

An Improved English to Chinese Translation Search Engine of Technical Text

by

Xiangyu Zhang

A master thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
April 17, 2017

Keywords: Machine Translation, Regular Expression, Industrial Electronics Dictionary, IEEE
Xplore Search Gateway, Web Pages, Search Engine

Copyright 2017 by Xiangyu Zhang

Approved by

Bogdan M. Wilamowski, Chair, Professor of Electrical and Computer Engineering
Vishwani D. Agrawal, Professor of Electrical and Computer Engineering
Thaddeus Roppel, Associate Professor of Electrical and Computer Engineering

Abstract

The English to Chinese automatic translators cannot always satisfy the requirement of users especially in technical fields. Through research, the main reason is wrong translation of technical terminologies. The researchers summarized common mistakes of technical translation and then developed an IE (Industrial Electronics) dictionary that is a professional technical dictionary correcting the wrong translations by Google translation. With the Python platform, researchers can automatically replaces incorrect translation by Google translation to get improved translations. Finally, we develop web pages to show translation results that can help Chinese engineers read industrial electronic papers and this approach can be also used in other technical areas. This master thesis (1) introduces the development of machine translation and how machine translation works; (2) reviews the necessary programming languages to complete the project; (3) edits the Industrial Electronics Dictionary;(4) develops an English and Chinese search engine which can search key words in webpages.

Acknowledgments

First and foremost, I would like to express my gratitude to my advisor, Dr. Wilamowski, for his generous support in the past three years. Without his patience and guidance I would not finish my graduate research. From him I have learned a lot, his perseverance in academic research affects me profoundly during my study here. Secondly, I would like to express my thanks to Auburn University and the ECE department for giving me the good environment to study.

I want to thank Dr. Agrawal and Dr. Roppel for serving on my committee. I really appreciate that you took time from your busy schedule to read my thesis. Your insightful comments have helped increase readability and reduce ambiguity of the thesis.

I also want to express my thanks to my parents and family, who always give me support whenever I face difficulties. I also would like to thank all my friends, who are willing to share happiness and unhappiness with me and help me in my life.

Table of Contents

Abstract.....	ii
Acknowledgements.....	iii
List of Tables	vi
List of Figures.....	vii
Chapter 1 Introduction.....	1
1.1 History of Machine Translation	1
1.2 Machine Translation General Problems and Approaches	3
1.3 Motivation	6
1.4 Goals	7
Chapter 2 Literature Review.....	8
2.1 Regular Expression Search Algorithm	8
2.2 Programming Techniques: HTML, Perl, Python.....	16
2.3 Google API Service: Building Applications	28
Chapter 3 Industrial Electronics (IE) Dictionary.....	29
3.1 General Mistakes of Traditional Technical Text Translation.....	29
3.2 Proposed Solution of IE Dictionary.....	31
3.3 Related Work.....	31
Chapter 4 Code Description.....	34
4.1 Download Titles and Abstracts by IEEE Xplore Search Gateway	34
4.2 Translate English Technical Text to Chinese by Goslate	36
4.3 Parse the document.....	37

4.4 Make Valid Web Pages and Save Them.....	39
4.5 Words and Phrase in IE Dictionary Replace Google Translation	41
4.6 Search Key Words in Html Web Pages.....	42
Chapter 5 Experimental Results	44
5.1 Search Engine Website.....	44
5.2 Journal Titles and Abstracts English to Chinese translation examples	45
5.3 Search Result Examples of Key Words.....	46
Chapter 6 Conclusion and Future Work	47
References	51
Appendix 1 Download and Translate Journals Document and Create Web Pages	53
Appendix 2 Home Pages of Selected IES Document.....	60
Appendix 3 Search Engine for Journals and Conference Documents.....	63

List of Tables

Table 1 POSIX Basic Metacharacter Description	13
Table 2 POSIX Extended Metacharacter Description	15
Table 3 Perl Data Types.....	23
Table 4 IEEE Xplore Search Gateway Query Parameters.....	34

List of Figures

Figure 1 Search Engine English Homepage	44
Figure 2 Search Engine Chinese Homepage.....	44
Figure 3 IEEE Transactions on Industrial Informatics 2012	45
Figure 4 IEEE Transactions on Industrial Informatics 2012 English to Chinese Translation....	45
Figure 5 English Search Results	46
Figure 6 Chinese Search Results.....	46
Figure 7 2014 Industry Application Abstract	47
Figure 8 Google Translation 1	47
Figure 9 IE Dictionary Translation 1	48
Figure 10 2014 Industrial Electronics Abstract	48
Figure 11 Google Translation 2	49
Figure 12 IE Dictionary Translation 2	49

Chapter 1

Introduction

Machine translation (MT) is a subfield of computational linguistics and refers to an automated translation from one natural language into another by way of a computerized system without human intervention [1].

1.1 History of Machine Translation

The development of machine translation has gone through a long and tortuous process. The earliest machine translation concept, which René Descartes proposed a universal language that can be used by different language, originated in the 17th century. There is no conception of translation via computers, until Dr. Warren Weaver who “ started thinking and talking about the possibility of using computer-like machinery for carrying out part of all of the operations involved in translating from one language to another.” [2] However, the first concrete proposals were made in 19th and full-time researcher, Yehosha Bar-Hillel, began his MT research at Massachusetts Institute of Technology (MIT) in 1951. A year later he organized the first machine translation conference, where the sketches of future research were already becoming comprehensible [3]. Then, there is a milestone that the first public demonstration of a MT system was presented at Georgetown University Leon Dostert collaborated with IBM in January 1954. Although it only selected 49 Russian sentences as an sample was translated into English, with a very restricted vocabulary of 250 words and just 6 grammar rules, it was adequately impressive to simulate the large-scale funding of MT research in the United States and to inspire the beginning of MT projects

elsewhere in the world [3]. This time is the start-up period of machine translation, and the approach mainly adopted word replacement based on dictionary translation and some basic grammar rules. In the next decade, it was a period that machine translation developed from the rise to the depression. Researchers continued to join the field as the Association for Machine Translation and Computational Linguistics which was formed in the United States in 1962 and the National Academy of Sciences formed the Automatic Language Processing Advisory Committee (ALPAC) to study MT in 1964. In 1966, ALPAC published a report to examine the prospects of MT, and concluded that the ten-year-long research had failed to fulfill expectations based on the fact that MT was slower, less accurate and more expensive than human translation [3]. The ALPAC report said that “there is no immediate or predictable prospect of useful machine translation” [4] which resulted in intensely reduced funding and took a vital end to MT research in the U.S. for over a decade. During the 1970s and 1980s, with the development of new kinds of interlingua systems and the investigation of artificial intelligence (AI), research interest on machine translation revived [3]. Moreover, as computational power increased and became the cost reduced, more interest was shown in statistical models for machine translation. Meanwhile, various MT companies were launched, including Trados which was the first to develop and market translation memory technology in 1989, and then the first commercial translator came to the market in 1990 [5]. From the 1999 to the present, the statistical machine translation and neural machine translation method has made a breakthrough development that is by far the most widely studied machine translation in recent years, and is still rapidly developing. Finally, in the 21th century, machine translation has become an online service on the internet, and there are some companies do machine translation at the forefront of the industry, the most common used are Google and Microsoft translator.

1.2 Machine Translation General Problems and Approaches

There is a scientific consensus that the understanding of natural language is a key to the success of machine translation. The general problems of MT systems are mainly about the lexical and structural analysis, both within languages (monolingual ambiguity) and between languages (bilingual ambiguity) [3].

It is difficult to translate for any monolingual ambiguity in translation, since there will be more than one possible equivalent. Homographs and polysemes (English cry; cook) must be resolved before translation (Chinese 哭泣 or 叫喊; 烹调 or 厨子); ambiguities of grammatical category (English light as noun, adjective or verb) must likewise be resolved for choice between Chinese 光, 明亮的 or 点燃, etc. Examples of monolingual structural ambiguities exist when a word or phrase can potentially modify more than one part of a sentence. In young boys and girls, the adjective young may refer only to boys or to both boys and girls. Prepositional phrases can modify nearly any previous verb or noun phrase, e.g. (a) The car was driven by the doctor with good skill, (b) The car was driven by the doctor with flawed tyres and (c) The car was driven by the doctor with white hair. Lexical and structural ambiguities may and often do combine: She saw him shaking hands, where shaking can be either an adjective ('hands which were shaking') or a verb component ('that he was shaking hands'). Bilingual lexical ambiguities occur primarily when the target language makes distinctions absent in the source language: English river can be Chinese 江 or 河 (Yangtze River 长江 or Yellow River 黄河); English boil can be Chinese 煮, 沸 or 烧; English fry can be Chinese 煎, 炒 or 炸. A more typical and common example is illustrated by the translation of wear from English to Chinese. English wear can be used to describe any parts of dressing and jewelry on the body no matter hats, coats, pants, shoes or necklace. However, in

Chinese, English wear ought to be translated to 穿 (clothes, pants, shoes) or 戴 (hats, rings, necklace) according to describe different wearing.

There are several common machine translation approaches: rule-based machine translation, dictionary-based machine translation, statistical machine translation, neural machine translation.

The rule-based machine translation paradigm includes transfer-based machine translation, interlingual machine translation and dictionary-based machine translation paradigms. This type of translation is used mostly in the creation of dictionaries and grammar programs. Unlike other methods, RBMT involves more information about the linguistics of the source and target languages, using the morphological and syntactic rules and semantic analysis of both languages. The basic approach involves linking the structure of the input sentence with the structure of the output sentence using a parser and an analyzer for the source language, a generator for the target language, and a transfer lexicon for the actual translation. RBMT's biggest downfall is that everything must be made explicit: orthographical variation and erroneous input must be made part of the source language analyzers in order to cope with it, and lexical selection rules must be written for all instances of ambiguity. Adapting to new domains in itself is not that hard, as the core grammar is the same across domains, and the domain-specific adjustment is limited to lexical selection adjustment.

Machine translation can use a method based on dictionary entries, which means that the words will be translated as a dictionary does – word by word, usually without much correlation of meaning between them. Dictionary lookups may be done with or without morphological analysis or lemmatisation. While this approach to machine translation is probably the least sophisticated, dictionary-based machine translation is ideally suitable for the translation of long lists of phrases

on the subsentential (i.e., not a full sentence) level, e.g. inventories or simple catalogs of products and services.

Statistical machine translation tries to generate translations using statistical methods based on bilingual text corpora, such as the Canadian Hansard corpus, the English-French record of the Canadian parliament and EUROPARL, the record of the European Parliament. Where such corpora are available, good results can be achieved translating similar texts, but such corpora are still rare for many language pairs. The first statistical machine translation software was CANDIDE from IBM. Google used SYSTRAN for several years, but switched to a statistical translation method in October 2007. In 2005, Google improved its internal translation capabilities by using approximately 200 billion words from United Nations materials to train their system; translation accuracy improved. Google Translate and similar statistical translation programs work by detecting patterns in hundreds of millions of documents that have previously been translated by humans and making intelligent guesses based on the findings. Generally, the more human-translated documents available in a given language, the more likely it is that the translation will be of good quality. Newer approaches into Statistical Machine translation such as METIS II and PRESEMT use minimal corpus size and instead focus on derivation of syntactic structure through pattern recognition. With further development, this may allow statistical machine translation to operate off of a monolingual text corpus. SMT's biggest downfall includes it being dependent upon huge amounts of parallel texts, its problems with morphology-rich languages (especially with translating into such languages), and its inability to correct singleton errors.

Neural machine translation (NMT) is an approach to machine translation in which a large neural network is trained by deep learning techniques. It is a radical departure from phrase-based statistical translation approaches, in which a translation system consists of subcomponents that are

separately engineered. Google has announced that its translation services are now using Google Neural Machine Translation (GNMT) in preference to its previous statistical methods. NMT models apply deep representation learning. They require only a fraction of the memory needed by traditional statistical machine translation (SMT) models. Furthermore, unlike conventional translation systems, all parts of the neural translation model are trained jointly (end-to-end) to maximize the translation performance. A bidirectional recurrent neural network (RNN), known as an encoder, is used by the neural network to encode a source sentence for a second RNN, known as a decoder, that is used to predict words in the target language[7].

1.3 Motivation

According to Science and Engineering Indicators 2014, China is now the world's third largest producer of peer-reviewed research articles, and a growing number of authors are from China within industrial electronics journals. However, the difference of language is a barrier between Chinese and western academia and industry. Many Chinese readers need a help of translator to read English papers. The quality of translation is measured depends on how it close to a professional human translation. The closer a machine translation is to a professional human translation, the better it is [6]. The improvement of output quality can be achieved by human intervention. For instance, some systems are able to translate more accurately if the user has unambiguously identified which words in the text are proper names. With the assistance of these techniques, machine translation has proven useful as a tool to assist human translators and it can even produce output that can be used as is in some cases. Several IEEE societies are trying to provide technical information to Chinese engineers. For example, the Power and Energy Society is in the process of translation their journals into Chinese. They are using the manual translation.

The IES tried to translate abstract of their journal and conference papers but none of existing automatic translation were provided adequate results. The purpose of this work is to create an improved English to Chinese translator within industrial electronics field and the same approach can be used in other areas.

1.4 Goals

The English to Chinese automatic translators cannot always satisfy the requirement of users especially in technical fields. Through research, the main reason is wrong translation of technical terminologies. The researchers summarized common mistakes of technical translation. Then developed an IE (Industrial Electronics) dictionary that is a professional technical dictionary correcting the wrong translations by Google translator. With the Python platform, researchers can automatically replaces incorrect translation by Google translator to get improved translations. Finally, we develop web pages to show translation results that can help Chinese engineers read industrial electronic papers and this approach can be also used in other technical areas.

The rest of the thesis is outlines as follows. In the chapter 2, the fundamentals of the regular expression search algorithm; programming techniques: HTML, Perl, Python; Google API Service: Building Applications is reviewed.

In the chapter 3, the general mistakes of traditional technical text translation is analyzed and an industrial electronics dictionary is edited. An English and Chinese search engine for technical texts is developed.

In the chapter 4, the experiment to prove that the translation results of technical contents obtain improved is conducted.

In the chapter 5, major results of the thesis are summarized and further research are discussed.

Chapter 2

Literature Review

This chapter reviews the background information of necessary algorithms and programming techniques to develop an English and Chinese web search engine and illustrates how they work.

2.1 Regular expression search algorithm

Regular expressions are the commonly used and powerful tool for text processing. In theoretical computer science and formal language theory, a regular expression is a sequence of characters that define a search pattern. In most cases, this pattern like a mini programming language which allow users to parse text, and especially used by string searching algorithms for finding and replacing operations on strings [8].

The concept of regular expressions originated in the 1950s, when the American mathematician Stephen Cole Kleene formalized the description of a regular language using his mathematical notation called regular sets. The concept came into common use with Unix text-processing utilities. Today, different syntaxes for writing regular expressions exist, one being the POSIX standard and another, widely used, being the Perl syntax.

Currently, regular expressions are widely supported in programming languages, text processing programs advanced text editors, and some other programs. Regex support is a part of the standard library of many programming languages, including Java and Python, and is built into the syntax of others, including Perl and ECMAScript. Implementations of regex functionality is often called a regex engine, and a number of libraries are available for reuse.

2.1.1 Basic concepts

A regular expression, often called a pattern, is an expression used to specify a set of strings required for a particular purpose. A simple way to specify a finite set of strings is to list its elements or members. However, there are often more concise ways to specify the desired set of strings. For example, the set containing the three strings "Handel", "Händel", and "Haendel" can be specified by the pattern `H(ä|æ|e?)ndel`; It shows that this pattern matches each of the three strings. In most formalisms, if there exists at least one regular expression that matches a particular set then there exists an infinite number of other regular expressions that also match it—the specification is not unique. Most formalisms provide the following operations to construct regular expressions.

Boolean "or": A vertical bar separates alternatives. For example, `bed|bad` can match "bed" or "bad". **Grouping:** parentheses are used to define the scope and precedence of the operators (among other uses). For example, `bed|bad` and `b(a|e)d` are equivalent patterns which both describe the set of "bad" or "bed". **Quantification:** A quantifier after a token (such as a character) or group specifies how often that preceding element is allowed to occur. The most common quantifiers are the question mark `?`, the asterisk `*`, and the plus sign `+`.

The question mark `“?”` indicates zero or one occurrences of the preceding element. For example, `colou?r` matches both "color" and "colour". The asterisk `“*”` indicates zero or more occurrences of the preceding element. For example, `de*f` matches "df", "def", "deef", "deef", and so on. The plus sign `“+”` indicates one or more occurrences of the preceding element. For example, `de+f` matches "def", "deef", "deef", and so on, but not "df".

$\{x\}$ The preceding item is matched exactly x times. $\{\min,\}$ The preceding item is matched \min or more times. $\{\min,\max\}$ The preceding item is matched at least \min times, but not more than \max times.

These constructions can be combined to form arbitrarily complex expressions, much like one can construct arithmetical expressions from numbers and the operations $+$, $-$, \times , and \div .

2.1.2 Formal language theory

Regular expressions is good at describing regular languages in formal language theory.

Formal definition

Regular expressions consist of constants and operator symbols that mean sets of strings and operations over these sets, individually. Given a finite alphabet Σ , the following constants are defined as regular expressions:

(empty set) \emptyset denoting the set \emptyset .

(empty string) ε denoting the set containing only the "empty" string, which has no characters at all.

(literal character) a in Σ denoting the set containing only the character a .

Given regular expressions R and S , the following operations over them are defined to produce regular expressions:

(concatenation) RS denotes the set of strings that can be obtained by concatenating a string in R and a string in S . For example, $\{"op", "q"\}\{"r", "st"\} = \{"opr", "opst", "qr", "qst"\}$.

(alternation) $R | S$ denotes the set union of sets described by R and S . For example, if R describes $\{"de", "f"\}$ and S describes $\{"de", "g", "hi"\}$, expression $R | S$ describes $\{"de", "f", "g", "hi"\}$.

2.1.3 Syntax

A regex pattern matches a target string. The pattern is composed of a sequence of atoms. An atom is a single point within the regex pattern which it tries to match to the target string. The simplest atom is a literal, but grouping parts of the pattern to match an atom will require using $()$ as metacharacters. Metacharacters help form: atoms; quantifiers telling how many atoms (and whether it is a greedy quantifier or not); a logical OR character, which offers a set of alternatives, and a logical NOT character, which negates an atom's existence; and backreferences to refer to previous atoms of a completing pattern of atoms. A match is made, not when all the atoms of the string are matched, but rather when all the pattern atoms in the regex have matched. The idea is to make a small pattern of characters stand for a large number of possible strings, rather than compiling a large list of all the literal possibilities.

Depending on the regex processor there are about fourteen metacharacters, characters that may or may not have their literal character meaning, depending on context, or whether they are "escaped", i.e. preceded by an escape sequence, in this case, the backslash \backslash . Modern and POSIX extended regexes use metacharacters more often than their literal meaning, so to avoid "backslash-osis" it makes sense to have a metacharacter escape to a literal mode; but starting out, it makes

more sense to have the four bracketing metacharacters () and { } be primarily literal, and "escape" this usual meaning to become metacharacters. Common standards implement both. The usual metacharacters are {}[]()^\$.|*+? and \. The usual characters that become metacharacters when escaped are dswDSW and N.

Delimiters

When entering a regex in a programming language, they may be represented as a usual string literal, hence usually quoted; this is common in C, Java, and Python for instance, where the regex `re` is entered as `"re"`. However, they are often written with slashes as delimiters, as in `/re/` for the regex `re`. This originates in `ed`, where `/` is the editor command for searching, and an expression `/re/` can be used to specify a range of lines (matching the pattern), which can be combined with other commands on either side, most famously `g/re/p` as in `grep` ("global regex print"), which is included in most Unix-based operating systems, such as Linux distributions. A similar convention is used in `sed`, where search and replace is given by `s/re/replacement/` and patterns can be joined with a comma to specify a range of lines as in `/re1/,re2/`. This notation is particularly well-known due to its use in Perl, where it forms part of the syntax distinct from normal string literals. In some cases, such as `sed` and Perl, alternative delimiters can be used to avoid collision with contents, and to avoid having to escape occurrences of the delimiter character in the contents.

Standards

The IEEE POSIX standard has three sets of compliance: BRE (Basic Regular Expressions), ERE (Extended Regular Expressions), and SRE (Simple Regular Expressions). SRE is

deprecated, in favor of BRE, as both provide backward compatibility. The subsection below covering the character classes applies to both BRE and ERE.

Perl regexes have a rich and powerful set of atomic expressions. Perl has no "basic" or "extended" levels, where the () and { } may or may not have literal meanings. They are always metacharacters, as they are in "extended" mode for POSIX. To get their literal meaning, you escape them. Other metacharacters are known to be literal or symbolic based on context alone. Perl offers much more functionality: "lazy" regexes, backtracking, named capture groups, and recursive patterns, all of which are powerful additions to POSIX BRE/ERE.

2.1.3.1 POSIX basic and extended

In the POSIX standard, Basic Regular Syntax (BRE) requires that the metacharacters () and { } be designated `\(` and `\{`, whereas Extended Regular Syntax (ERE) does not.

Table 1: POSIX Basic Metacharacter Description

Metacharacter	Description
.	Matches any single character (many applications exclude newlines, and exactly which characters are considered newlines is flavor-, character-encoding-, and platform-specific, but it is safe to assume that the line feed character is included). Within POSIX bracket expressions, the dot character matches a literal dot. For example, <code>d.f</code> matches "def", etc., but <code>[d.f]</code> matches only "d", ".", or "f".

[]	<p>A bracket expression. Matches a single character that is contained within the brackets. For example, [def] matches "d", "e", or "f". [a-z] specifies a range which matches any lowercase letter from "a" to "z". The - character is treated as a literal character if it is the last or the first (after the ^, if present) character within the brackets: [abc-], [-abc]. Note that backslash escapes are not allowed. The] character can be included in a bracket expression if it is the first (after the ^) character: []abc].</p>
[^]	<p>Matches a single character that is not contained within the brackets. For example, [^abc] matches any character other than "a", "b", or "c". [^a-z] matches any single character that is not a lowercase letter from "a" to "z". Likewise, literal characters and ranges can be mixed.</p>
^	<p>Matches the starting position within the string. In line-based tools, it matches the starting position of any line.</p>
\$	<p>Matches the ending position of the string or the position just before a string-ending newline. In line-based tools, it matches the ending position of any line.</p>
()	<p>Defines a marked subexpression. The string matched within the parentheses can be recalled later (see the next entry, \n). A marked subexpression is also called a block or capturing group. BRE mode requires \(\).</p>

\n	Matches what the nth marked subexpression matched, where n is a digit from 1 to 9. This construct is vaguely defined in the POSIX.2 standard. Some tools allow referencing more than nine capturing groups.
*	Matches the preceding element zero or more times. For example, ab*c matches "ac", "abc", "abbbc", etc. [xyz]* matches "", "x", "y", "z", "zx", "zyx", "xyzy", and so on. (ab)* matches "", "ab", "abab", "ababab", and so on.
{m,n}	Matches the preceding element at least m and not more than n times. For example, a{3,5} matches only "aaa", "aaaa", and "aaaaa". This is not found in a few older instances of regexes. BRE mode requires \{m,n\}.

2.1.3.2 POSIX extended

The meaning of metacharacters escaped with a backslash is reversed for some characters in the POSIX Extended Regular Expression (ERE) syntax. With this syntax, a backslash causes the metacharacter to be treated as a literal character. So, for example, \(\) is now () and \{\} is now { }. Additionally, support is removed for \n backreferences and the following metacharacters are added:

Table 2: POSIX Extended Metacharacter Description

Metacharacter	Description
---------------	-------------

?	Matches the preceding element zero or one time. For example, <code>ab?c</code> matches only "ac" or "abc".
+	Matches the preceding element one or more times. For example, <code>ab+c</code> matches "abc", "abbc", "abbbc", and so on, but not "ac".
	The choice (also known as alternation or set union) operator matches either the expression before or the expression after the operator. For example, <code>abc def</code> matches "abc" or "def".

2.2 Programming techniques: HTML, Perl, Python

The demonstration of this project is primarily based on the web pages and web applications. Therefore, it is necessary to use programming techniques to achieve this work. First of all, HyperText Markup Language (HTML) is the programming language which we should learn and master.

The following sections describe the details of these three programming languages.

2.2.1 HyperText Markup Language

HyperText Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS), and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a webserver or from local storage and render them into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document [9].

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects, such as interactive forms, may be embedded into the rendered page. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as `` and `<input />` introduce content into the page directly. Others such as `<p>...</p>` surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

Markup

HTML markup consists of several key components, including those called tags (and their attributes), character-based data types, character references and entity references. HTML tags most commonly come in pairs like `<h1>` and `</h1>`, although some represent empty elements and so are unpaired, for example ``. The first tag in such a pair is the start tag, and the second is the end tag (they are also called opening tags and closing tags).

Another important component is the HTML document type declaration, which triggers standards mode rendering.

The following is an example of the classic Hello world program, a common test employed for comparing programming languages, scripting languages and markup languages. This example is made using 9 lines of code:

```
<!DOCTYPE html>
```



```
<html>

<head>

  <title>This is a title here</title>

</head>

<body>

  <p>Hello world!</p>

</body>

</html>
```

(The text between `<html>` and `</html>` describes the web page, and the text between `<body>` and `</body>` is the visible page content. The markup text "`<title>This is a title</title>`" defines the browser page title.)

Elements

HTML documents imply a structure of nested HTML elements. These are indicated in the document by HTML tags, enclosed in angle brackets thus: `<p>`.

In the simple, general case, the extent of an element is indicated by a pair of tags: a "start tag" `<p>` and "end tag" `</p>`. The text content of the element, if any, is placed between these tags.

Some elements, such as the line break `
`, do not permit any embedded content, either text or further tags. These require only a single empty tag (akin to a start tag) and do not use an end tag. Many tags, particularly the closing end tag for the very commonly used paragraph element `<p>`, are optional. An HTML browser or other agent can infer the closure for the end of an element from the context and the structural rules defined by the HTML standard. These rules are complex and not widely understood by most HTML coders.

Element examples

Header of the HTML document: `<head>...</head>`. The title is included in the head, for example:

```
<head>

<title>The Title is Here</title>

</head>
```

Headings: HTML headings are defined with the `<h1>` to `<h6>` tags:

Paragraphs:

```
<p>Paragraph 1</p> <p>Paragraph 2</p>
```

Line breaks:`
`. The difference between `
` and `<p>` is that "br" breaks a line without altering the semantic structure of the page, whereas "p" sections the page into paragraphs. Note also that "br" is an empty element in that, although it may have attributes, it can take no content and it may not have an end tag.

`<p>This
 is a paragraph
 with
 line breaks</p>`

Comments:

`<!-- This is a comment -->`

Comments can help in the understanding of the markup and do not display in the webpage.

Hypertext markup makes parts of a document into links to other documents

An anchor element creates a hyperlink in the document and its href attribute sets the link's target URL. For example, the HTML markup, `Google Search`, will render the word " Google Search" as a hyperlink. To render an image as a hyperlink, an "img" element is inserted as content into the "a" element. Like "br", "img" is an empty element with attributes but no content or closing tag. ``.

Attributes

Most of the attributes of an element are name-value pairs, separated by "=" and written within the start tag of an element after the element's name. The value may be enclosed in single or double quotes, although values consisting of certain characters can be left unquoted in HTML (but not XHTML). Leaving attribute values unquoted is considered unsafe. In contrast with name-value pair attributes, there are some attributes that affect the element simply by their presence in the start tag of the element, like the ismap attribute for the img element. There are several common attributes that may appear in many elements:

The id attribute provides a document-wide unique identifier for an element. This is used to identify the element so that stylesheets can alter its presentational properties, and scripts may alter, animate or delete its contents or presentation. Appended to the URL of the page, it provides a globally unique identifier for the element, typically a sub-section of the page. The class attribute provides a way of classifying similar elements. This can be used for semantic or presentation purposes. For example, an HTML document might semantically use the designation class="notation" to indicate that all elements with this class value are subordinate to the main text of the document. In presentation, such elements might be gathered together and presented as footnotes on a page instead of appearing in the place where they occur in the HTML source. Class attributes are used semantically in microformats. Multiple class values may be specified; for example class="notation important" puts the element into both the "notation" and the "important" classes.

An author may use the style attribute to assign presentational properties to a particular element. It is considered better practice to use an element's id or class attributes to select the element from within a stylesheet, though sometimes this can be too cumbersome for a simple, specific, or ad hoc styling.

The title attribute is used to attach subtextual explanation to an element. In most browsers this attribute is displayed as a tooltip.

Data types

HTML describes several data types for element content, such as script data and stylesheet data, and a plethora of types for attribute values, including IDs, names, URIs, numbers, units of length,

languages, media descriptors, colors, character encodings, dates and times, and so on. All of these data types are specializations of character data.

Document type declaration

HTML documents are required to start with a Document Type Declaration (informally, a "doctype"). In browsers, the doctype helps to define the rendering mode—particularly whether to use quirks mode.

The original purpose of the doctype was to enable parsing and validation of HTML documents by SGML tools based on the Document Type Definition (DTD). The DTD to which the DOCTYPE refers contains a machine-readable grammar specifying the permitted and prohibited content for a document conforming to such a DTD. Browsers, on the other hand, do not implement HTML as an application of SGML and by consequence do not read the DTD.

This declaration references the DTD for the "strict" version of HTML 4.01. SGML-based validators read the DTD in order to properly parse the document and to perform validation. In modern browsers, a valid doctype activates standards mode as opposed to quirks mode.

2.2.2 Perl

Perl is a popular dynamic programming language, extremely rich with regular expressions, freely and readily obtainable, easily approachable by the beginner, and available for a remarkably wide variety of platforms, including pretty much all flavors of Windows, Unix, and the Mac. The way you wield Perl to solve a problem is different from traditional languages. The overall layout of a Perl program often uses traditional structured and object-oriented concepts, but data

processing often relies heavily on regular expressions. In fact, I believe it is safe to say that regular expressions play a key role in virtually all Perl programs [10].

2.2.2.1 Perl's Strength and Weakness

The richness of variety and options among Perl's operators and functions is perhaps its greatest feature. They can change their behavior depending on the context in which they're used, often doing just what the author naturally intends in each differing situation. In fact, O'Reilly's *Programming Perl* goes so far as to boldly state "In general, Perl operators do exactly what you want...." The regex match operator `m/regex/`, for example, offers an amazing variety of different functionality depending upon where, how, and with which modifiers it is used [10].

This concentrated richness in expressive power is also one of Perl's least-attractive features. There are innumerable special cases, conditions, and contexts that seem to change out from under you without warning when you make a subtle change in your code that you've just hit another special case you weren't aware of. The *Programming Perl* quote in the previous paragraph continues "...unless you want consistency" [10].

2.2.2.2 Perl Data Types

Perl has a number of fundamental data types. The most commonly used and discussed are scalars, arrays, hashes, filehandles, and subroutines:

Table 3: Perl Data Types

Type	Sigil	Example	Description
------	-------	---------	-------------

Scalar	\$	\$foo	A single value; it may be a number, a string, a filehandle, or a reference.
Array	@	@foo	An ordered collection of scalars.
Hash	%	%foo	A map from strings to scalars; the strings are called keys, and the scalars are called values. Also known as an associative array.
Filehandle	none	\$foo or FOO	An opaque representation of an open file or other target for reading, writing, or both.
Subroutine	&	&foo	A piece of code that may be passed arguments, be executed, and return data.
Typeglob	*	*foo	The symbol table entry for all types with the name 'foo'.

2.2.2.3 Common Gateway Interface

In computing, Common Gateway Interface (CGI) proposals a standard protocol for web servers to execute programs that execute like console applications (also called command-line interface programs) running on a server that generates web pages dynamically. Such programs are known as CGI scripts or simply as CGIs. The specifics of how the script is executed by the server are determined by the server. In the common case, a CGI script essentially executes at the time a request is made and generates HTML.

2.2.2.3.1 Purpose of the CGI standard

Each web server runs HTTP server software, which responds to requests from web browsers. Usually, the HTTP server has a directory (folder), which is chosen as a document collection — files that can be sent to Web browsers connected to this server.[8] For example, if the Web server has the domain name example.com, and its document collection is stored at /usr/local/apache/htdocs in the local file system, then the Web server will respond to a request for http://example.com/index.html by sending to the browser the (pre-written) file /usr/local/apache/htdocs/index.html.

For pages constructed on the fly, the server software may defer requests to separate programs and relay the results to the requesting client (usually, a web browser that displays the page to the end user). In the early days of the web, such programs were usually small and written in a scripting language; hence, they were known as scripts.

Originally, diverse server software would use different ways to exchange this information with scripts. As a result, it wasn't possible to write scripts that would work unchanged for different server software, even though the information being exchanged was the same. Therefore, it was decided to establish a standard way for exchanging this information: CGI (the Common Gateway Interface, as it defines a common way for server software to interface with scripts). Web pages generating programs invoked by server software that run according to the CGI standard are named as CGI scripts.

An early use of CGI scripts was to process forms. HTML forms typically had an "action" attribute and a button designated as the "submit" button. When the submit button is pressed the URI specified in the "action" attribute would be sent to the server with the information from the

form sent as a query string. If the "action" identifies a CGI script then the CGI script would be executed and it would create a HTML page.

2.2.2.3.2 Using CGI scripts

A web server allows its owner to configure which URLs shall be handled by which CGI scripts. This is usually done by marking a directory within the document collection as containing CGI scripts - its name is often cgi-bin. For example, /usr/local/apache/htdocs/cgi-bin could be designated as a CGI directory on the web server. When a Web browser requests a URL that points to a file within the CGI directory then, instead of simply sending that file to the Web browser, the HTTP server runs the specified script and passes the output of the script to the Web browser. That is, anything that the script sends to standard output is passed to the Web client instead of being shown on-screen in a terminal window.

When a web server executes a CGI script it provides input to the console/shell program using environment variables or "standard input". Standard input is like typing data into a console/shell program; in the case of a CGI script, the web server does the typing. The CGI script writes data out to "standard output" and that output is sent to the client (the web browser) as a HTML page. Standard output is like the output you see in a console/shell program except the web server reads it and sends it out.

2.2.3 Python

Python is a mostly used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy which emphasizes code readability (particularly using whitespace indentation

to delimit code blocks rather than curly braces or keywords), and a syntax which allows programmers to express concepts in fewer lines of code than in other languages such as C++ or Java. The language provides builds intended to enable writing concise programs on both a small and large scale [10].

Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library.

2.2.3.1 The Advantages of Python:

Python is robust: Python is solid and powerful. Python has a relative small quantity of lines of code, which makes it less prone to issues, easier to debug, and more maintainable [18].

Python is flexible: Python is used in a wide array of industries and for a long list of different usages, from websites and web applications to systems administration, voice over IP, and desktop apps.

Python is easy to learn and use: "Python in particular emerges as a near ideal candidate for a first programming language", says John M. Zelle, in the Department of Mathematics, Computer Science, and Physics at Wartburg College in Iowa. People do not have to look up references frequently, nor overwhelmed by the formalities of the language, like they would in Java or C++.

Python reduces time to market: Python makes it possible to get applications to market faster in part due to the fact that it has a huge standard library and is often referred to as coming with

"batteries included". In addition, Python stays out of my way. Therefore I can be more productive than if I was using Java/XML: the same task will require less code using Python.

Python can be compiled into executable files. A Python program is compiled automatically by the interpreter into platform independent byte code which is then interpreted [12].

Python is fast: The average run time of Python is shorter than Perl and PHP. [13].

2.3 Google API Service: Building Applications

Application program interface (API) is a series of routines, protocols, and tools for creating software applications. An API identifies how software components should interact. Moreover, APIs are used when programming graphical user interface (GUI) components. Google Cloud Translation API provides a simple programmatic interface for translating an arbitrary string into another supported language. Translation API is highly responsive, so websites and applications can integrate with Translation API for fast, dynamic translation of source text from the source language to a target language (e.g., English to Chinese). Language detection is also available in cases where the source language is unidentified. The fundamental technology pushes the boundary of Machine Translation and is updated constantly to improve translations and present new languages and language pairs.

Chapter 3

Industrial Electronics (IE) Dictionary

The researchers summarized common mistakes of technical translation. Then developed an IE (Industrial Electronics) dictionary that is a professional technical dictionary correcting the wrong translations by Google translator [18].

3.1 General mistakes of traditional technical text translation

Through the analysis of 15,000 technical terms' translation. There are several mainly mistakes which Google translation usually has as following:

A Technical Terminology mistakes

The Google translation can hardly tell some technical terminology. For example, 'multi-agent', Google translate into '多代理', actually it should be '多智能' in Chinese.

B Length

Some phrases of Google translation in Chinese are too long or too simple. For instance, 'orthogonal array', Google translate into '正交', it is better to translate into '正交矩阵' or '正交列阵' [14]. The missing of translation of 'array' might lead to misunderstand. This is an example of translation too simple.

There is an example of translation too long. ‘Permanent magnet array’ Google translate into ‘永久磁体列阵’, this translation is too tedious. Due to the reference, it is better to translate into ‘阵列永磁’ [15].

C Grammar

Some Google translations do not match Chinese grammar. The most common mistake is confusion of adjective with noun or verb with noun. For instance, ‘inverter’ Google translates into ‘逆变’, it is totally wrong in Chinese grammar which confuse the noun with verb. The right translation is ‘逆变器’

Another mistake is misuse ‘the’ in Chinese. For example, ‘power imbalance’ Google translate into ‘功率的不平衡’, better translation is ‘功率不平衡’.

D Abbreviation

Some professional abbreviation Google cannot tell. For example, ‘MUD’ is abbreviation of ‘Multiple User Dimension’, it is also a common word ‘mud’. Google chooses the most common meaning of this word then translates into ‘泥巴’, which actually should be ‘多用户层面’ [16].

E Strange Translation

There are some translation of Google is obviously strange. For example, ‘nonlocal means’ Google translate into ‘非本地均值’, which actually is ‘非局部均值’ [17]. Another example, ‘open shop’, Google translates into ‘开网店’, which means ‘open online shop’ in English. The right translation is ‘开店’.

3.2 Proposed Solution of IE Dictionary

It is known that the main problem that leads to bad translation results is due to wrong translation of technical terminologies. If there is a professional technical dictionary which contains the accurate translations, it will be helpful to solve this problem. Therefore, it is necessary to make an industrial electronics (IE) dictionary.

In order to write the IE dictionary, the researchers consult the professional technical Chinese dictionary, search the journals and conferences papers to get the correct translation results. After reference and comparison, a better professional technical dictionary is made that is a Chinese to Chinese technical terminologies dictionary based on wrong translation terms by Google translation.

With the aim of improving the quality of translation, the researchers use the IE dictionary that replace the incorrect translation by Google translator. By this method, the translation results can be improved. Then we develop web pages to show translation results that can easily see the translation results and compare new translation results to Google translation. According to results, it is proved that the new translations based on IE dictionary is improved. With improvement of translation, Chinese engineers can get better understanding when they read industrial electronic papers and this approach can be also used in other technical areas.

3.3 Related Work

In this section, first we describe the work steps of making IE dictionary. There are details about how to achieve this job. Next, we build our webpages to show the translation results in several IEEE journals and conferences papers such as IEEE Transactions on Industrial

Electronics, IEEE Transactions on Industrial Informatics, Industrial Electronics Society Annual Conference, etc. According to compare and analyze these webpages translation results, we can find that the English to Chinese translation of technical text is improved.

3.3.1. Making the IE dictionary

First of all, we get keywords list by finding match words using a common technical terminology list comparing to 20000 abstracts and titles. There is a common technical terminology list that contains more than 15000 technical words or phrases within industrial electronics. It is feasible to download more than 20000 abstracts and titles through IEEE Xplore Search Gateway using Python platform. IEEE Xplore is a digital library providing full text access to the world's highest quality technical literature in electrical engineering, computer science, electronics, and related disciplines. IEEE Xplore contains full-text documents from IEEE journals, magazines, conference proceedings, and standards, as well as eBooks and educational courses. From IEEE Xplore Search Gateway, users can query parameters about articles like article number, author, abstract, publication number and year, etc. and download these information as XML files. Then extract the titles and abstracts saved as text files which are easy to compare to a technical terminology list by writing programs on Python. So that a keywords list can be generated.

Secondly, the keywords list can be translated by Google translator API by Python. With Google translate, people can dynamically translate text between thousands of language pairs. The Google translate API lets websites and programs integrate with Google translate programmatically. It can be realized by the programs to use Google translate API through Python platform.

Finally, it is doable to find the wrong translation and manually correct them by consulting the professional technical Chinese dictionary, search the journals and conferences papers to get the correct translation results to develop IE dictionary. By this method, a fixed technical dictionary can be generated which correct the inaccurate translation by Google translator.

3.3.2. Webpages and search engine development

At first, it is necessary to download IEEE journals and conferences papers by using IEEE Xplore Search Gateway on Python platform. Then parse the files and extract required information like titles, abstracts, authors, publish year, links, etc. saved as html format files that all could be done by Python programming. Next, translate all the titles and abstracts by Google translatorAPI. After that we can get automatic improvement of the translation in key words using a developed English to Chinese IE dictionary. Hence, we can get both English and improved Chinese titles and abstracts webpages of IEEE journals and conferences. Finally, a CGI script that can be made by perl is an essential part to achieve searching function and generate HTML webpages.

Chapter 4

Code Description

4.1 Download Titles and Abstracts by IEEE Xplore Search Gateway

IEEE Xplore is a digital library providing full text access to the world's highest quality technical literature in electrical engineering, computer science, electronics, and related disciplines. IEEE Xplore contains full-text documents from IEEE journals, magazines, conference proceedings, and standards, in addition to eBooks and educational courses. The query parameters can be found at <http://ieeexplore.ieee.org/gateway/>.

Table 4: IEEE Xplore Search Gateway Query Parameters

Parameter	Description	Boolean query field
Parameter specifying single article		
an	Article number	"Article Number"
Parameters specifying search fields and terms		
au	Terms to search for in Author	Author
ti	Terms to search for in Document Title	"Document Title"
ab	Terms to search for in Abstract	Abstract
doi	Terms to search for in DOI	DOI
cs	Terms to search for in Affiliations	"Author Affiliation"
jn	Terms to search for in Publication Title	"Publication Title"
isbn	Terms to search for in isbn	"isbn"
issn	Terms to search for in issn	"issn"
py	Terms to search for in Publication Year	"Publication Year"
partnum	Terms to search by Part Number	"Part Number"
thsrterms	Terms to search for in Thesaurus Terms	"Thesaurus Terms"
cntrlterms	Terms to search for in Controlled Index Terms	"Inspec Controlled Terms"
idxterms	Terms to search for in Index Terms	"Search Index Terms"
md	Terms to search for in all configured metadata fields and abstract. Accepts complex queries involving field names and boolean operators.	
querytext	Terms to search for in all configured metadata fields, abstract and document text. Accepts complex queries involving field names and boolean operators.	
Filtering parameters		

oa	Open Access only (1 - true; 0 - false)
pn	Publication number
pys	Start value of Publication Year to restrict results by.
pye	End value of Publication Year to restrict results by.
pu	Publisher. One of: IEEE/AIP/IET/AVS/IBM
ctype	Content Type. One of: Conferences/Journals/Books/Early Access/Standards/Educational Courses
Sorting parameters	
sortfield	Field name on which to sort. One of: au/ti/cs/jn/an/py
sortorder	asc (for ascending sort) or desc (for descending sort)
Paging parameters	
hc	Number of records to fetch. Default: 25; Maximum: 1000
rs	Sequence number of first record to fetch. Default: 1
Open Facets parameters	
facet	Open Facet
Open Facets Possible Values	
d-au	Open Author Facet
d-year	Publication Year Facet
d-pubtype	Content Type Facet
d-publisher	Publisher Facet

Related codes to define all the journals and conference:

```
#pyquery allows user to make jquery queries on xml documents.
from pyquery import PyQuery as pq
#Query Parameters: pn, society url, early access vol number, startyear
TransCode = {'IEM': [4154573, 'http://iee-ies.org/', 0, 2007],
             'TIE': [41, 'http://iee-ies.org/', 62, 1954],
             'TII': [9424, 'http://iee-ies.org/', 11, 2005]}

confInfo = {'IECON': {'name': 'Industrial Electronics Society, Annual
Conference of',
                    'code': 1000352,
                    'baseYear': 1974,
                    'society_url': 'http://iee-ies.org/',
                    'year': {2014: [7036020, 'Oct.29 - Nov.1 '],
                             2013: [6683943, '10-13 Nov. '],
                             2012: [6373889, '25-28 Oct. '],
                             2011: [6109934, '7-10 Nov. '],
                             2010: [5661635, '7-10 Nov. ']}},
            'ISIE': {'name': 'Industrial Electronics (ISIE), IEEE
International Symposium on',
                    'code': 1000354,
                    'baseYear': 1991,
```

```

        'society_url': 'http://iee-ies.org/',
        'year': {2014: [6851787, '1-4 June '],
                 2013: [6554304, '28-31 May '],
                 2012: [6230783, '28-31 May '],
                 2011: [5976319, '27-30 June '],
                 2010: [5609073, '4-7 July ']}},
    'ICIT': {'name': 'Industrial Technology (ICIT), IEEE
International Conference on',
            'code': 1000355,
            'baseYear': 1993,
            'society_url': 'http://iee-ies.org/',
            'year': {2015: [7108493, '17-19 March'],
                     2014: [6880451, 'Feb. 26 - March 1 '],
                     2013: [6495638, '25-28 Feb. '],
                     2012: [6198915, '19-21 March '],
                     2011: [5750115, '14-16 March '],
                     2010: [5465897, '14-17 March ']}},
    'INDIN': {'name': 'IEEE International Conference on Industrial
Informatics (INDIN)',
             'code': 1001443,
             'baseYear': 2002,
             'society_url': 'http://iee-ies.org/',
             'year': {2014: [6926648, '27-30 July '],
                      2013: [6599026, '29-31 July '],
                      2012: [6294017, '25-27 July '],
                      2011: [6029399, '26-29 July '],
                      2010: [5538458, '13-16 July ']}},
    'ETFA': {'name': 'Emerging Technologies and Factory Automation
(ETFA), International Conference on',
            'code': 1000260,
            'baseYear': 1995,
            'society_url': 'http://iee-ies.org/',
            'year': {2014: [6994138, '16-19 Sept. '],
                     2013: [6636144, '10-13 Sept. '],
                     2012: [6479732, '17-21 Sept. '],
                     2011: [6045288, '5-9 Sept. '],
                     2010: [5623495, '13-16 Sept. ']}},
    'AIM': {'name': 'Advanced Intelligent Mechatronics, IEEE/ASME
(AIM) International Conference on',
           'code': 1000006,
           'baseYear': 1996,
           'society_url': 'http://iee-ies.org/',
           'year': {2015: [7190945, '7-11 July '],
                    2014: [6869121, '8-11 July '],
                    2013: [6578014, '9-12 July '],
                    2012: [6257544, '11-14 July '],
                    2011: [6019107, '3-7 July '],
                    2010: [5685170, '6-9 July ']}
}

```

4.2 Translate English Technical Text to Chinese by Goslate

Goslate is an official translation API which can provide users free Python API to Google translation service by querying Google translation website. Related codes to translate English content into Chinese:

```
import goslate

gs = goslate.Goslate(service_urls=['http://translate.Google.de']); #for
large-scale translation

if(document.find('title') is not None):
    tmp = document.find('title').text
    if(tmp):
        title = tmp
        time.sleep(3)
        title_zh = gs.translate(title,'zh')
if(document.find('abstract') is not None):
    tmp = document.find('abstract').text
    if(tmp):
        abstract = tmp
        abstract=HTMLParser().unescape(abstract) # transfer into
normal characters
        #chinese version
        abstract_re = abstract.replace("&#x201C;", "")
        abstract_re = abstract_re.replace(", &#x201D;", "")
        abstract_re = abstract_re.replace("&#x201D;", "")
        abstract_re = abstract_re.replace("# ", "No.")
        abstract_re = abstract_re.replace(" & ", " and ")
        time.sleep(3)
        abstract_zh = gs.translate(abstract_re,'zh')
```

4.3 Parse the document

XML is an integrally hierarchical data format, and the most natural way to represent it is with a tree. ET has two classes for this purpose - ElementTree represents the whole XML document as a tree, and Element represents a single node in this tree. Interactions with the whole document (reading and writing to/from files) are usually finished on the ElementTree level. Interactions with a single XML element and its sub-elements are completed on the Element level. Related codes to parse the document:

```
try:
```

```

import xml.etree.cElementTree as ET
except ImportError:
import xml.etree.ElementTree as ET

# parse the document, extract requited data
for document in Sorted_doc:
#root.findall('document'):
title = ''
authors = ''
pubtitle = ''
vol = ''
issue = ''
spage = ''
epage = ''
abstract = ''
abslink = ''
pdflink = ''
abstract_zh=''

if(document.find('title') is not None):
tmp = document.find('title').text
if(tmp):
title = tmp
time.sleep(3)
title_zh = gs.translate(title,'zh')

if(document.find('authors') is not None):
tmp = document.find('authors').text
if(tmp):
authors = ReOrder(tmp)

if(document.find('pubtitle') is not None):
tmp = document.find('pubtitle').text
if(tmp):
pubtitle = tmp
#change position between ','
if pubtitle.find(','):
temp_f=pubtitle[0:pubtitle.find(',')]
temp_b=pubtitle[pubtitle.find(',')+1:len(pubtitle)]
pubtitle=temp_b+' '+temp_f

if(document.find('volume') is not None):
tmp = document.find('volume').text
if(tmp):
vol = tmp

if(document.find('issue') is not None):
tmp = document.find('issue').text
if(tmp):
issue = tmp

if(document.find('spage') is not None):
tmp = document.find('spage').text
if(tmp):
spage = tmp

if(document.find('epage') is not None):

```

```

tmp = document.find('epage').text
if(tmp):
    epage = tmp

if(document.find('abstract') is not None):
    tmp = document.find('abstract').text
    if(tmp):
        abstract = tmp
        abstract=HTMLParser().unescape(abstract) # transfer into
normal characters
        #chinese version
        abstract_re = abstract.replace("&#x201C;", "")
        abstract_re = abstract_re.replace("&#x201D;", "")
        abstract_re = abstract_re.replace("&#x201E;", "")
        abstract_re = abstract_re.replace("# ", "No.")
        abstract_re = abstract_re.replace(" & ", " and ")

        time.sleep(3)
        abstract_zh = gs.translate(abstract_re, 'zh')
        #w.write(authors + ',' + '' + title + '' + ' ' +
pubtitle+' '+ py +',pp.' + spage + '-' + epage + '\n' + '\n' + ' ' +abstract
+ '\n'+'\n')

if(document.find('mdurl') is not None):
    tmp = document.find('mdurl').text
    if(tmp):
        abslink = tmp

if(document.find('pdf') is not None):
    tmp = document.find('pdf').text
    if(tmp):
        pdflink = tmp
#transfer ASCII into UTF-8
reload(sys)
sys.setdefaultencoding('utf-8')

```

4.4 Make Valid Web Pages and Save Them

```

# make valid (not empty) web pages to html and save to temp folder
if(vol != '' and spage.startswith('C')==False and vol != 'PP' and
issue != '' and abstract !='' and abstract_zh!='' and authors!=''): #trans
has papers
    fname = '../temp/' + vol + '_' + issue + '.htm'
    fname_zh = '../temp/' + vol + '_' + issue + '_zh.htm'
    counter[int(issue)-1] += 1
    volNum = vol
    print 'Volume #:'+' '+ vol
    # make html files according to issue number of papers
    print issue
    print issueCollect
    if issue not in issueCollect:
        issueCollect.append(issue)
    if op.isfile(fname):

```

```

line
        fh = open(fname, 'a') #'a' means appending at the last
        fh_zh = open(fname_zh, 'a')
        # if issue is not in the issueCollect then create new files as
following format
    else:
        if(abslink):
            d = pq(abslink)
            tmp = d('div.article-info').find('dl')
            name = pq(tmp[1]).find('dt')
            for i in range(1, len(name)+1):
                if re.search(r'Issue Date', pq(name[i-1]).text()):
                    break
            tmp1 = pq(tmp[1]).find('dd')
            issueDate[int(issue)-1] = pq(tmp1[i-1]).text()
            print issueDate[int(issue)-1]

        fh = open(fname, 'w')
        fh_zh = open(fname_zh, 'w')
        dict_head = {'name': subtitle,
                    'vol': vol,
                    'issue': issue,
                    'date': issueDate[int(issue)-1],
                    'code': TransCode[trans][0],
                    'society_url': TransCode[trans][1]}
        fp = open('../template/head.html', 'r')
        fp_zh = open('../template/head_zh.html', 'r')
        # read template file: head.html from temp folder
        t = Template(fp.read())
        t_zh = Template(fp_zh.read())
        fp.close()
        fp_zh.close()
        html = t.render(Context(dict_head))
        html_zh = t_zh.render(Context(dict_head))
        fh.write(html)
        fh_zh.write(html_zh)

        fh.close()
        fh_zh.close()
        fh = open(fname, 'a')
        fh_zh = open(fname_zh, 'a')

    item_dict = {'vol': vol,
                'issue': issue,
                'counter': counter[int(issue)-1],
                'authors': authors,
                'title': title,
                'journal_name': subtitle,
                'spage': spage,
                'epage': epage,
                'date': issueDate[int(issue)-1],
                'abslink': abslink,
                'pdflink': pdflink,
                'abstract': abstract}
    item_dict_zh = {'vol': vol,
                   'issue': issue,
                   'counter': counter[int(issue)-1],

```

```

        'authors': authors,
        'title': title,
        'title_zh': title_zh,
        'journal_name': pubtitle,
        'spage': spage,
        'epage': epage,
        'date': issueDate[int(issue)-1],
        'abslink': abslink,
        'pdflink': pdflink,
        'abstract': abstract_zh}

    fp = open('../template/item.html')
    fp_zh = open('../template/item_zh.html')
    t = Template(fp.read())
    t_zh = Template(fp_zh.read())
    fp.close()
    fp_zh.close()
    html = t.render(Context(item_dict))
    html_zh = t_zh.render(Context(item_dict_zh))
    fh.write(html)
    fh_zh.write(html_zh)
    fh.close()
    fh_zh.close()
else:
    title + '\t' + vol + '\t' + issue + '\n'

for issue in issueCollect:
    fname = '../temp/' + volNum + '_' + issue + '.htm'
    fh = open(fname, 'a')
    fh.write('</table>')
    fh.close()
    fname_zh = '../temp/' + volNum + '_' + issue + '_zh.htm'
    fh_zh = open(fname_zh, 'a')
    fh_zh.write('</table>')
    fh_zh.close()

```

4.5 Words and Phrase in IE Dictionary Replace Google Translation

In Python, the method `replace()` returns a copy of the string in which the occurrences of old have been replaced with new. The syntax for `replace()` method is : `str.replace(old, new)`. Related codes:

```

# -*- coding: utf-8 -*-

replacements = {'3 电平逆变器': '三级逆变器', 'AC 转换器': '交流转换器', 'AC 机': '交流电机', '交流电机驱动器': '变频器', 'AC-DC 转换': '交流直流转换', '交流-交流电力转换': '交流-交流功率变换', 'AC-DC': '交流-直流', '交流传动': '交流变频器', '加速衰老': '加

```



```
速老化', '王牌框架':'ace 框架', 'ACO':'蚁群算法', '吸音降噪':'隔音降噪', '动作识别': '行为识别', '主动钳位电路':'有源箝位正激变换器', '主动阻尼控制':'有源阻尼控制', '积极布局网络':'主动配电网', '活动网络':'有源网络', '主动网络':'有源网络', '有功功率脱钩': '有功功率解耦', '有源电力滤波器':'有源滤波器', '主动整流器':'有源整流', '活跃的三次谐波注入':'有源三次谐波注入', '演员评论家':'评估-决策方法', '执行器饱和':'执行机构饱和', '专案和传感器网络':'无线自组传感器网络', 'ad-hoc 网络':'自组网', '自适应神经元':'适应机', '自适应通量观测':'自适应观测器', '自适应输出复发性小脑模型控制':'自适应递归小脑神经网络', '添加剂制造':'增材制造', 'AIC':'现用指令计数器', 'AOI':'自动化运算符介面', 'ASIC':'特定应用集成电路', 'ASON':'自动交换光网络', '包包的话':'词袋模型', '球轴承':'滚珠轴承', '滚珠丝杠':'滚珠螺杆', '银行生态系统':'金融生态系统', '棒破损':'断条', '指路明灯':'信标', '标杆':'评量基准', '分岔':'分支', '生物地理学为基础的优化': '生物地理分布优化', 'BIST':'内建自测试', 'DC-DC 降压转换器':'直流-直流降压转换器', '直流母线电容电压平衡':'直流环节电容电压平衡', 'DC / AC 转换':'交/直流转换', '僵局':'死锁',}
```

```
with open('ISIE2014_zh.txt') as infile, open('ISIE2014_zh_changed.txt',
'w') as outfile:
    for line in infile:
        for src, target in replacements.iteritems():
            line = line.replace(src, target)
        outfile.write(line)
```

4.6 Search Key Words in Html Web Pages

Search key words for titles and abstracts in Html web pages by using regular expressions.

Related codes:

```
use CGI

for($i=$year_begin-$startyear;$i<=$year_end-$startyear;$i++)
{if($titorabs eq "titles and
abstracts"){ $htmlfile=$directory.$i.".htm";&search;}
}
sub search{
open (TEST,$htmlfile) || return;
@test=<TEST>;
close(TEST);
$L=@test;
for($v=0;$v<$L;$v++)
{
if($test[$v]=~ m/<\/td><td>/)
{
$u=$test[$v];
@z=split('<\/td><td>', $u);
```


Chapter 5

Experimental results

5.1 Search Engine Website

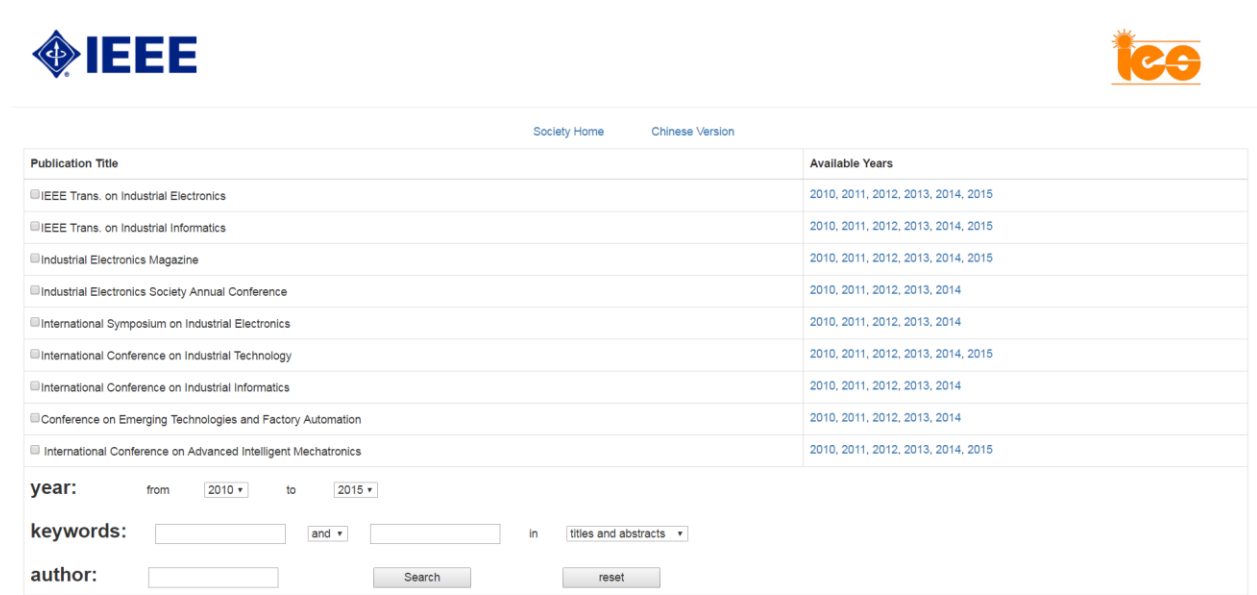


Figure.1. Search Engine English Homepage

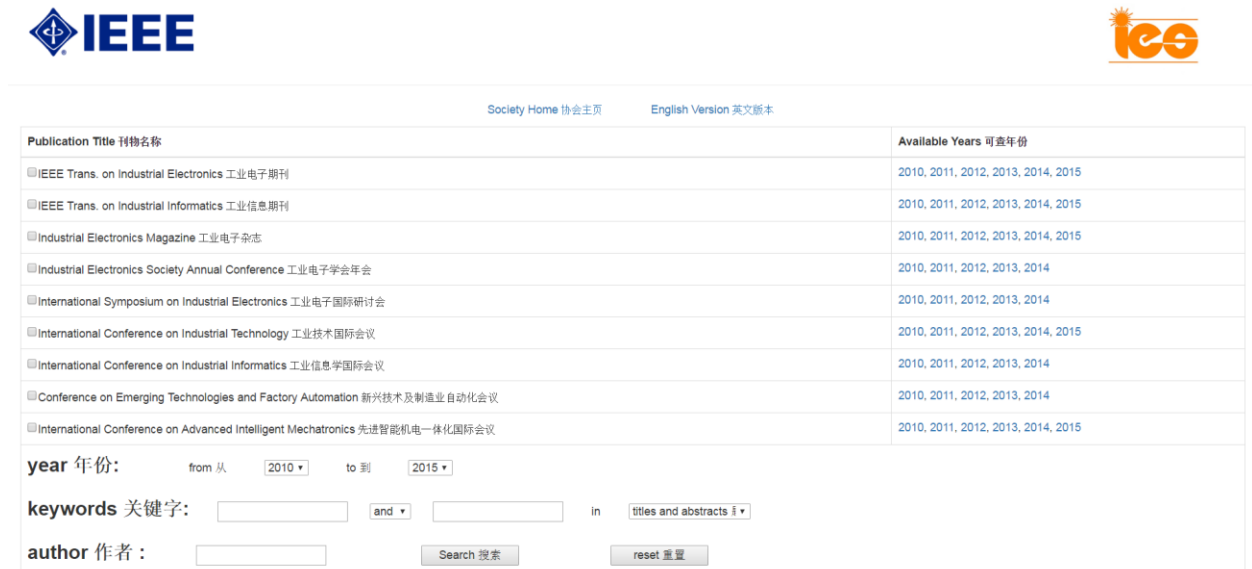
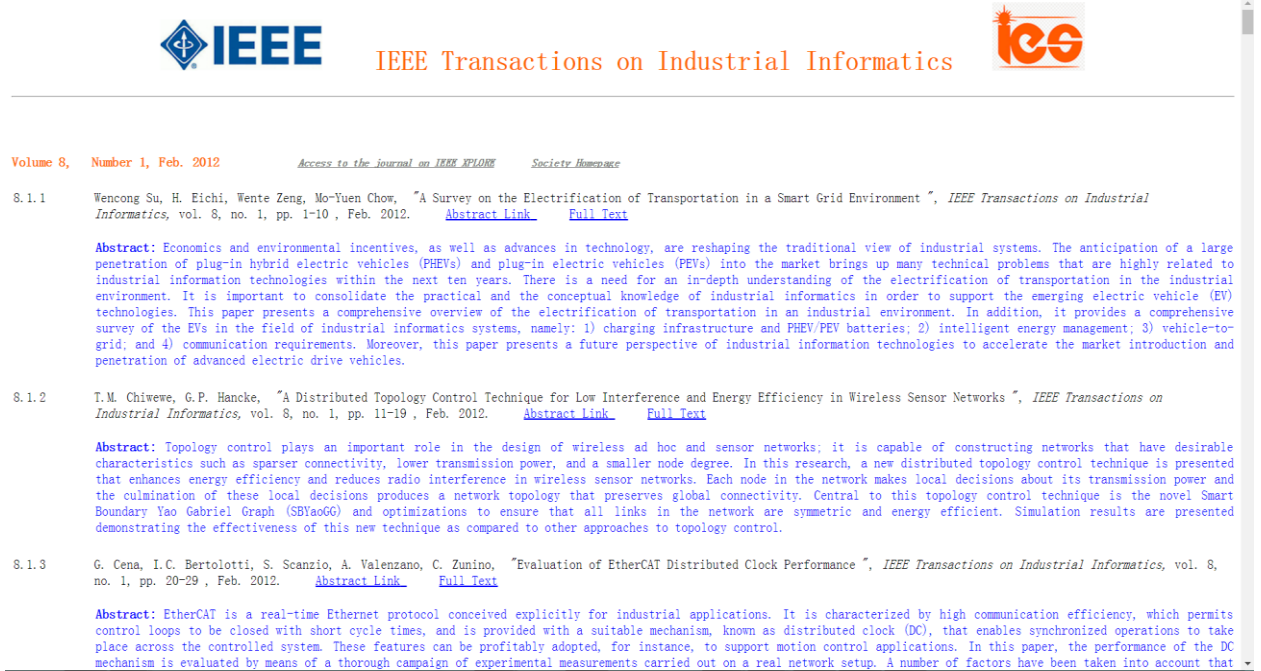


Figure.2. Search Engine Chinese Homepage

5.2 Journal titles and abstracts English to Chinese translation examples



Volume 8, Number 1, Feb. 2012

Access to the Journal on IEEE Xplore Society Homepage

8.1.1 Wencong Su, H. Eichi, Wentz Zeng, Mo-Yuen Chow, "A Survey on the Electrification of Transportation in a Smart Grid Environment", *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 1-10, Feb. 2012. [Abstract Link](#) [Full Text](#)

Abstract: Economics and environmental incentives, as well as advances in technology, are reshaping the traditional view of industrial systems. The anticipation of a large penetration of plug-in hybrid electric vehicles (PHEVs) and plug-in electric vehicles (PEVs) into the market brings up many technical problems that are highly related to industrial information technologies within the next ten years. There is a need for an in-depth understanding of the electrification of transportation in the industrial environment. It is important to consolidate the practical and the conceptual knowledge of industrial informatics in order to support the emerging electric vehicle (EV) technologies. This paper presents a comprehensive overview of the electrification of transportation in an industrial environment. In addition, it provides a comprehensive survey of the EVs in the field of industrial informatics systems, namely: 1) charging infrastructure and PHEV/PEV batteries; 2) intelligent energy management; 3) vehicle-to-grid; and 4) communication requirements. Moreover, this paper presents a future perspective of industrial information technologies to accelerate the market introduction and penetration of advanced electric drive vehicles.

8.1.2 T.M. Chiuwee, G.P. Hancke, "A Distributed Topology Control Technique for Low Interference and Energy Efficiency in Wireless Sensor Networks", *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 11-19, Feb. 2012. [Abstract Link](#) [Full Text](#)

Abstract: Topology control plays an important role in the design of wireless ad hoc and sensor networks; it is capable of constructing networks that have desirable characteristics such as sparser connectivity, lower transmission power, and a smaller node degree. In this research, a new distributed topology control technique is presented that enhances energy efficiency and reduces radio interference in wireless sensor networks. Each node in the network makes local decisions about its transmission power and the culmination of these local decisions produces a network topology that preserves global connectivity. Central to this topology control technique is the novel Smart Boundary Yao Gabriel Graph (SBYaGG) and optimizations to ensure that all links in the network are symmetric and energy efficient. Simulation results are presented demonstrating the effectiveness of this new technique as compared to other approaches to topology control.

8.1.3 G. Cena, I.C. Bertolotti, S. Scanzio, A. Valenzano, C. Zunino, "Evaluation of EtherCAT Distributed Clock Performance", *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 20-29, Feb. 2012. [Abstract Link](#) [Full Text](#)

Abstract: EtherCAT is a real-time Ethernet protocol conceived explicitly for industrial applications. It is characterized by high communication efficiency, which permits control loops to be closed with short cycle times, and is provided with a suitable mechanism, known as distributed clock (DC), that enables synchronized operations to take place across the controlled system. These features can be profitably adopted, for instance, to support motion control applications. In this paper, the performance of the DC mechanism is evaluated by means of a thorough campaign of experimental measurements carried out on a real network setup. A number of factors have been taken into account that

Figure.3. IEEE Transactions on Industrial Informatics 2012



Volume 8, Number 1, Feb. 2012

IEEE Xplore 文章入口 Society 主页

8.1.1 Wencong Su, H. Eichi, Wentz Zeng, Mo-Yuen Chow, "A Survey on the Electrification of Transportation in a Smart Grid Environment", *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 1-10, Feb. 2012. [Abstract Link](#) [Full Text](#)

题目: 在智能电网环境的调查研究交通运输的电气化 [摘要链接](#) [全文下载](#)

摘要: 经济与环境的激励机制,以及技术的进步,正在重塑工业系统的传统观点,插电式混合动力电动汽车(P混合动力汽车)和插电式电动车(电动汽车)进入市场的大普及的预期带来了那些高度相关的工业信息技术在未来十年内的许多技术问题,有必要进行一个深入的了解运输在工业环境中的电气化,它巩固实用和工业信息学,以支持新兴电动汽车(EV)技术的概念性知识是重要的,本文提出的交通电气化的工业环境的全面概述,此外,它还提供了电动汽车的全面调查工业信息学系统领域,分别是:1)充电基础设施和P混合动力汽车/PEV电池;2)智能能源管理;3)车辆到电网,和4)通信需求。此外,本文提出了工业信息技术,加快市场引进先进的电力驱动汽车和渗透的未来前景。

8.1.2 T.M. Chiuwee, G.P. Hancke, "A Distributed Topology Control Technique for Low Interference and Energy Efficiency in Wireless Sensor Networks", *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 11-19, Feb. 2012. [Abstract Link](#) [Full Text](#)

题目: 分布式拓扑控制技术低干扰和能源效率在无线传感器网络 [摘要链接](#) [全文下载](#)

摘要: 拓扑控制起着无线自组织和传感器网络的设计提供了重要的作用,它是能够构造网络,具有理想的特性,例如稀疏连通,低功率传输,一个较小的节点度,在这项研究中,一种新的分布式拓扑控制技术,提出了增强的能源效率,并减少在无线传感器网络中的无线电干扰,网络中的每个节点,使局部决策有关其传输功率以及这些局部决策的顶点产生,可以保留全球连接的网络拓扑,这方面的核心拓扑控制技术是新型的智能边界就加布显埃图(SBYaGG)和优化,以确保网络中的所有链接都是对称的,能源效率。仿真结果表明,相对于其他方法拓扑控制这种新的技术的有效性。

8.1.3 G. Cena, I.C. Bertolotti, S. Scanzio, A. Valenzano, C. Zunino, "Evaluation of EtherCAT Distributed Clock Performance", *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 20-29, Feb. 2012. [Abstract Link](#) [Full Text](#)

题目: 评估的EtherCAT分布式时钟性能 [摘要链接](#) [全文下载](#)

摘要: EtherCAT是明确地设想用于工业应用实时以太网协议。它的特点是高通效率,其允许控制周期短被关闭,并且设置有一合适的机构,称为分布式时钟(DC)时,一种能够同步操作,以发生在整个受控系统,这些功能可以获利的采用,例如,以支持运动控制应用。在本文中,DC机构的性能由上进行真正的网络设置实验测量的透明运动的装置评价,许多因素都被考虑到可能影响精度和精度,并且它们的影响进行了深入研究。

8.1.4 C. Caione, D. Brunelli, L. Benini, "Distributed Compressive Sampling for Lifetime Optimization in Dense Wireless Sensor Networks", *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 30-40, Feb. 2012. [Abstract Link](#) [Full Text](#)

题目: 分布式压缩采样在密集无线传感器网络生命周期的优化 [摘要链接](#) [全文下载](#)

摘要: 作为大型网络部署在无线传感器网络(WSN)数据采集和收集的问题变得越来越重要,增加网络规模带来显著的数据收集的挑战,所关注的采样和传输协调工作,以及网络的生命周期。为了解决这些问题,在网络没有集中协调的压缩技术正在成为重要的解决方案,以延长使用寿命。在本文中,我们考虑在一个大的无线传感器网络,基于ZigBee协议,用于监测(如建筑,工业等)的场景,我们提出了一个新的算法在网络压缩对更长的网络生命周期,我们的方法是完全分布式的:每个节点自主需要关于压缩和转发方案决策,以减少分组要发送的数量。性能进行了研究相对于利用一个现实生活中的部署收集的数据集的网络规模,该算法的增强版本也被引入到考虑到压缩花费的能量。实验结果表明,该方法有助于寻找在传输和数据压缩消耗的能量之间的最佳平衡。

8.1.5 Shengyong Chen, Z. Wang, "Acceleration Strategies in Generalized Belief Propagation", *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 41-48, Feb. 2012. [Abstract Link](#) [Full Text](#)

Figure.4. IEEE Transactions on Industrial Informatics 2012 English to Chinese Translation

5.3 Search Result Examples of Key Words

IEEE Industrial Electronics Society Publications

[Back to the search page](#) [Access to the journal on IEEE XPLORÉ](#)

Search Result: 29 papers

author:none keywords:network and electronic from 2010 to 2015 titles and abstracts

Industrial Electronics Magazine: 8 papers

5.3.6 Xinghuo Yu, C. Cecati, T. Dillon, Simó, M.G. es, "The New Frontier of Smart Grids", *IEEE Industrial Electronics Magazine*, vol. 5, no. 3, pp. 49-63, Sept. 2011. [Abstract Link](#) [Full Text](#)

Abstract:The power grid is a massive interconnected network used to deliver electricity from suppliers to consumers and has been a vital energy supply. To minimize the impact of climate change while at the same time maintaining social prosperity, smart energy must be embraced to ensure a balanced economical growth and environmental sustainability. There fore, in the last few years, the new concept of a smart grid (SG) became a critical enabler in the contemporary world and has attracted increasing attention of policy makers and engineers. This article introduces the main concepts and technological challenges of SGs and presents the authors' views on some required challenges and opportunities presented to the IEEE Industrial Electronics Society (IES) in this new and exciting frontier.

5.3.10 M.P. Kazmierkowski, "On-Chip Integration and Industrial Electronics[review of "Communication Architectures for Systems-on-Chip" (Ayala, J.L.; 2011) [Book News] ", *IEEE Industrial Electronics Magazine*, vol. 5, no. 3, pp. 85-85, Sept. 2011. [Abstract Link](#) [Full Text](#)

Abstract:This book provides a reference for the broad range of professionals, researchers, and students interested in the design of Systems-on-a-chip (SoC), in particular with a clear emphasis on the technologies and mechanisms used to perform data communication in these devices. The book covers from what could be considered traditional communication architectures (buses) and novel architectures (networks-on-chip) to recent technologies (nanoelectronics, optics, RF, and so on) and design issues (security) found in SoC communications. Both research and implementation aspects are covered. This book is organized into eight chapters that provide a comprehensive knowledge of the following covered areas: Introduction to SoC communication; SoC communication architectures; network-on-chip architectures; quality of service in network-on-chip; emerging interconnect technologies; HeTERO: hybrid topology exploration for RF based on chip networks; intra/interchip optical communications; and security issues in SoC communication.

6.4.1 X. Roboam, B. Sareni, A.D. Andrade, "More Electricity in the Air: Toward Optimized Electrical Networks Embedded in More-Electrical Aircraft", *IEEE Industrial Electronics Magazine*, vol. 6, no. 4, pp. 6-17, Dec. 2012. [Abstract Link](#) [Full Text](#)

Abstract:Along with the main trends and future challenges of electrical networks embedded in more-electrical aircraft, this article also focuses on optimization efforts in

Figure.5. English Search Results

IEEE Industrial Electronics Society Publications

[Back to the search page](#) [Access to the journal on IEEE XPLORÉ](#)

Search Result: 4235 papers

author:none keywords:电 or 工程 from 2010 to 2015 titles and abstracts

Industrial Electronics Society Annual Conference (IECON): 1749 papers

1 Nihal Kularatna, "Rechargeable batteries and battery management systems design", *IECON 2010*, pp. 1-2, 7-10 Nov. 2010. [Abstract Link](#) [Full Text](#)

题目: 可充电电池和电池管理系统的设计 [摘要链接](#) [全文下载](#)

摘要: 预计全球销售的可充电电池,是US \$ 37十亿左右,2008年,这是预计到2013年将增长对US \$ 45十亿按市场报告,对一次和二次电池美国的需求将会以每年2.5%,2012年提高到16.8十亿,而一次电池将占到5.8十亿以3%的增长率。对于较小的轻巧便携电子设备永不满足的需求急剧增加,需要的可充电化学电池的研究。除了实现对铅酸,镍镉(NiCd)电池提高性能,许多新的化学品已推出了近25年,如镍氢(NiMH),锂离子(Li-Ion)电池,锂聚合物,充电碱性电池,银锌,锌空气。本教程详细介绍电池家庭终端等特点,密封铅酸,镍镉,镍氢电池,锂LON/锂聚合物/LiFePO 4,和可充电碱性以及电池管理系统和芯片使用的现代技术,但没有具体说明的化学电池。介绍到充电终止技术和放电检测结束将与现代的电池管理IC处理器为基础的方法来提供。对电池的运行时间预测模型的讨论。从电池制造商数据表剩余额量,发展电池模型的精确预测将是本教程的一个重要子集。介绍SiBus的智能电池规格,并在IEEE 1625标准电池安全概要也将提供。高温应用及电池组设计用于极端温度范围将是另一个子主题。简单介绍在较小的电池组和监测技术用于大型电池银行预测功能将包括在内。超级电容器技术和超级电容器电池混合动力车和超级电容器的一些有创意的应用程序也将discussed。总体列报将根据适用的技术,相关的国际标准,现有的技术和工业实践技术和未来方向的必要的均衡搭配,通过研究出版物的一组选定的支持。

2 M. McQueen, "Software and human vulnerabilities", *IECON 2010*, pp. 1-85, 7-10 Nov. 2010. [Abstract Link](#) [Full Text](#)

题目: 软件和人的脆弱性 [摘要链接](#) [全文下载](#)

摘要: 幻灯片的收藏涵盖了以下主题:电网系统,变电站的数据,软件漏洞,人类的脆弱性,风险分析,控制系统,和软件的安全性。

3 B. Fahimi, S. Pekarek, "Design and control of electric machines utilizing a field reconstruction method", *IECON 2010*, pp. 1-2, 7-10 Nov. 2010. [Abstract Link](#) [Full Text](#)

题目: 设计并利用现场重建方法电机控制 [摘要链接](#) [全文下载](#)

摘要: 电-磁-机械能量转换过程的准确评估,通常需要一个多物理有限元(FEA)分析,因为有限元分析是一个耗时的过程的数值。集总参数模型,简化的等效磁路中,并分离场分析通常用于设计和电动机驱动的驱动器控制。这个计算负担是在发电机风力能量收获及电动机用于电动和插电式混合动力电动汽车,其中显著字段统计在设计过程中经常使用的推进设计特别明显。场重建方法(FRM)是在其中的一组字段分析小用于建立基础函数在机器的磁通密度的技术。一旦基函数是建立,机器的性能是根据任意的速度和励磁预测。它已经显示了FRM可以数量级的两个或三个数量,同时保持精度,通常完成二维有限元分析相同的水平减少计算时间。由于其高计算效率,FRM创造了健康监测,容错操作,电机的多物理设计的新机遇。在过去的6年中,该技术已成功应用于永磁,感应和开关磁阻电机驱动器。在本教程中,主持人将解释永磁同步电机(PMSM),感应电机的背景下,FRM的基本面,和开关磁阻电机。作为演示的一部分的冲击和必要的修改纳入磁饱和的将被讨论。了FRM在计算系统引起的振动和消除噪声和振动的永磁同步电机,每对感应电动机驱动变频操作的最大扭矩,和消除应用在双馈感应发电机present- - 版本说明该技术的潜力为优化设计和电机的运行。此外,突出于磁通量估计和多相永磁电机驱动的容错操作最近正在进行的的研究将与观众共享。最后,在下一代微控制器和集成电力电子变换器的FRM的潜在影响进行讨论。教程将总结了一系列的实验结果证实了FRM供将来可变速电机驱动应用的实用性。

Figure.6. Chinese Search Results

Chapter 6

Conclusion and Future Work

After using IE dictionary to improve Google translation, some translation are more accuracy due to replacement of incorrectly translated technical terms [18]. For example:

Title: Observer-Based State-Space Current Control for a Three-Phase Grid-Connected Converter Equipped With an LCL Filter
[Abstract Link](#) [Full Text](#)

Kukkola, J.; Hinkkanen, M., "Observer-Based State-Space Current Control for a Three-Phase Grid-Connected Converter Equipped With an LCL Filter", IEEE Trans. on Industry Applications, 2014

Abstract: This paper presents a state-space current control method for **active damping** of the resonance frequency of the LCL filter and setting the dominant dynamics of the converter current through direct pole placement. A state observer is used, where upon additional sensors are not needed in comparison with the conventional L filter design. The relationship between the system delay and instability caused by the resonance phenomenon is considered. Nyquist diagrams are used to examine the parameter sensitivity of the proposed method. The method is validated with simulations and experiments.

Figure.7. 2014 Industry Application Abstract

标题: 观察基于状态空间电流控制的三相电网连接器，配备的LCL滤波器
[Abstract Link](#) [Full Text](#)

Kukkola, J.; Hinkkanen, M., "观察基于状态空间电流控制的三相电网连接器，配备的LCL滤波器", IEEE Trans. on Industry Applications, 2014

摘要: 本文提出了LCL滤波器的共振频率的**主动阻尼**，并通过直接极点设定转换器电流的主要动力状态空间电流控制方法。状态观测的情况下，其中在附加的传感器不需要与常规的L个滤波器设计的比较。系统延迟和不稳定性所造成的共振现象之间的关系被认为是。奈奎斯特图是用来检查该方法的参数敏感性。该方法的验证，仿真和实验。

Figure.8. Google Translation 1

标题: 观察基于状态空间电流媒体控制的三个相电力网络连接器, 配备的LCL滤波器 [Abstract Link](#) [Full Text](#)

Kukkola, J.; Hinkkanen, M., "观察基于状态空间电流媒体控制的三个相电力网络连接器, 配备的LCL滤波器", IEEE Trans. on Industry Applications, 2014

摘要: 本文构成了LCL滤波器的共振频率的有源阻尼, 并通过直接极点设定转换器电流媒体的主要推进状态空间电流媒体控制方法。Luenberger观察者的情况下, 它的中在附加的无传感器不需要与常规的L个滤波器设计的比较。系统延迟和不稳定性所造成的共振现象之间的关系被认为是。奈奎斯特图是用来检查该方法的参数敏感性。该方法的验证, 模块拟和实验。

Figure.9. IE Dictionary Translation 1

In this example, Google Translation made a mistake which translated the key word ‘active damping’ incorrectly into ‘主动阻尼(dynamic damping)’. IE dictionary replaces ‘主动阻尼(dynamic damping)’ with correct translation of ‘有源阻尼(active damping)’.

Another example:

Title: Speed Sensorless and Sensor-Fault Tolerant Optimal PI Regulator for Networked DC Motor System With Unknown Time-Delay and Packet Dropout [Abstract Link](#) [Full Text](#)

Ahmadi, A.-A.; Salmasi, F.R.; Noori-Manzar, M.; Najafabadi, T.A., "Speed Sensorless and Sensor-Fault Tolerant Optimal PI Regulator for Networked DC Motor System With Unknown Time-Delay and Packet Dropout", IEEE Trans. on Industrial Electronics, 2014

Abstract: Sensorless and sensor-fault-resilient control of a networked dc motor system (NDCMS) with an optimal integral-square-error proportional-integral (PI) controller is considered, while network-induced delays and packet dropouts are taken into account. A sliding-mode observer is developed to estimate rotor speed and unknown load torque for the networked system. Then, a PI controller is designed such that the overall NDCMS with complete or partial sensor failure is stabilized and a linear quadratic cost function is sufficiently minimized. Optimal controller parameters are determined by solving bilinear matrix inequalities. The numerical and experimental tests are performed to evaluate the feasibility and applicability of the networked sensorless or sensor-fault-tolerant controller. The results show good performance in both estimation and control objectives.

Figure.10. 2014 Industrial Electronics Abstract

标题: 速度传感器和传感器容错最优PI调节器联网直流电动机系统的未知**延时**和丢包 [Abstract Link](#) [Full Text](#)

Ahmadi, A.-A.; Salmasi, F.R.; Noori-Manzar, M.; Najafabadi, T.A., "速度传感器和传感器容错最优PI调节器联网直流电动机系统的未知**延时**和丢包", IEEE Trans. on Industrial Electronics, 2014

摘要: 传感器和联网直流电动机系统 (NDCMS) 配有一个最佳的积分平方误差的比例积分 (PI) 控制器的传感器故障复原控制被认为是, 当网络诱导延迟和分组丢失都考虑在内。滑动模式观测器开发的估计转子转速和未知的负载转矩的网络系统。然后, PI控制器被设计为使得与全部或部分的传感器故障的整体NDCMS稳定和线性二次代价函数被充分地最小化。优化控制器参数, 通过求解双线性矩阵不等式确定。数值和实验测试被执行以评估网络传感器或传感器的容错控制器的可行性和适用性。结果表明, 在这两个估计和控制目标良好的性能。

Figure.11. Google Translation 2

标题: 速度无传感器和无传感器容错最优PI调节器网络直流媒体电汽车器系统的未知**时滞**和包丢失 [Abstract Link](#) [Full Text](#)

Ahmadi, A.-A.; Salmasi, F.R.; Noori-Manzar, M.; Najafabadi, T.A., "速度无传感器和无传感器容错最优PI调节器网络直流媒体电汽车器系统的未知**时滞**和包丢失", IEEE Trans. on Industrial Electronics, 2014

摘要: 无传感器和网络直流媒体电汽车器系统 (N直流媒体MS) 配有一个最优的集成区平方误差的比例集成区 (PI) 控制器的无传感器的错康多层次的处理路复用控制被认为是, 当网络诱导延迟和分区组丢失都考虑在内。滑动模式观测器开发的估计转子转速和未知的负载转矩的网络系统。然后, PI控制器被设计为使得与全部或部分区的无传感器的错的整体N直流媒体MS稳定和线性微系统公司光次代价函数被充分区地最小化。最优控制器参数, 通过求解方案双线性矩阵不等式确定。数值和实验测试被执行以评价网络无传感器或无传感器的容错控制器的可行性和适用性。结果表明, 在这两个个评价和控制目标好的性能。

Figure.12. IE Dictionary Translation 2

In this example, Google translated ‘time-delay’ into “延时 (delay)” which is not technical. IE dictionary modified the result into “时滞 (time-delay)” and made this translation more accuracy.

From the results, it demonstrates that the main problem that leads to bad translation results in technical field has corrected well. The Industrial Electronics dictionary has a positive effect on improving the professional terminologies from English to Chinese. The web pages and search engine can achieve the aim to show the journals and conference documents and search the information such as key words, years and author in data pages. Therefore, the project was completed successfully. With the improvement of translation, Chinese engineers can get better understanding when they read industrial electronic papers and this approach can be also used in other technical areas.

Besides the improvement, one shortcoming of this method is unselective substitution by IE dictionary. IE dictionary automatically replaces all discernible incorrect keywords of translation. Some words which contain the same word will be incorrectly substituted. In the following work, the researchers will develop the method of substitution and revise IE dictionary.

Reference

- [1] Slocum, Jonathan. "A survey of machine translation: its history, current status, and future prospects." *Computational linguistics* 11.1 (1985): 1-17.
- [2] Hutchins, John. "From first conception to first demonstration: the nascent years of machine translation, 1947–1954. a chronology." *Machine Translation* 12.3 (1997): 195-252.
- [3] Hutchins, John. "Machine translation: History and general principles." *The encyclopedia of languages and linguistics* 5 (1994): 2322-2332.
- [4] Hutchins, John. "ALPAC: the (in) famous report." *Readings in machine translation* 14 (2003): 131-135.
- [5] Chan, Sin-wai. *Routledge encyclopedia of translation technology*. Routledge, 2014.
- [6] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu "BLEU: a Method for Automatic Evaluation of Machine Translation" *Proceedings of the 40th annual meeting of the association for computational linguistics(ACL)*, Philadelphia, July 2002.B.
- [7] Wu, Yonghui, et al. "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation." *arXiv preprint arXiv:1609.08144* (2016).
- [8] Friedl, Jeffrey EF. *Mastering regular expressions*. " O'Reilly Media, Inc.", 2006. [9] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, 1 edition. Reading, Mass: Addison-Wesley Professional, 1989.
- [9] Musciano, Chuck, and Bill Kennedy. *HTML, the definitive Guide*. O'Reilly & Associates, 1996.
- [10] Srinivasan, Sriram. *Advanced perl programming*. " O'Reilly Media, Inc.", 1997.
- [11] Kuhlman, Dave. *A Python Book: Beginning Python, Advanced Python, and Python Exercises*. Dave Kuhlman, 2009.
- [12] M.F.Sanner "Python: a programming language for software integration and development" *The scripps research institute*, 1999.J.
- [13] Zahariev I. *C++ vs. Python vs. Perl vs. PHP performance benchmark*[J]. 2011
- [14] Tang B. *Orthogonal Array-Based Latin Hypercubes*[J]. *Journal of the American Statistical Association*, 1993, 88(424):1392-1397.
- [15] Junhui Chen, Fengyu Yang, Long Yang. "Automatic equalization for digital communication," *Bell Syst. Tech. J.*, vol. 44, no. 4, pp. 547–588, Apr. 1965.
- [16] Schmalstieg D, Fuhrmann A, Hesina G. "Bridging multiple user interface dimensions with augmented reality" *Augmented Reality, 2000.(ISAR 2000).*, IEEE and ACM International Symposium on. IEEE, 2000: 20-29.

- [17] G. W. Juetten and L. E. Zeffanella. "Radio noise currents in short sections on bundle conductors (Presented Conference Paper style)," presented at the IEEE Summer power Meeting, Dallas, TX, June 22–27, 1990, Paper 90 SM 690-0 PWRS.
- [18] Zhang, Xiangyu, et al. "An improved English to Chinese translation of technical text." Intelligent Engineering Systems (INES), 2015 IEEE 19th International Conference on. IEEE, 2015.

Appendix 1: Python Codes -- Download and Translate Journals Document and Create HTML Web Pages

```
#!/usr/bin/env python
#-*-coding:utf-8-*-
# __time__ = 9/14/15
import urllib2
import os.path as op
from pyquery import PyQuery as pq
from django.template import Context, Template
from HTMLParser import HTMLParser # unescape
import sys
import goslate
import os
import re #regular expression
import time
import random
from ftplib import FTP
from datetime import datetime

stdout = sys.stdout
print sys.getdefaultencoding() #ascii

try:
    import xml.etree.cElementTree as ET
except ImportError:
    import xml.etree.ElementTree as ET
# pn, society url, early access vol number, startyear
TransCode = {'IEM': [4154573, 'http://ieeexplore.ieee.org/', 0, 2007],
             'TIE': [41, 'http://ieeexplore.ieee.org/', 62, 1954],
             'TII': [9424, 'http://ieeexplore.ieee.org/', 11, 2005],
             ...}
Step: Translate abstract from EN into ZH
'''
def ReOrder(author):
    Alist = []
    for a in author.split(';'):
        name = a.split('.')
        if len(name)>1:
            Alist.append(name[1].strip() + ' ' + name[0].strip())
        else:
            Alist.append(a)
    return ' '.join(Alist)
# download data, create webpages and upload them
def download(trans, year, upload):
    # clearTemp()
    gs = goslate.Goslate(service_urls=['http://translate.Google.de']); #for large-scale translation
    url = 'http://ieeexplore.ieee.org/gateway/ipsSearch.jsp?pn=' + str(TransCode[trans][0]) + '&py=' + str(year) + '&hc=5000' #hc shows paper amount
    xmldata = urllib2.urlopen(url).read()
    root = ET.fromstring(xmldata)
    number = root.find('totalfound').text
    print 'total papers found: ' + str(number)
    issueDate = ['']*12
    counter = [0]*12
    issueCollect = []
    volNum = 0

    Sorted_doc_ori = root.findall('document')
    Sorted_doc = sorted(Sorted_doc_ori, key=lambda d: int(d.find('spage').text) if d.find('spage').text.startswith('C')==False else sys.maxint) #sys.maxint
    '''#just for debug
    dd=len(Sorted_doc)-1
    print dd
    for i in range(dd):
        print Sorted_doc_ori[i].find('spage').text
    for i in range(dd):
        print Sorted_doc[i].find('spage').text
    '''
# parse the document, extract required data
for document in Sorted_doc:
    #root.findall('document'):
    title = ""
    authors = ""
    subtitle = ""
    vol = ""
    issue = ""
    spage = ""
    epage = ""
    abstract = ""
    abslink = ""
    pdflink = ""
    abstract_zh=""
```

```

if(document.find('title') is not None):
    tmp = document.find('title').text
    if(tmp):
        title = tmp
        time.sleep(3)
        title_zh = gs.translate(title, 'zh')

if(document.find('authors') is not None):
    tmp = document.find('authors').text
    if(tmp):
        authors = ReOrder(tmp)

if(document.find('pubtitle') is not None):
    tmp = document.find('pubtitle').text
    if(tmp):
        pubtitle = tmp
        #change position between ','
        if pubtitle.find(',') :
            temp_f=pubtitle[0:pubtitle.find(',')]
            temp_b=pubtitle[pubtitle.find(',')+1:len(pubtitle)]
            pubtitle=temp_b+ ',' +temp_f

if(document.find('volume') is not None):
    tmp = document.find('volume').text
    if(tmp):
        vol = tmp

if(document.find('issue') is not None):
    tmp = document.find('issue').text
    if(tmp):
        issue = tmp

if(document.find('spage') is not None):
    tmp = document.find('spage').text
    if(tmp):
        spage = tmp

if(document.find('epage') is not None):
    tmp = document.find('epage').text
    if(tmp):
        epage = tmp

if(document.find('abstract') is not None):
    tmp = document.find('abstract').text
    if(tmp):
        abstract = tmp
        abstract=HTMLParser().unescape(abstract) # transfer into normal characters
        #chinese version
        abstract_re = abstract.replace("&#x201C;", "")
        abstract_re = abstract_re.replace("&#x201D;", "")
        abstract_re = abstract_re.replace("&#x201D;", "")
        abstract_re = abstract_re.replace("# ", "No.")
        abstract_re = abstract_re.replace(" & ", " and ")

        time.sleep(3)
        abstract_zh = gs.translate(abstract_re, 'zh')
        #w.write(authors + ',' + '' + title + '' + '' + pubtitle+ ' ' + py +',pp.' + spage + '-' + epage + '\n' + '\n' + ' ' +abstract + '\n'+'\n')

if(document.find('mdurl') is not None):
    tmp = document.find('mdurl').text
    if(tmp):
        abslink = tmp

if(document.find('pdf') is not None):
    tmp = document.find('pdf').text
    if(tmp):
        pdflink = tmp
#transfer ASCII into UTF-8
reload(sys)
sys.setdefaultencoding('utf-8')
# make valid (not empty) web pages to html and save to temp folder
if(vol != "" and spage.startswith('C')==False and vol != 'PP' and issue != "" and abstract != "" and abstract_zh!=" and authors!="): #trans has
papers
    fname = './temp/' + vol + '_' + issue + '.htm'
    fname_zh = './temp/' + vol + '_' + issue + '_zh.htm'
    counter[int(issue)-1] += 1
    volNum = vol
    print 'Volume #' + vol
    # make html files according to issue number of papers

```

```

print issue
print issueCollect
if issue not in issueCollect:
    issueCollect.append(issue)
if op.isfile(fname):
    fh = open(fname, 'a') # 'a' means appending at the last line
    fh_zh = open(fname_zh, 'a')
# if issue is not in the issueCollect then create new files as following format
else:
    if(abslink):
        d = pq(abslink)
        tmp = d('div.article-info').find('dl')
        name = pq(tmp[1]).find('dt')
        for i in range(1, len(name)+1):
            if re.search(r'Issue Date', pq(name[i-1]).text()):
                break
        tmp1 = pq(tmp[1]).find('dd')
        issueDate[int(issue)-1] = pq(tmp1[i-1]).text()
        print issueDate[int(issue)-1]

    fh = open(fname, 'w')
    fh_zh = open(fname_zh, 'w')
    dict_head = {'name': pubtitle,
                'vol': vol,
                'issue': issue,
                'date': issueDate[int(issue)-1],
                'code': TransCode[trans][0],
                'society_url': TransCode[trans][1]}
    fp = open('./template/head.html', 'r')
    fp_zh = open('./template/head_zh.html', 'r')
    t = Template(fp.read())
    t_zh = Template(fp_zh.read())
    fp.close()
    fp_zh.close()
    html = t.render(Context(dict_head))
    html_zh = t_zh.render(Context(dict_head))
    fh.write(html)
    fh_zh.write(html_zh)

    fh.close()
    fh_zh.close()
    fh = open(fname, 'a')
    fh_zh = open(fname_zh, 'a')

item_dict = {'vol': vol,
            'issue': issue,
            'counter': counter[int(issue)-1],
            'authors': authors,
            'title': title,
            'journal_name': pubtitle,
            'spage': spage,
            'epage': epage,
            'date': issueDate[int(issue)-1],
            'abslink': abslink,
            'pdflink': pdflink,
            'abstract': abstract}
item_dict_zh = {'vol': vol,
               'issue': issue,
               'counter': counter[int(issue)-1],
               'authors': authors,
               'title': title,
               'title_zh': title_zh,
               'journal_name': pubtitle,
               'spage': spage,
               'epage': epage,
               'date': issueDate[int(issue)-1],
               'abslink': abslink,
               'pdflink': pdflink,
               'abstract': abstract_zh}

fp = open('./template/item.html')
fp_zh = open('./template/item_zh.html')
t = Template(fp.read())
t_zh = Template(fp_zh.read())
fp.close()
fp_zh.close()
html = t.render(Context(item_dict))
html_zh = t_zh.render(Context(item_dict_zh))
fh.write(html)
fh_zh.write(html_zh)

```



```

pubtitle = ""
abstract = ""
abslink = ""
pdflink = ""
year = ""

if(document.find('title') is not None):
    tmp = document.find('title').text
    if(tmp):
        title = tmp

if(document.find('authors') is not None):
    tmp = document.find('authors').text
    if(tmp):
        authors = ReOrder(tmp)

if(document.find('pubtitle') is not None):
    tmp = document.find('pubtitle').text
    if(tmp):
        pubtitle = tmp

if(document.find('abstract') is not None):
    tmp = document.find('abstract').text
    if(tmp):
        abstract = tmp
#         reload(sys)
#         abstract = translate(abstract, to_langage='zh')

if(document.find('mdurl') is not None):
    tmp = document.find('mdurl').text
    if(tmp):
        abslink = tmp

if(document.find('pdf') is not None):
    tmp = document.find('pdf').text
    if(tmp):
        pdflink = tmp

if(document.find('py') is not None):
    tmp = document.find('py').text
    if(tmp):
        year = tmp

if counter==1:
    AbsH = open(AbsDir, 'w')
    NoAbsH = open(NoAbsDir, 'w')
    dict_head = {'pubtitle': pubtitle,
                'number': number,
                'time': datetime.now().strftime('%b %d, %Y'),
                'code': TransCode[trans][0],
                'society_url': TransCode[trans][1]}
    fp = open('../template/earlyHead.html', 'r')
    t = Template(fp.read())
    fp.close()
    html = t.render(Context(dict_head))
    AbsH.write(html)
    NoAbsH.write(html)
    AbsH.close()
    NoAbsH.close()

AbsH = open(AbsDir, 'a')
NoAbsH = open(NoAbsDir, 'a')

dict_item = {'counter': counter,
            'authors': authors,
            'title': title,
            'pubtitle': pubtitle,
            'year': year,
            'abslink': abslink,
            'pdflink': pdflink,
            'abstract': abstract,
            'abs': 1}
fp = open('../template/earlyItem.html', 'r')
t = Template(fp.read())
fp.close()
html = t.render(Context(dict_item))
AbsH.write(html)
AbsH.close()

dict_item['abs'] = 0

```

```

    html = t.render(Context(dict_item))
    NoAbsH.write(html)
    NoAbsH.close()

AbsH = open(AbsDir, 'a')
AbsH.write("</table>")
AbsH.close()

NoAbsH = open(NoAbsDir, 'a')
NoAbsH.write("</table>")
NoAbsH.close()

# ftp
if upload:
    ftp = FTP('tii.ieee-ies.org')
    ftp.login(user='jiaoy',passwd='_xJ!Ag')
    ftp.cwd(trans+pub)

    AbsH = open(AbsDir, 'rb')
    ftp.storbinary('STOR %s' % AbsFile, AbsH)
    AbsH.close()
    os.remove(AbsDir)

    NoAbsH = open(NoAbsDir, 'rb')
    ftp.storbinary('STOR %s' % NoAbsFile, NoAbsH)
    NoAbsH.close()
    os.remove(NoAbsDir)

ftp.quit()

def merge(trans, Syear, Eyear, upload):
    Svol = Syear - TransCode[trans][3] + 1
    Evol = Eyear - TransCode[trans][3] + 1
    print Svol, Evol

    AbsFile = str(Syear) + '_' + str(Eyear) + '.htm'
    AbsDir = './temp/' + AbsFile
    AbsH = open(AbsDir, 'w')
    AbsH.close()
    AbsH = open(AbsDir, 'a')
    for vol in range(Svol, Evol+1):
        url = 'http://tii.ieee-ies.org/' + trans + 'pub/' + str(vol) + '.htm'
        html = urllib2.urlopen(url).read()
        AbsH.write(html)
    AbsH.close()

    NoAbsFile = str(Syear) + '_' + str(Eyear) + 's.htm'
    NoAbsDir = './temp/' + NoAbsFile
    NoAbsH = open(NoAbsDir, 'w')
    NoAbsH.close()
    NoAbsH = open(NoAbsDir, 'a')
    for vol in range(Svol, Evol+1):
        url = 'http://tii.ieee-ies.org/' + trans + 'pub/' + str(vol) + 's.htm'
        html = urllib2.urlopen(url).read()
        NoAbsH.write(html)
    NoAbsH.close()

    if upload:
        ftp = FTP('tii.ieee-ies.org')
        ftp.login(user='jiaoy',passwd='_xJ!Ag')
        ftp.cwd(trans+pub)

        AbsH = open(AbsDir, 'rb')
        ftp.storbinary('STOR %s' % AbsFile, AbsH)
        AbsH.close()
        os.remove(AbsDir)

        NoAbsH = open(NoAbsDir, 'rb')
        ftp.storbinary('STOR %s' % NoAbsFile, NoAbsH)
        NoAbsH.close()
        os.remove(NoAbsDir)

    ftp.quit()

if __name__ == '__main__':
    os.environ.setdefault("DJANGO_SETTINGS_MODULE", "IEEE3.settings")
    # after runing, comment it as record
    download('IEM', 2013, False)
    # download('TIE', 2013, False)
    # download('TII', 2013, False)

```

Appendix 2: Html Codes – Home Pages of Selected IES Document

```
<!DOCTYPE html>
<!-- saved from url=(0048)http://vps.ieee-ies.org/zh/tr/static/en/TIA.html -->
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<meta charset="utf-8">
<link rel="stylesheet" type="text/css" href="bootstrap.min.css">
<meta http-equiv="Content-Language" content="en-us">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<meta http-equiv="Cache-Control" CONTENT="private,must-revalidate">
<title>Recent Issues</title>

<script type="text/javascript">
<!--
function Warn() {
    var pattern=/^\w$/g;
    if(document.search.TIE.checked==false && document.search.TII.checked==false && document.search.IEM.checked==false &&
document.search.ALL_journal.checked==false && document.search.IECON.checked==false && document.search.ISIE.checked==false &&
document.search.ICIT.checked==false && document.search.INDIN.checked==false && document.search.ETFA.checked==false &&
document.search.AIM.checked==false && document.search.ALL_conference.checked==false)
        {alert("Please select publication!");
        return false;}
    if(pattern.test(document.search.author.value))
        {alert("Please enter a name, not a letter!");
        return false;}
    if(pattern.test(document.search.keywords.value))
        {alert("Please enter keywords,not a letter!");
        return false;}
    if(pattern.test(document.search.keywords0.value))
        {alert("Please enter keywords,not a letter!");
        return false;}
}
var start=new Array(3000,3000,3000,3000,3000,3000,3000,3000,3000,3000);
var end=new Array(0,0,0,0,0,0,0,0,0);
var TIE=new Array(2010,2015);
var TII=new Array(2010,2015);
var IEM=new Array(2010,2015);
var IECON=new Array(2010,2014);
var ISIE=new Array(2010,2014);
var ICIT=new Array(2010,2015);
var INDIN=new Array(2010,2014);
var ETFA=new Array(2010,2014);
var AIM=new Array(2010,2015);
var startyear,endyear;
function changeyear(first,second){
var myEle;
var x;
switch(first.value)
{
case "TIE":if(first.checked){start[0]=TIE[0];end[0]=TIE[1];}
else{start[0]=3000;end[0]=0;} break;
case "TII":if(first.checked){start[1]=TII[0];end[1]=TII[1];}
else{start[1]=3000;end[1]=0;} break;
case "IEM":if(first.checked){start[2]=IEM[0];end[2]=IEM[1];}
else{start[2]=3000;end[2]=0;} break;
case "IECON":if(first.checked){start[3]=IECON[0];end[3]=IECON[1];}
else{start[3]=3000;end[3]=0;} break;
case "ISIE":if(first.checked){start[4]=ISIE[0];end[4]=ISIE[1];}
else{start[4]=3000;end[4]=0;} break;
case "ICIT":if(first.checked){start[5]=ICIT[0];end[5]=ICIT[1];}
else{start[5]=3000;end[5]=0;} break;
case "INDIN":if(first.checked){start[6]=INDIN[0];end[6]=INDIN[1];}
else{start[6]=3000;end[6]=0;} break;
case "ETFA":if(first.checked){start[7]=ETFA[0];end[7]=ETFA[1];}
else{start[7]=3000;end[7]=0;} break;
case "AIM":if(first.checked){start[8]=AIM[0];end[8]=AIM[1];}
else{start[8]=3000;end[8]=0;} break;
}
startyear=Math.min(start[0],start[1],start[2],start[3],start[4],start[5],start[6],start[7],start[8]);
endyear=Math.max(end[0],end[1],end[2],end[3],end[4],end[5],end[6],end[7],end[8]);
for(var p=second.options.length;p>0;p--)second.options[p]=null;
for(x=startyear;x<=endyear;x++)
{
myEle = document.createElement("option");
myEle.value = x;
myEle.text = x;
second.add(myEle);
}
}
}

```


Appendix 3: Perl Codes – Search Engine for Journals and Conference Documents

```
#!/C:\xampp\perl\bin\perl.exe"

use CGI;

$q=CGI->new;
$author=$q->param('author');
$keywords=$q->param('keywords');
$keywords0=$q->param('keywords0');
$year_begin=$q->param('year_begin');
$year_end=$q->param('year_end');
$andor=$q->param('andor');
$titorabs=$q->param('titorabs');
#journals
$TIE=$q->param('TIE');
$TII=$q->param('TII');
$IEM=$q->param('IEM');
#major conference
$IECON=$q->param('IECON');
$ISIE=$q->param('ISIE');
$ICIT=$q->param('ICIT');
$INDIN=$q->param('INDIN');
$ETFA=$q->param('ETFA');
$AIM=$q->param('AIM');
#####
$number=0;
$allnumber=0;
$htmlfile="";
$headdir="";
@dir=();
@publication_code=();
$publication_number=0;
if($TIE eq "TIE"){push(@dir,$headdir."TIEpubzh/");}
if($IEM eq "IEM"){push(@dir,$headdir."IEMpubzh/");}
if($TII eq "TII"){push(@dir,$headdir."TIIpubzh/");}
if($IECON eq "IECON"){push(@dir,$headdir."confzh/IECON/");}
if($ISIE eq "ISIE"){push(@dir,$headdir."confzh/ISIE/");}
if($ICIT eq "ICIT"){push(@dir,$headdir."confzh/ICIT/");}
if($INDIN eq "INDIN"){push(@dir,$headdir."confzh/INDIN/");}
if($ETFA eq "ETFA"){push(@dir,$headdir."confzh/ETFA/");}
if($AIM eq "AIM"){push(@dir,$headdir."confzh/AIM/");}
foreach $directory (@dir)
{
    $name="";
    $startyear=0;
    @code=();
    SWITCH:{
        $directory eq $headdir."TIEpubzh/" && do{$startyear=1953;$name="Transactions of Industrial Electronics";}
        $directory eq $headdir."IEMpubzh/" && do{$startyear=2006;$name="Industrial Electronics Magazine";}
        $directory eq $headdir."TIIpubzh/" && do{$startyear=2004;$name="Transactions of Industrial Informatics";}
        $directory eq $headdir."confzh/IECON/" && do{$startyear=1974;$name="Industrial Electronics Society Annual Conference (IECON)";}
        $directory eq $headdir."confzh/ISIE/" && do{$startyear=1991;$name="International Symposium on Industrial Electronics (ISIE)";}
        $directory eq $headdir."confzh/ICIT/" && do{$startyear=1993;$name="International Conference on Industrial Technology (ICIT)";}
        $directory eq $headdir."confzh/INDIN/" && do{$startyear=2002;$name="International Conference on Industrial Informatics (INDIN)";}
        $directory eq $headdir."confzh/ETFA/" && do{$startyear=1995;$name="Conference on Emerging Technologies and Factory Automation (ETFA)";}
        $directory eq $headdir."confzh/AIM/" && do{$startyear=1996;$name="International Conference on Advanced Intelligent Mechatronics (AIM)";}
    }
    for($i=$year_begin-$startyear;$i<=$year_end-$startyear;$i++)
    {
        if($titorabs eq "titles and abstracts"){$htmlfile=$directory.$i.".htm";&search;}
        if($titorabs eq "titles only"){$htmlfile=$directory.$i.".s.htm";&search;}
    }
    $publication_code[$publication_number]="<table><html><tr color='red'><td align='center'><strong>$name &nbsp;&nbsp;$number&nbsp;&nbsp;papers</strong></td></tr></table>";
    $allnumber=$allnumber+$number;
    $number=0;
    if(@code){
        $temp=join("\n<td></td>\n</html> <html><tr>\n",@code);
        $publication_code[$publication_number]=$publication_code[$publication_number]."<html><tr>\n".$temp."</td></tr>\n</html></table>";
    }
    $publication_number++;
}
#Doctype and meta charset did changes below
$head1="
<!DOCTYPE html>
<html>
<head>
```