**CSEN1082 Visual Programming** **(Due: 02/05/20)**

# FP7R Language Project Proposal

*Instructor:* Dr. Nada Sharaf *Team ID :* Team 16

***Team Members*** **:**

- Omar Osama Hassan, 37-9279.
- Omar Hesham, 37-4450.
- Mohamed Adel Fahim, 37-6669.
- Omar Amr Shaker, 37-2403.
- Hesham Hamdy Morgan, 37-6934.

***FP7R Concept*** **:**

A graph-based visual general programming language to help teach beginners basic programming principles.

***Syntax*** **:**

***Constructs*** **:**

Nodes in the program are connected by arrows, which indicates the program flow. Nodes can be categorized into start, end, conditional, operational, iterative, value, and print nodes. The properties of the valid nodes can be defined as follows:

- Start node indicates the start of the program and has a single outgoing arrow.
- End node indicates the end of the program has a single in-going arrow.
- Exactly one Start and End node are required for a program to be valid.
- Value nodes have an input field and a single outgoing value arrow.
- Conditional nodes have two in-going value arrows holding numerical values, one in-going flow arrow, and two outgoing flow arrows, one in case the condition is true and the other in case it's false.
- Operational nodes have two in-going value arrows holding numerical values and a single outgoing value arrow holding the result of the operation.
- Iterative (repeat) nodes have one in-going flow arrow and one in-going value arrow holding the number of iterations and a single outgoing flow arrow.
- Print nodes have one in-going flow arrow and one in-going value arrow holding the value to be printed and one outgoing flow arrow.

The properties of the valid arrows/connections can be defined as follows:

- Flow arrows indicate the sequential flow of the program.
- Value arrows carry values from one node to the next.

## ***Grammar*** :

### ***Attributes*** :

The attributes of the non-terminals are as follows :

$START[next]$
$END[prev]$
$CONDITIONAL[prev, nextTrue, nextFalse, input1, input2]$
$OPERATIONAL[prev, input1, input2, output]$
$REPEAT[prev, next, input]$
$VALUE[input, output]$
$PRINT[prev, next, input]$

### ***CFG*** :

$$\begin{aligned}
STARTNODE &\rightarrow next(START, NODES)\\
NODES &\rightarrow next(NODENODES)|END\\
NODE &\rightarrow \begin{matrix} CONDITIONALNODE|OPERATIONALNODE| \\ REPEATNODE|VALUENODE|PRINTNODE \end{matrix}\\
CONDITIONALNODE &\rightarrow nextCond(CONDITIONALinput1input2, NODES, NODES)\\
OPERATIONALNODE &\rightarrow next(OPERATIONALinput1input2, NODES)\\
REPEATNODE &\rightarrow next(REPEATinput, next(NODES, ENDREPEAT))\\
VALUENODE &\rightarrow nextIO(VALUEinput, NODES)\\
PRINTNODE &\rightarrow next(PRINTinput, NODES)
\end{aligned}$$

### ***Conditions*** :

The conditions of the grammar rules are stated as follows :
$next(A, B)$, where $A.next == B.prev$.
$nextIO(A, B)$, where $A.output == B.input|A.output == B.input1|A.output == B.input2$.
$nextCond(A, B, C)$, where $A.nextTrue == B.prev\&A.nextFalse == C.prev$

## ***Semantics*** :

### ***State*** :

A *FP7R* state is defined as a tuple $\langle V, R, IR, \mathcal{S}\rangle_n$, where

- $V$ is the set of nodes,
- $R$ is the set of tuples that represents the binary accessibility relation, $V \times V$,
- $IR$ is the set of tuples that represents the binary association relation, $V \times \mathbb{Z}$,
- $S$ is the screen which the data is displayed in.

Note that $V$ is the representation of all nodes present in a given state. A node can be categorized as follows,

- Start node is the node denoting the start of the program,
- End node is the node denoting the end of the program,
- Value nodes are the nodes that represents the values present in the program,
- Conditional nodes are the nodes that represents the conditions that should be satisfied,

- Operational nodes are the nodes that represents the operations that are computed,
- Iterative nodes are the nodes that represents the iterations present in the program,

$n$ is the next free identifier, used to identify new nodes. As for the set $R$, it consists of tuples that represents the accessibility relation, the accessibility relation is a directed relation that represents the edges between the nodes.

***Operations*** **:**

**Add**: $\langle V, R, IR, \mathcal{S}\rangle_n \rightarrow_{Add} \langle \{v\#n\} \uplus V, R, IR, \mathcal{S}\rangle_{n+1}$
given that $v$ is a newly created node, and $v$ can not be either start or end if they already exist in the set $V$.

---

**Delete**: $\langle \{v\#i\} \uplus V, R, IR, \mathcal{S}\rangle_n \rightarrow_{Delete} \langle V, R, IR, \mathcal{S}\rangle_n$
given that $v$ is an existing node in the previous state.

---

**Connect**: $\langle \{v_1\#i, v_2\#j\} \uplus V, R, IR, \mathcal{S}\rangle_n \rightarrow_{Connect} \langle \{v_1\#i, v_2\#j\} \uplus V, \{(v_1\#i, v_2\#j)\} \uplus R, IR, \mathcal{S}\rangle_n$
given that both $v_1, v_2$ are existing nodes, and not accessible from each other in the previous state.

---

***Transitions*** **:**

**Conditional Process**: $\langle \{v_1\#i, v_2\#j\ , v_3\#x\ , v_4\#y\ , v_5\#z\} \uplus V, \{(v_1\#i, v_3\#x), (v_2\#j, v_3\#x), (v_3\#x, v_4\#y), (v_3\#x, v_5\#z)\} \uplus R, IR, \mathcal{S}\rangle_n \rightarrow_{Conditional} \langle \{v_1\#i, v_2\#j\ , v_3\#x\ , v_4\#y\ , v_5\#z\} \uplus V, \{(v_1\#i, v_3\#x), (v_2\#j, v_3\#x), (v_3\#x, v_4\#y), (v_3\#x, v_5\#z)\} \uplus R, \{(v_4\#y, 1), (v_5\#z, 0)\} \uplus IR, \mathcal{S}\rangle_n$

given that $v_1\#i, v_2\#j\ , v_3\#x\ , v_4\#y\ , v_5\#z$ are existing nodes.

---

**Operational Process**: $\langle \{v_1\#i, v_2\#j\ , v_3\#x\ , v_4\#y\} \uplus V, \{(v_1\#i, v_3\#x), (v_2\#j, v_3\#x), (v_3\#x, v_4\#y)\} \uplus R, IR, \mathcal{S}\rangle_n \rightarrow_{Operational} \langle \{v_1\#i, v_2\#j\ , v_3\#x\ , v_4\#y\} \uplus V, \{(v_1\#i, v_3\#x), (v_2\#j, v_3\#x), (v_3\#x, v_4\#y)\} \uplus R, \{(v_4\#y, k)\} \uplus IR, \mathcal{S}\rangle_n$

given that $v_1\#i, v_2\#j\ , v_3\#x\ , v_4\#y$ are existing nodes, and $\{(v_4\#y, k)\}$ is a representation of the association binary relationship between $v_4\#i$ and a value $k$, where $k \in \mathbb{Z}$ .

---

**Iterative Process**: $\langle \{v_1\#i, v_2\#j\ , v_3\#x\} \uplus V, \{(v_1\#i, v_3\#x), (v_2\#j, v_3\#x)\} \uplus R, IR, \mathcal{S}\rangle_n \rightarrow_{Iterative} \langle \{v_1\#i, v_2\#j\ , v_3\#x\} \uplus V, \{(v_1\#i, v_3\#x), (v_2\#j, v_3\#x)\} \uplus R, IR, \mathcal{S}\rangle_n$

given that $v_1\#i, v_2\#j\ , v_3\#x$ are existing nodes.

---

**Display**: $\langle V, R, IR, \mathcal{S}\rangle_n \rightarrow_{Add} \langle V, R, IR, \mathcal{S}'\rangle_{n+1}$

$\mathcal{V} \wedge \mathcal{R} \wedge \mathcal{IR} \wedge \mathcal{S} \iff \mathcal{S}'$ given that the current screen S does not consist of all the elements in the new screen $\mathcal{S}'$

---