad \text{identifiers} \]
\[ D2 \quad \text{PT} \quad \text{activation markers} \]
\[ D3 \quad \text{REF} = \text{ID} \times \text{PT} \]

the interpretation of \((id, i) \in \text{REF}\) is id is known in activation i.

the following abbreviation is used:

\[ \text{ID}^{P, \text{REF}} \overset{\text{df}}{=} \{ \text{id} \mid (\text{id}, \text{p}) \in \text{REF} \} \]

relation:

\[ \text{REF}\overset{\text{REF}}{=} \text{an equivalence relation;} \]
\[ \text{the interpretation of } r1\overset{\text{REF}}{=} r2, \text{ for } r1, r2 \in \text{REF}, \]
\[ \text{is } r1 \text{ and } r2 \text{ refer to the same entity.} \]

values:

\[ D5 \quad \text{a value is an object associated with a coset of } \overset{\text{REF}}{=} \]
\[ \text{and, if a procedure introduced in activation } r, \text{ ID}^{P} \text{ can be determined.} \]

initial:

\[ I1 \quad \text{REF}^{1} = \{ \} \]
block: suppose the block to be interpreted is encountered in activation p.
let: \( \text{REF}, \mathcal{R}^\text{REF} \) be the set and relation at activation p
\( D \) be the set of names declared in the block

B1 choose \( q \in \text{PT} \) such that \( \neg (\exists r)(r \in \text{REF} \land 2\text{nd}(r) = q) \)

then the block is interpreted with:

B2 \( \text{ID}^q = \text{ID}^p, \text{REF} \cup D \)
B3 \( \text{REF}' = \text{REF} \cup \{(id,q) \mid id \in \text{ID}^q\} \)

extend \( \mathcal{R}^\text{REF} \) to \( \mathcal{R}^{\text{REF}'} \) as follows:

B4 \( \alpha, \beta \in \text{REF} \Rightarrow (\alpha \mathcal{R}^{\text{REF}'} \beta \equiv \alpha \mathcal{R}^{\text{REF}'} \beta) \)
B5 \( id \notin D \Rightarrow ((id,q) \mathcal{R}^{\text{REF}'} (id,p)) \)
B6 \( id \in D \Rightarrow ((id,q) \mathcal{R}^{\text{REF}'} = \{(id,q)\}) \)
B7 execution of the nested block has no influence on \( \text{REF} \) or \( \mathcal{R} \) of activation p
B8 for \( id \in D \): value associated with \( (id,q) \mathcal{R}^{\text{REF}'} \) is introduced in q.
procedure: suppose the call is encountered in activation \( p \), and the procedure invoked was introduced in activation \( r \).

let: \( \text{REF}, \mathcal{K}^{\text{REF}} \) be the set and relation at activation \( p \)
\( \mathcal{P} \) be the names of the parameter list
\( \mathcal{A} \) be the names of the argument list \( \{ \) used both as sets and lists

P0 if id is the name of the procedure called, then:
\( (\text{id}, p) \in \text{REF} \)
value associated with \( (\text{id}, p) \mathcal{K}^{\text{REF}} \) is the denotation of the invoked procedure.

P1 choose \( q \in \mathcal{P} T \) such that \( \gamma(\emptyset) r (r \in \text{REF} \land \text{2nd}(r) = q) \)

then the procedure is interpreted with:

P2 \( \text{ID}^q = \text{ID}^r, \mathcal{P} T \cup \mathcal{P} \)

P3 \( \text{REF}' = \text{REF} \cup \{ (\text{id}, q) \mid \text{id} \in \text{ID}^q \} \)

extend \( \mathcal{K}^{\text{REF}} \) to \( \mathcal{K}^{\text{REF}'} \) as follows:

P4 \( \alpha, \beta \in \text{REF} \Rightarrow (\alpha \mathcal{K}^{\text{REF}'} \beta) \notin \mathcal{K}^{\text{REF}} \)

P5 \( \text{id} \notin \mathcal{P} \Rightarrow ((\text{id}, q) \mathcal{K}^{\text{REF}'} (\text{id}, r)) \)

P6 \( \text{id} = \text{P}_i \Rightarrow ((\text{P}_i, q) \mathcal{K}^{\text{REF}'} (\text{A}_i, p)) \)

P7 execution of the called procedure has no influence on \( \text{REF}, \mathcal{K} \) of activation \( p \).

To show that a model satisfies these properties, it is necessary to define \( D1 - D5 \) in terms of its state and show that these realizations satisfy \( I1; B1 - B8 \) and \( P0 - P7 \).
First Model

State:

s1  \text{is-state}(\xi) \Rightarrow (\text{is-dump}(s \cdot d(\xi)) \land \\
     \text{is-dendir}(s \cdot dn(\xi)) \land \\
     s \cdot U(\xi) \subseteq \text{UN}) \tag{S1}

s2  \text{is-dump}(d) \Rightarrow \text{is-de}(d_i) \quad \text{for } 1 \leq i \leq l(d) \tag{S2}

s3  \text{is-de}(de) \Rightarrow (s \cdot tp(de) \in TP \land \\
     \text{is-env}(s \cdot e(de))) \tag{S3}

s4  \text{is-env}(e) \Rightarrow (D(e) \subseteq \text{ID} \land \\
     R(e) \subseteq \text{UN}) \tag{S4}

s5  \text{is-dendir}(dn) \Rightarrow (D(dn) \subseteq \text{UN} \land \\
     R(dn) \subseteq \text{DEN}) \tag{S5}

s6  \text{is-proc-den}(den) \Rightarrow (\text{den} \in \text{DEN} \land \\
     s \cdot tp(den) \in TP \land \\
     \text{is-env}(s \cdot e(den))) \tag{S6}

State transitions:

Interpretation of a program begins with \text{init}

Interpretation of a block consists of \text{term} \ldots \cdot \text{bloc}

and \text{term} \ldots \cdot \text{proc}

Only elements of a computation formed by \text{init}, \text{bloc}, \text{proc}

or \text{term} are considered below.

\text{init}: \Rightarrow \xi^1 \tag{T1}

i1  l(d^1) = 1 \tag{T2}

i2  e^1 = \Omega \tag{T3}

i3  \text{dn}^1 = \Omega \tag{T4}

i4  U^1 = \{\} \tag{T5}
bloc : \( \xi \Rightarrow \xi' \)

b1 \( D(eo-b) = D \) (T8)
b2 \( R(eo-b) \cap U = \{ \} \) (T9)
b3 \( id1(eo-b) = id2(eo-b) \neq \Omega \Rightarrow id1 = id2 \)
b4 \( \text{rest}(d') = d \) (T10)
b5 \( \text{for } i \leq l(d) : u \in R(s-e(d_i)) \Rightarrow u(dn') = u(dn) \) (T13)
b6 \( U' = R(eo-b) \cup U \) (T14)
b7 \( e' = \text{update}(e, eo-b) \) (T17)
b8 \( \text{for } u \in R(eo-b) \land \text{is-proc-den}(u(dn')): 
                      \quad s-e(u(dn')) = e' \) (T19b)

proc: \( \xi \Rightarrow \xi' \)

p1 \( \text{id-p} \in D(e) \) (T20)
p2 \( u-p = \text{id-p}(e) \) (T22)
    \( e-p = s-e(u-p(dn)) \) (T21)
p3 \( \text{is-proc-den}(u-p(dn)) \) (T21)
p4 \( D(eo-b) = P \) (T8)
p5 \( \text{for } 1 \leq i \leq l(P) : P_i(eo-b) = A_i(e) \) (T8)
p6 \( \text{rest}(d') = d \) (T10)
p7 \( \text{for } i \leq l(d) : u \in R(s-e(d_i)) \Rightarrow u(dn') = u(dn) \) (T13)
p8 \( U' = U \) (T14)
p9 \( e' = \text{update}(e-p, eo-b) \) (T23)

term: \( \xi \Rightarrow \xi' \)

t1 \( d' = \text{rest}(d) \) (T26)
t2 \( \text{for } i \leq l(d') : u \in R(s-e(d_i')) \Rightarrow u(dn') = u(dn) \) (T27)
t3 \( U' = U \) (T28)

---

1) Strictly, the model should exhibit a way of storing \( P \) with the denotation, the omission derives from the use of /1/ as base.
Further, let (for some state $\xi$):

$m1$ \quad ID^i = D(e_i)

$m2$ \quad PT = N \quad \text{indexes to the dump}

$m3$ \quad \text{REF} = \{(id, i) \mid id \in ID^i \land 0 \leq i \leq l(d)\}

$m4$ \quad (id_1, i) \not\in \text{REF} \quad \text{if} \quad id1(e_i) \neq \emptyset \land id1(e_i) = id2(e_j)

Two lemmas can be proved about the model:

$l1$ \quad for $\xi$: \quad u \in U \Rightarrow \neg (\exists id, j)(j \leq l(d) \land u = id(e_j))

A proof by induction (similar to L5 of /1/) of a strengthened proposition can be given using:

basis \quad i4
induction bloc \quad b4, b6, b7, b8
\quad proc \quad p5, p6, p8, p9, P0, m3
\quad term \quad t1, t3

$l2$ \quad for \quad \text{for a procedure introduced in activation } r \text{ such that id(e) gives its unique name:}

\quad s-e(id(e)(dn)) = e_r

A proof by induction (similar to L6 of /1/, but made simpler by the differences b5 to T13 etc.) can be given using:

basis \quad i2
induction bloc \quad b4, b5, b7, b8
\quad proc \quad p5, p6, p7, p9, P0, m3
\quad term \quad t1, t2
Theorem: The above model satisfies the properties required by the definition.

Proof:

D1, D2, D3: The basic sets are defined

D4: \( \mathcal{K} \) is an equivalence relation over the given sets

D5: values are associated with the cosets of \( \mathcal{K} \) via unique names, and \( \text{ID}^p \) for procedures can be found.

I1: \( \text{ID}^1 = \{\} \)

\[ \text{REF}^1 = \{\} \]

B1: \((id,i) \in \text{REF} \Rightarrow i \leq l(d)\)

\[ \therefore \ q = l(d') \text{ fulfils the conditions} \]

B2: \( \text{ID}^q = D(e') \)

\[ = D(e_p) \cup D(eo-b) \]

\[ = \text{ID}^p; \text{REF} \cup D \]

B3: \( \text{REF}' = \text{REF} \cup \{ (id,q) \mid id \in \text{ID}^q \} \)

B4: \( \alpha, \beta \in \text{REF} \Rightarrow 2\text{nd}(\alpha) \leq l(d) \land 2\text{nd}(\beta) \leq l(d) \)

\[ \therefore \alpha ^{REF}' \beta = \alpha \cap \text{REF} \beta \]

B5: \( \text{id} \notin D \Rightarrow \text{id}(e') = \text{id}(e) \)

\[ \therefore (id,q) \notin \text{REF}' (id,p) \]

B6: \( \text{id} \epsilon D \Rightarrow \text{id}(e') = \text{id}(eo-b) \)

\[ \text{id}(e') \notin U \]

\[ \neg (\exists id,j)(j \leq l(d) \land \text{id}(e') = \text{id}(e_j)) \]

\[ \therefore (id,q) \notin \text{REF}' = \{ (id,q) \} \]

B7: \text{term*bloc} is an identity with respect to:

\[ \text{REF} \]

and thus \( \mathcal{K} \)
P1: \((id,i) \in \text{REF} \Rightarrow i \leq l(d)\)  
\[\therefore q = l(d')\] fulfils the condition

P2: \(ID^q = D(e')\)
\[= D(e'_r) \cup D(eo-b)\]
\[= ID^q,\text{REF} \cup P\]

P3: \(\text{REF}' = \text{REF} \cup \{ (id,q) \mid id \in ID^q \}\)

P4: \(\alpha, \beta \in \text{REF} \Rightarrow 2nd(\alpha) \leq l(d) \land 2nd(\beta) \leq l(d)\)
\[\therefore \alpha \not\equiv \beta\]

P5: \(id \notin P \Rightarrow id(e'_q) = id(e'_r)\)
\[\therefore (id,q) \not\equiv \text{REF}'(id,r)\]

P6: \(id \notin P \Rightarrow P_i(e'_q) = P_i(eo-b)\)
\[= A_i(e)\]
\[\therefore (P_i,q) \not\equiv \text{REF}'(A_i,p)\]

P7: term·proc is an identity with respect to:
\[\text{REF}\]
and thus \[\not\equiv\]
Second Model:

State:

\[ s_1 \quad \text{is-state}(\xi) \Rightarrow \text{is-dump}(s \cdot d(\xi)) \] (S1)

\[ s_2 \quad \text{is-dump}(d) \Rightarrow \text{is-de}(d_i) \quad \text{for } 1 \leq i \leq l(d) \] (S2)

\[ s_3 \quad \text{is-de}(de) \Rightarrow (s \cdot \text{tp}(de) \in TP \wedge \\
\text{is-dendir}(s \cdot \text{dn}(de)) \wedge \text{s-epa}(de) \in N) \] (S3)

\[ s_4 \quad \text{is-dendir}(dn) \Rightarrow (D(dn) \in ID \wedge \\
R(dn) \in DEN) \] (S4)

\[ s_5 \quad \text{is-proc-den}(den) \Rightarrow (\text{den} \in DEN \wedge \\
s \cdot \text{tp}(den) \in TP \wedge \\
s \cdot \text{epa}(den) \in N) \] (S5)

\[ s_6 \quad \text{is-parm-den}(den) \Rightarrow (\text{den} \in DEN \wedge \\
s \cdot \text{id}(den) \in ID) \] (S6)

State transitions:

\[ \text{init: } \quad \longrightarrow \xi' \]

\[ i_1 \quad l(d_1) = 1 \] (T1)

\[ i_2 \quad \text{dn}_1 = \Omega \] (T4)

\[ i_3 \quad \text{epa}_1 = 0 \] (T5)

\[ \text{bloc: } \xi \Rightarrow \xi' \]

\[ b_1 \quad D(dn') = D \] (T8)

\[ b_2 \quad \neg (\exists dn)(dn \in R(dn') \wedge \text{is-parm-den}(dn)) \] (T9)

\[ b_3 \quad \text{rest}(d') = d \] (T10)

\[ b_4 \quad \text{epa}' = l(d) \] (T18)

\[ b_5 \quad \text{for } \text{is-proc-den}(id(dn')): \\
\quad \text{s-epa}(id(dn')) = l(d) \] (T19)
proc: $\xi \mapsto \xi'$

p1 \hspace{1cm} \text{is-proc-den(find-d(id-p,d))} \hspace{1cm} (T21)

p2 \hspace{1cm} \text{epa-p = s-epa(find-d(id-p,d))} \hspace{1cm} (T22)

p3 \hspace{1cm} D(dn') = P \hspace{1cm} (T8)

p4 \hspace{1cm} \text{for } 1 \leq i \leq l(P): \text{s-id}(P_i(dn')) = A_i \hspace{1cm} (T10)

p5 \hspace{1cm} \text{rest}(d') = d \hspace{1cm} (T24)

p6 \hspace{1cm} \text{epa'} = \text{epa-p} \hspace{1cm} (T26)

t1 \hspace{1cm} d' = \text{rest}(d) \hspace{1cm} (T25)

Further, let (for some state $\xi$):

m0 \hspace{1cm} \text{find(id,d) = l(d) = 1 $\mapsto \Omega$}

\hspace{1cm} \text{is-param-den(id(dn)) $\mapsto \text{find(id(dn),rest(d))}$}

\hspace{1cm} \text{id } \in \text{D(dn)} $\mapsto \text{id(dn)}$

\hspace{1cm} T $\mapsto \text{find(id,rest(d,epa))}$

where: dn = s-dn(top(d))

\hspace{1cm} epa = s-epa(top(d))

To simplify the equivalence relation $m4$, find-p is used which differs from find only in that the third case distinction yields the pair ($id, l(d)$).

m1 \hspace{1cm} \text{ID}^i = \{ id \mid \text{find(id,rest(d,i)) } \neq \Omega \}$

m2 \hspace{1cm} \text{PT} = N

m3 \hspace{1cm} \text{REF} = \{ (id,j) \mid \text{id } \in \text{ID}^j \land 0 \leq j \leq l(d) \}$

m4 \hspace{1cm} (id1,i) \notin (id2,j) \hspace{1cm} \text{if } \text{find-p(id1,rest(d,i)) } \neq \Omega \land

\hspace{4cm} \text{find-p(id1,rest(d,i)) = find-p(id2,rest(d,j))}$
The following lemma can be proved about the model:

for $\xi$: for a procedure introduced in activation $r$ such that $\text{find}(id,d)$ gives its denotation:

$$s\text{-epa}(\text{find}(id,d)) = r$$

A proof by induction can be given:

basis $i1,m0$

induction bloc $b3,b4,b5,m0$
proc $p2,p3,p4,p5,p6,P0,m3$
term $t1$

**Theorem**: The above model satisfies the properties required by the definition.

**Proof:**

D1,D2,D3: The basic sets are defined $m1,m0,s4,m2,m3$

D4: $R$ is an equivalence relation over the given sets $m1,m3,m4$

D5: values are associated with the cosets of $R$ $m4,m0,s4$
via find, and $ID^R$ for procedures can be found. $P0,11$

I1: $ID^1 = \{\}$ $m1,m0,i1$

$\therefore$ REF$^1 = \{\}$ $m3,i1$
B1: \((id,i) \in \text{REF} \Rightarrow i \leq l(d)\)
\[\Rightarrow q = l(d') \text{ is satisfactory}\]

B2: \(\text{ID}^q = \{id \mid \text{find}(id, d') \neq \emptyset\}\)
\(= \{id \mid \text{find}(id, \text{rest}(d)) \neq \emptyset\} \cup D(dn')\)
\(= \text{ID}^p, \text{REF} \cup D\)

B3: \(\text{REF}' = \text{REF} \cup \{(id, q) \mid id \in \text{ID}^q\}\)

B4: \(\alpha, \beta \in \text{REF} \Rightarrow 2\text{nd}(\alpha) \leq l(d) \land 2\text{nd}(\beta) \leq l(d)\)
\[\Rightarrow \alpha \preceq \beta \preceq \text{REF}' \preceq \alpha \preceq \beta\]

B5: \(id \not\in \text{ID} \Rightarrow \text{find-p}(id, d') = \text{find-p}(id, \text{rest}(d'))\)
\[\Rightarrow (id, q) \preceq \text{REF}'(id, p)\]

B6: \(id \in D \Rightarrow \text{find-p}(id, d') = (id, q)\)
\((id, i) \in \text{REF} \Rightarrow i \neq q\)
\[\Rightarrow (id, q) \preceq \text{REF}' = \{(id, q)\}\]

B7: term*bloc is an identity with respect to:
\[\text{REF}\]
and thus \(\preceq\)

\(m3, b3, t1\)

\(m4, b4, b5, b6\)
P1: \((id, i) \in \text{REF} \Rightarrow i \leq l(d)\)  
\[
\therefore q = l(d') \text{ is satisfactory}
\]

P2: \(\text{ID}^q = \{id \mid \text{find}(id, d') \neq \emptyset\} = \{id \mid \text{find}(id, \text{rest}(d, r)) \neq \emptyset\} \cup D(dn')\)
\[
\text{ID}^r, \text{REF} \cup P
\]

P3: \(\text{REF}' = \text{REF} \cup \{(id, q) \mid id \in \text{ID}^q\}\)

P4: \(\alpha, \beta \in \text{REF} \Rightarrow 2\text{nd}(\alpha) \leq l(d) \wedge 2\text{nd}(\beta) \leq p\)
\[
\therefore \alpha \preceq \beta, \text{REF}' \preceq \alpha \preceq \text{REF}' \preceq \beta
\]

P5: \(id \notin P \Rightarrow \text{find-p}(id, d') = \text{find-p}(id, \text{rest}(d', r))\)
\[
\therefore (id, q) \preceq \text{REF}'(id, r)
\]

P6: \(id \in P \Rightarrow \text{find-p}(P_i, d') = \text{find-p}(A_i, \text{rest}(d))\)
\[
\therefore (P_i, q) \preceq \text{REF}'(A_i, p)
\]

P7: term-proc is an identity with respect to \(\text{REF}\) and thus \(\preceq\)

The remaining steps of the proof to obtain the same result as in 1/1 consist of:

show \(\text{find}(id, d) = id(s\text{-eo}(d, \text{index}(s(id, d), d)))\) by R.I.
then (L9), Theorem II, (L10), (L11), Theorem III

Notice that the transition made above, of incorporating the denotations in the stack was not formally justified in 1/1.
Discussion

The previous proofs of block implementations (e.g. /1/ and /2/) have taken as the definition an abstract machine giving a, hopefully simple, model. The disadvantages of this approach can be seen:

(a) It is not clear where the division between the essential and inessential properties of the defining model lays. It is possible that this difficulty could be minimized by supplying additional notes.

(b) Proving the correctness (i.e. equivalence) of an implementation often requires that difficult lemmas on the base model are established. Notice that Lemmas 5 and 6 of /1/ occur above in the proof of the base model not in the proof of the implementation. It is possible that this difficulty could be overcome by proving a suitable set of lemmas about the base model and incorporating them with the definition.

(c) If a convenient common range for the results of the definition and implementation functions is not available, the equivalence may be difficult to state. For example, a formal proof of the inclusion of the denotations in the stack in /1/ would not have had the set of unique names as a common range.

This paper is an attempt to go beyond noting what is important in, or what lemmas can be deduced from, the defining model: facts like these replace the model as a definition. Naturally, it is not possible to simply write down these defining properties. Not only is it an easier task to formulate one model but the study of a range of models is the key to finding the essential properites. (In fact the idea to use equivalence relations came to the author when trying to formulate a correctness condition between the first two models of /2/.)
Although the above definition fulfils the required role for a number of models, it is certainly not general enough. In particular, it would be necessary to change the equivalence relation to one references, as in the text, in order to cover the copy rule. At least for this reason the above definition cannot be claimed to be machine-free.

As well as being easier to formulate, and possibly to read, definitions by models have the advantage that when an implementation uses precisely the same algorithm as the definition for some, or all, sub-problems, these points can be ignored by the proof (this is done at several points in /1/). However, the author suspects that to obtain, in general, more straightforward proofs of implementations, the more promising approach is to develop properties which models could be shown to fulfil. To substantiate this suspicion other areas of languages must be investigated.
REFERENCES


/3/ LANDIN, P.: A Correspondence between Algol 60 and Church's Lambda-Notation.— (Part 1) CACM 8 (1956), No. 2, p. 89.

ACKNOWLEDGEMENT

The author is grateful to W. Henhapl and P. Lucas for stimulating discussions. The latter has also pointed out that P. Landin in /3/ uses an equivalence relation to informally define parts of his Sharing Machine for interpretation of IAEs.