**On page 106, remove the following sentence:**

Symbols may contain spaces.

**On page 94, add the following rules to the S-FEEL grammar:**

63. white space = vertical space | \u0009 | \u0020 | \u0085 | \u00A0 | \u1680 | \u180E | [\u2000-\u200B] | \u2028 | \u2029 | \u202F | \u205F | \u3000 | \uFEFF ;

64. vertical space = [\u000A-\u000D]

**On page 109, add the following rules to the FEEL grammar:**

63. white space = vertical space | \u0009 | \u0020 | \u0085 | \u00A0 | \u1680 | \u180E | [\u2000-\u200B] | \u2028 | \u2029 | \u202F | \u205F | \u3000 | \uFEFF ;

64. vertical space = [\u000A-\u000D]

**On page 109, Additional syntax rules, DELETE the following 3 bullet items:**

* A name may contain spaces but may not contain a sequence of 2 or more spaces.
* A name start (grammar rule 28) SHALL NOT be a language keyword. (Language keywords are enclosed in double quotes in the grammar rules, for example, "and", "or", "true", "false".)
* A name part (grammar rule 29) MAY be a language keyword.

**On page 109, BEFORE 10.3.1.3, INSERT new 10.3.1.3, renumbering as needed**

**10.3.1.3 Tokens, Names, and White space**

A FEEL expression consists of a sequence of tokens, possibly separated with white space (grammar rule 63). A token is a sequence of Unicode characters, either

* A literal terminal symbol in any grammar rule other than grammar rule 30. Literal terminal symbols are enclosed in double quotes in the grammar rules, *e.g.* “and”, “+”, “=”, or
* a sequence conforming to grammar rule 28, 29, 35, or 37

White space (except inside strings) acts as token separators. Most grammar rules act on tokens, and thus ignore white space (which is not a token).

A name (grammar rule 27) is defined as a sequence of tokens. I.e. the name Income Taxes Amount is defined as the list of tokens **[ Income, Taxes, Amount ]**. The name Income+Expenses is defined as the list of tokens **[ Income, + , Expenses ]**. A consequence of this is that a name like Phone Number with one space in between the tokens is the same as Phone Number with several spaces in between the tokens.

A name start (grammar rule 28) SHALL NOT be a literal terminal symbol.

A name part (grammar rule 29) MAY be a literal terminal symbol.

**REPLACE section 10.3.1.5 Ambiguity WITH the following new section:**

**10.3.1.6 Ambiguity**

Because names are a sequence of tokens, and some of those tokens can be FEEL operators and keywords, context is required to resolve ambiguity. For example, the following could be names or other expressions:

* a-b
* a - b
* what if?
* Profit and loss

Ambiguity is resolved using the scope. Name tokens are matched from left to right against the names in-scope, and the longest match is preferred. In the case where the longest match is not desired, parenthesis or other punctuation (that is not allowed in a

name) can be used to disambiguate a FEEL expression. For example, to subtract b from a if *a-b* is the name of an in-scope context entry, one could write *(a)-(b).* Notice that it does not help to write *a - b*,using space to separate the tokens, because the space is not part of the token sequence and thus not part of the name.