IMPROVEMENTS IN ELECTRONIC DOCUMENT ANALYSIS

The present invention relates to improvements in electronic document analysis. More particularly, embodiments of the invention provide apparatus for and a method of determining a characteristic of a block segment in the electronic document. Alternative embodiments relate to determining whether first and second block segments within the electronic document are arranged in a recognisable pattern. Embodiments of the invention provide apparatus for and methods of analysing document layouts and extracting information from electronic documents.

With recent vast advances in computer and internet technologies, a very large volume of information is created and exchanged electronically in the form of documents and files. Word processing applications, such as Microsoft Word, and the widespread use of HTML have enabled text documents to be created in a variety of styles and formats. Electronic text documents are no longer limited to being generated using only a few font types, styles, sizes or layouts. With just a few menu commands, the style and size of a document and its component parts (for example, the heading) can easily be changed to be presented in bold and enlarged font, so as to differentiate the title from the main body of text in the document. As a further example, tables can be inserted into electronic documents with headers in either vertical or horizontal orientation, table cells can be merged to insert spanning labels and all of this can be completed with just a few clicks of the mouse of a PC. Varying presentation formats are also possible. For example, a field label and its value can be differentiated by presenting the former in bold font, or simply by separating the two with a colon. Although the same text information can be presented in a variety of styles in a single electronic document, a human reader can differentiate the function of each text part or block by its style and/or layout. The layout of a particular document may reproduce the style of a technical journal, a report, or a newspaper report etc. each of which typically consist of paragraphs of text with a heading above. Text can be presented either in single or multiple columns. A document's layout may also be of table-form or data sheet types, where information is presented with field labels on one side and its field values on the other. In the case of tables, field labels are presented either at the top or side (or both), with the

corresponding values displayed in rows or columns. Each value in a table cell constitutes a text block, and several field values may share the same label.

Considerable efforts have been made in the art of document layout analysis in attempts to analyse the layout direction and composition of the document including analysing, for example, the number of columns of text, the correct reading order of the document, the logical order of text blocks within the document, correct location of a table within the document etc. The same or similar analysis can be performed on document images as well as on documents comprising text produced by word processing applications. In this analysis, electronic documents are often broken down into text blocks: independent block segments of text with a specific function. For example, a text block with a font size significantly larger than the rest of the text, and that spans substantially across the top of the first page of a document, may very well have the function of a document title. A text block that is displayed in bold font and/or to the left of another text block in the normal style of reading (i.e. Western style reading left-to-right and top-to-bottom), may very well be a field label, with the latter being its associated field value. Text blocks can have many functions including being a header, footer, section heading, paragraph, field label, field value, table header, table cell, etc.

Further, the format and layout of electronic text documents can be highly varied. Documents can range from short text in electronic mail, bulletin board postings, news articles, legal documents, scientific research papers, complete news magazines or journals, and even whole books or encyclopaedias to name but a few. The document layout can also vary greatly, even for the same type of documents. The documents can differ in font size, font style, the number of columns used or the types of presentation style used, such as journal style or tables. The invention of word processing tools has enabled the production of heterogeneous (i.e. non-uniform) layout documents by providing numerous styling options. The difficulty in analysing a document is thus highly dependent on the complexity and predictability of the layout.

Document layout analysis is an important part of an automated document extraction system, especially when the layout of the document is heterogeneous. Embodiments of

the present invention are related *inter alia* to two aspects of document analysis: document layout analysis and document information extraction.

Document layout analysis is concerned primarily with image processing and consequently is applied primarily but not exclusively to image data. Information extraction is concerned primarily with natural language processing and machine learning and deals mainly with the character codes used to represent electronic text. The techniques disclosed herein may be employed equally on documents created and processed electronically or paper "hard" copies scanned to, for example, PDF (Portable Document Format) or TIFF (Tagged Image File Format) and subsequently processed according to the techniques discussed below.

A common standard for character encoding is ASCII (American Standard Code for Information Interchange) but the techniques disclosed herein apply equally well to any standard character encoding of text. In embodiments of the present invention, techniques that are traditionally confined to the image processing domain are applied to the analysis of text documents. These techniques include image processing techniques such as image pixel/region quantisation and/or aggregation based on a surrounding neighbourhood of pixels. The concept of any localised pixel region being processed autonomously from the other pixels of an image is well known in image processing. Text documents here refer to electronic documents, comprising primarily of textual content. The text content can exist within each document as an image or in any native text document format (including as a plain ASCII document) in which the documents were originally written. In the case of the former, a pre-processing step involving Optical Character Recognition (OCR) is performed on the image text (e.g. in the TIFF format or other pure image formats such as bitmap, JPEG, GIF and PNG) to recreate as accurately as possible the ASCII equivalent of the original document from which the image was derived. In the case of the latter, a pre-processing step involving conversion of the native format document (i.e. TIFF, etc.) into a common ASCII format document is performed. In both cases, it is expected that any font and layout styling that may exist in the original document is extracted together with the ASCII text.

A known algorithm for segmentation of an electronic document into block segments is disclosed in "Table Structure Recognition Based on Robust Block Segmentation" by T.G. Kieninger, *Proceedings of SPIE, Vol. 3305, Document Recognition V, pp. 22-32, 1998*. Application of this or equivalent segmentation algorithms to, for example, document extracts 10 and 14 of the material data sheets of Figure 1a and 1b typically results in the block segments 12 and 16 respectively. However, by parsing the document extracts 10, 14 in this fashion, it can be seen that it may not be possible to perform document layout analysis to process the document further electronically to determine the correct reading order or to extract further information from the block segments as block segments 12, 16 do not distinguish between section titles and sections and attribute labels and values (e.g. "Physical Form" and "Liquid" respectively).

A common problem arises in the art of analysis of documents containing domain dependent information, such as those in the field of chemistry, legal papers, and so on. Typically such systems provide for training examples specific to each layout type and the implementing algorithm must "learn" the layout type prior to analysing any further or "new" documents. US 6,694,053 requires the provision of a set of manually-created domain-specific rules for each class of document layout types to be handled.

An example of a system for domain dependent layout document analysis is illustrated in Figure 2. The process 20 takes document images at step 22 and pre-processes them at step 24 to prepare the document 22 to an internal structure suitable for the algorithm, perhaps in the ASCII text-coordinate format as is known in the art. An example of a product for preparing the document is the PDW program commercially available from PDF Tools AG. Analysis of text documents may then be performed on the ASCII equivalent of their text content, accompanied by any associated font and layout styling as applied within each original document when it was written. Where the original text content is an image, the ASCII text and its associated styling may only be approximations of their true values on account of any limitations in OCR techniques. Image text documents are first converted to their equivalent ASCII representation via OCR software, while native text documents have their equivalent ASCII representation extracted via text conversion software. In either case each piece of ASCII text is

accompanied by its styling information and page coordinates within the original document.

A block segmentation algorithm 26 such as that of Kieninger mentioned above decomposes the elements of a page of the document into blocks of elements. Subsequently, a block differentiation step 30 takes training documents and/or domain specific templates 28 to assist in the analysis of the document. Finally, a block-attribute-value labelling step 32 associates the attributes with the values found in the blocks of elements.

Considerable effort in the field has been spent in the area of analysis of documents of a homogeneous layout; i.e. those documents which conform to a style or formatting guideline or whose layout can be approximately predicted. Heterogeneous layout documents are those whose layout is not confined or restricted to a pre-defined formatting rule. Examples of homogeneous layout documents include technical journals and business letters. Technical journals are considered relatively uniform layout documents since they are typically either one or two-column style, have a title, an abstract, an introduction consisting of a heading and a paragraph text, followed by the body with similar formatting. Similarly, styling and formatting rules guide business-letter writing.

Heterogeneous layout documents can consist of a hybrid of information presentation components, such as horizontal tables, vertical tables, forms, paragraph text, headings and lists, all within the same page and varying in an unpredictable fashion. A corpus of such documents can contain similar information but in varying layouts. One such example is the worldwide collection of documents known as Materials Safety Data Sheets (MSDS). MSDS are documents containing information about the safety and properties of a chemical, prepared by the manufacturers of the chemical. As these documents are prepared by different chemical manufacturers from all around the world, little can be done to control the layout and format of these documents. The Globally Harmonized System for the Classification and Labelling of Chemicals (GHS) standardization committee within the United Nations represents the most

comprehensive attempt at providing guidelines on what information a MSDS should contain. However, the standard's objective is to achieve completeness and correctness of information within an MSDS, not to achieve uniformity in how exactly that information is to be presented. This means that while the number of MSDS sections and their titles may be constant across all MSDS, the style of presentation will vary greatly from one MSDS to another. In fact, the reality is that even those parts of an MSDS that have had their naming conventions standardised still display significant variance across producers. To exemplify further, the layout of one MSDS document may consist of multiple horizontal table-forms in several sections, while another may consist of hybrids of both vertical and horizontal tables in the sections, and yet a third may consist of only vertical tables. The layout of MSDS information is thus free-formed, and no international standard or style guide exists for presenting such information. In order to automate the extraction of such documents, a system must be able to handle the analysis of diverse layout documents.

US 6,336,124 to Alam *et al* discloses a computer implemented method of converting a document in an input format to a document in a different output format. This document discloses locating data in the input document, grouping into one or more intermediate format blocks in an intermediate format document and converting the intermediate format document to the output format document using the intermediate format blocks. However, the system disclosed by this document is unable to process as required documents of heterogeneous layout or documents of unanticipated domains. This is because Alam's system does not attempt to determine any associations between segmented blocks. In the case of a heterogeneous layout document where the logical flow of the text blocks changes directions, a significant change in the output document page width could result in associated pairs of text blocks no longer being placed beside each other.

A typical known process for analysis of a homogeneous layout document is illustrated in Figure 3. The process takes documents of homogeneous or non-heterogeneous layout and pre-processes them at step 42 in a fashion as described above in relation to Figure 2. The steps of 46 and 50 are also similar to that of Figure 2. In this system, however, the

user must specify a text file at step 48 containing a list of fields of which the user wishes the algorithm to extract information. The block differentiation step 50 does, at this point, compare the block segments and extracts the relevant information to form at step 52.

Hitherto, there have been no systems which can provide for the required analysis of heterogeneous documents and/or domain-independent algorithms for determining automatically the correct layout or to extract relevant information from these documents. It is at least an object of embodiments of the present invention to overcome the drawbacks of known systems.

The invention is defined in the independent claims. Some preferred features of the invention are defined in the dependent claims.

Embodiments of the invention allow analysis of documents to be domain independent. By establishing a characteristic of a block segment with reference to its surrounding region or neighbourhood in the document, the overall layout of the document is irrelevant and analysis of the block segment can be performed with respect to a localised region around the block segment and without the provision of learning templates or training examples. As such, the algorithm for performing the disclosed techniques is domain independent and does not require learning or training examples.

Other embodiments of the invention allow analysis of documents of heterogeneous layout to be performed. These embodiments of the invention introduce the concept of detecting recognisable patterns in the layout of the block segments or detecting one or more regions with block segments in a uniform or substantially uniform layout. The block segments are processed and those which are laid out in a recognisable pattern can be processed first. The remainder of the document (i.e. with the uniform region masked from the remainder of the process) can then be re-segmented and a further search for a recognisable pattern in the documents can then be undertaken.

Embodiments of the present invention will now be described, by way of example only, and with reference to the accompanying drawings in which:

Figure 1 illustrates schematic examples of document segmentation results produced by known block segmentation algorithms;

Figure 2 is a flow chart illustrating a process employed by existing block differentiation schemes applicable to documents containing domain-specific data;

Figure 3 is a flow chart illustrating a process employed by existing block differentiation schemes applicable to documents with a homogeneous layout;

Figure 4 is a flow chart illustrating a first process for analysis of an electronic document containing unanticipated domain information;

Figure 5 illustrates segmentation of the document of Figure 1 when subjected to the block segmentation algorithm of the process of Figure 4;

Figure 6 demonstrates a selection of schematic examples of variations in primitive characteristic strength assigned by the process of Figure 4;

Figure 7 is a flowchart illustrating one implementation of the auto-block attribute-value association process of Figure 4;

Figure 8 is a schematic diagram illustrating implementation of the logical template casting process within the auto-block attribute-value association process of Figure 4;

Figure 9 illustrates an example of a sample document page input to the system;

Figure 10 is an example of an output of the process of Figure 4 when operating on the sample document of Figure 9;

Figure 11 illustrates an example of a sample document page input to the system;

Figure 12 is an example of an output of the process of Figure 4 when operating on the sample document of Figure 11;

Figure 13 is an example of a sample output of the process of Figure 4, whereby some attributes of the sample document of Figure 9 are specified via a template;

Figure 14 is a flow chart illustrating a second process for analysis of an electronic document of a heterogeneous layout;

Figure 15 is a flow chart illustrating a process in which features of the processes of Figures 4 and 14 are used in conjunction;

Figure 16 is a sequence of schematic diagrams illustrating the process of Figure 15;

Figure 17 illustrates two possible attribute label to attribute value association templates;

Figure 18 illustrates a segment of a page from a document of heterogeneous layout and a problem arising from the application of the process of Figure 4 to the text segment;
Figure 19 is a sequence of schematic diagrams illustrating the result of applying the process of Figure 15 as applied to the same page segment of Figure 18;
Figure 20 is first example of a document of heterogeneous layout consisting of separate table-forms and tables with boundary lines;
Figure 21 is a second example of a document of heterogeneous layout typical of a technical journal, consisting of headings and paragraph text;
Figure 22 is a third example of a document of heterogeneous layout comprising three columns, consisting of headings and paragraph text;
Figure 23 is a fourth example of a document of heterogeneous layout presented in plain text with little formatting, consisting of table-forms and tables without boundary lines;
Figure 24 is a fifth example of a document of heterogeneous layout presented with each section in separate boxes, the section headings being to the right of the box instead of at the top;
Figure 25 is a sixth example of document of heterogeneous layout, the information being presented with each section in separate boxes and using lists to present information;
Figure 26 is an example of a hardware system for implementation of embodiments of the invention; and
Figure 27 is a block diagram illustrating the software architecture of an embodiment of the present invention.

Referring to Figure 4 a domain-independent process for fine-grained analysis of an electronic document will now be described.

Steps 62 to 66 are similar to the corresponding steps described above in relation to the prior art processes of Figures 2 and 3. The pre-processing step 64 may comprise the step of normalising all the common electronic document formats into a single common format without any loss of the coordinate and font information that is related to all the text content. For example, various non-PDF file formats, such as Microsoft Word, HTML and ASCII text may be converted to PDF format files, which are then each

converted into a text format structure comprising the text and their corresponding co-ordinates and font-related information. The Constrained Block Segmentation step 68 identifies possible boundaries between text blocks. The underlying process that is repeated for each case exemplified is similar to that used in the Kieninger algorithm identified above. This process within the Constrained Block Segmentation step 68 involves systematically testing each and every block of text on a page (no particular order is required) and determining: if the block is vertically aligned with either of its neighbouring blocks to its left or right; if it horizontally overlaps any neighbouring blocks within the rows immediately above or below it; and if its distance from a neighbouring block is within threshold vertical and horizontal distances (determined statistically by step 68 via an initial scan of all blocks); and if it has the same set of primitive attributes as its neighbour (with certain exception cases, one example of which is given in the following paragraph). If any current block and its neighbouring block satisfy the above criteria, the block(s) are tagged as being part of the same composite block. Where either of the pair has already been tagged for merger with another block, the same tag value is used. Otherwise, a new unique tag is generated. When all blocks have been tested or tagged within the page, all blocks sharing the same tag are merged into composite blocks. The Constrained Block Segmentation step 68 recalculates the inter-block gaps between neighbouring blocks to determine new threshold distances and the cycle is repeated until no further composite blocks are formed. The Constrained Block Segmentation step 68 also detects the orientation of the page, detects the number of columns, processes columns, collects statistics on font characteristics and, then, performs the merging of characters, words and lines into text blocks of interest in a page. The Constrained Block Segmentation step 68 applies a segmentation algorithm that produces a more fragmented set of text blocks 12, 16 than those produced by prior art algorithms such as Kieninger's mentioned above and as illustrated in Figure 1. That is to say, whereas prior art segmentation algorithms commonly perform block segmentation based primarily on horizontal overlap and vertical closeness of words on a page with simple consideration of font style, font size and capitalisation, the process of Figure 4 imposes stricter constraints by disallowing merging of blocks with any different properties or primitive attributes, such as font styles, font size, different casing patterns, underlining, special punctuation marks, etc. A typical result is illustrated in

Figure 5 which shows the result of the Constrained Block Segmentation algorithm 68 operating on the same extracts of documents illustrated in Figure 1. As shown, the results are more fragmented than those known in the art.

For example, and with reference to Figure 5a, although the individual lines under the section have normal spacing between all their words, each line is still segmented into two blocks as the presence of the colon character (i.e. the ":" at the end of "Physical Form") suggests a change in function type from label to value, with "Physical Form being the label and "Liquid" being the value. As another example, in Figure 5b, the section title "4. First Aid Measures" is segmented as a separate block because it is set in bold while the text adjacent and below is not. The process of Figure 4 can, however, include certain exception conditions whereby for example: above-average horizontal spacing between two blocks does not prevent merger; or short text blocks of small font are allowed to merge with preceding larger font blocks if they are very close to, and appear to be super- or sub-scripted to the latter. For example, in Figure 5a, the section number "9." is still segmented as a common block with the section title "PHYSICAL AND CHEMICAL PROPERTIES" even though there is a large horizontal gap between the two. This is because an exception case is applied that a numeric text string is more meaningful prefixed to an alphabetic text string than as a standalone block. Even with such exception conditions, the result will still be a segmentation algorithm which will produce, on average, a significantly more fragmented set of text blocks than prior art algorithms. The increased fragmentation in text blocks produced by the Constrained Block Segmentation algorithm 68 is desirable for the Block Feature Extraction step 70 to prevent text blocks of differing primitive characteristics (as will be explained further below) from being merged together, where important information would be lost.

Next the process applies the Block Feature Extraction Step 72 and Auto-Block Attribute Differentiation and Labelling 74 (which performs an Auto-Block Attribute-Value association as will be described below), collectively known as the Fine Grain Block Differentiation step 70. Step 70 analyses each text block for their function types and classifies them into potential blocks of attributes or value blocks. This step also

performs an association of the individual blocks into pairs of blocks containing potential attribute-value pairs of blocks, to output 78.

The Block Feature Extraction step 72 uses a generic set of deterministic rules to assign various primitive or low-level characteristics to candidate text block segments. Each characteristic indicates a block segment's potential to represent any number of different abstract block types that a human reader would associate with different parts of a document. The step of Auto-Block Attribute-Value Association 74 takes the candidate text blocks from the Block Feature Extraction step 72 and casts them into one or more generic logical templates as will be described below in relation to Figure 8. Each logical template represents a distinctive logical path that traces out the proper reading sequence for a region of text blocks. A non-trivial template would typically cover more than one abstract block type in representing its reading sequence. The process can use recursively the minimum combination of just two abstract block types to represent almost all documents as a logical tree of information. (A tree is a standard type of programming data structure for representing a series of recursive compositions of information whereby the single trunk of the tree branches out into increasingly smaller branches of information. The information represented by any one branch will be equivalent to the composition of all its immediate sub-branches. In the context of block segmentation on documents, the whole document/page may be considered the trunk of the information tree, with the main branches representing, say, the sections of the document/page, and sub-sequent branches representing sub-sections until eventually single words/characters represent the smallest units of information at the terminating branches of the tree.) Here, we refer to these two abstract types as an attribute and its value (or values). With recursion, the value can itself have a sub-value (or sub-values), whereupon the former becomes the attribute to the latter's value. Within the context of attribute-value block pairs, some possible logical paths represented by these generic logical templates would include: reading strictly from left-to-right; reading top-to-bottom for each pair but left-to-right between pairs; reading top-to-multiple bottoms (i.e. one-dimensional tables); and, reading both top-to-multiple bottoms and left-to-multiple rights (i.e. two-dimensional tables). Examples of attribute-value pairs include the literal association of a field label with its value, as well as the abstract association of: a section with a section

title and all text blocks following it (in the logical sense) up to the next section title; a header with the recurring text block at the top of most pages; or, a footer with the recurring text block at the bottom of most pages. Although text blocks are used extensively in the above description, the process can also be used with image blocks, for example a value block can be an image block and the associating label block is a text caption about the image. However, the roles can of course be reversed so that it is the image block which is the label and the text caption which is the value. The tendency is to put the object with greater variations as the value. So a standard text caption paired with a digital photograph that could represent any scene is more natural as a label-value pair, while the same would be the case for a standard graphical bullet marking a paragraph of text.

Figure 6 shows a selection of schematic examples of the variations in primitive characteristic strength assigned by the Block Feature Extraction step 72. The Block Feature Extraction step applies a first block characteristic extraction rule to a first text block. In the illustrated embodiment, input parameters to the block characteristic extraction rule are a property of the first block segment and a property of a region of the electronic document in which the first block segment is located. Examples of the property of the first block characteristic are the font size of text within the block segment, font weighting, size of the block segment, whether a trailing colon (indicative of function of the block as a label) is present, etc. The property of the region of the document in which the first block segment could be the distance to a second block segment (i.e. a measure of a distance between the first and second block segments) or to a page edge, or an analysis of the contents of a second block segment within the region. In embodiments of the process, the two inputs to the first block characteristic extraction rule comprise a relative comparison of the properties; e.g. comparison of font size of two block segments in the region. As such, it will be seen that, in embodiments of the process of Figure 4, execution of the first block characteristic returns a result of differences in the first block's own representation, and/or a neighbouring (second) block's representation, or even the representation of the intervening space between the two blocks. In Figure 6, the process of Figure 4 determines a characteristic of the first block segment (e.g. block segment 80a, 80b, 80c etc) by setting a "L-likelihood"

parameter which refers to the quantitative likelihood that the first block segment comprises a field label; the parameter "V-likelihood" refers to the quantitative likelihood that the first block segment is a field value. The example in Figure 6a illustrates how the presence of a larger-than-median spacing between the first and second block segments 80a and 82a causes the process of Figure 4 to increase the L-likelihood parameter of block 80a, while determining a characteristic of the second block segment, e.g. the L-likelihood of block 82a, after taking consideration of the characteristic (the L-likelihood parameter) of the first block segment; that is, the L-likelihood parameter of block segment 82a is reduced. In embodiments of the process, following determination that block segment 80a is likely to be a label (the L-likelihood is increased), the process determines that, as a consequence of this and if the two block segments are sufficiently closely spaced, the V-likelihood parameter of the second block segment 82a is raised as it is likely the second block segment 82a is a value. In Figure 6b, it is the representation of the text in first block segment 80b in uppercase that leads to the process of Figure 4 increasing in that block's L-likelihood parameter with a corresponding increase in the V-likelihood characteristic of second block segment 82b. In Figure 6c, the process increases the L-likelihood parameter of first block segment 80c as its text in bold. In Figure 6d, it is the presence of a trailing colon (:) character in the text of first block segment 80d that leads to the process to increase the L-likelihood value of the block with a corresponding increase in the V-likelihood value of block 82d. In Figure 6e, it is the absence of a larger-than-median vertical spacing between the two text blocks 84e, 86e that causes the L-likelihood of the lower block to be reduced. In Figure 6f, it is the representation of the text of block 80f in a larger-than-median font size that causes the L-likelihood of the upper block to be boosted.

The block features are recognised and extracted using known techniques familiar to those skilled in the art. Discrete pieces of text extracted from the original document (e,g, into the PDW data format) contain information that indicates if the fragment of text is bold/underlined/italicised, the font style/size it uses and the character components that make up the fragment.

Various other Block Feature Extraction rules may be encoded and the selection of rules found to be applicable to any text block in a document may have their prescribed changes in primitive characteristic strength combined to determine the overall L-likelihood and V-likelihood parameters/characteristic, or any other abstract quantifier, of that block. It will be clear to anyone skilled in the art that the size of the increment or decrement in characteristic strength recommended by different block characteristic extraction rules can be varied, so that some rules can have a stronger influence while other rules can have a weaker influence. For example, the process may recognise that the presence of a colon in the first block segment provides a better assessment of whether the block segment is a label, than a comparison of font sizes of text in the first block segment with that of another block segment. In embodiments of the invention the process recognises that if two block segments are relatively far apart, then the probability they are related may be less and, as such, the influence the second block segment has on the result of the first block characteristic extraction rule is consequently less; that is the influence varies according to the distance between the block segments.

By applying the rules to localised regions of text blocks, the Block Feature Extraction step 72 is therefore applicable to all document layout types and realises a domain-independence not found in the prior art.

Figure 7 illustrates a possible implementation of the Auto-Block Attribute-Value Association step 74. Depending on the logical template into which the output 90 from Block Feature Extraction step 72 of Figure 4 is to be cast, a starting block is selected from the set of text blocks to be analysed and at this point an arbitrary block segment in the document can be chosen. A logical template will typically specify a minor axis and a major axis that together describe the path to be followed through all blocks in a region, as well as a finite sequence of abstract block types which is to be concurrently cycled through (steps 106 – 110, with the sequence of blocks following the aforementioned path) as the path is traced out. Following the pattern encoded by the template, the next block (i.e. first block of the next row of minor-axis blocks) in the input set can be determined (step 96) and the process is repeated (steps 98 – 104) until all blocks have been analysed. In casting any block into part of the logical template (step 112), a

specific primitive characteristic is used to quantise the quality of the cast. This characteristic is determined by the part of the abstract type sequence that is currently aligned with the block. So for example, the first block in a page may contribute its L-likelihood rating to the logical template cast (in the first pass through step 108) because the first abstract type of the template's repeating sequence is a field label. Extending the example, if the second abstract type of the template's repeating sequence is a field value, then the second block in the page would contribute its V-likelihood value (in the second pass through step 108 for same minor axis). In this way, the quality of the page's fit to the logical template is represented quantitatively via the summation of the relevant primitive characteristic rating of each block within the page. By comparing the fitness of a page for different templates (step 114) the process is thus able to determine the true logical path that is being expressed within the page of a document. Finally, by mapping back the abstract associations encoded within the logical template (as represented by the repeating type sequence) any associations between text blocks on a page can be transcribed and produced as the output 118 of the Auto-Block Attribute-Value Association step 74 of Figure 4, for any document layout type encountered.

It will be clear to a person skilled in the art that the process of Figure 4 will encounter situations while tracing out the path described by the logical template, whereby the path cannot be continued without ignoring the absence or presence of certain text blocks which do not conform with expectations/requirements of the logical template. For example, consider the case where a logical template specifies that blocks must occur in pairs of field labels and values along its minor axis, without line-wrapping (i.e. without continuation onto the next minor axis along the major axis), and an odd number of text blocks are found on the current minor axis. Then an inconsistency arises whereby the final text block on the current minor axis is identified as a field label, but has no corresponding field value text block following it. To put it another way, the type sequence loop encoded by the logical template is not completed for the current minor axis. Under circumstances such as these, a penalty (step 104) can be imposed on the logical template's overall fitness by way of some deduction (either derived through a formula or heuristic) from the accumulated primitive characteristic ratings of all blocks traversed so far.

Figure 8a shows a schematic diagram illustrating the Auto-Block Attribute-Value Association step 74 of Figure 4 through the application of a left-to-right logical template of attribute label/value pairs to a region of text block segments that has been produced by Block Feature Extraction step 72. Figure 8b shows contrasting diagrams where an alternative application of a logical template that enforces a top-to-bottom, left-to-right label-to-value mapping, is applied to the same region of text blocks as in Figure 8a. In Figures 8a and 8b, the values of the form L_{XX} and V_{XX} respectively represent the L-likelihood and V-likelihood parameters of the text block identified by the characters XX. The overall fitness of the template is then calculated from the scores of the L- and V-likelihood parameters, before deducting any penalties for instances where the page does not fit with the requirements of the logical template (i.e. blocks 118 are missing from the points where the logical templates expect to find blocks). In the example of Figure 8, it can be seen that the logical template of Figure 8a is a better "fit" than the template of Figure 8b; the template of Figure 8a imposes four penalties on the fitness score, while the template of figure 8b imposes eight penalties on the fitness score. By applying the logical templates to the document, an interpretation of the correct reading sequence or at least an approximation of this can be determined by the process of Figure 4.

Figure 10 shows an example of the output 78 from the Auto-block attribute differentiation and labelling step 74 of Figure 4 when the algorithm has processed an input document exemplified by that illustrated in Figure 9. At this point the process uses extraction techniques at process steps 76 and 78 known in the art. The document of Figure 9 uses colons to differentiate between attributes and values. Figure 11 shows another document where bold fonts and spacing are used to indicate attributes and values. Figure 12 shows its output from Fine Grain Block Differentiation step 70 after processing.

If the user does not want to see all the possible list of associations, a template of selected attributes can be provided, and the system provides a simple match of the selected attributes with the corresponding values from its list of generic associations.

Alternatively (not shown in Figure 4) attribute-value pairs can optionally be fed to a set of template attributes to match selected attributes and a simple step of matching selected attributes from the template with values text block is performed to provide an output. Figure 13 shows an example of the output when a template of attributes is fed to the system given an input document like Figure 9.

With reference now to Figure 14, a process for analysing a document of heterogeneous layout will be described. The process begins with provision of a document of heterogeneous layout 120 for pre-processing in a manner similar to that described above in relation to Figure 4. The steps of conversion to ASCII text co-ordinate format and Block Segmentation 124 and 126 are also similar to the corresponding steps in Figure 4.

Using the co-ordinate information about the basic block segments obtained in step 124, the block segmentation algorithm 126 connects ASCII primitive block text words with similar font size, font style and in close proximity to one another into composite blocks using a "bottom up" approach. What is meant by a "bottom up" approach, is that the block segments are built up (recursively composited) from the basic text coordinate information obtained by the process at step 124. By effecting the process this way, a collection of segmented text blocks for each page of a document is created. Subsequently, the process uses a top-down approach to detect all regions of segmented text blocks that demonstrate some regular pattern in their relative positions in method step 128 and records each region to a prioritised list sorted by e.g. size of region. Bigger or regular regions are processed first as smaller regions have a greater likelihood of appearing regular only by coincidence.

The step of uniform region detection 128 is effected by scanning the segmented blocks on a page for a recognisable pattern. In its most elementary form, the process operates on a document containing first and second block segments and scans the document to determine whether the first and second block segments are arranged in a recognisable pattern. In doing so, method step 128 cross-checks the arrangement of the first and second block segment diagrams with a set of logical templates. In its most elementary

form, the set of logical templates comprises first and second templates as illustrated in Figure 17 as will be further explained in relation to Figure 16.

In this embodiment, the process relies on a concept of not processing all segmented blocks on a page simultaneously, but instead, of detecting and quantifying patterns of regularity within sub-regions of blocks on a page. The process also introduces a block differentiation threshold to distinguish quantitatively between well- and poorly-differentiated regions on a page, as will be described below. The process also adds a feedback step that allows block segmentation (or block feature extraction in the process of Figure 15) to be revisited on selected regions on a page. By detecting patterns of regularity based on the functional pre-disposition of blocks, the described processes are able to apply selectively block differentiation to the most regular sub-regions of blocks on a page first. Whereupon the first sub-region of blocks is found that has a differentiation measure above the differentiation threshold, any other sub-regions that are found to fail the threshold have their differentiation outcome rejected and fed back to the block segmentation or block feature extraction stage of the process. Block segmentation or block feature extraction is then performed only on the rejected regions and has the potential to produce different output values due to the missing influence of those regions that have passed the differentiation threshold. In this way the feedback loop is repeated until all regions eventually satisfy the differentiation threshold or an invariant set of rejected blocks is reached, whereupon they are no longer subjected to the threshold test and their differentiation results accepted.

Returning to the process of Figure 14, more sophisticated embodiments of this process scan a plurality of segmented blocks within the document to determine whether the document contains successive rows of blocks that contain the same number of blocks and/or that form a grid-like pattern by their relative positions. When and if no such rows can be found, the process may apply less stringent criteria whereby, for example, any of the following criteria may be allowed for in the search for uniform regions: the first or last rows may be allowed to contain fewer blocks; columns of irregular blocks across candidate rows may be excluded from either end of each row; two consecutive rows with no vertically-overlapping blocks may be merged to produce a row that forms a

highly-uniform region with a third row; or, a grid-like pattern is not required. In scanning the block segments, the process applies an overall "fitness" score for each logical template, similar to that described above in relation to the process of Figure 4, to determine which template is the best "fit" for the segmented blocks. This is also described further below. When and if the process finds regions of uniform layout, scores will be added to a candidate region being scanned if the rows within it demonstrate uniformity. Areas of exceptional uniformity, such as a grid-like pattern of blocks in which all blocks within each column are perfectly aligned horizontally with respect to their left or right edge, or their horizontal mid-points may be scored highly. Conversely, with any relaxation of criteria for any candidate sequence of rows, a corresponding penalty is associated with the candidate rows which do not fit, or fit well, the logical template applied. Such a scoring enables a quantitative ranking of the uniformity of each region of blocks. If the process scans the document according to a plurality of templates, the overall best fit template can be determined in this way by calculating the scores and then determining which logical template score is the highest or satisfying any other user-defined criteria for fitness.

For each candidate block region, and with reference to the list of field labels that the user is interested in, the system proceeds to differentiate the text blocks at step 130 to separate the blocks likely to represent attribute labels from those likely to represent attribute values. The user-defined list of attribute labels that the process scans is entered in a conventional way and stored in memory of the device upon which the algorithm is executed. The process then stores in memory of the device a list of block segments determined to be likely to be labels if the contents of the block segment match, either wholly or partially, an entry in the list of user-defined attribute labels. Block segment pairs matching the template of, say, one of the templates of Figure 17, one of which is known to be a label, can then be assumed to be a label-value pair. These pairs are then masked from further processing of the document. The process then determines whether further segmentation and uniform region detection is required at step 134 before remitting the remainder of the document (less the masked regions) for re-segmentation at step 126.

To support the decision at process step 134 whether re-segmentation at step 126 is required, a number of tests may be applied. These include assessing the quantitative fitness of the differentiation performed for each candidate region (and as scored in the fitness score) and may be compared against some differentiation threshold. The block differentiation step 130 associates certain strengths to each text block that matches (fully or partially) any entry in the user-defined list of labels 132. An example of how this may be done is for the process to pair each entry of the list 132 with a score. This score reflects an assessment of which alternatives among the labels is most commonly used across documents of the corpus, or which labels have the potential also to match text in attribute value blocks and so are not as reliable indicators of a block being an attribute label. In addition, this score from the matching label may be reduced if the label only covers part of the text block.

The process of Figure 14 applies a scoring range in the examples of Figures 16a to 16m that are used to illustrate a few cycles through the loop of steps 146, 148, 150, 152, 154 and back to 146. The process assigns the score L_{High} (likely to be a label) to the L-likelihood parameter for those text blocks that match exactly a label in the list 156. In embodiments of the process, the score for this could be set to "1". The process assigns the score L_{Medium} (e.g. 0.5 as this may be a label) to the L-Likelihood parameter for those text blocks that partially match a label in the list 156, and a score of L_{low} (e.g. "0" as the block segment is unlikely to be a label, but assign a score of V_{High} (i.e. "1") to the V-Likelihood (likely to be a value) parameter for those text blocks that do not match any labels in the list 156. That is to say, in embodiments of the process of Figure 14, all labels in the list 156 are assigned the same score, and text blocks that exactly match a single label in the list 156 are highly likely to be field labels, text blocks that only partially match a single label are moderately likely to be field labels, and text blocks that do not match any labels are highly unlikely to be field labels and thus highly likely to represent field values.

Figure 16a shows a sample segment of text 170 from a page of an MSDS document. A human reader may be able to determine the correct reading sequence for data in the sample text segment 170 can be segmented logically into label-value pairs 172a,b,

174a,b etc. of block segments as shown in Figure 16b, with the associations from field labels to field attributes illustrated by the arrows 178 between pairs of blocks. To achieve the same result of the human reader, the process of Figure 14 executes the sequence of steps illustrated in Figure 16c to Figure 16m.

In the first pass through that the process executes to process/analyse the text segment, the Block Segmentation step 126 receives an input similar to that depicted in Figure 16c, where each individual word is represented as its own primitive text block 180, 182, 184 etc. as a result of the text-coordinate association step 124. Based on this input, the block segmentation algorithm 126 measures the horizontal and vertical spacing between neighbouring primitive blocks in an effort to determine statistically the median for intra-block spacing H1, V1 and inter-block spacing H2, V2 for both vertical and horizontal axes. After the median distances for intra- and inter-block distances have been determined, the block segmentation algorithm 126 determines whether primitive text blocks should be grouped together. The process does this by determining whether the spacing between blocks is above or below the respective median and, on that basis, determines whether the spacing is an intra- or inter-block spacing and groups the blocks accordingly. Looking at Figure 16c, the process determines that, roughly speaking, there are two values, V1 and V2 that represent the variation in vertical spacing and three values, H1, H2 and H3 to represent the variation in horizontal spacing (where H3 represents the open-ended range of all spacing significantly greater than H2). As there are only two values for vertical inter- and intra-block spacing in the illustrated embodiment, the threshold between vertical intra-block and vertical inter-block spacing can be calculated as being between V1 and V2. In the case of the horizontal spacing, certain assumptions must be made by the illustrated process. If it is assumed that H2 is double the length of H1 and H3 is at least four times the length of H1, then the process of locating the mid-point of a density plot of horizontal values may take the form of an approximate equation of the form: half of density plot = (29 x H1 + 3 H2 + 10 H3) / 2 = (29 x H1 + 3 x 2 H1 + 10 x 4 x H1) / 2 = 37.5 x H1 (where, for example, 29 x H1 denotes that the process counts there are 29 gaps of size H1 in the sample text segment 170). Since 37.5 x H1 is greater than (29 x H1 + 3 x H2) but less than (29 x H1 + 3 x H2 + 10 H3), the process determines that the threshold between horizontal intra-block

and horizontal inter-block spacing lies between H2 and H3. That is to say, any horizontal gaps of size H1 or H2 are intra-block gaps and any horizontal gap of greater than the point between H2 and H3 is an inter-block gap. Thus, primitive text blocks separated by intra-block gaps should be grouped as block segments 190. The result of the determination of the threshold spacing for vertical and horizontal axes would be the segmentation of the primitive text blocks of Figure 16c by the Block Segmentation step 126 of Figure 14 into the composite block segments depicted by Figure 16d. Assuming that all the field labels of Figure 16b are "perfect" labels (i.e. they match labels in list 132 exactly), the process then calculates a score assignment to the segmented blocks as that shown in Figure 16e (where the default score for the unlabelled blocks is V_{High}). In embodiments of the process, the process makes the further assumption that all choices of the pattern of associations between label and value blocks are limited to horizontal-left-to-right and vertical-top-to-bottom mappings like those shown in Figure 17a and Figure 17b respectively. In embodiments of the process, the dotted boundary between each pair of associations in Figures 17a and b to be equivalent to the perforations in a sheet of postage stamps, thereby allowing the process to match whatever outline of uniform text blocks that may be returned by the Uniform Region Detection step 128. Thus, based on the segmentation results in Figure 16e, the process of Figure 14 returns a result from the Uniform Region Detection step 128 of four uniform regions as shown (within the dashed boxes 192) in Figure 16f, with the lower two uniform regions representing the better fits to the label-value mappings of Figures 17a and b.

As an alternative example, embodiments calculate the mean of the distance between blocks instead of the median as described above.

Determining that an invariant state has not yet been reached (i.e. a number of block segments 194 in Figure 16f remain un-associated), the decision step 134 directs the process flow to follow the feedback loop 138 back to the Block Segmentation step 126. Effectively, the regions 192 are now "masked" from further processing of the text segment 172 by the process of Figure 14 and, thus, are shown as being "greyed" out in Figure 16g as regions whose differentiation results have been finalised and are excluded from consideration in any additional cycles through the feedback loop 138. The Block

Segmentation algorithm 126 is re-applied to the reduced set of primitive text blocks 196 of Figure 16g and the thresholds between intra-block and inter-block spacing are recalculated in view of the reduction in the input set of primitive blocks. In this iteration of the feedback loop, the equation to determine the mid-point of the density plot of horizontal spacing takes the form of: half of density plot = (16 x H1 + 3 H2 + 5 H3) / 2 = (16 x H1 + 3 x 2 H1 + 5 x 4 x H1) / 2 = 21 x H1. Now, the result of the equation (21 x H1) is greater than (16 x H1) but less than (16 x H1 + 3 x H2), and, as such, the process determines that the threshold between horizontal intra-block and horizontal inter-block spacing now lies between H1 and H2. This results in the block segmentation output together with the attribute scoring as shown in Figure 16h. The second pass through Uniform Region Detection step 128 then locates four new candidate regions as shown in Figure 16i, and only three uniform regions 198 match either the horizontal or vertical label-value association templates of Figure 17. In the illustrated example, the feedback cycle is repeated one more time as shown in Figures 16j to Figure 16l, whereupon the final fully differentiated state of Figure 16m is reached. Comparing Figure 16m with Figure 16b, it is demonstrated that the algorithm has successfully reached the same block differentiation conclusion on a heterogeneously laid out text segment, as a human reader would have been expected to reach. Once the process determines that all the blocks are differentiated into attribute-value pairs (i.e. the process determines that there are no block segments of the document not associated as part of a label-value pair) the attribute labels are then extracted to a form by step 136.

Thus, it can be seen that the process described returns accurate results without dependence on the prior art methods of providing learning examples, but instead focuses on identifying uniform regions and differentiating block regions, the system is able to analyse documents with drastically different layout within a corpus.

Referring to Figure 15, a second approach to performing layout analysis and extraction on heterogeneous documents is described. This approach is based on a combination of the fine-grained information extraction and heterogeneous layout analysis components described above with reference to Figures 4 and 14 respectively. As can be seen, the process of Figure 15 employs steps common to both of the processes of Figures 4 and

14: pre-processing of an electronic document (140), conversion of the document to ASCII text, font style and co-ordinate format, block differentiation/labelling and extraction to form, numbered as method steps 140, 142, 144, 152 and 158 of Figure 15 respectively. Also employed in this process of Figure 15 are: the method steps of constrained block segmentation, block feature extraction and auto-block attribute differentiation and labelling of the process of Figure 4, numbered as method steps 146, 148 and 152 of Figure 14; and the method step of uniform region detection, checking whether there are blocks still to be re-segmented, the feedback loop to the block segmentation method step, and scanning a user-defined list of fields/labels to be extracted numbered as method steps 150, 154, 156 and 160 in the method of Figure 14.

One significant difference between the fine-grained information extraction process described and the prior art methods for analysis and extraction of non-heterogeneous layout documents mentioned in the first approach of Figure 4, is the presence of the additional Block Feature Extraction step 148 which introduces the concept of multiple primitive characteristics that can be assigned to each block produced by the Constrained Block Segmentation step 146. The strength of each primitive characteristic of each text block is dependent not only on the block itself, but also on the characteristics of all its neighbours as well. The influence of each neighbouring block is not equal but is taken to be inversely proportional to their proximity to the block in question because the farther apart between blocks, the less likely they are to be related. Therefore, based on a text block's font information such as font style, font size, casing pattern, special punctuation etc., as well as those of its neighbours, each text block is assigned a quantitative measure of each of its possible function types (e.g. section title, field label, field value).

The process of Figure 15 is demonstrated with reference to Figure 19 operating on the text segment of Figure 18a. Because this sample text region 270 only contains the abstract text block types of field label and field value, the process similarly restricts its demonstration sequence of Figures 19a to 19e to the use of only two fitness measures, "L-likelihood" and "V-likelihood", that refer to the quantitative likelihood that an associated text is a field label or a field value respectively. However, it will be

appreciated that the principles of the process can be expanded to embrace a higher number of fitness measures. The L-likelihood and V-likelihood values are equivalent to the L_{High} (or L_{Medium}) and V_{High} values respectively that were used in the example of Figures 16a to 16m that illustrated the approach of Figure 14 of implementing heterogeneous layout analysis without fine-grained information extraction. The difference is that the L-likelihood and V-likelihood values are able to reflect a much wider range of values e.g. scaled from 0 to 1 in increments of 0.1, or finer, such as 0.5 (i.e. not just "High", "Medium" or "Low") than their counterparts of the approach of Figure 14. This is because the likelihood values are derived from the multiple primitive characteristics assigned to the text blocks in the Constrained Block Segmentation step 146, as opposed to the single list of scored labels used in the example of Figure 14.

Within the Auto-block attribute differentiation and labelling step 74 of the process of Figure 4, text blocks are cast into one or more generic logical templates. Each logical template represents a distinctive logical path that traces out the proper reading sequence for a region of text blocks. A non-trivial template typically covers more than one abstract block type in representing its reading sequence. The process can implement a minimum combination of two abstract block types used recursively to represent almost all documents as a logical tree of information. Here, these two abstract types are referred to as an attribute and its value (or values). With recursion, the value can itself have a sub-value (or sub-values), whereupon the former becomes the attribute to the latter's value. Within the context of attribute-value block pairs, some possible logical paths represented by these generic logical templates would include: reading strictly from left-to-right; reading top-to-bottom for each pair but left-to-right between pairs; reading top-to-multiple bottoms (i.e. one-dimensional tables); and, reading both top-to-multiple bottoms and left-to-multiple rights (i.e. two-dimensional tables). Heterogeneous layout analysis adopts the same concept of casting groups of blocks into candidate templates and this has been demonstrated previously during the detailed description of the approach of Figure 14. The difference between the logical templates of the heterogeneous layout analysis component and those of the fine-grained information extraction component is that the templates in the former method are not restricted to being cast a whole page at a time but can be applied to groups of blocks so

that within a page, different types of layouts (e.g. form, journal, or table) can be detected. This difference has been demonstrated previously during the detailed description of Figure 14.

Without the concept of applying the logical templates to regions of blocks instead of a whole page of blocks, the fine-grained information extraction component of Figure 4 will not be able to process a heterogeneous text segment like that shown in Figure 18a. Whereas the human reader may be able to deduce the correct set of associations between the field labels and field values of the example, as shown in Figure 18b, the fine-grained information extraction component will not. For example, considering the application of a horizontal, left-to-right logical template as shown in Figure 18c, we see that the logical path 272 traced for blocks 1a, 1b, 1c, 2a, 3a and 3b will be an incorrect one and consequently so will their label-value associations in step 74. Conversely, if a vertical, top-to-bottom logical template were to be applied (not illustrated), then it would be the turn of the lower blocks, 4a-4d and 5a-5d to have an incorrect logical path traced through them.

Figures 19a to 19e illustrate how the incorporation of heterogeneous layout analysis into a fine-grained information extraction system into the process of Figure 15 will overcome the limitations highlighted. After the first pass through the Block Feature Extraction step 148, the Uniform Region Detection step 150 will determine the most uniform region of blocks to be that of blocks 4a-4d, 5a-5d. Applying a horizontal, left-to-right template on this region in the associate step 152 enables the process of Figure 15 to trace a much better fitness measure compared to a vertical, top-to-bottom template, on account of the terminating colons (:) in the blocks of Figure 18a giving each field label an increased L-likelihood value The result of the first pass through steps 148 and 150 is illustrated in Figure 19a.

The feedback loop 160 to the constrained block segmentation step 146 will then return a reduced set of primitive text blocks to the Block Feature Extraction step 148. The confirmation of the logical path through blocks 4a-4d, 5a-5d removes them from consideration and essentially creates a virtual boundary by virtue of the masking of

these blocks just below blocks 3a and 3b as shown in Figure 19b. Or to put it another way, the disappearance of neighbouring blocks 4b and 4c causes the likelihood values of 3a and 3b to undergo the following reevaluations: L_{3a} (the L-likelihood parameter of block 3a) $\rightarrow L'_{3a}$; $V_{3a} \rightarrow V'_{3a}$; $L_{3b} \rightarrow L'_{3b}$; and, $V_{3b} \rightarrow V'_{3b}$. The process of Figure 15 recognises that since blocks 3a and 3b are in the last row of a group of blocks not masked, they are thus likely to be field values, and so L'_{3a} and L'_{3b} will be lower that L_{3a} and L_{3b} respectively, while V'_{3a} and V'_{3b} will be higher than V_{3a} and V_{3b} respectively.

Figure 19c illustrates an attempt to map the horizontal logical template onto the remaining blocks 1a-1c, 2a 3a and 3b. Given that the blocks do not form a uniform grid and that they collectively trace out a three-column region, the overall fitness of the horizontal template will need to be penalised for the merger of rows 2 and 3 and the two value blocks expected by the template in the missing fourth column.

Figure 19d illustrates an attempt to map the vertical logical template onto the remaining blocks 1a-1c, 2a, 3a and 3b. As with the case of the horizontal mapping of Figure 19c the vertical mapping also suffers a penalty for the merger of rows 2 and 3. However, it is not liable to any missing-block penalty. Observing that the field labels of blocks 1a-1c also terminate with a special character (:) as shown in Figure 18a, it would not be unexpected to find that L-likelihood values of blocks 1a-1c are likely to be higher than their V-likelihood values, while the reverse is true for blocks 2a, 3a and 3b. Thus the probability is that the vertical template map will show a better fitness compared to the horizontal template map. This is to say, we will find that: $L_{1a} + V_{2a} + L_{1b} + V'_{3a} + L_{1c} + V'_{3b}) - (1 \text{ x row-merge Penalty})) > (L_{1a} + V_{1b} + L_{1c} + L_{2a} + V'_{3a} + L'_{3b})$ - row-merge Penalty – (2 x Penalty).

Therefore, the final composite logical path traced through the blocks of Figure 18a will be one in which the top half of the blocks in the page segment follow a vertical, top-to-bottom logical path, while the lower half of the blocks in the page segment follow a horizontal, left-to-right logical path. This is as illustrated in Figure 19e and corresponds exactly to the label-value associations that a human reader would be expected to deduce

Figure 18b.

As with the case of the first approach the final step 158 of the process flow duplicates the step 78 of the non-heterogeneous embodiment of the process illustrated by Figure 4, and complete set of differentiated blocks, is then extracted to a form via an extraction list 156.

The processes described above can be implemented in software running on suitable machine such as a desktop computer. The invention can be coded in PERL which adapts itself particularly well for applications required to look for patterns. The processes described above could also be coded in alternative languages such as C++ or Java.

The heterogeneous layout documents referred to here are exemplified by Figures 20 to 25. The figures show that the layouts of the documents in the corpus are not confined to a pre-defined formatting rule. Documents can consist of a hybrid of information presentation components, such as horizontal tables, vertical tables, forms, paragraph text, headings and lists all within the same page. Within a corpus, one document or page can contain a hybrid of a horizontal form in one section, a vertical table in another and a list in a third section, while a second document can differ totally in the layout from the first. Non-heterogeneous or uniform layout documents are also easily analysed by the process described above. Thus technical journals, business letters, invoices, financial statement and resume layouts are considered subsets of the layout type recognised by the described processes. It will be clear to anyone skilled in the art that none of the existing prior art methods described earlier will be able to extract reliably the multiple pieces of information contained within documents such as these. Considering prior art which utilises learning methods, it is clear that such methods will fail to extract properly the information contained in all six types of layouts if it has not been given training examples of each. For example, the learning methods will not be able to properly extract the information from document with a layout format like that shown in Figure 20, if they have only been trained on documents with layouts like those of Figures 21 to 25.

The processes described above can generically be applied to documents of heterogeneous layout, including both technical journal style or table-form type of documents. Varying formats or styles of document layout can be handled automatically by the processes to analyse the document's layout and to differentiate each text block's function within the document.

Both top-down and bottom-up approaches are used. The top-down analysis establishes the geometric layout of the document, while the bottom-up approach establishes the logical layout of the document. The document is first segmented into broad blocks to identify the section boundary, headings, table location and reading order of the document, table or form. The number of columns in the document, table and forms are also identified. Characters that form a word and words that form a sentence are stitched together. The text blocks are then further segmented into small units and differentiated for their function, such as to identify the field label and value. At the end of the algorithm, the text blocks are identified and labelled as section heading, section text, field label, field value, table header, table value, etc.

Referring now to Figure 26, a hardware system 300 for implementing the disclosed techniques will now be described.

The system 300 comprises a Central Processing Server/Database Server/Document Store 302 comprising the server/store 304 itself, and administrator interface devices, display 306 and keyboard 308. Connected to the server/store 304 is a local external storage device 309 and a local document scanner 310 for scanning documents for analysis. The server 302 can optionally be connected to a Remote Database Server/Remote Document Store 314 over a suitable network such as the internet or a WAN (Wide Area Network) or LAN (Local Area Network) 316a. The Remote Database Server/Remote Document Store 314 can function either as an alternative or complementary database server and/or document store to the server 304.

The server 302 is also connected over the internet/WAN/LAN 316b to multiple client PCs 318 which, in turn, may have client external storage devices 320 and client document scanners 322.

Referring now to Figure 27, software architecture 350 for implementing the disclosed techniques will now be described.

The User Interface 352 comprises:

- a Client Graphical User Interface 354;
- a Document Submission Module 356;
- a Document Viewer/Retrieval Module 358; and
- a Results Search Module 360.

The Document Store 362 (which may be implemented in any or all of the server 304, local external storage device 309, remote database server/document store 314, client PCs 318 or client external storage devices 320 of Figure 26) comprises storage for:

- Original Format Documents 364;
- Standard Viewing Format Documents 366;
- Text Context Format Documents 368;
- Raw Text and Coordinate Information Format Documents 370; and
- XML Format Documents 372.

The Relational Databases 374 comprise:

- an Extraction Field Label/Pattern List 376;
- an Extracted Meta Data Database 378; and
- a User Account Database 380.

The architecture 350 also comprises:

- a Layout Analysis Module 382;
- a Document Format Converter 384; and
- a Server User Interface 386.

The Client Graphical User Interface 354 module in the User interface 352 is responsible for presenting the graphical user interface to the (possibly remote) end user (not shown) who is submitting a document for layout analysis.

The Client Graphical User Interface module 354 interfaces with the User Account Database 380 in the Relational Databases module 374 to determine the document submitter's access privileges to the system.

The Document Submission Module 356 in the Client Graphical User Interface module 354 interfaces with the Document Store 362 to upload documents submitted by the document submitter for layout analysis.

The Document Submission Module 356 uploads the submitted document to the Document Store 362 as a new addition to the set of Original Format Documents 364.

The Document Viewer/Retrieval Module 358 in the Client Graphical User Interface module 354 interfaces with the Document Store 362 to enable the retrieval of Original Format Documents 364 or Standard Viewing Format Documents 366.

The Results Search Module 360 in the Client Graphical User Interface module 354 interfaces with: the Document Store 362 to enable the search and retrieval of Text Content Format Documents 368 from the Document Store 362; and the Extracted Meta Data Databases 378 in the Relational Databases module 374 to enable the search and retrieval of the document layout analysis results produced by the Layout Analysis Module 382.

The Results Search Module 360 interfaces with the Document Viewer/Retrieval Module 358 to enable the search results of either the Text Content Format Documents 368 or the Extracted Meta Data Database 378 – which do not contain the documents' original layout information – to be paired with the corresponding document in either the set of Original Format Documents 364 or Standard Viewing Format Documents 366 – which do contain the documents' original layout information.

The Document Format Converter module 384 converts Original Format Documents 364 into: Standard Viewing Format Documents 366; Text Content Format Documents 368; and Raw Text and Coordinate Information Format Documents 370 within the Document Store 370.

The Layout Analysis Module 382 performs layout analysis on the Raw Text and Coordinate Information Format Documents 370 in the Document Store 362 and creates XML Format Documents 372 representing the results of its analysis, in the Document Store 362.

The Layout Analysis Module 382 also stores the results of its analysis on the Raw Text and Coordinate Information Documents 370 to the Extracted Meta Data Database 378 in the Relational Databases module 374, optionally appending domain-specific tagging information loaded from the Extraction Field Label/Pattern List 376 in the Relational Databases module 374.

The Server User Interface module 386 is responsible for presenting the (possibly) graphical user interface 352 to the (possibly remote) system administrator (not shown) who is responsible for maintaining the Document Store 362 and Relational Databases 374 and operating/configuring the Layout Analysis Module 382.

It will be appreciated that features of one embodiment of the invention can be combined with the feature(s) of another embodiment.

Claims:

1. A method of analysing an electronic document having a first block segment, the method comprising determining a first block segment characteristic from application of a first block characteristic extraction rule to the document, wherein input parameters of the first block characteristic extraction rule comprise a property of the first block segment and a property of a region of the electronic document in which the first block segment is located.

2. The method of claim 1, wherein the property of the region in which the first block segment is located comprises a property of a second block segment located in the region.

3. The method of claim 2, wherein the first block segment characteristic is determined from a comparison of the first block segment property and the second block segment property.

4. The method of claim 3, wherein the first block segment characteristic is determined from a measure of a distance between the first block segment and the second block segment.

5. The method of any of claims 2 to 4, wherein an influence the second block segment has on a result of the first block characteristic extraction rule is dependent upon a distance between the first block segment and the second block segment.

6. The method of any preceding claim, the method further comprising determining a second block segment characteristic from application of a second block characteristic extraction rule, the second block characteristic extraction rule taking consideration of the first block segment characteristic.

7. The method of any preceding claim, wherein the method comprises applying a plurality of block characteristic extraction rules and varying one of the first block

segment characteristic and the second block segment characteristic in dependence of results returned from the plurality of block characteristic extraction rules.

8. The method of any of claims 2 to 7, the method further comprising determining an associative relationship between the first block segment and the second block segment.

9. The method of any of claims 2 to 8, the method further comprising applying a plurality of logical templates to the first and second block segments for determination of a reading sequence of the block segments.

10. The method of claim 9, the method further comprising calculating a fitness score for the application of each logical template to the first and second block segments to determine a best fit logical template for the reading sequence of the block segments.

11. The method of any preceding claim, wherein, prior to application of the first block characteristic extraction rule, the method further comprises segmentation of the electronic document by application of a segmentation algorithm which disallows merging of text blocks with different block properties.

12. The method of claim 11, wherein the document comprises primitive text blocks in text co-ordinate format and the step of segmentation of the electronic document comprises:

compiling a log of gaps between primitive text blocks in the document,

determining, from the log, whether a gap between two primitive text blocks satisfies a threshold criterion; and

upon determination the threshold criterion is satisfied, determining whether the two primitive text blocks should form a block segment.

13. The method of claim 12, wherein the step of determining whether a gap between two primitive text blocks satisfies a threshold criterion comprises determining the median or mean of the gaps in a particular direction between the primitive block

segments and if a candidate gap is less than the median or mean, determining that the gap is an intra-block gap and segmenting the primitive text blocks accordingly.

14. The method of any of claims 2 to 13, the method further comprising determining whether the first and second block segments are arranged in a recognisable pattern.

15. A method of analysing an electronic document having first and second block segments, the method comprising determining whether the first and second block segments are arranged in a recognisable pattern.

16. The method of claim 15, the method further comprising determining whether contents of at least one of the first or second block segment comprises an attribute label.

17. The method of claim 16, wherein the step of determining whether contents of at least one of the first or second block segment comprises an attribute label comprises checking the contents of any of the first or second block segment with contents of a stored list of user-defined attribute labels for a match or partial match.

18. The method of claim 16 or claim 17, the method further comprising, assigning a characteristic value score to any of the first and second block segments if determined that the block segment comprises an attribute label.

19. The method of any of claims 15 to 18, the method further comprising applying a plurality of logical templates to the first and second block segments to determine a reading sequence of the block segments.

20. The method of claim 19, the method further comprising determining a fitness score for each of the plurality of logical templates and determining a best fit template is from the scores determined for each logical template.

21. The method of any of claims 15 to 20, the method further comprising:

determining that the electronic document contains a third block segment not found to be arranged in a recognisable pattern;

masking the first and second block segments from further processing of the electronic document;

applying a segmentation algorithm to the remainder of the electronic document excluding the masked first and second block segments to re-segment the remainder of the document; and

determining whether any of the re-segmented blocks are arranged in a recognisable pattern.

22. Apparatus for analysing an electronic document having a first block segment, the apparatus being operable to determine a first block segment characteristic from application of a first block characteristic extraction rule to the document, wherein input parameters of the first block characteristic extraction rule comprise a property of the first block segment and a property of a region of the electronic document in which the first block segment is located.

23. Apparatus operable to perform the method of any of claims 1 to 14.

24. Apparatus for analysing an electronic document having first and second block segments, the apparatus being operable to determine whether the first and second blocks are arranged in a recognisable pattern.

25. Apparatus operable to perform the method of any of claims 15 to 21.

26. A computer program product having stored thereon computer program code configured to analyse an electronic document having a first block segment, the computer program code being further configured to determine a first block segment characteristic from application of a first block characteristic extraction rule to the document, wherein input parameters of the first block characteristic extraction rule comprise a property of the first block segment and a property of a region of the electronic document in which the first block segment is located.

27. A computer program product having stored thereon computer program code configured to execute the method of any of claims 1 to 14.

28. A computer program product having stored thereon computer program code configured to analyse an electronic document having first and second block segments, the computer program code being further configured to determine whether the first and second blocks are arranged in a recognisable pattern.

29. A computer program product having stored thereon computer program code configured to execute the method of any of claims 15 to 21.