Grammar for unlooped unsubstituted SSL

An **unlooped substituted SSL script** is a list of **blocks.**

A **block** is

- A **definition block**
- An **SSL-BAF block**
- An **SSL block**

Definition blocks define SSL actions and don't generate any BAF output. SSL-BAF blocks are basically the same as BAF blocks in ordinary scripting and basically generate themselves (with a couple of augmentations). SSL blocks generate large numbers of closely related BAF blocks as output.

A **definition block** is

```
BEGIN_ACTION_DEFINITION
    Name(string)
    TRIGGER
        Definition trigger list
    ACTION
        Definition action block list
END
```

Each definition block defines an Action, which can be called by SSL blocks and provides the template for the BAF blocks it generates. The value of Name() is the name of the Action thus defined, and is used to call that action.

A **definition trigger** is a BAF trigger, except that scstarget can be used in place of any object, and scsargumentN (for any value of N) can be used in place of any variable (so that HPPercentLT(scstarget,scsargument2) is a BAF trigger, for instance).[1] The scstarget variable is set by whatever the target is for the relevant BAF block being generated (this is determined by the Target, TargetBlock, and ConditiionalTargetBlock commands in the SSL block calling the action). The scsargument variables are set by the Action() call in the SSL block. If not defined, they are just entered as text, normally causing the BAF file output to be malformed.

A **definition action block** is

```
RESPONSE #probability
  Definition action list
```

[1] Actually, as far as SSL is concerned a definition trigger is just a string; SSL doesn't care if it's correctly-formatted BAF. But the result will only be a BAF file if these definitions are adhered to (and not always then, if the arguments haven't been properly defined). This is true for any place in the grammar where BAF scripting is called for.

A **probability** is

- A positive integer
- “scsprob1”
- “scsprob2”

The variables scsprob1 and scsprob2 are set when the Action is called. If they are not defined by the action call, they are set to 100 and 0 respectively.

A **definition action** is a BAF action, except that scstarget and scsargumentN are allowed as arguments. See **definition trigger**s for more information.

An **SSL-BAF block** is

```
IF
        SSL-BAF trigger list
THEN
        SSL-BAF action block list
END
```

An **SSL-BAF trigger** is

- An **IgnoreBlock()** command
- A **RequireBlock()** command
- A standardly-formatted BAF trigger line

A **IgnoreBlock()** command is IgnoreBlock(**string**1|**string**2| ...)

The strings are the arguments of the command. If any of them are the string “True” (case-sensitive, I think), as in “IgnoreBlock(True|Refinements)”, then the whole SSL-BAF block is skipped and produces no output. Otherwise, the IgnoreBlock is removed from the trigger list before the block is outputted.

A **RequireBlock()** command is RequireBlock(**string**1|**string**2|...)

The strings are the arguments of the command. If none of them are the string “True”, then the whole SSL-BAF block is skipped and produces no output. Otherwise, the RequireBlock is removes from the trigger list before the block is outputted.

An **SSL-BAF action block** is

```
RESPONSE #positive_integer
        BAF action list
```

A **BAF action** is a standardly-formatted BAF action line.

An **SSL block** is

```
IF TRIGGER
        concatenated SSL trigger list
THEN DO
        SSL action list
END
```

A **concatenated SSL trigger** is **SSL trigger**&**SSL trigger**&...

(Put another way, SSL triggers can be separated either by spaces or by "&". This is to help with substitutions.)

An SSL trigger is

- An **augmented BAF trigger**
- An **IgnoreBlock()** or **RequireBlock()** command, format as above
- A **Target()** command
- A **TargetBlock()** command
- A **ConditionalTargetBlock()** command
- A **TriggerBlock()** command

An **augmented BAF trigger** is a standardly-formatted BAF trigger, except that scstarget can be used as an object
Example: CheckStatGT(scstarget,10,Level)

A **Target()** command is Target(**object**1|**object**2|...), where **object**N is a legally-formed BAF object .
Example: Target(Player1|Player2|NearestEnemyOf(Myself))

A **TargetBlock()** command is TargetBlock**(targetlist**1|**targetlist**2|...), where **targetlist**N is a string. Each **targetlist**N must be the name of a target list in a previously-loaded library file, or the SSL run will fail.
Example: TargetBlock(PCsPreferringWeak|EnemiesInOrder)

A **ConditionalTargetBlock()** command is TargetBlock(**targetlist**1|**targetlist**2|...;**concatenated augmented BAF trigger**), where **targetlist**N is as above and a **concatenated augmented BAF trigger** is **augmented BAF trigger**1&**augmented BAF trigger**2& ...
Example:ConditionalTargetBlock(PCsPreferringWeak;HPPercentLT(scstarget,40))

A **TriggerBlock()** command is TriggerBlock**(triggerlist**1|**triggerlist**2|...), where **triggerlist**N is a string. Each **triggerlist**N must be the name of a trigger list in a previously-loaded library file, or the SSL run will fail.

An **SSL action** is

- An **augmented BAF action**
- An **Action()** command
- An **ActionCondition()** command
- An **OnContinue()** command
- Combine()

An **augmented BAF action** is a standardly-formatted BAF trigger, except that scstarget can be used as an object
Example: Attack(scstarget)

An **Action()** command is

- Action(actionname,scsargument1,scsargument2,...scsargumentN)
- Action(actionname,scsargument1,scsargument2,...scsargumentN|scsprob1)
- Action(actionname,scsargument1,scsargument2,...scsargumentN|scsprob1|scsprob2)

In each case, actionname and scsargumentN are strings; scsprob1 and scsprob2 are positive integers. Unless actionname is the name of a previously-defined Action (or "Literal") the SSL run fails. Each of argument1, ... argumentN and prob1, prob2 is used to set the variables in the action. Note that before doing so, the backslash \ in each argument is replaced by a comma. (This is to allow BAF actions to be used as arguments; yes, it's a kludge.)

An **ActionCondition()** command is

- ActionCondition(actionname,scsargument1, ... scsargumentN;**concatenated augmented BAF trigger)**
- The same, but with scsprob1 or scsprob1|scsprob2, as above for Action().

An **OnContinue()** command is OnContinue(**concatenated augmented BAF action**). A **concatenated augmented BAF action** is a sequence of **augmented BAF actions** separated by "&".