**SEMANTICS**

Semantics is the study of meaning expressed by elements of a language, characterizable as a symbolic system.

The spreadsheet is a client, JavaScript web application working with a server application. Entries in the spreadsheet are analysed by semantic parsers on the server, and the representation and recognition of spreadsheet semantics are detailed.

Semantic spreadsheets will work differently. There are three key points to understand. First, because the coordinates of the information in a semantic spreadsheet are based on meaning rather than the presentation coordinates, linking is far less brittle.  The linking of information is driven by the information's [semantics](http://xbrl.squarespace.com/journal/2010/6/1/differentiating-syntax-and-semantics.html)(meaning). The meaning does not change.  For example, "total Sales for fiscal year 2012 for the consolidated group" is always the same and the details of that total is likewise always the same.  You may have different aspects or breakdowns of that total, but the breakdown will always add up to the total.

Second, the semantics are enforced by [business rules](http://xbrl.squarespace.com/journal/2009/10/18/business-rules-what-are-they.html) which could be included within the spreadsheet or  seperate from the semantic spreadsheet and therefore usable across many spreadsheets or all spreadsheets throughout an organization.  This both makes the bindings between spreadsheets stronger and improves information quality because one set of business rules common to any spreadsheet can be used to make sure the information is always correct.

**LEXEME**

*A lexeme is a sequence of characters in the source program that matches the pattern for a token and is identified by the lexical analyser as an instance of that token. In general, there is a set of string in the input for which the same token is produced as output. This set of strings is described by a rule called a pattern associated with the token. In a summarized way, we can say “Lexeme is matched against pattern to generate token.”*

The notion of lexeme is frequently used as a technical and less ambiguous equivalent of

the notion of word. It has the advantage of allowing the integration in the lexicon of

many lexical units which are not words in the ordinary sense (for instance because they

are formed of smaller units which are also words). Lexemes may thus be a cover term so

as to refer to all the semantic units stored in the lexicon, and may thus refer to lexical

units (e.g. milk), infra-lexical units (e.g. units as micro-, affixes – if we admit they have

a signification – and bound bases) and supra-lexical units which behave like syntactic

atoms (e.g. pomme de terre – potato, litterally apple of ground – or phrases like tout à

fait, lexicalised idioms).

lexemes which minimally are morpheme/construction pairs and are associated with

lexicalised meanings (i.e. senses),

**TOKEN**

*A token is a pair consisting of a token name and an optional attribute value. The token name is an abstract symbol representing a kind of lexical unit. E.g., a particular keyword, or a sequence of input characters denoting an identifier. The token names are input symbols that the parser processes. We often refer to a token by its token name.*

**LANGUAGE RECOGNIZER**

Intellexer Language Recognizer [identifies the language](http://demo.intellexer.com/) and character encoding of incoming documents. It supports more than 30 languages, covering major European and Asian languages.

Intellexer Language Recognizer can be successfully used:

- as a pre-filtering step to improve the quality of input text data (because of most natural processing algorithms deal with monolingual texts and inclusion of other languages can decrease the performance of document management systems);

- in mining bilingual texts for machine translation from online resources;

- for retrieval, grouping and understanding relevant information (user’s texts, emails and etc.) in multilingual environment.

Intellexer Language Recognizer combines statistic and linguistic technologies in order to obtain the highest recognition results. Our language detection algorithm is based on strong mathematical model of vector spacing algorithm. It creates multidimensional space of vectors scanning document contests and uses N-grams notion for calculating their frequencies. Afterwards the algorithm analyzes the positions of the necessary vectors in space to determine their similarity. Besides, for correction of the statistical algorithm results, we use special linguistic rules developed by our experts.

Intellexer Language Recognizer accurately determines not only the language of the whole document, but also the language of each text fragment.

*This recognition devices are like filters separating correct sentences from those that are incorrectly. A recognizer is used in the syntax analysis part of the compiler. The syntax analyser determines whether the given programs are syntactically correct.*

**LANGUAGE GENERATOR**

*Is a device that can be used to generate the sentences of a language. It generates unpredictable sentences which makes a generator seems to be a device of limited usefulness as a language descriptor. However, people prefer certain forms of generators over recognizer because they can be more easily read.*

References:

semantics

<http://udel.edu/~dlarsen/ling101/slides/Semantics.pdf>

<https://www.um.edu.mt/__data/assets/pdf_file/0014/137111/Week_1_Lecture.pdf>

tokens

<https://www.denovosoftware.com/site/manual/spreadsheet_token.htm>

lexeme

<https://www.scribd.com/doc/45799053/Lexeme-Generator-5th-Sem-2009-REPORT>

<https://www.researchgate.net/publication/228597765_Morphemes_and_Lexemes_versus_Morphemes_or_Lexemes\>

language recognizer and generator

<https://www.intellexer.com/language_recognizer.html>

<https://www.youtube.com/watch?v=ZctI5HzfY0A>