Appendix: Reference Guide

A.1 EBNF for the Agent Language

```
agent             → init_bels init_goals plans
init_bels         → beliefs rules
beliefs           → ( literal "." )*
rules             → ( literal ":-" log_expr "." )*
init_goals        → ( "!" literal "." )*
plans             → ( plan )*
plan              → [ "@" atomic_formula ] triggering_event
                     [ ":" context ]
                     [ "<-" body ] "."
triggering_event  → ( "+" | "-" ) [ "!" | "?" ] literal
literal           → [ "~" ] atomic_formula
context           → log_expr  |  "true"
log_expr          → simple_log_expr
                  |  "not" log_expr
                  |  log_expr "&" log_expr
                  |  log_expr "|" log_expr
                  |  "(" log_expr ")"
simple_log_expr   → ( literal | rel_expr | <VAR> )
body              → body_formula ( ";" body_formula )*
                  |  "true"
body_formula      → ( "!" | "!!" | "?" | "+" | "-" | "-+" ) literal
                  |  atomic_formula
                  |  <VAR>
                  |  rel_expr
atomic_formula    → ( <ATOM> | <VAR> )
                     [ "(" list_of_terms ")" ]
                     [ "[" list_of_terms "]" ]
list_of_terms     → term ( "," term )*
term              → literal
                  |  list
                  |  arithm_expr
                  |  <VAR>
                  |  <STRING>
```

```
list          → "[" [ term ( "," term )* [ "|" ( list | <VAR> ) ] ] "]"

rel_expr      → rel_term
                [ ("<"|"<="|">"|">="|"=="|"\=="|"="|"=..") rel_term ]+
rel_term      → (literal|arithm_expr)
arithm_expr   → arithm_term
                [ ( "+" | "-" | "*" | "**" | "/" | "div" | "mod" )
                  arithm_term ]*
arithm_term   → <NUMBER>
              | <VAR>
              | "-" arithm_term
              | "(" arithm_expr ")"
```

It is worth noting that this grammar has been slightly simplified to make it more readable; the actual grammar used to generate the parser used by ***Jason*** is available in the distribution (file `doc/AS2JavaParser.html`). On particular thing that this grammar does not reflect is that initial beliefs, initial goals and plans do not necessarily need to appear in this order, *even though it is very much recommended that this order is kept*, at least within an AgentSpeak file. One situation, for example, where these constructs will unavoidably appear interleaved in the final AgentSpeak code for an agent is when various AgentSpeak files are included in an agent program and each contributes to the initial beliefs, initial goals and plan library.

A.2 EBNF for the Multi-Agent Systems Language

```
mas           → "MAS" <ID> "{"
                   [ infrastructure ]
                   [ environment ]
                   [ exec_control ]
                   agents
                "}"
infrastructure → "infrastructure" ":" <ID>
environment   → "environment" ":" <ID> [ "at" <ID> ]
exec_control  → "executionControl" ":" <ID> [ "at" <ID> ]
agents        → "agents" ":" ( agent )+
agent         → <ASID>
                [ filename ]
                [ options ]
                [ "agentArchClass" <ID> ]
                [ "beliefBaseClass" <ID> ]
                [ "agentClass" <ID> ]
                [ "#" <NUMBER> ]
                [ "at" <ID> ]
                ";"
filename      → [ <PATH> ] <ID>
options       → "[" option ( "," option )* "]"
option        → "events" "=" ( "discard" | "requeue" | "retrieve" )
              | "intBels" "=" ( "sameFocus" | "newFocus" )
              | "nrcbp" "=" <NUMBER>
```