

# Appendix A

## Abbreviations, Notations, Symbols

### A.1 Abbreviations

<b><i>αFORTH.</i></b>	Agent Forth (AFL)
<b><i>AI.</i></b>	Artificial Intelligence
<b><i>AoC.</i></b>	Agent-on-Chip (Agent Platform Architecture)
<b><i>AAPL.</i></b>	Activity-Transition Graph based Agent Programming Language
<b><i>AFL.</i></b>	Agent Forth Programming Language
<b><i>AI.</i></b>	Artificial Intelligence
<b><i>AML.</i></b>	Agent Forth Machine Language
<b><i>ANN.</i></b>	Artificial Neural Network
<b><i>APP.</i></b>	Agent Processing Platform
<b><i>ALAP.</i></b>	As Late As Possible
<b><i>ASAP.</i></b>	As Soon As Possible
<b><i>AST.</i></b>	Abstract Syntax Tree
<b><i>ATG.</i></b>	Activity Transition Graph
<b><i>BDI.</i></b>	Belief Desire Intention (Agent Model)
<b><i>CF.</i></b>	Code Frame (Agent Forth Machine)
<b><i>CPS.</i></b>	Cyber Physical System
<b><i>CS.</i></b>	Code Segment (Processor)
<b><i>CSP.</i></b>	Communication Sequential Processes
<b><i>DB.</i></b>	Database
<b><i>DOS.</i></b>	Distributed Operating System

<b>DSN.</b>	Distributed Sensor Network
<b>DT.</b>	Decision Tree
<b>DTL.</b>	Decision Tree Learner
<b>FEM.</b>	Finite Element Model (Mechanics)
<b>FIFO.</b>	First-In First-Out (Data processing order)
<b>FSM.</b>	Finite-State Machine
<b>GUI.</b>	Graphical User Interface
<b>HLS.</b>	High-Level Synthesis
<b>JAM.</b>	JavaScript Agent Machine (platform)
<b>JS.</b>	JavaScript
<b>LIFO.</b>	Last-In First-Out (Data processing order)
<b>LM.</b>	(Structural/Mechanical) Load Monitoring
<b>LUT.</b>	Look-up Table
<b>MAS.</b>	Multi-Agent System
<b>ML.</b>	Machine Learning (AI)
<b>PAVM.</b>	Pipelined Agent Forth Virtual Machine (PAFVM)
<b>PCSP.</b>	Pipelined Communicating Sequential Processes
<b>PID.</b>	Proportional-Integral-Derivative Controller
<b>PN.</b>	Petri Net
<b>PRS.</b>	Procedural-Reasoning-System
<b>PU.</b>	Processing Unit
<b>QoS.</b>	Quality of Service
<b>ROI.</b>	Region Of Interest
<b>RPCSP.</b>	Reconfigurable Pipelined Communicating Processes
<b>RT.</b>	Real Time
<b>RTL.</b>	Register Transfer (Logic) Level
<b>SEM.</b>	Smart Energy Management (with AI)
<b>SHM.</b>	Strutural Health Monitoring
<b>SoC.</b>	System-on-Chip (Hardware Architecture)
<b>SoPC.</b>	System-on-Programmable Chip (Hardware Architecture)
<b>SoMas.</b>	Self-organizing Multi-Agent System
<b>SoS.</b>	Self-organizing System
<b>SQL.</b>	Standard Query Language (Database)
<b>TS.</b>	Tactile Sensing (Application)
<b>TS.</b>	Tuple Space (Communication)
<b>μC.</b>	Microcontroller
<b>VM.</b>	Virtual Machine

## A.2 Symbols

**A.** Set of activities of an ATG

$\mathcal{T}$  Set of transitions of an ATG

## A.3 Notation

### A.3.1 AAPL Short Notation

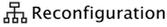
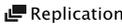
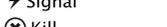
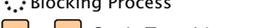
<p><b>Tuple Space</b></p> <p><math>\nabla^+(TP)</math> Add tuple <math>TP</math> to database</p> <p><math>\nabla^-(TP)</math> <math>\nabla^{?}</math>(TMO, <math>TP</math>) Read and remove tuple <math>TP</math> from database (or try it <math>?</math>)</p> <p><math>\nabla^{\%}(TP)</math> <math>\nabla^{? \%}</math>(<math>TP</math>) Read tuple (or try) <math>TP</math> from database</p> <p><math>\nabla^X(TP)</math> Remove tuple <math>TP</math> from database</p> <p><math>\nabla^T(TMO, TP)</math> Add marking tuple <math>TP</math> to database with lifetime <math>TMO</math></p> <p><math>?\nabla(TP)</math> Test for existence of tuple <math>TP</math></p> <p><math>x?</math> Formal Parameter</p> <hr/> <p><b>Mobility</b></p> <p><math>?\Lambda(\omega)</math> <math>?\Lambda(\delta)</math> Test for connection link in direction <math>\omega/\delta</math></p> <p><math>\omega</math> Set of directions {NORTH,SOUTH,...}</p> <p><math>\delta</math> Relative position vector and hop vector relative to root node</p> <p><math>\delta(\omega)</math> Numeric direction-to-position difference vector</p> <p><math>\varpi</math> Opposite direction</p> <p><math>\Leftrightarrow(\omega) \Leftrightarrow(\delta)</math> Migrate to direction <math>\omega / \delta</math></p> <hr/> <p><b>Agents</b></p> <p><math>\Psi AC: (x, y, \dots) \rightarrow \{ \dots \}</math> Define an agent class <math>AC</math> with parameters <math>x, y, \dots</math></p> <p><math>\varphi ac: \{ \dots \}</math> Define a sub-class <math>ac</math></p> <p><math>\Sigma: \{ \dots \}</math> Definition of body variables, <math>\sigma: \{ \dots \}</math> none-persistent</p> <p><math>\alpha A: \{ \dots \}</math> Definition of activity <math>A</math></p> <p><math>F: (x, y, \dots) \rightarrow \{ \dots \}</math> Definition of a function <math>F</math> with parameters <math>x, y, \dots</math></p> <p><math>\zeta S: (P) \rightarrow \{ \dots \}</math> Definition of a signal handler <math>S</math> with parameter <math>P</math></p> <p><math>\Pi: \{ \dots \}</math> Definition of transitions, <math>\pi: \{ \dots \}</math> : Define sub-class transitions</p> <p><math>\downarrow: \{ x, y, \dots \}</math> Subclass Import of variables, activities,...</p> <p><math>\Theta^+(v1, v2, \dots)</math> Fork agent with arguments</p> <p><math>\Theta^X(A)</math> Destroy agent <math>A</math></p> <p><math>\Theta^+AC(v1, v2, \dots)</math> Create new agent from agent class <math>AC</math> with arguments</p> <hr/> <p><b>Timer</b></p> <p><math>\tau^+(TMO, S)</math> Add timer with timeout <math>TMO</math> and signal <math>S</math></p> <p><math>\tau^-(S)</math> Remove timer for signal <math>S</math></p> <hr/> <p><b>Reconfiguration</b></p> <p><math>\pi^+(T_1, T_2, C)</math> Add transition <math>T_1 \rightarrow T_2</math> with condition <math>C</math></p> <p><math>\pi^*(T_1, T_2, C)</math> Update transitions <math>T_1 \rightarrow T_2</math> with condition <math>C</math></p> <p><math>\pi^-(T_1, T_2)</math> Remove transitions <math>T_1 \rightarrow T_2</math></p> <p><math>\alpha^+(A)</math> Add activity <math>A</math></p> <p><math>\alpha^-(A)</math> Remove activity <math>A</math></p> <hr/> <p><b>Signals</b></p> <p><math>\zeta S(V) \Rightarrow A</math> Send a signal <math>S</math> with argument <math>V</math> to agent <math>A</math></p> <hr/> <p><b>Values</b></p> <p><math>\\$V</math> Agent reference variable (<math>V=self, parent, \dots</math>)</p> <p><math>\mathfrak{R}\{SET\}</math> Random element selection from a set or in the range {min .. max}</p> <p><math>x \leftarrow</math> Change value of variable <math>x</math></p> <p><math>R = (x, y, \dots)</math> Definition of a record tuple <math>R</math> with elements <math>x, y, \dots</math></p>	<p><b>Symbols</b></p> <p> Timer</p> <p> Reconfiguration</p> <p> Replication</p> <p> Move</p> <p> Tuple Database</p> <p> Signal</p> <p> Kill</p> <p> Blocking Process</p> <p> Static Transition</p> <p> Dynamic Transition</p> <hr/> <p><b>Set Iteration</b></p> <p><math>\forall \{ x \in X \mid c(x) \}</math> do <math>I</math></p> <p><math>\forall \{ x \mid c(x) \}</math> do <math>I</math></p> <p>Repeat the following statement <math>I(x)</math> (using <math>x</math>) for each element <math>x</math> of the set <math>X</math> for which the condition <math>c(x)</math> is satisfied.</p> <hr/> <p><b>Interval set Iteration</b></p> <p><math>\forall x \in \{ a \dots b \}</math> do <math>I</math></p> <p><math>\forall \{ x \mid x \in \{ a \dots b \} \}</math> do <math>I</math></p> <p>Repeat the following statement <math>I(x)</math> (using <math>x</math>) for each element <math>x</math> of the interval set <math>\{ a \dots b \}</math></p>
--	---

Fig. A.1

AAPL Short Notation and symbols related to the dynamic ATG and Agent Interaction model

## A.3 Notation

## A.3.2 Rules

**Synthesis Transformation Rule**

A pattern A (the prefix) is transformed in pattern B (the transformation outcome).

$$\frac{\text{Pattern } A}{\text{Pattern } B}$$

**Reduction Rule**

A rule A (the prefix) enables a transition to rule b (the transition outcome).

$$\frac{\text{Rule } A}{\text{Rule } B}$$

