US 20150248379A1

(19) **United States**

(12) **Patent Application Publication**

(10) **Pub. No.: US 2015/0248379 A1**

(43) **Pub. Date: Sep. 3, 2015**

Medlock et al.

(54) **FORMATTING MODULE, SYSTEM AND METHOD FOR FORMATTING AN ELECTRONIC CHARACTER SEQUENCE**

(71) Applicant: **TOUCHTYPE LIMITED**, London (GB)

(72) Inventors: **Benjamin Medlock**, London (GB); **David Martinez Del Corral**, Kent (GB)

(21) Appl. No.: **14/428,972**

(22) PCT Filed: **Sep. 18, 2013**

(86) PCT No.: **PCT/GB2013/052443**

§ 371 (c)(1),
(2) Date: **Mar. 17, 2015**

(30) **Foreign Application Priority Data**

Sep. 18, 2012 (GB) 1216640.1

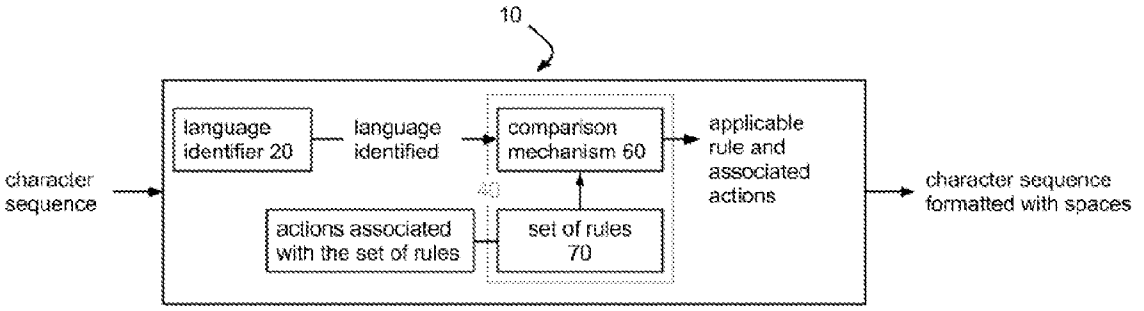
Publication Classification

(51) **Int. Cl.**
G06F 17/22 (2006.01)
G06F 17/30 (2006.01)

(52) **U.S. Cl.**
CPC ***G06F 17/22*** (2013.01); ***G06F 17/30507*** (2013.01)

(57) **ABSTRACT**

There is provided a formatting module configured to format spaces in an electronic character sequence. The formatting module supports at least one language and comprises a language identifier configured to identify whether the electronic character sequence is written in a supported language, and a character identifier configured to identify a particular character or a particular sequence of characters in the electronic character sequence. The formatting module is configured to format spaces in the electronic character sequence on the basis of the language identified and the particular character identified or the particular sequence of characters identified, when a supported language is identified. A system and method for formatting text are also provided.



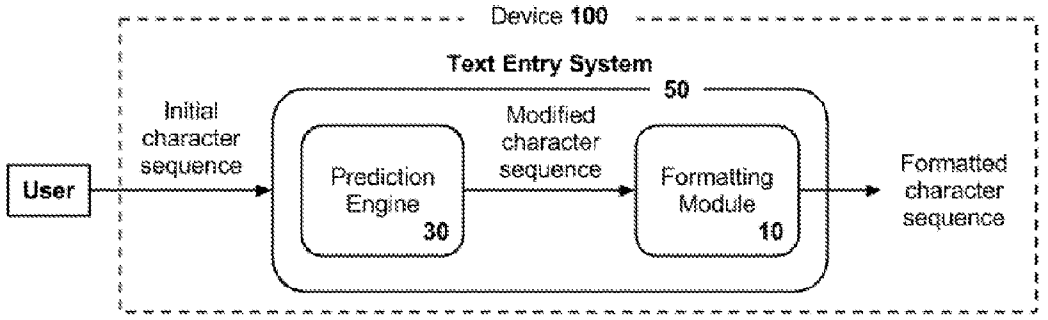


Fig. 1

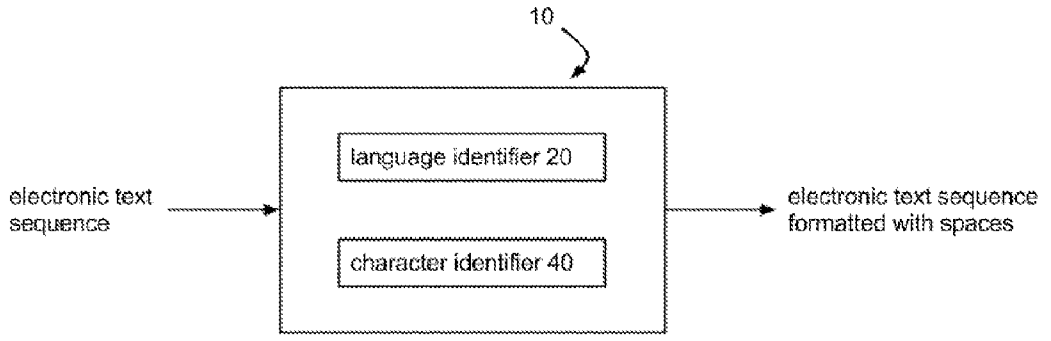


Fig. 2

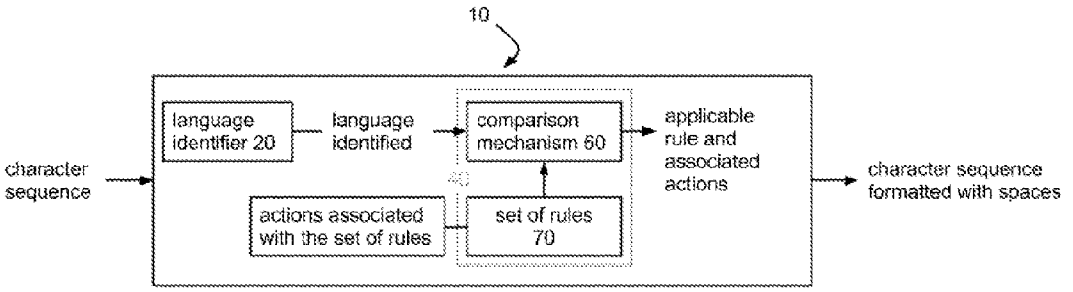


Fig. 3

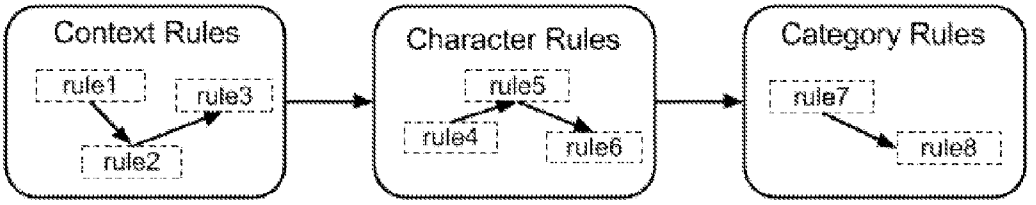

Context Rules
rule1
rule3
rule2
Character Rules
rule5
rule4
rule6
Category Rules
rule7
rule8

Fig. 4

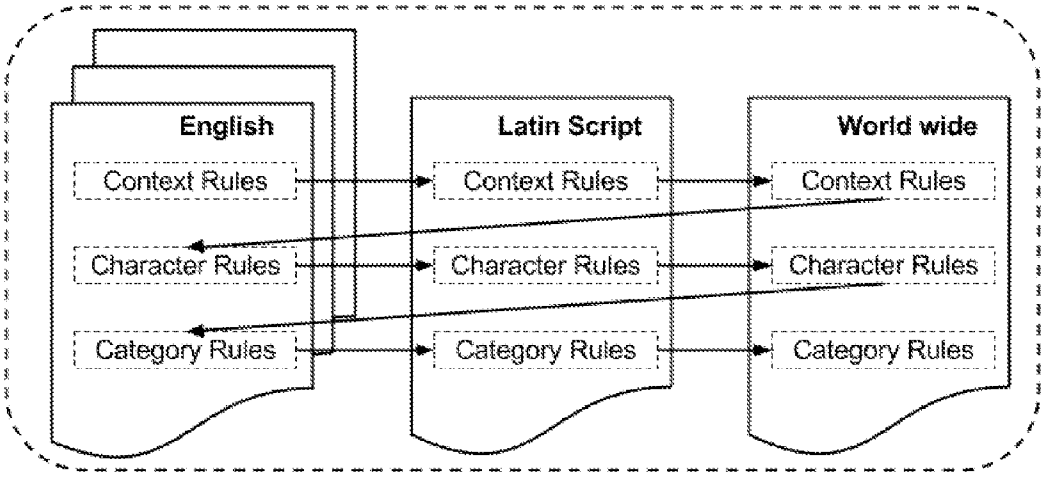

English
Context Rules
Character Rules
Category Rules
Latin Script
Context Rules
Character Rules
Category Rules
World wide
Context Rules
Character Rules
Category Rules

Fig. 5

FORMATTING MODULE, SYSTEM AND METHOD FOR FORMATTING AN ELECTRONIC CHARACTER SEQUENCE

FIELD OF THE INVENTION

[0001] The present invention relates to the formatting of spaces in an electronic character sequence. In particular, it relates to a formatting module, system and method for formatting spaces in an electronic character sequence.

BACKGROUND

[0002] Punctuation marks are symbols that indicate the structure and organization of written language, as well as intonation and pauses to be observed when reading aloud. The appearance and usage of punctuation marks varies between languages and scripts but in most cases they are vital to disambiguate the meaning of sentences. The use and interpretation of punctuation marks can be heavily context-dependent. For example, a full stop "." can be used as sentence-ending punctuation, an abbreviation indicator, a decimal point, and so on. Punctuation is also present in mathematical and scientific formulae.

[0003] Some punctuators appear in pairs and one cannot exist without the other. For example, left parenthesis '(' and right parenthesis ')'. However, in some scenarios a single character is used to represent two punctuators, creating ambiguity, for example in the case of the single quote mark: '.

[0004] A space is a blank area, often used to separate words, letters, numbers, and punctuation. Conventions for the formatting of spaces vary among languages. For example, the correct formatting of spaces around a question mark "?" in English is "word?", with no space between the word and the question mark, and a space following the question mark. However, in French the convention is "word ?", where a space is inserted either side of the question mark.

[0005] A number of current-market text input systems exhibit some form of space formatting. For example, when a user enters one of the following characters [? ! : ; , .] after entering a space, the Android default keyboard formats spaces either side of the punctuation mark by removing the leading space and adding a trailing space, irrespective of the language in which the text is being entered.

[0006] It is an object of the present invention to provide a means for formatting automatically the spaces in an electronic character sequence, such that a user can concentrate on the content of a message without worrying about whether the spaces are correctly formatted in the electronic character sequence. It is also an object of the invention to provide a means for correctly formatting spaces in an electronic character sequence on the basis of the conventions of the language in which the electronic character sequence is written.

SUMMARY OF THE INVENTION

[0007] In a first aspect of the present invention, there is provided a formatting module supporting at least one language and configured to format spaces in an electronic character sequence written in a supported language, the formatting module comprising:

[0008] a language identifier configured to identify whether the electronic character sequence is written in a supported language;

[0009] a character identifier configured to identify a particular character or a particular sequence of characters in the electronic character sequence;

[0010] wherein the formatting module is configured to format spaces in the electronic character sequence on the basis of the language identified and the particular character or sequence of characters identified, when a supported language is identified.

[0011] Preferably, formatting spaces in the electronic character sequence comprises inserting and/or deleting spaces in the electronic character sequence.

[0012] In a preferred embodiment, the character identifier comprises:

[0013] at least one set of rules, each rule relating to a particular character or sequence of characters to be identified in the electronic character sequence; and

[0014] a comparison mechanism configured to compare each rule of one of the at least one set of rules to the electronic character sequence to identify whether a rule is applicable;

[0015] wherein each rule is associated with one or more actions which describe the format of spaces to be applied by the formatting module to the electronic character sequence given a supported language and the particular character or sequence of characters; and

[0016] wherein the formatting module is configured to format spaces in the electronic character sequence by applying the one or more actions associated with the applicable rule to the electronic character sequence.

[0017] The comparison mechanism is preferably configured to compare each rule of one of the at least one set of rules to the electronic character sequence only when a supported language is identified.

[0018] Preferably, the formatting module supports a plurality of languages and the language identifier is configured further to identify the most likely language of the supported languages that the electronic character sequence is written in.

[0019] The character identifier may be configured to identify a punctuation mark and the formatting module may be configured to format the spaces either side of the punctuation mark on the basis of the punctuation mark.

[0020] The character identifier may be configured to identify a particular context in the electronic character sequence and the formatting module may be configured to format the spaces in the electronic character sequence on the basis of the context.

[0021] The character identifier may be configured to identify a punctuation mark in the electronic character sequence, and the formatting module may be configured to format the spaces either side of the punctuation mark on the basis of the category of punctuation mark.

[0022] The one or more actions may comprise a sequence of actions, wherein when a rule is found to be applicable, the comparison mechanism is configured to apply the sequence of actions to the electronic character sequence.

[0023] When the formatting module is configured to support a plurality of languages, the character identifier preferably comprises a plurality of sets of rules, one set of rules for each language that is supported, where the comparison mechanism is configured to compare each rule of the set of rules that corresponds to the most likely language to the electronic character sequence.

[0024] The formatting module may comprise sets of rules relating to each language, each family of languages, and all

languages in the world, wherein the rules are applied in a hierarchal structure such that, once a supported language has been identified, the comparison mechanism first compares each rule from the set of rules specific to that language, followed by each rule from the set of rules applicable to the family of languages to which that language belongs, followed by each rule of the set of rules which are applicable to all languages until an applicable rule is identified or no applicable rule is identified and all rules are exhausted.

[0025] The comparison mechanism is preferably configured to compare the rules in a specific predetermined order. The set of rules preferably comprises context rules, character rules and category rules and the comparison mechanism is preferably configured to compare the rules in the following order until an applicable rule is identified or no applicable rule is identified and all rules are exhausted: context rules, character rules, and then category rules.

[0026] In a second aspect of the invention there is provided a formatting module supporting at least one language and configured to format spaces in an electronic character sequence, the formatting module comprising:

[0027] a punctuation mark identifier configured to identify a punctuation mark in the electronic character sequence;

[0028] wherein the formatting module is configured to format spaces in the electronic character sequence on the basis of the language in which the electronic character sequence is written, the punctuation mark identified, and a context of the punctuation mark, when a supported language is identified,

[0029] In a third aspect of the invention there is provided a system for inputting text into an electronic device comprising:

[0030] a text prediction engine configured to receive an electronic character sequence as input and configured to generate and output a corrected electronic character sequence; and

[0031] a formatting module as described above, wherein the formatting module is configured to receive the modified electronic character sequence as input, and to generate a formatted character sequence by formatting spaces in the modified electronic character sequence when a supported language is identified.

[0032] In a fourth aspect of the invention there is provided a system for inputting text into an electronic device comprising:

[0033] a text prediction engine configured to receive an electronic character sequence as input, the text prediction engine comprising:

[0034] a language identifier configured to identify which language the electronic character sequence is most likely written in, and to correct the electronic character sequence on the basis of the identified language;

[0035] wherein the text prediction engine is configured to generate and output a corrected electronic character sequence and to output the language identified;

[0036] a formatting module supporting at least one language and configured to receive the language identified and the corrected electronic character sequence, and configured to format spaces in the electronic character sequence when the identified language is supported, the formatting module comprising:

[0037] a character identifier configured to identify a particular character or a particular sequence of characters in the electronic character sequence;

[0038] wherein, the formatting module is configured to format spaces in the electronic character sequence on the basis of the language identified and the particular character or the particular sequence of characters identified.

[0039] In a fifth aspect of the invention there is provided a method of formatting, with a formatting module supporting at least one language and having a character identifier, spaces in an electronic character sequence, the method comprising:

[0040] identifying whether the electronic character sequence is written in a language supported by the formatting module;

[0041] identifying, with the character identifier, a particular character or a particular sequence of characters in the electronic character sequence;

[0042] formatting, with the formatting module, spaces in the electronic character sequence on the basis of the language identified and the particular character or sequence of characters identified, when a supported language is identified.

[0043] The formatting module may comprise a language identifier to identify whether the electronic character sequence is written in a language supported by the formatting module. Preferably, the formatting module supports a plurality of languages and the method further comprises identifying with the language identifier the most likely language of the electronic character sequence.

[0044] The most likely language of the electronic character sequence may be identified by a text prediction engine, where the method further comprises transmitting the most likely language to the formatting module which identifies whether the most likely language is supported by the formatting module.

[0045] The language identifier preferably comprises at least one set of rules and a comparison mechanism, each rule defining the formatting of spaces in the electronic character sequence, wherein the method further comprises:

[0046] comparing, with the comparison mechanism, each rule of one of the at least one set of rules to the electronic character sequence to identify whether a rule is applicable to the character sequence;

[0047] identifying, with the comparison mechanism, that a particular rule is applicable to the character sequence; and

[0048] applying the applicable rule to the electronic character sequence to format the spaces in the electronic character sequence.

[0049] Preferably, the comparison mechanism compares each rule of one of the at least one set of rules to the electronic character sequence only when a supported language is identified.

[0050] Each rule may relate to a particular character or sequence of characters to be identified and each rule is associated with one or more actions which describe the format of spaces to be applied by the formatting module to the electronic character sequence given a supported language and the particular character or sequence of characters. In the method, the step of applying the applicable rule preferably comprises applying the one or more actions associated with that applicable rule to the electronic character sequence.

[0051] Identifying a particular character may comprise identifying a punctuation mark and formatting the spaces in the electronic character sequence may comprise formatting the spaces either side of the punctuation mark on the basis of the form of the punctuation mark.

[0052] Identifying a particular sequence of characters may comprise identifying a particular context and formatting the spaces in the electronic character sequence may comprise formatting the spaces on the basis of the context.

[0053] Identifying a particular character may comprise identifying a punctuation mark and formatting the spaces in the electronic character sequence may comprise formatting the spaces either side of the punctuation mark on the basis of the category of punctuation mark.

[0054] Where each rule is associated with one or more actions, the one or more actions may comprise a sequence of actions, wherein the sequence of actions is applied sequentially to the electronic character sequence.

[0055] Where the formatting module supports a plurality of languages, the language identifier may comprise a plurality of sets of rules, one set of rules for each language supported, and comparing each rule to the electronic character sequence comprises comparing each rule of the set of rules that corresponds to the most likely language.

[0056] The formatting module may comprise sets of rules relating to each supported language, each family of languages, and all languages in the world, and the method comprises applying the rules in a hierarchal structure such that, once a language has been identified, the comparison mechanism first compares each rule from the set of rules specific to that language, followed by each rule from the set of rules applicable to the family of languages to which that language belongs, followed by each rule of the set of rules which are applicable to all languages until an applicable rule is identified or no applicable rule is identified and all rules are exhausted.

[0057] The comparison mechanism preferably compares the rules in a specific predetermined order.

[0058] The set of rules may comprise context rules, character rules and category rules, and the method preferably comprises comparing the rules in the following order until an applicable rule is identified or no applicable rule is identified and all rules are exhausted: context rules, character rules, and then category rules.

[0059] In a sixth aspect of the invention there is provided a computer program product comprising a computer readable medium having stored thereon computer program means for causing a processor to carry out a method as described above.

BRIEF DESCRIPTION OF THE DRAWINGS

[0060] The present invention will now be described in detail with reference to the accompanying drawings, in which:

[0061] FIG. **1** is a schematic of a system comprising a prediction engine and a formatting module in accordance with the present invention;

[0062] FIG. **2** is a schematic of a formatting module in accordance with the present invention;

[0063] FIG. **3** is a schematic of the formatting module of FIG. **2** shown in greater detail;

[0064] FIG. **4** is an illustration of a structure of specific types of rules within a set of rules for a given language, and shows the order in which a comparison mechanism compares the rules, in accordance with the present invention;

[0065] FIG. **5** is an illustration of how the rules are structured for the English language and the order in which the comparison mechanism compares the rules, in accordance with the present invention.

DETAILED DESCRIPTION

[0066] The present invention provides a formatting module that is configured to format the spaces for a particular sentence on the basis of the conventions for the language in which the sentence is written. The formatting module formats the spaces by inserting and/or deleting spaces in the electronic character sequence.

[0067] Preferably, but not necessarily, the formatting module **10** is part of a system, such as an electronic device **100**, comprising a text prediction engine **30**, as shown in FIG. **1**. The electronic device is preferably a mobile device, such as a PDA, tablet, laptop computer or mobile phone. The formatting module may be used to format the spaces in an electronic character sequence entered by a user for a text message. The user interacts with a text entry system **50** of the electronic device **100** by entering text via an input mechanism such as a virtual keyboard. In the particular case of a predictive text entry system, the text prediction engine **30** may be configured to correct mistyped or misspelt words and may also be configured to predict what the user is going to write next, thus improving the performance and quality of the text input into the device. An example of such a text prediction engine **30** is described in PCT/GB2011/001419, which is hereby incorporated by reference in its entirety.

[0068] As can be seen from FIG. **1**, a character sequence is input into the device **100**. The character sequence is passed to a text prediction engine **30** which may modify that character sequence to correct misspelt words and/or to predict words. The character sequence, so modified by the text prediction engine **30**, is passed to the formatting module **10**. The formatting module **10** is configured to output a space formatted version of the modified character sequence, as shown in FIGS. **1** and **2**. The formatting module formats the spaces of a character sequence by inserting and/or deleting spaces in the sequence. The formatting module **10** formats the spaces for an electronic character sequence, if the language in which that character sequence is written is supported by the formatting module **10**.

[0069] A formatting module **10** in accordance with the present invention is shown in FIG. **2**. The formatting module **10** is configured to support at least one language. The formatting module **10** comprises a language identifier **20** configured to identify whether an electronic character sequence is written in a language supported by the formatting module **10**. The language identifier **20** makes use of one or more statistical language models, the general properties of which are known in the art, in order to identify whether the electronic character sequence is written in a language supported by the formatting module **10**.

[0070] In a preferred embodiment, the formatting module **10** supports a plurality of languages. Thus, in the preferred embodiment, the language identifier **20** comprises a plurality of statistical languages models, each statistical language model corresponding to a different language supported by the formatting module **10**, and the language identifier **20** is configured further to identify the most likely supported language of the electronic character sequence. At any given stage, the

formatting module **10** is configured to maintain a list of "active languages", each of which is associated with a language model.

[0071] One process for identifying the most likely current language is to maximize the probability of a language, given a context, i.e. maximizing P(language|context), according to the following expression (using Bayes rule):

$$P(\text{language} \mid \text{context}) = \frac{P(\text{language} \mid \text{context})P(\text{language})}{P(\text{context})}$$

[0072] As the absolute values of P(language|context) are not important, since only the ranking of languages matters, the term P(context), which does not depend on language, may be dropped from the expression. Additionally, a uniform prior over languages, P(language)=k, may also be dropped since it is constant with respect to language. With these assumptions, the only quantity that the language identifier is required to estimate is P(context|language). Typically context is just a sequence of words, therefore to estimate P(context|language), the language identifier preferably uses a 'chain' of conditional probability estimates, making a 'Markovian' conditional independence assumption:

$$P(\text{language} \mid \text{context}) \approx \prod_{i=0}^{N_{words}} P(word_i \mid word_{i-1} \ \ldots \ word_{i-N+1}, \text{language})$$

[0073] Each language is therefore separately modelled by a smoothed n-gram language model (known in the art and as described in WO 2012/042217), capable of estimating the probability of a word, given local context.

[0074] There are other ways of estimating P(context|language), using different types of language models, e.g. those that include syntactic and/or semantic information. Another possibility would be to use a Hidden Markov Model (HMM) to estimate a progression of unobserved language "states". A further possibility would be to use a supervised discriminative classification model to predict language, e.g. a support vector machine (SVM) or neural network.

[0075] To transform the incoming sequence of characters into a sequence of terms the language identifier **20** uses a tokenizer as is known in the art.

[0076] In a system such as that illustrated in FIG. **1**, the prediction engine **30** may comprise a language identifier, rather than it being provided in the formatting module **10**. As described above, the language identifier will comprise a tokeniser and a plurality of language models, which may already be present in the prediction engine, such as the prediction engine described in WO 2012/042217, which is hereby incorporated by reference in its entirety.

[0077] To estimate the most likely language given context, the language identifier **20** is configured to calculate the likelihood of the context in each language which is supported in turn, and selects the language with the maximum likelihood. The likelihood of the context (a sequence of terms) is the product of the probability of each term, given preceding terms, which is computed by a smoothed n-gram model, as has been described in relation to a text prediction engine in WO 2012/042217.

[0078] If the user switches languages whilst typing, the formatting of the spaces around the punctuation marks may differ between the sentences, dependent on the language in which it is written, e.g. "Bonjour mon ami ! How are you doing? Talk to you soon."

[0079] To provide a formatting module **10** that is capable of identifying a change in language, for example where a user has switched languages between sentences, the language identifier **20** is preferably configured to limit the amount of context used to make the estimate of the most likely language. This provides a basic form of recency in the model for identifying the most likely language—languages used more recently are intuitively more likely than languages used much earlier in a document. For instance, in one embodiment, the language identifier **20** may use the six most recent words of context. However, the number of most recent words of context could be chosen dependent on the frequency at which a user switches between languages and the length of their input stream in any given language.

[0080] The language identifier **20** is preferably configured to identify whether the language in which the electronic character sequence is written is supported by the formatting module **10**. By way of a non-limiting example, the language identifier **20** may identify that the electronic character sequence is written in an unsupported language if none of the context terms of the sequence are present in one of the language identifier's language models, where each language model corresponds to a supported language. Thus, if one or more of the context terms are determined to be present in one of the language models, the language identifier determines that the electronic character sequence is written in a supported language. A variation on this example is one in which the language identifier **20** is configured to identify whether a certain fraction or ratio of the context words are present in a language model, e.g. a quarter, two-thirds or any other fraction or ratio of the context terms are present in one of the language models, in order to determine that the electronic character sequence is written in a supported language. Any other suitable method for determining whether the language of the electronic character sequence is supported can be used.

[0081] As shown in FIG. **3**, the character identifier **40** preferably comprises a set of rules **70**, each rule relating to a character or particular sequence of characters to be identified, and a comparison mechanism **60** configured to compare each rule of the set of rules **70** to the electronic character sequence to determine whether a rule is applicable. If the rule is applicable, then a character or particular sequence of characters is identified, e.g. if the rule relates to a particular punctuation mark and the rule is found to be applicable, it is because that punctuation mark is within the electronic character sequence. The electronic character sequence is preferably passed to the formatting module **10** sequentially, e.g. a character at a time, with the comparison mechanism **60** comparing each rule to the last character or last sequence of characters received.

[0082] Thus, the character identifier **40** uses the rules to identify when a particular character or sequence of characters, such as a punctuation mark, occurs in the electronic character sequence. Furthermore, the rules define, by one or more actions associated with the rule, the space formatting to apply to the electronic character sequence, i.e. whether spaces should be inserted and/or deleted. Thus, once a rule has been found to be applicable to a particular character or sequence of characters, the actions associated with that rule are applied to the electronic character sequence to format the spaces within the electronic character sequence, e.g. in the case of the particular character being a punctuation mark, the actions

may define the formatting of the spaces either side of the punctuation mark, as will be described in more detail below.

[0083] The set of rules **70** preferably comprises a plurality of sets of rules, a set of rules for each language supported by the formatting module **10**. The comparison mechanism **60** is configured to compare the set of rules relating to the language identified by the language identifier **2** as the most likely supported language. In an embodiment in which the language identifier **20** supports a single language, the comparison mechanism **60** comprises a single set of rules **70** corresponding to that language, and the comparison mechanism **60** is configured to compare the set of rules **70** to the electronic character sequence if the language of the character sequence is identified as being the supported language. If the language of the character sequence is not identified as a supported language, the comparison mechanism **60** does not search for applicable rules.

[0084] The formatting module **10** is configured such that a system designer is able to manually add new rules, with associated actions, to the formatting module. The rules and associated actions can be updated without affecting the other components of the formatting module.

[0085] A rule is preferably defined by a four-tuple, as follows: Rule :: (C, s, A, S)

[0086] :: is an operator that can be read "has type of".

[0087] C is a condition taking the form of a regular expression, implementing a function of type F :: [character]→{true, false}, e.g. taking the incoming character sequence and returning a boolean denoting whether or not a rule is applicable and thus whether or not to apply the sequence of actions associated with that rule. The comparison mechanism **60** identifies a particular character or sequence of characters in an electronic character sequence by implementing the function of the type F :: [character]→{true, false}. This field is therefore essential and is never empty.

[0088] s represents a state that allows the system to "remember" previous rule applications in some cases. For example, the state may be "None" when the system is not required to maintain a status, or the state may be "Open" or "Close" where punctuators appear in pairs and one cannot exist without the other, e.g. left parenthesis '(' and right parenthesis ')'.

[0089] A is a sequence of Actions, i.e. A :: [Action]. In special cases this could be an empty sequence represented by [ ]. Actions are the means by which the formatting module **10** describes the space formatting that should be applied to, for example, a punctuation mark given a particular character sequence context (e.g. where the punctuation mark is found in the context of a mathematical equation). When a punctuation mark of the electronic character sequence is determined by the comparison mechanism **60** to match one of the rules, each action held by the rule is applied, preferably sequentially, to the punctuation mark to ensure the correct formatting of the spaces either side of the punctuation mark. For example, if the punctuation mark is a full stop, the Action might be to delete the space before the full stop (if such a space is present) and to insert a space after the full stop (if such a space is missing), where the most likely language is English.

[0090] There are two types of actions that the formatting module may comprise: type A and type B.

[0091] An action of type A is a function that operates on a sequence of characters and returns a formatted sequence of characters, without changing the sequence of characters, other than by formatting them:

[0092] Action A :: [character]→[character]

[0093] For example, in the case of "word.word"→"word. word"

[0094] An action of type B is a function that given a sequence of characters returns a code that represents the state of the system, without changing the sequence of characters:

[0095] Action B :: [character]→new state

[0096] The new state is any of the possible states that the system might be in, e.g. the shift state to define whether the next character should be capitalised or not, e.g. "Word. "→"shift state of system".

[0097] S is a recursive sequence of rules, known as "secondary rules", i.e. S :: [Rule]. When the Rule does not describe any secondary rules, S will be represented by Ø. The secondary rules will be checked before the actions of the parent rules are applied, allowing an alternative behaviour for condition C depending on factors described by the secondary rules. The input for the secondary rules is the same electronic character sequence as for the parent rules; however, the focus of the condition C for the secondary rule is the character in the sequence that precedes the character that triggered the parent rule.

[0098] For example, in the preferred embodiment where the electronic character sequence is passed to the formatting module sequentially, e.g. a character at a time, the comparison mechanism compares each parent rule to the last character received. If a parent rule is found to be applicable, and that parent rule comprises at least one secondary rule, the comparison mechanism compares the at least one secondary rule to the penultimate character in the sequence (since the condition C for the parent rule is focused on the final character, whereas for the secondary rule the focus is on the penultimate character).

[0099] The application of secondary rules will be described in more detail below. Since secondary rules are not essential, the general form of the Rule could omit this field.

[0100] When designing the formatting module **10**, the sequence of actions associated with a rule can be selected by a designer from a predetermined set of candidate actions. The sequence of actions may contain any number of the candidate actions in any order and with any number of repetitions. As stated above, the formatting module **10** allows a system designer of the formatting module **10**, to manually extend and adapt the associated actions to the requirements of the languages or the text entry system.

[0101] In a preferred embodiment, the formatting module comprises three specialisations of the Rule described above: Context Rules, Category Rules, and Character Rules. The specialised rules provide a powerful tool to capture the way punctuation is used in natural language.

[0102] A context rule is a rule of the form: Context Rule :: (C, None, A, Ø). The regular expression present in C is applied only to the context, e.g. the regular expression corresponds to a particular character sequence in the context of the electronic character sequence, for example "www". Since the state is "None", a Context Rule will never have or maintain state. The Context Rules have no "secondary rules".

[0103] An example of a context rule is a rule for URLs which states that when "www" is in the context, no spaces should be inserted automatically on either site of the punctuator ".", e.g. "www.site.com"

[0104] Thus, an example of a context rule is:

[0105] Context Rule :: ('www', None, [DeleteSpaceBefore, DeleteSpaceAfter], Ø).

[0106] A Category Rule preferably takes the form: Category Rule :: (C, None, A, S)

[0107] This rule will match the Unicode category of the character in the electronic character sequence to the Unicode category defined by the rule, e.g. the Unicode category of a punctuation mark.

[0108] C is a regular expression that is limited to matching the Unicode category of the punctuation mark. Therefore, this type of Rule is only applied to a single character. S is a sequence of secondary rules, e.g. a context rule, a character rule or a category rule. Alternatively this field can be empty, Ø, in the case where no secondary rules are defined.

[0109] An example of a category rule is:

[0110] Category Rule :: ('P', [DeleteSpaceBefore, InsertSpaceAfter], Ø) where P corresponds to a category of punctuation marks, e.g. a category that includes '!' and '?' because they should be formatted with the same spaces.

[0111] As is known in the art, characters within the Unicode standard have a range of properties associated with them. One of these properties is the category to which a character belongs. The condition C of a category rule can relate to a Unicode category. The General Category value for a character serves as a basic classification of that character, based on its primary usage. The property extends the widely used subdivision of ASCII characters into letters, digits, punctuation, and symbols—a useful classification that needs to be elaborated and further subdivided to remain appropriate for the larger and more comprehensive scope of the Unicode standard.

[0112] Each Unicode code point is assigned a normative General Category value. Each value of the General Category is given a two-letter property value alias, where the first letter gives information about a major class and the second letter designates a subclass of that major class. In each class, the subclass "other" merely collects the remaining characters of the major class. For example, the subclass "No" (Number, other) includes all characters of the Number class that are not a decimal digit or letter. These characters may have little in common besides their membership in the same major class.

[0113] A character rule preferably takes the form: Character Rule :: (C, s, A, S). This rule matches a character defined by the rule to a character in the electronic character sequence. Therefore, in this type of rule, C can only contain a single character. C is a regular expression consisting of a single character matched against a character of the electronic character sequence. C may define the Unicode for the particular character of interest, this Unicode being matched to the Unicode in the electronic character sequence.

[0114] s preferably defines two new states, in addition to the None state: {Open, Close, None}. s therefore dictates actions for ambiguous pairs that might be in an Open or Close state, and also includes no state for non-ambiguous characters. By this definition, if for one punctuation mark two different sequences of Actions are required for different states, the system will define two rules. e.g. for the English language to format the following sentence correctly:

[0115] And he said "Goodbye" and left. It was surprising.

[0116] The character rules that define the formatting of spaces in this sentence are as follows:

[0117] rule1→Character Rule :: ('"', Open, [InsertSpaceBefore, DeleteSpaceAfter], Ø)

[0118] rule2→Character Rule :: ('"', Close, [DeleteSpaceBefore, InsertSpaceAfter], Ø)

[0119] rule3→Character Rule :: ('.', None, [DeleteSpaceBefore, InsertSpaceAfter], Ø)

[0120] S is a sequence of secondary rules, which may include any of the three types of rules: Context, Category or Character. It can also define no further rules, in which case this field is denoted by Ø.

[0121] An example to explain the interaction of the rules and, in particular, how secondary rules are applied is now provided. In French, a space is placed either side of an exclamation mark "!" or a question mark "?", e.g. Bonjour ! Ça va ? . However, there is an exception to this rule, when another exclamation mark precedes the current one, e.g. Bonjour !!! Ça va ? . In order for the system to deal with this situation properly secondary rules can be defined:

[0122] rule1→Character Rule :: ('!', None, [InsertSpaceBefore, InsertSpaceAfter], [rule2])

[0123] rule2→Category Rule :: ('P', None, [DeleteSpaceBefore, InsertSpaceAfter], Ø)

[0124] In this example, P relates to Category punctuation in accordance with the Unicode standard which occurs prior to the current character of interest e.g. if the formatting module **10** receives "!" in the sequence "?!", P is the category that encompasses "?". In the example above, when the user types the first '!' rule1 will be triggered. Within the trigger routine of rule1 all the secondary rules will be checked but no matches will happen, since there is no punctuation mark preceding the '!', so the default actions for rule1 will be applied by the formatting module **10**. On subsequent insertions of the exclamation mark, the step for matching secondary rules will trigger rule2 as '!' is within the Category Punctuation by the Unicode standard and the actions defined in rule2 will be applied.

[0125] The outcome if rule2 did not exist the formatting would be: Bonjour ! ! ! Ça va ? Thus, resulting in an incorrect formatting of the text.

[0126] For a given language it is generally required to have multiple rules defined to ensure correct formatting of spaces in an electronic character sequence. The different types of rules, i.e. context, category and character, are preferably applied using a priority scheme, such that the formatting module **10** can succinctly specify the formatting patterns of the spaces for a given language.

[0127] A couple of examples which demonstrate why it is preferable to prioritise the application of the types of rules are provided below.

[0128] For the specific case of URLs, assume that there are two rules: a context rule which defines that when "www" is in the context, no space should be inserted automatically; and a character rule that says that when the full stop "." punctuation mark is introduced, a space should be inserted afterwards. In this situation, if the character rule is applied first and the user enters "www.site.com", the result from the punctuator will be "www. site. com", because the character rule for the full stop will have preference. To format such a URL correctly, the context rule should have preference over the character rule and should therefore be applied first.

[0129] In another example, where a formatting module comprises two rules: a first category rule that states that all the characters in a Maths category will have spaces on either side of the Maths character, and a second character rule that defines that the character "−" (minus) will not have any spaces either side of it, because it is most likely to be used as a

hyphen. If the user were to insert '–', and the category rule was prioritised over the character rule, the character rule would never be triggered. Thus, to format the sequence correctly, the character rule should be prioritised over the category rule.

[0130] The rule that is prioritised is applied, and the comparison mechanism **60** stops the search for applicable rules. However, as described above, a different rule type may be applied as part of a secondary set of rules.

[0131] Thus, in the preferred embodiment, as illustrated in FIG. **4**, the comparison mechanism is configured to compare, and the formatting module is configured to apply, the rules for an individual language in accordance with the following prioritisation structure:

[0132] Context Rules→Character Rules→Category Rules

[0133] To implement the prioritisation structure, the comparison mechanism **60** is preferably configured to identify the type of rule. The comparison mechanism **60** can be configured to identify the rule type by any suitable means. For example, each rule can be labelled with its rule type, where the comparison mechanism **60** is configured to identify all of the rules of a first rule type before comparing those rules to the electronic character sequence to see if one of them is applicable. The rules of a given type can be placed in a container, so that the comparison mechanism **60** compares all rules in a given container, before moving on to the next container. In another embodiment, the comparison mechanism **60** may comprise code to identify the different rule types.

[0134] Alternatively, the rules themselves could be ordered according to the prioritisation structure, e.g. listed in accordance with the prioritisation structure.

[0135] As will be apparent from the description above, if the comparison mechanism **60** finds that a rule is applicable, it does not continue through the prioritisation structure, e.g. if the category rule is found to be applicable to www.site.com, then the character rule is not compared or applied, since the comparison mechanism **60** has stopped searching for applicable rules. Otherwise this character rule, if applied after the context rule, would result in the incorrect formatting "www. Site. Com" as described above. However, the rule that is applied may comprise secondary rules of the other rule types, e.g. the formatting module dealing with repeated punctuation where the triggered character rule comprises a secondary category rule:

[0136] rule1→Character Rule :: ('!', None, [InsertSpaceBefore, InsertSpaceAfter], [Rule2])

[0137] rule2→Category Rule :: ('P *', None, [DeleteSpaceBefore, InsertSpaceAfter], Ø)

[0138] Rules may be applicable to a particular language, e.g. English, and the family of languages to which that language belongs, e.g. Latin, or to all languages in the world. There are multiple conventions for punctuation that are common to a number of languages. For example, in all languages URLs are written the same way and therefore they all must have the necessary rules for the correct formatting of these elements.

[0139] In the preferred embodiment in which the formatting module **10** supports a plurality of languages, the language identifier **20** is configured to pass the identified language to the comparison mechanism **60**, and the comparison mechanism **60** is configured to compare the rules from the set of rules **70** that are relevant given the particular language so identified. The set of rules is preferably ordered into a hierarchal structure, in order to avoid repeating the same rules.

[0140] Thus, in addition to the comparison mechanism **60** being configured to compare the rules according to the rule prioritisation structure, e.g. context rule→character rule→category rule, as described above, the comparison mechanism **60** is further configured to compare the rules in a particular order of increasing generality:

[0141] language specific rules→language family rules→worldwide rules

[0142] To enable the comparison mechanism **60** to compare the rules in this order, the comparison mechanism is preferably configured to identify the language generalisation rule i.e. whether the rule is a language specific rule, a language family rule or a worldwide rule. The comparison mechanism **60** may be coded to recognise the language generalisation rule or each rule may be labelled to identify the type of language generalisation rule, and containers may be used, as explained above when discussing the rule type prioritisation structure. As stated above, an alternative could be to order the rules into the generalisation structure.

[0143] Thus, the comparison mechanism **60** is preferably configured to identify the rule type and the language generalisation rule, e.g. context rule applicable to French (language specific rule).

[0144] As can be seen from FIG. **5**, the comparison mechanism **60** compares the rules in accordance with the priority system described above, until a rule is found to be applicable: first all the "context rules" will be compared in order of increasing generality of language, e.g. the context rules are checked first for language specific rules, then for family rules, and then for worldwide rules; the comparison mechanism **60** then proceeds to compare the next type of rule, character rules, through increasing generality in language, and then compares the category rules in the same way, until a rule is found to be applicable, at which point the comparison mechanism **60** stops the search for an applicable rule. Alternatively, the comparison mechanism **60** compares all of the rules to find that no rule is applicable and all rules are exhausted.

[0145] Preferably, the comparison mechanism **60** is configured to compare each of the rules to each character in the electronic character sequence in turn. Thus, if the comparison mechanism **60** discovers that a rule is applicable to a character of the character sequence, the formatting module **10** applies this rule to the electronic character sequence to format the spaces of the electronic character sequence, and the comparison mechanism moves on to comparing the rules to the next character in the character sequence. Likewise, if no rule is found to be applicable to that character, the comparison mechanism **60** moves on to comparing the rules to the next character in the electronic character sequence.

[0146] As will be understood from above, the language identifier **20** is configured to identify whether the language in which the electronic character sequence is being written is supported and, preferably, which is the most likely supported language. The language identifier **20** may be configured to identify the current language periodically, e.g. for every term (where the electronic character sequence is converted into a sequence of terms or words by a tokeniser) or for example every three terms, in order to identify whether the language has been switched by the user and thus to change the set of rules that are being compared by the comparison mechanism **60** to the electronic character sequence. Any other frequency of checking may be used. If the language identifier **20** determines that the language of the character sequence is not

supported by the formatting module **10**, the comparison mechanism **60** stops searching for an applicable rule.

[0147] A formatting module **10** or system **100** comprising a formatting module **10** in accordance with the present invention provides language detection and rule mechanisms that provide automatic dynamic punctuation. Unlike existing systems which neglect the possibility of having different behaviours for the same punctuation mark depending on the context in which the punctuation mark occurs, the formatting module **10** of the present invention is able to format the spaces either side of a punctuation mark on the basis of the context of the punctuation mark.

[0148] The formatting module **10** of the present invention is therefore able to increase the productivity of the user by reducing the interaction required to produce correctly formatted punctuation appropriate to the target language. For multilingual users, the formatting module **10** is preferably able to automatically adjust the space formatting to the language currently being entered. This allows the user to focus on the message being delivered rather than formatting conventions specific to various target languages.

[0149] Furthermore, the formatting module **10** of the present invention provides a separate layer that defines the behaviour of the formatting of the spaces for the punctuation, i.e. the rules and their associated actions. This allows independent manual updates of the rules and their associated actions for a particular language, to change the space formatting for that language, without affecting the space formatting for the other languages or requiring an upgrade of the entire formatting module **10**.

[0150] The present invention also provides a corresponding method for formatting spaces in an electronic character sequence that has preferably been entered by a user. Turning to FIG. **1** and the above described formatting module **10** and system **100** comprising a formatting module **10**, the method comprises identifying whether the electronic character sequence is written in a language supported by the formatting module; identifying, with the character identifier **40** (see FIG. **2**), a particular character or a particular sequence of characters in the electronic character sequence; and formatting, with the formatting module **10** if a supported language is identified, spaces in the electronic character sequence on the basis of the language identified and the particular character or sequence of characters identified. As will be apparent from the description of the formatting module **10** and the system **100** comprising a formatting module **10**, the formatting module preferably supports a plurality of languages, and the most likely supported language can be identified by a language identifier **20** of the formatting module **10** or a language identifier of the prediction engine **30** of the system **100**.

[0151] Other aspects of the method of the present invention can be readily determined by analogy to the above system description. For example, the formatting module comprises a language identifier to identify whether the electronic character sequence is written in a language supported by the formatting module and to identify the most likely language of the electronic character sequence. The method, by analogy to the formatting module, will also comprise selecting, with a comparison mechanism **10**, the set of rules that correspond to the most likely language identified, etc.

[0152] The present invention also provides a computer program product comprising a computer readable medium having stored thereon computer program means for causing a processor to carry out the method according to the present invention.

[0153] The computer program product may be a data carrier having stored thereon computer program means for causing a processor external to the data carrier, i.e. a processor of an electronic device, to carry out the method according to the present invention. The computer program product may also be available for download, for example from a data carrier or from a supplier over the internet or other available network, e.g. downloaded as an app onto a mobile device (such as a mobile phone) or downloaded onto a computer, the mobile device or computer comprising a processor for executing the computer program means once downloaded.

[0154] It will be appreciated that this description is by way of example only; alterations and modifications may be made to the described embodiment without departing from the scope of the invention as defined in the claims.

1. A system, comprising:

a processor;

memory storing instructions that, when executed by the processor, configure the processor to:

identify whether an electronic character sequence is written in a supported language;

identify a particular character or a particular sequence of characters in the electronic character sequence; and

format spaces in the electronic character sequence on the basis of the language identified and the particular character or sequence of characters identified, when the supported language is identified.

2. The system of claim **1**, wherein the instructions that format spaces in the electronic character sequence insert and/or delete spaces in the electronic character sequence.

3. The system of claim **1**,

wherein the memory stores:

at least one set of rules, each rule relating to a particular character or sequence of characters to be identified in the electronic character sequence;

wherein each rule is associated with one or more actions which describe the format of spaces to be applied to the electronic character sequence given a supported language and the particular character or sequence of characters;

wherein the instructions configure the processor to:

compare each rule of one of the at least one set of rules to the electronic character sequence to identify whether a rule is applicable;

format spaces in the electronic character sequence by applying the one or more actions associated with the applicable rule to the electronic character sequence.

4. The system of claim **3**, wherein the instructions configure the processor to compare each rule of one of the at least one set of rules to the electronic character sequence only when a supported language is identified.

5. The system of claim **1**, wherein the system supports a plurality of languages and the instructions further configure the processor to identify the most likely language of the supported languages that the electronic character sequence is written in.

6. The system of claim **1**, wherein the instructions configure the processor to identify a punctuation mark and is configured to format the spaces either side of the punctuation mark on the basis of the punctuation mark.

7. The system of claim 1, wherein the instructions configure the processor to identify a particular context in the electronic character sequence, and format the spaces in the electronic character sequence on the basis of the context.

8. The system of claim 1, wherein the instructions configure the processor to identify a punctuation mark in the electronic character sequence and format the spaces either side of the punctuation mark on the basis of the category of punctuation mark.

9. The system of claim 3, wherein the one or more actions comprise a sequence of actions, wherein when a rule is found to be applicable, the instructions configure the processor to apply the sequence of actions to the electronic character sequence.

10. The system of claim 5, wherein the memory comprises a plurality of sets of rules, one set of rules for each language that is supported, and the instructions configure the processor to compare each rule of the set of rules that corresponds to the most likely language to the electronic character sequence.

11. The system of claim 10, wherein the memory comprises sets of rules relating to each language, each family of languages, and all languages in the world and wherein the rules are applied in a hierarchal structure such that, once a supported language has been identified, the instructions configure the processor to first compare each rule from the set of rules specific to that language, followed by each rule from the set of rules applicable to the family of languages to which that language belongs, followed by each rule of the set of rules which are applicable to all languages until an applicable rule is identified or no applicable rule is identified and all rules are exhausted.

12. The system of claim 3, wherein the instructions configure the processor to compare the rules in a specific predetermined order, wherein the set of rules comprises context rules, character rules and category rules and the processor is configured to compare the rules in the following order until an applicable rule is identified or no applicable rule is identified and all rules are exhausted: context rules, character rules, and then category rules.

13. (canceled)

14. The system of claim 1,

wherein the particular character identified in the electronic character sequences is a punctuation mark; and

the instructions configure the processor to format spaces in the electronic character sequence on the basis of the language in which the electronic character sequence is written, the punctuation mark identified, and a context of the punctuation mark, when a supported language is identified.

15. The system of claim 1, wherein the instructions configure the processor further

to generate a corrected electronic character sequence from the electronic character sequence; and

format spaces in the corrected electronic character sequence when a supported language is identified.

16. A system, comprising:

a processor;

memory storing instructions that, when executed by the processor, configure the processor to;

receive an electronic character sequence;

identify which language the electronic character sequence is most likely written in;

correct the electronic character sequence on the basis of the identified language

generate a corrected electronic character sequence;

identify a particular character or a particular sequence of characters in the corrected electronic character sequence; and

format spaces in the corrected electronic character sequence on the basis of the language identified and the particular character or the particular sequence of characters identified.

17. A method, comprising:

identifying whether an electronic character sequence is written in a supported language;

identifying a particular character or a particular sequence of characters in the electronic character sequence;

formatting spaces in the electronic character sequence on the basis of the language identified and the particular character or sequence of characters identified, when a supported language is identified.

18. (canceled)

19. The method of claim 17, further comprising identifying the most likely language of the electronic character sequence.

20. (canceled)

21. The method of claim 17, further comprising:

comparing each rule of one of at least one set of rules, each rule defining the formatting of spaces in the electronic character sequence, to the electronic character sequence to identify whether a rule is applicable to the character sequence;

identifying that a particular rule is applicable to the character sequence; and

applying the applicable rule to the electronic character sequence to format the spaces in the electronic character sequence.

22. (canceled)

23. (canceled)

24. The method of claim 17, wherein identifying a particular character comprises identifying a punctuation mark and formatting the spaces in the electronic character sequence comprises formatting the spaces either side of the punctuation mark on the basis of the form of the punctuation mark.

25. The method of claim 17, wherein identifying a particular sequence of characters comprises identifying a particular context and formatting the spaces in the electronic character sequence comprises formatting the spaces on the basis of the context.

26. The method of claim 17, wherein identifying a particular character comprises identifying a punctuation mark and formatting the spaces in the electronic character sequence comprises formatting the spaces either side of the punctuation mark on the basis of the category of punctuation mark.

27. (canceled)

28. (canceled)

29. (canceled)

30. The method of claim 17, wherein the rules are compared in a specific predetermined order, wherein the set of rules comprises context rules, character rules and category rules, and the method comprises comparing the rules in the following order until an applicable rule is identified or no applicable rule is identified and all rules are exhausted: context rules, character rules, and then category rules.

31. (canceled)

32. (canceled)

33. A non-transitory computer readable medium containing program instructions which, when executed by a processor, configure the processor to:

identify whether an electronic character sequence is written in a supported language;

identify a particular character or a particular sequence of characters in the electronic character sequence; and

format spaces in the electronic character sequence on the basis of the language identified and the particular character or sequence of characters identified, when a supported language is identified.

* * * * *