EUROSTAT GRANT
Preparation on the implementation of ISCO-08

Development of a tool for the automatic coding of the variable ‘occupation’ implementing the new classification ISCO-08

FINAL REPORT
INDEX

1. Introduction 5

2. Automatic coding algorithm 7

2.1. Dictionary 7

2.1.1. Description and construction 7

2.1.2. Weights given to the main words in the dictionary 8

• Weight of the word associated with the dictionary
• Weight of the word associated with the category

2.1.3. Dictionary tables: Contents and structures 8

• Table of texts
• Table of words
• Category/word table
• Table of synonyms
• Table of fictitious codes

2.2. Steps in the automatic coding process 13

Step 1.- Entering the text to be coded
Step 2.- Cleaning of the text to be coded
Step 3.- Matching the cleaned text to be coded with cleaned texts in the table of texts
Step 4.- Matching the cleaned text to be coded with cleaned texts in the fictitious code table
Step 5.- Selection of NPDs
Step 6.- Applying the algorithm

Step 6.1 Calculating the indicators for discriminating between the NPDs

• Indicators of the distance between texts
• Indicator of the distance between a text and a category
• Indicator of the distance between the values of the auxiliary variable and the classification codes

Step 6.2.- Coding algorithm

• Number of words in the text u to be coded, in the table of words
• Number of words in the text u to be coded, used in the selection of NPDs
• Number of NPD’s categories
• Usage of the distance indicators

Step 7.- Reiteration process: the text has not been coded and not all its words were used

Step 7.1 Determination of the filter word in the reiteration process
Step 7.2 Obtaining indicators
Step 7.3 Integration of the information derived from both processes
Step 7.4 Applying the algorithm
3. Automatic coding software tool 25

3.1. IT development 25
- Summary of functions
- Technical summary
- Technical glossary

3.2. User’s interaction for every function: interfaces and options 27
  3.2.1. Automatic coding 27
    - Automatic coding using the standard options
    - Automatic coding using the advanced options
  3.2.2. Management of dictionaries and coherence tables 33
    - Management of dictionaries
    - Management of coherence tables
  3.2.3. Management of users and units 37
    - Management of users
    - Management of units

3.3. Web service 40

4. A real-life experiment: recoding the 2010 LFS quarterly samples to ISCO08 using the automatic coding tool 42

  4.1. Stating the problem 42
  4.2. Re-coding process 42
  4.3. Assessment of the results 45

5. Availability of the automatic coding tool for other organizations and countries: Adaptation to other languages 47

  5.1. What is in the CD-ROM? 47
  5.2. What additional works are required for the adaptation to other languages? 48
    5.2.1. Adaptation of the interface to the required language (screens, help files, etc.) 48
    5.2.2. Preparation of the dictionary tables 48
    5.2.3. Adjustment of the algorithm parameters 48
5.2.4. Checking the dictionary coverage and adjusting the algorithm parameters:
   Trial-and-error process 50

5.3. Conclusions 50

6. Bibliography 51
Development of a tool for the automatic coding of the variable ‘occupation’ implementing the new classification ISCO-08

FINAL REPORT

1. Introduction

The main objective of the project was to facilitate the implementation of ISCO-08 in the social surveys as well as in the administrative registers using this variable, especially the public employment services.

The variable occupation is usually collected in statistical surveys by means of a free text. The free texts collected in the questionnaires have to be codified in terms of the national version of ISCO\(^1\). The implementation of the new ISCO-08 entails the need of coding in terms of a classification in which the coding staff is not experienced yet. Due to this, an automatic coding tool would help in the ISCO-08 implementation process.

To this aim, a free software tool has been developed. It can be used by the units in charge of surveys (or administrative registers) collecting the occupation by means of a free text from the respondent in order to automatically code these texts, giving as a result a single code when the text has been coded, or else, a set of candidate codes.

Considering that the questionnaires usually collect other information that can be useful to codify the texts (for instance, when coding a text describing an occupation, the economic activity in which the person is working, the field of education of the degree hold by the person or the employment status can provide some information allowing to discriminate between two or more different ISCO-08 categories), the software tool allows to take advantage of this information in a very flexible way.

This tool and the algorithm behind it can be used in two different ways. On the one hand, it can be used for coding a batch of texts, so a specific interface has been developed to this aim. On the other hand, a web service has been developed so that it can be used in the data collection, when it is carried out by electronic means (CAPI, CAWI or CATI). This will allow guaranteeing that every collected text has enough information to be automatically coded, preventing sub-specification, as well as to keep not only the text but also the corresponding code.

This report includes three different parts. First, the automatic coding algorithm is presented, including how the candidate codes are selected, the different weights used in the discrimination among them, as well as the circumstances under which a description text is codified. The next chapter of this document is devoted to the application itself, its IT development, how the user can interact with it when coding a batch of texts and how the web service is thought to work once implemented in the computer-assisted data collection. Finally, a summary report of a real experience in

\(^1\) The national versión of ISCO88 is CNO94 (Clasificación Nacional de Ocupaciones, 1994). The national versión of ISCO08 is CNO09 (Clasificación Nacional de Ocupaciones 2009).
using this software tool, specifically the re-coding of the four LFS quarterly samples in terms of ISCO-08, is included.
2. Automatic coding algorithm

Automatic coding of the variable *occupation* is part of the Text Categorization, defined in Sebastiani (2002) as “the task of assigning a Boolean value to each pair \((d,c)\in D\times C\), where \(D\) is a domain of documents and \(C=\{c_1,..,c_j\}\) is a set of predefined categories. A value of \(T\) assigned to \((d,c)\) indicates a decision to file \(d\) under \(c\), while a value of \(F\) indicates a decision not to file \(d\) under \(c\)“.

The purpose of this type of categorization is that each response text for this variable be assigned a single code. Furthermore, the structure of the categories remains fixed, it being necessary to assign each text to one of these categories.

This part of the report is devoted to the explanation of the coding algorithm designed for the automatic coding tool. The values in the algorithm have been set through a trial-and-error process, considering the 3-digit level of the classification, as well as basing on real free texts obtained from the Labour Force Survey. Some characteristics of the free texts usually defining occupations collected in statistical surveys, especially in the Labour Force Survey, make them special:

- the texts are usually made up of 2 or 3 words, those of more than 4 words being rare.

- the descriptions are usually very similar, since each sector has a jargon used by the people working in it.

However, the automatic coding tool will be flexible enough to allow the user modifying the values used in the algorithm to adapt it to both the kind of definitions to be coded and the level of the classification to be considered.

2.1. Dictionary

2.1.1. Description and construction

In this document, a *dictionary* means a reference set of texts describing occupations that have been associated to different specific codes of the classification, and it includes some additional information that will be useful in the automatic coding process. The information in the dictionary is organized in different tables (see 2.1.3 *Dictionary tables: Contents and structure*).

The base for the construction of the dictionary in terms of the national version of ISCO08 was a set of free texts describing the occupation gathered in the last Census, the SES or the LFS. All these texts had already been included in the dictionary of words and descriptions used in AYUDACOD, the assisted coding tool already...
available in the INE’s website\textsuperscript{2}. Every text in the AYUDACOD dictionary was coded in terms of the national version of ISCO-88. In order to avoid any delay in designing the algorithm, and since this dictionary was foreseen as being the starting point for the construction of the dictionary of texts in terms of the national version of ISCO-08, it was deemed acceptable that the results obtained using this dictionary in the trial-and-error process would also be valid for the new classification. In deed, the values fixed for the algorithm have been checked once the ISCO-08 dictionary was ready.

In the first dictionary, each text in the set was assigned a CNO94 code, and then they were standardized, basically by means of identifying “main words”, replacing each word identified as a “synonym” by the main word associated to it, and deleting “empty words” (words without relevant information: articles, some prepositions...).

Initially the CNO94 dictionary included:

- 4,523 different main words
- 9,644 different texts
- 6,915 synonyms

Currently the ISCO-08 dictionary includes:

- 4,625 different main words
- 9,905 different texts
- 8,670 synonyms

Although it could be thought that the dictionary is rather limited, its use in AYUDACOD as well as the tests carried out showed that its coverage was acceptable. Anyway, recognizing that the coverage could be improvable, it was decided, as will be pointed out below, that when a text contained more than 1/3 of its words not appearing in the dictionary, it would be directly sent to the coding queue, i.e. it would have to be coded manually by an agent.

2.1.2. Weights given to the main words in the dictionary

An essential element of the automatic coding is to select the candidate texts from the dictionary (called Neighbour Potential Donors\textsuperscript{3} or NPDs in this document) as close as possible to the text to be coded. To this end, it is important to chose the

\textsuperscript{2} http://www.ine.es/EX_INICIOAYUDACOD

\textsuperscript{3} The experience of the assisted coded tool (AYUDACOD) shows that selecting the dictionary texts that have as many as possible words in common with the text to be coded, in most cases at least one text is liable to code the text correctly. Consequently, it was decided that the Nearest Neighbour (Winkler, 2003) method would be used, for which it was necessary to assign a number of weights to the words in the dictionary, which in a later step would be used to define some indicators to measure the distance among the texts. The selected texts from the dictionary are called neighbour potential donors (NPDs) in this document due to the used method. Besides, the use of complete words instead of streams of characters allows to keep and take advantage of the semantic content in the word.
word with most discriminating power (filter word) among those making up the text to be coded.

Two different weights, therefore, were assigned to the words in the dictionary:

- **Weight of the word associated with the dictionary**

  Let’s consider a dictionary including \( m \) texts (derived from surveys) and \( k \) words stemming from these texts. Each text \( t_i (i=1..m) \) in the dictionary may be represented by an array \( t_i=(X_{i1},X_{ik}) \), where:

  \( X_{ij}=1 \) if the \( j^{\text{th}} \) word in the dictionary is included in the text \( t_i \) and
  \( X_{ij}=0 \) otherwise.

  The first weight associated to each word is defined as:

  \[
  w_j = \sum_{i=1}^{m} X_{ij} \quad \forall \quad j = 1...k
  \]

  In other words, the first weight associated to each word is the number of texts in the dictionary including this word, i.e. its frequency in the dictionary.

- **Weight of the word associated with the category**

  It was deemed desirable to give a weight to the words for their discriminating power in relation to the classification categories.

  Let \( c_1,c_2,\ldots,c_l \) be the categories of the classification at an specific level, e.g. 3-digit level. Taking into account that each text \( t_i \) is associated with a single category \( c_{i'} \), the words \( p_j \) that belong to text \( t_i \) can be associated with category \( c_{i'} \).

  As in the above case, an array \( c_{i'}=(X_{i1},X_{ik},X_{i1l}) \) may be defined, where:

  \( X_{i1}=1 \) if the \( j^{\text{th}} \) word in the dictionary is included in text \( t_i \) and the latter is associated with category \( c_{i'} \) and

  \( X_{i1}=0 \) otherwise.

  With each word \( p_j \), a weight per category \( c_{i'} \) is associated, which is defined by

  \[
  w_{jh} = \sum_{l=1}^{l} X_{ihj} \quad \forall \quad j = 1...k, \quad h = 1...l
  \]
To give an index comparable among words, the index

\[ wc_j = \max_h \frac{w_{jh}}{w_j} \quad \forall \quad j = 1 \ldots k \]

is defined.

This weight was calculated for every classification level, for instance, in the case of ISCO-08 for 4 different levels (major groups, sub-groups, minor groups and unit groups)

### 2.1.3. Dictionary tables: Contents and structures

The dictionary has been basically defined as a set of texts associated to the different categories in the classification. These texts were broken down in words. Every word was analyzed in order to identify its relevant semantic content.

A word with no relevant content was categorized as an empty word and included in a specific list. Most empty words were articles, prepositions, conjunctions and similar words. The empty word list is valid for any dictionary, so it was decided to include it in the software tool as part of the general program, instead of including it in every single dictionary.

The remaining words, i.e. the words with a relevant semantic content, were grouped in sets of words with the same meaning. Two words were considered to have the same meaning when the semantic content in each word was very close, considering the variable to be coded using the dictionary in question. For instance, when coding occupations the Spanish word ‘conductor’ (=driver) is considered to have the same meaning as ‘chófer’ (=chauffeur). However, when coding economic activities ‘conductor’ and ‘chófer’ are not considered synonyms because ‘conductor’ also means ‘something that allows heat or electricity to travel along or through it’. This second meaning is not relevant regarding occupations, but it is when describing economic activities (e.g. manufacturing of lightning conductors).

One word in each set of words with the same meaning was categorized as the main word and the rest were considered synonyms of that main word. Each main word was assigned the weights described in section 2.1.2.

Considering these three word categories, every single text was transformed in a standardized or cleaned text. The empty words in the text were deleted, every synonym was replaced by its associated main word, duplicated main words were deleted and the remaining words were alphabetically ordered.

All this information was stored in the following tables:
• Table of texts

This table includes all the texts in the dictionary, the classification code assigned to it and the corresponding standardized or cleaned text. The fields in this table are:

INTEGRATED CODE:  
Dividing blank: First position: 0  Length: 15
CLEANED TEXT: First position: 16  Length: 175
ORIGINAL TEXT: First position: 192  Length: 175

The integrated code included in this table will be used to construct the different codes corresponding to the different levels in the classification. Therefore, the code in this table is not exactly the code in the classification. The integrated code structure is fixed regarding the family of classifications to which the dictionary is related.

There is a system table including the classification families and how to construct the codes for every level. Initially there are two families available, occupations and economic activities. However, new classification families can be added by accessing directly the system table (see the maintenance manual).

Regarding occupations, the integrated code has five positions, in order to include an extra level of the national version between the 1 and 2-digit levels of ISCO08. The system table of classification families specifies how to construct every level code. The 1-digit code is made by selecting the first digit. The national 1-letter code is formed by the second character. When coding at 2-digit level, the code is constructed by selecting the first and the third characters. Adding the fourth character, the 3-digit level is obtained, and the 4-digit one includes also the fifth position. Let’s give an example:

Integrated code:

<table>
<thead>
<tr>
<th>INTEGRATED CODE</th>
<th>LEVEL 1 1-digit</th>
<th>LEVEL 2 1-letter</th>
<th>LEVEL 3 2-digit</th>
<th>LEVEL 4 3-digit</th>
<th>LEVEL 5 4-digit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1A509</td>
<td>1</td>
<td>A</td>
<td>15</td>
<td>150</td>
<td>1509</td>
</tr>
</tbody>
</table>

• Table of words

This table includes all the main words in the dictionary, as well as the number of cleaned texts in the dictionary including every main word, i.e. its frequency in the dictionary (\(w\)) or its weight associated with the dictionary. This information is stored in the following fields:
Development of a tool for the automatic coding of the variable ‘occupation’ implementing the new classification ISCO-08
FINAL REPORT

MAIN WORD: Wc
Dividing blank: First position: 0 Length: 25

• Category/word table

This table includes all the main words in the dictionary and the weight associated with the category calculated for each level in the classification (wc). The following fields are included in this table:

MAIN WORD: Wc, 1-digit level: Wc, 2-digit level: Wc, 3-digit level: Wc, 4-digit level: Wc, 5-digit level:
Dividing blank: First position: 25 Length: 1
Dividing blank: First position: 26 Length: 4
Dividing blank: First position: 32 Length: 1
Dividing blank: First position: 33 Length: 6
Dividing blank: First position: 39 Length: 1
Dividing blank: First position: 40 Length: 6
Dividing blank: First position: 46 Length: 1
Dividing blank: First position: 47 Length: 6
Dividing blank: First position: 53 Length: 1
Dividing blank: First position: 54 Length: 6

• Table of synonyms

This table includes all the words in the texts considered as synonyms and the equivalent main word. This information is kept in two fields:

SYNONYM: MAIN WORD:
Dividing blank: First position: 25 Length: 1
Dividing blank: First position: 26 Length: 25

• Table of fictitious codes

The experience shows that quite a number of very high frequency texts collected in surveys are not going to be coded. The problem lay in the fact that these texts, although perfectly defined, did not contain enough information for them to be coded at the established level (sub-specification) Some examples could be ‘Civil servant’, ‘teacher’, ‘manager’, etc. Removing this kind of texts from the file of texts to be coded would also allow to save time and resources.

To this end, it is possible to include a table of fictitious codes in the dictionary set of tables. This table would include standardized sub-specified texts frequently collected in surveys accompanied by a fictitious code, i.e. a code that is not one of those used in the classification. The transformation of these fictitious codes in classification codes is out of the scope of this project, but it can be done in a later step by defining transformation rules considering all the available information in the questionnaire (e.g. economic activity, sex, age, level of education, field of education, employment
status or even geographical region), and in some occasions, by means of probabilistic models.

The texts in this table are sub-specified for a given level of the classification. For instance, department director could not be classified at 4-digit level in ISCO-08, but at 1-digit level it is clearly included in Major Group 1 Managers. Bearing this in mind, a dictionary including a table of fictitious code should only be used to classify texts at the classification levels at which the texts in the table are sub-specified.

After analysing the free texts collected in the LFS and in the last Census, both coding at 3-digit level of CNO94, 988 texts were included in the table of fictitious codes.

The table of fictitious codes has the following fields:

| FICTITIOUS CODE: | First position: 0 | Length: 15 |
| Dividing blank: | First position: 15 | Length: 1 |
| STANDARDIZED TEXT: | First position: 16 | Length: 175 |

2.2. Steps in the automatic coding process

The objective of the automatic coding is basically to give a classification code to a text describing an occupation. To this aim, the steps of the coding process are:

1) Identification of the text to be coded.
2) Cleaning of the text, locating main words, synonyms and empty words and obtaining a standardized text as a result.
3) Selection of the candidate texts (NPDs) from the dictionary.
4) Applying an algorithm to the selected NPDs in order to select the best one (when possible).
5) Reiteration of the process when a text has not been coded.

Step 1.- Entering the text to be coded

Texts to be coded can be provided in two different ways:

1.- When using the batch coding tool, a set of text from a specific source are submitted to the process. For instance, all the text collected by the Labour Force Survey can be submitted to the process, and the result of the process will be a file of texts, some of them already codified in terms of the national version ISCO-08, and some of them not codified but accompanied by a set of candidate codes.

2.- When using the web service, the data collection software will send the text collected as soon as it has been typed by the respondent or the survey agent, and it will get back either a code, when the text has being coded, or a set of candidate codes.

INE Spain, April 28th, 2011
codes which can be used by the collecting tool to offer them to the respondent, or as a key for popping up a message asking for a more precise description.

Every single text is submitted to the same automatic coding process, independently from the way it has been entered (as an element in a batch or by means of the web service).

**Step 2.- Cleaning of the text to be coded**

In the first place, the text \( u \) to be coded is cleaned:

a) Words without semantic content, called **empty words**, are eliminated.

b) Words included in the **table of synonyms** are replaced by the main word assigned to it.

c) Duplicated main words are eliminated.

d) It is assumed that words are independent and that their order has no influence (naïve Bayes), so the remaining words (hopefully most of them unduplicated main words, but it can also include words not in the dictionary) were alphabetically ordered to build the **standardized or cleaned text** to be coded.

**Step 3.- Matching the cleaned text to be coded with cleaned texts in the table of texts**

In order to reduce the number of texts to be treated, what means a reduction in terms of time and information resources, and regarding the fact that the table of texts also includes a cleaned version of every text, a pre-step to the automatic coding process consists in a direct match of the cleaned text to be coded with the cleaned texts in the table of texts. In case an exact coincidence is found, the code corresponding to this text in the dictionary is directly assigned to the text to be coded, so it is not involved in the next steps of the coding process.

**Step 4.- Matching the cleaned text to be coded with cleaned texts in the fictitious code table**

As explained above, when collecting the information in a specific survey, there is quite a number of very highly frequent texts that are not going to be coded. The problem lay in the fact that these texts, although perfectly defined, did not contain enough information for them to be coded at the established levels (sub-specification). These texts can optionally be kept in a table of fictitious codes.

If the cleaned text to be coded is identical to one in the fictitious code table, it is given the fictitious code, without being involved in the next steps of the automatic coding process.
These fictitious codes have to be analyzed one by one and solved using ancillary variables (for instance level of education, socio-economic status and economic activity when coding occupations) or maybe using a probabilistic imputation model based on external information. This transformation of the fictitious codes into classification codes is not part of the automatic coding tool, and has to be defined by every user considering the auxiliary information available in each case.

**Step 5. Selection of NPDs**

In case the cleaned text to be coded does not match neither a text in the table of texts nor a text in the table of fictitious codes, it is submitted to the actual automatic coding process, being the first step the selection of **NPDs**.

The experience of the assisted coded tool (AYUDACOD) shows that selecting the dictionary texts that have as many as possible words in common with the text to be coded, in most cases at least one text is liable to code the text correctly. Consequently, it was decided that the Nearest Neighbour (Winkler, 2003) method would be used. The selected texts from the dictionary are called **neighbour potential donors (NPDs)** in this document due to the used method.

In the first place the **filter word** of the text **u** to be coded is selected according to the system explained below:

From the words of the text to be coded **u** that are included in the table of words, the above defined two weights (**w_j**, **w_C**) are considered, the **filter word** being determined according to the following conditions:

1. From the words of the text to be coded **u** with **w_C** > 0.3 and **w_j** > 4, the word of maximum **w_j** is selected as the **filter word**, the aim being the achievement of a **filter word** that is representative in some code and appears very frequently.

2. If the above condition is not fulfilled, we have one of the following cases:
   2.1. **w_C** ≤ 0.3 y **w_j** > 4
   2.2. **w_C** > 0.3 y **w_j** ≤ 4
   2.3. **w_C** ≤ 0.3 y **w_j** ≤ 4

If the conditions set in 1 are not fulfilled by any word in the text, the word of maximum **w_j** is selected, provided that **w_j** < 100.

3. If all the words have **w_j** ≥ 100, the first word in alphabetical order is selected.

Once the **filter word** has been selected, we obtain a first batch of **NPDs** made up of those texts of the table of texts that contained the **filter word** in question. From this batch of texts, a selection is then made of those that have the greater amount of words in common with the text **u** to be coded. Among these **NPDs**, different
combinations of words in common with the text to be coded could be included. The final set of NPDs is made up of those including the most frequent combination.

**Step 6.- Applying the algorithm**

The algorithm is based on the calculation of several indicators using the weights assigned to the words in the table of words, as well as on the grouping of NPDs associated to the same classification code. Thus, different values are assigned to each candidate code according to its proximity to the text to be coded.

**Step 6.1 Calculating the indicators for discriminating between the NPDs**

After step 5, a batch of $z$ NPDs has been selected. It is possible to define several different indicators. Two of them measure the distance from the NPD $t_i$ ($i=1...z$) to the text $u$ to be coded. The other two indicators are one measuring the distance between the text $u$ to be coded and every category in the classification $c_h$ ($h=1...l$), and another one measuring the distance between a value of a defined auxiliary variable and a code in the classification. Two different auxiliary variables were used when testing the automatic coding of occupations in CNO11 codes, the economic activity in terms of CNAE09 (National version of NACE Rev.2) and a previously given CNO94 code. Nevertheless, every user will be able to decide the auxiliary variable to be used, considering its own available information, being even able to select a set of different variables. This will be explained below.

- **Indicators of the distance between texts**

In order to measure the distance from a NPD $t_i$ to the text to be coded $u$, three different weights are considered: the weight assigned to the NPD, the weight assigned to the text to be coded and the weight assigned to the words used in the NPD selection.

The **weight assigned to the NPD $t_i$** can be defined as the addition of the weights assigned to the words of the cleaned text in the table of words (for a definition of the used values, see 1. Dictionary):

$$w_{ti} = \sum_{j=1}^{k} w_j x_{ij} \quad \forall \ t_i \text{ liable neighbour}$$

When considering the text to be coded $u$, the corresponding array can be defined as $u=(x_{ui},..,x_{uk})$ where:

$X_{ui}=1$ if $j^{th}$ word in the table of words is included in text $u$, and

$X_{ui}=0$ otherwise.

The **weight assigned to the text $u$ to be coded** can be defined as:

---

INE Spain, April 28th, 2011
\[ W_U = \sum_{j=1}^{k} w_j x_{uj} \]

In order to assign a weight to the set of common words used in the selection, a new array is defined as \( (Y_{ui}, Y_{uj}) \), where:

- \( Y_{ui} = 1 \) if \( g^{th} \) word of the text \( u \) to be coded is included in the \( NPD \) \( t_i \), and
- \( Y_{ui} = 0 \) otherwise.

The **weight assigned to the common words** between the text \( u \) to be coded and the \( NPD \) \( t_i \) can be defined as:

\[ W_A = \sum_{g=1}^{#(u)} \sum_{j=1}^{k} w_j x_{ij} Y_{gi} \]

Since the set of common words is the same in all the selected \( NPDs \), \( W_A \) will have the same value for all of them.

The **indicator of the distance between a \( NPD \) \( t_i \) and the text to be coded \( u \)** is defined as:

\[ IND3 = \frac{W_A}{W_U} \times \frac{W_A}{W_{ti}} = \frac{\sum_{g=1}^{#(u)} \sum_{j=1}^{k} w_j x_{ij} Y_{gi}}{\sum_{j=1}^{k} w_j x_{ij}} \times \frac{\sum_{g=1}^{#(u)} \sum_{j=1}^{k} w_j x_{ij} Y_{gi}}{\sum_{j=1}^{k} w_j x_{ij}} \]

This indicator varies from 0 to 1. \( IND3 \) equals 1 when the text to be coded coincides with the \( NPD \), and it equals 0 when there are no words in common. Then, an indicator value closer to 1 represents a shorter distance between texts.

This indicator entails a problem generated by the words with a high frequency in the dictionary, which have a great influence on the indicator value. In this sense, it is possible to define an analogous indicator for measuring the distance from one text to another, considering the weights of the words in relation to the category\(^4\) \( wc \);

\( \text{\textsuperscript{4}} \text{This category belongs to a specific level of the classification} \)
Let $w_{ca}$ be the weight assigned to the common words between the NPD and the text to be coded:

$$WC_A = \sum_{g=1}^{#(u)} \sum_{j=1}^{K} w_{cj} X_{ij} Y_{gj}$$

Let $w_{ct}$ be the weight assigned to the NPD $t$:

$$WC_T = \sum_{j=1}^{K} w_{cj} X_{ij}$$

Let $w_{cu}$ be the weight assigned to the text to be coded $u$:

$$WC_U = \sum_{j=1}^{K} w_{cj} X_{uj}$$

The new indicator of the distance between a NPD $t$, and the text to be coded $u$ is defined as:

$$IND4 = \frac{WC_A}{WC_U} \times \frac{WC_A}{WC_T} = \frac{\sum_{g=1}^{#(u)} \sum_{j=1}^{K} w_{cj} X_{ij} Y_{gj}}{\sum_{j=1}^{K} w_{cj} X_{uj}} \times \frac{\sum_{g=1}^{#(u)} \sum_{j=1}^{K} w_{cj} X_{ij} Y_{gj}}{\sum_{j=1}^{K} w_{cj} X_{ij}}$$

This indicator varies from 0 to 1. As it happens with IND3, an indicator value closer to 1 represents a shorter distance between texts.

- **Indicator of the distance between a text and a category**

Let $n$ be the number of NPDs selected from the table of texts. Every NPD is assigned to an only category of the classification.

It is possible to define the array $c_n=(Z_{ni}, Z_{ni})$, where:

$Z_{ni}=1$ if the $j^{th}$ selected NPD have assigned the category $c_{ni}$ and

$Z_{ni}=0$ otherwise.

Let $\alpha_n$ be the number of NPDs selected in the category $c_n$: 
\[ \alpha_h = \sum_{i=1}^{n} Z_{ih} \]

On the other hand, let \( \beta \) be the total number of selected NPDs:

\[ \beta = \sum_{i=1}^{l} \sum_{h=1}^{n} Z_{ih} \]

The indicator is defined as:

\[ IND_{2h} = \frac{\alpha_h}{\beta} = \frac{\sum_{i=1}^{n} Z_{ih}}{\sum_{h=1}^{l} \sum_{i=1}^{n} Z_{ih}} \]

This indicator varies from 0 to 1. \( IND_{2h} \) equals 1 when all the selected NPDs are classified in category \( c_{i*} \) and it equals 0 when no selected NPDs are assigned to this category. Therefore, an indicator value closer to 1 represents a shorter distance between the text to be coded and the category.

- **Indicator of the distance between the values of the auxiliary variable and the classification codes**

A rather frequent situation is that of the file to be coded not only including texts, but also some auxiliary information. It is possible to make good use of that auxiliary information, especially when it is coded in terms of its own classification (for instance economic activity in terms of NACE Rev.2, or its national version CNAE09).

Regarding the fact that the auxiliary information available can broadly vary from a user to another, the value of this indicator is directly collected from a table, called **Coherence table**, which includes pairs formed by one code of occupation (ISCO-08, or its national version, CNO11) and one code of the ancillary variable, as well as the value of \( IND_{1} \) assigned to this pair. \( IND_{1} \) values should be fixed between 0 (impossible combination) to 1 (perfect correlation). The values of \( IND_{1} \) can be set freely by the user, but in the tests we have calculated them considering the frequency of every ISCO08/NACE Rev.2 pair (or more precisely, pairs of their national versions CNO11/CNAE09) in the LFS sample.

The coherence table includes the following fields:
Some coherence tables are available in the system for the most common auxiliary variables. For instance, when coding in terms of CNO11 at 3-digit level, CNAE09 (or NACE Rev.2) at 3-digit level is a good auxiliary information usually available in surveys, so a coherence table is available for any user in order to code in CNO11 using CNAE09 as auxiliary variable, or in case a file already coded in CNO94 has to be recoded to CNO11, a specific coherence table is also available using CNO94 as auxiliary information.

However, there are so many different possibilities regarding the auxiliary information and the classification levels that it was deemed desirable that any user could use its own coherence table, built according to its own auxiliary information and fulfilling some specifications (structure of the table, IND1 values between 0 (impossible combination) to 1 (perfect correlation)). In that way, a user could create its own coherence table, regarding the auxiliary information available, being even able of using a concatenation of codes corresponding to different variables as auxiliary code (for instance 3 digits of NACE Rev.2 and 2 digits of ISCED\(^5\) level of education).

If there is no auxiliary information available or the user decides not to use such a table, the coding tool gives IND1 a value of 0.01 in order to let the algorithm code the free text.

**Step 6.2.- Coding algorithm**

Through a trial-and-error process, an algorithm was created for decision making in the coding process. In addition to the values of the different indicators already mentioned, the algorithm uses other values:

- **Number of words in the text \(u\) to be coded, in the table of words**

For a starter and since there could be some doubts as to the coverage of the dictionary, it is deemed desirable to code only those texts whose majority of words are in the table of words.

It is possible to represent the array \(u=(U_1,\ldots,U_n)\) where:

\[ U_i=1 \text{ if the } i^{th} \text{ word of text } u \text{ is included in the table of words and} \]

Considered for the selection of

It is possible to consider two integer values:

a) The total number of words of the text to be coded $u$:

$$ D_1 = \#(u) $$

b) The number of words of the text to be coded $u$ that are in the table of words:

$$ D_2 = \sum_{i=1}^{\#(u)} U_i $$

If $D_2 \leq D_1/2$ (or $D_1/D_2 \leq 1/2$), then the text would be discarded and passed on to the coding queue. This means that a text of 2 words would only be coded if both words are in the dictionary; if a text has 3 words, it would be coded if at least 2 of them are in the dictionary and if the text has 4 words, at least 3 of them should belong to the dictionary.

- **Number of words in the text $u$ to be coded, used in the selection of NPDs**

Once the NPDs have been selected, the first important fact is the number of words considered for the selection of NPDs ($A$) as compared to the number of words in the text to be coded also included in the table of words ($B$). If it is relatively low ($A/B < 0.34$) the process is reiterated. On the contrary, if $A/B \geq 0.34$, the coding is attempted.

If all the words are used for coding ($A/B = 1$), whether in the first process or in the reiteration, the result obtained can be coded or not coded, being part in this last case of the coding queue.

- **Number of NPD’s categories**

In the next step, the number of different classification categories presented by the NPDs is tackled.

$\alpha_h$ ($h = 1..l$) is the number of NPDs selected in category $c_h$. If we consider the set

$$ N = \{ \alpha_h \mid \alpha_h > 0 \} $$

It is possible to define the number $\eta$ of NPD’s categories, as

$$ \eta = \#(N) $$
The treatment varies to the number of categories obtained.

- **Usage of the distance indicators**

In some cases, when taking a decision, account is taken not only of the best \( NPD \) having a given value but also of it being much better than the second \( NPD \), as it is shown in the algorithm on the next page. Different values are used in the algorithm:

\[
D_u \text{- Number of words of the text } u \text{ to be coded}
\]

\[
D_{uw} \text{- Number of words of the text to be coded } u \text{ that are included in the dictionary of words}
\]

\[
A_u \text{- Number of words of the text to be coded } u \text{ used for the selection of NPDs}
\]

\[
B = D_{uw} \text{- Number of words of the text to be coded } u \text{ that are included in the dictionary of words}
\]

\[
\eta \text{- Total number of NPD’s categories}
\]

\[
\beta \text{- Total number of selected NPDs}
\]

\[
IND_1 \text{- Indicator of distance between a pre-code and a code}
\]

\[
IND_2 \text{- Indicator of distance between a text and a category}
\]

\[
IND_3 \text{- Indicator of the distance between a NPD } t \text{ and the text to be coded } u \text{ considering the weights } w_i \text{ assigned to the words in the dictionary.}
\]

\[
IND_4 \text{- Indicator of the distance between a NPD } t \text{ and the text to be coded } u \text{ considering the weights } w_i \text{ assigned to the words in relation to the categories.}
\]

\[
IND_5 = IND_2 + IND_3
\]

\[
IND_6 = IND_1 + IND_4 + IND_5 = IND_1 + IND_2 + IND_3 + IND_4
\]

\[
R = 1.6 \cdot (IND_5(\text{max})-1)/4
\]

\[
X = 1.6 \cdot (IND_6(\text{max})-1)/4
\]

Every exit in the algorithm is identified with a key. The exit key is included in the corresponding yellow square. This information will be very useful when analysing the results of the automatic coding process.
Development of a tool for the automatic coding of the variable ‘occupation’ implementing the new classification ISCO-08

FINAL REPORT

D1  Exact matching of the cleaned text to be coded with a cleaned text in the corpus of texts

D2  Exact matching of the cleaned text to be coded with a text in the fictitious code table

---

LITERALES CANDIDATOS

COLA DE CODIFICACION

---

M1  CODING QUEUE

M2  CODING QUEUE

M3  CODING QUEUE

M4  CODING QUEUE

M5  CODING QUEUE

M6  CODING QUEUE

M7  CODING QUEUE

---

C1  CODED

C2  CODED

C3  CODED

C4  CODED

C5  CODED

C6  CODED

C7  CODED

C8  CODED

---

INE Spain, April 28th, 2011
Step 7.- Reiteration process: the text has not been coded and not all its words were used

As explained earlier, the incoming texts are usually short and it is, therefore, necessary to ensure that their entire information has been taken into account. To this end, if the text to be coded is not resolved in a coding process and if at least one of its words has not been considered, there will be a reiteration of the process and the resulting information will be integrated later.

Step 7.1 Determination of the filter word in the reiteration process

The selection of NPDs in the reiteration process follows the same steps as the initial process, except that the filter word is obtained from those that were not taken into account in the first coding process. If there are several words:

1. If there are words with \(4 < w_i \leq 100\), the filter word is that of them with maximum \(w_c\).
2. If there are only words with \(w_i < 4\) or \(w_i > 100\), the filter word is the one that having \(w_i < 4\) has maximum \(w_c\).
3. If there are only words with \(w_i > 100\), the reiteration would not be carried out because there would be too many non-relevant selected NPDs.

Step 7.2 Obtaining indicators

The same indicators are obtained as in the initial coding process.

Step 7.3 Integration of the information derived from both processes

After the reiteration process, two different files have to be joined, one with the information available before reiteration (FILE1) and one with the information obtained from reiteration (FILE2).

Those categories that comply with condition \(IND3 > 0.3\) or \(IND4 > 0.3\) or \(\alpha > 1\) in FILE1 or in FILE2 are selected as NPD's categories, the result being a new file F of NPDs.

For F the indexes are redefined naturally. In the case of the indicators of the distance between texts (IND3, IND4), the highest value of the values assigned in FILE1 and in FILE2 is taken.

Step 7.4 Applying the algorithm

The file resulting from step 7.3 is submitted to the automatic coding algorithm as described in step 6.2. If the result for a given text after this reiteration is one of those labeled as “REITERATION”, the process is not reiterated again, and the text is sent to the coding queue.
3. Automatic coding software tool

A software tool applying this algorithm was developed, and some tests were carried out to check the programming. In order to conduct these tests, a set of texts was defined, including examples of every different case in the algorithm. Every single indicator and value was manually calculated for every text in the set, and the algorithm scheme was applied. Then the set was submitted to the software process and the results were compared. When an error was detected, it was corrected and the test was repeated.

Once the programming of the algorithm had been checked, new tests were conducted using texts from SES, LFS and other surveys, aiming at checking the algorithm in a real situation, the percentage of coded texts and the quality of the coding, being the most relevant test the re-coding of the 2010 LFS to CNO11 (see 4. A real-life experiment: recoding the 2010 LFS quarterly samples to ISCO08 using the automatic coding tool).

A web automatic coding application has been developed for the batch process. It also includes interfaces for managing users, dictionaries and coherence tables (see 3.2).

A web service is also available, and it has been checked using an interface specifically developed to this end (see 3.3).

3.1. IT development

- **Summary of functions**

The automatic coding application is divided into three functional modules that correspond to the three interfaces, which shows:

**Automatic coding:** Carries out the central activity of the application, the automatic coding of texts. Users and administrators access this interface.

In order to begin coding texts, it is necessary to introduce a flat text file with the batch of texts, the dictionary and the coherence table, against which the coding of the texts shall be carried out. The selection of the coherence table is optional, and the user may use its own, through a flat text file, which shall be removed once the coding is complete.

---

\* For a more in-depth comprehension of the technological issues related to this tool, three different guides have been written: a User’s guide, a Software installation guide and a Software Maintenance guide. They are attached to this report.
There is an advanced-option section that enables the modification of the coding algorithm parameters and of the position and length of the entry file fields.

**Dictionaries and tables:** This allows managing the dictionaries and the coherence tables. Access to this interface is restricted to administrators.

In order to register a dictionary, flat text files must be introduced that correspond to the tables composing a dictionary. Registering the coherence table works along similar lines. Notification of registration or error is sent by email.

Deleting of dictionaries and/or coherence tables is carried out by selecting the dictionary or table that one wishes to delete.

There is not an option for modifications of dictionaries. Any change will need deleting the dictionary and adding the modified one.

**Users and units:** It is used for managing users and units. Access to this interface is restricted to administrators.

Registrations are made by entering the required fields of a user or a unit.

Deleting of users or units is carried out by selecting the user or unit to be deleted.

Modifications are carried out by selecting the user or unit to be changed, and entering the modifications.

For a thorough explanation of these three interfaces, see 3.2 User’s interaction for every function: interfaces and options.

- Technical summary

The "Automatic Coding" application is an online application that has been implemented with the following technologies: JSF, Spring and JAX-WS. Its structure is essentially the same as an online application that implements the MVC pattern, but specifically in which access to the model is through web services. It is displayed in the application server, through two modules:

**AutomaticEncoding:** This contains the view and control of the application, that is, it shows the results, based on JSF and Spring, and carries out the control of which calls to the model should be made, through web services using JAX-WS.

**AutomaticEncodingWS:** This contains the business logic, the model. There are four online services, one for each functional module and another for utilities. This has been performed using JAX-WS and Spring to manage the business logic and as support for the data access with JDBC. The model is prepared to be able to connect to any DBMS, adding the corresponding JDBC controller and the table creation files, such as are indicated in the maintenance manual.

INE Spain, April 28th, 2011
• Technical glossary

**API** (Application Programming Interface): Set of procedures and methods providing a specific library to be used by other software such as abstraction layer.

**JAX-WS** (Java API for XML Web Services): This is a Java API for creating web services.

**JDBC** (Java Database Connectivity): This is an API enabling the executing of operations on databases from Java, irrespective of the operating system from which it is executed or of the database being accessed.

**JSF** (JavaServer Faces): This is a web-based technology and framework for Java applications, which simplifies the development of user interfaces in applications.

**MVC** (Model-View-Controller): This is a software architecture pattern type that separates application data, the user interface, and control logic into three different components.

**DBMS** (Database Management System): Specific software dedicated to being used as an interface between the database, the user and the applications using it.

**Spring**: This is an open-source application development framework for the Java platform.

### 3.2. User’s interaction for every function: interfaces and options

This part of the report presents the general interfaces aiming at coding batch of texts and managing dictionaries and users. Part 3.3 is devoted to the web service.

#### 3.2.1. Automatic coding

All users and administrators can access this interface.

When coding a batch of texts, two options are offered to the user: on the one hand the user can provide the batch of texts on a flat text file with a given structure of fields and apply the algorithm taking the default values, or on the other hand, the user can define its own flat text file structure and give different values to the algorithm parameters.

• **Automatic coding using the standard options**

It is advisable for non-advanced users to choose the default options. In this case, the following steps have to be taken:
Development of a tool for the automatic coding of the variable ‘occupation’ implementing the new classification ISCO-08

FINAL REPORT

1. Fill the box ‘F. a codificar’ (File to be coded) with the full path of the file to be coded. To help in this, the user can press the button ‘Examinar’ (Explore) to open an explorer and search it through the tree of folders. The flat text file must follow the following structure of fields:

   **RECORD ID:**
   First position: 0  Length: 50
   Dividing blank: First position: 50  Length: 1

   **TEXT TO BE CODED:**
   First position: 51  Length: 175
   Dividing blank: First position: 226  Length: 1

   **VALUE OF THE AUXILIARY VARIABLE:**
   First position: 227  Length: 15

2. Optionally, the user can provide its own coherence table by filling the box ‘F. coherencia’ (Coherence file) with the full path of its own coherence file. To help in this, the user can press the button ‘Examinar’ (Explore) to open an explorer and search it through the tree of folders. The flat text file must follow the following structure of fields:

   **CODE OF THE CLASSIFICATION:**
   First position: 0  Length: 15
   Dividing blank: First position: 50  Length: 1

   **VALUE OF THE AUXILIARY VARIABLE:**
   First position: 16  Length: 15
   Dividing blank: First position: 50  Length: 1

   **IND1 VALUE FOR THIS PAIR:**
   First position: 32  Length: 8

3. Select an exit file format (Excel/Text). In order to save time and resources, when coding a batch of 200 or more texts it is advisable to select a text file.

4. Select the exit file option (Standard/Advanced). The **standard exit file** includes the basic information resulting from the coding process:
**Development of a tool for the automatic coding of the variable “occupation” implementing the new classification ISCO-08**

**FINAL REPORT**

| RECORD ID: | First position: 0 | Length: 50 |
| TEXT TO BE CODED: | First position: 50 | Length: 1 |
| Dividing blank: | |
| TEXT TO BE CODED: | First position: 51 | Length: 175 |
| Dividing blank: | |
| VALUE OF THE AUXILIARY VARIABLE: | First position: 227 | Length: 15 |
| Dividing blank: | |
| BEST CANDIDATE CODE: | First position: 243 | Length: 15 |
| Dividing blank: | |
| EXIT KEY: | First position: 259 | Length: 8 |
| Dividing blank: | |
| FIVE BEST CANDIDATE CODES: | First position: 268 | Length: 50 |

The BEST CANDIDATE CODE is the code assigned to the text when the EXIT KEY is identifying a “coded” exit (in this case, the exit key is “D1” or its first character is “C”). When the exit key is “D2”, it means that the best candidate code is not a real code, but a fictitious code (see 2.1.3).

The **advanced exit file** includes additional information. It could be more useful for an expert user, specially to check how the parameterized values of the algorithm are working. To this aim, every value used in the algorithm is included. The fields of this advanced exit file are:

| RECORD ID: | First position: 0 | Length: 50 |
| TEXT TO BE CODED: | First position: 50 | Length: 1 |
| Dividing blank: | |
| TEXT TO BE CODED: | First position: 51 | Length: 175 |
| Dividing blank: | |
| CLEANED TEXT TO BE CODED: | First position: 227 | Length: 175 |
| Dividing blank: | |
| VALUE OF THE AUXILIARY VARIABLE: | First position: 403 | Length: 15 |
| Dividing blank: | |
| BEST CANDIDATE CODE: | First position: 419 | Length: 15 |
| Dividing blank: | |
| SAMPLE DICTIONARY TEXT: | First position: 435 | Length: 175 |
| Dividing blank: | |
| D1: | First position: 610 | Length: 1 |
| Dividing blank: | First position: 611 | Length: 4 |
| Dividing blank: | First position: 615 | Length: 1 |
| D2: | First position: 620 | Length: 1 |
| Dividing blank: | First position: 621 | Length: 4 |
| Dividing blank: | First position: 625 | Length: 1 |
| A: | First position: 630 | Length: 1 |
| Dividing blank: | First position: 631 | Length: 4 |
| Dividing blank: | First position: 635 | Length: 1 |
| η: | First position: 636 | Length: 4 |
| Dividing blank: | First position: 640 | Length: 1 |
| α,: | First position: 641 | Length: 4 |
| Dividing blank: | First position: 645 | Length: 1 |
| IND1: | First position: 654 | Length: 8 |

INE Spain, April 28th, 2011

29
Development of a tool for the automatic coding of the variable ‘occupation’ implementing the new classification ISCO-08
FINAL REPORT

IND2:
Dividing blank: First position: 655 Length: 8

IND3:
Dividing blank: First position: 663 Length: 1

IND4:
Dividing blank: First position: 672 Length: 1

INDS:
Dividing blank: First position: 690 Length: 1

IND6:
Dividing blank: First position: 699 Length: 1

EXIT KEY:
First position: 700 Length: 3

5. Select the classification level in which the coding is going to be done. As an example, when coding in terms of ISCO08, level 1 should be used when trying to give a 1-digit code to the texts; if you want to want to give them a 2-digit code, you should select level 2, and so on.

6. Select a dictionary in the ‘Diccionarios’ (Dictionaries) area.

7. Optionally, and if the user has not provide its own coherence table, a pre-loaded coherence table can be selected in the ‘Tablas de coherencia’ (Coherence tables) area. If the user neither selects a coherence table nor provides its own one, the algorithm will apply the default value IND1=0.1, not making any discrimination between candidates for this IND1. For instance, this will be the case when no auxiliary variables are available.

8. Press the button ‘Codificar’ (Coding).

9. When the coding process is finished, the user will get a warning e-mail including a download link to the exit file. The file will be available there for a week, and once this period is over, it will be deleted from the server.

- Automatic coding using the advanced options

Users can define some advanced options. They can define its own flat text file structure and/or give different values to the algorithm parameters.

In order to be able to do this, the user has to tick the ‘Avanzado’ (Advanced) box. Then, two option sets are offered to the user, one for customizing the values in the algorithm and another one for locating the relevant fields in the entry file when it does not have the standard format.
A user who wants to customize the values in the algorithm must follow the following steps:

1. Click on the ‘Edit’ (Edit) button.
2. Type the values to use (up to 4 decimal positions).
3. Click on the ‘Save’ (Guardar) button.

There are two additional buttons in this area:

- The ‘Unmake’ (Deshacer) button gives back the original value to all the algorithm parameters, in other words, unmake all the changes made by the user.
The ‘Algoritmo’ (Algorithm) button shows a picture of the algorithm scheme, identifying the value to be modified in every text box:

If the entry flat text file does not fulfill the pre-fixed structure of fields (see page 24), the user has to specify the first position and the length of each relevant field (RECORD ID, TEXT TO BE CODED and VALUE OF THE AUXILIARY VARIABLE).

In order to be able to do this, the user has to follow the following steps:

1. Click on the ‘Editor’ (Edit) button.

2. Enter the first position and the length of every relevant field, being aware that the first position in every row is position 0. The last position will be updated automatically so that the user can check it.
3. Click on the ‘Guardar’ (Save) button.

The ‘Deshacer’ (Unmake) button gives back the default values, in other words, the values in text boxes are replaced by those corresponding to the pre-fixed file structure.

Apart from these two advanced options, the steps for coding a file are the same described in the standard-option part.

3.2.2. Management of dictionaries and coherence tables

Only users with an administrator role can access this interface.

The interface is divided in two different areas, one intended for the management of dictionaries and the other one for managing coherence tables.

- Management of dictionaries

The management of dictionaries considers two different actions: to add a new dictionary and to delete an existing one. There is no option for changing an existing dictionary.
In order to **add a new dictionary**, the administrator must organize all the dictionary information following the structure of tables described in 2.1.3 **Dictionary tables: Contents and structures.**

The administrator must follow the steps below:

1. Fill the box ‘F. Literales’ with the full path of the file containing the *Table of texts*. The provision of this table is compulsory. To help locating the corresponding file, the user can press the button ‘Examinar’ (Explore) to open an explorer and search it through the tree of folders.

2. Fill the box ‘F. Palabras’ with the full path of the file containing the *Table of words*. The provision of this table is compulsory.

3. Fill the box ‘F. Lit. Palabras’ with the full path of the file containing the *Category/word table*. The provision of this table is compulsory.

4. Fill the box ‘F. Sinónimos’ with the full path of the file containing the *Table of synonyms*. The provision of this table is compulsory.

5. Fill the box ‘F. Ficticios’ with the full path of the file containing the *Table of fictitious codes*. This box is labeled with ‘(*)’ in order to remark that the provision of this table is optional (non-mandatory). As explained in 2.1.3, the fictitious codes are designed to be valid for a specific level of the classification, so when a fictitious code table is provided, the dictionary will only be useful for coding at this or lower levels (for instance, if a text does not contain information enough to be coded at 3-digit level of ISCO08, it will not have information enough to be coded at 4-digit level, but it could have enough information to be coded at 2-digit level).

6. Fill the text box ‘Nombre’ *(Name)* with the name assigned to the dictionary. The name of the dictionary should follow a specific structure defined by the administrator.
7. Fill the text box ‘Descripción’ (Description) with a wider description of the dictionary. It would be advisable to include in this description the date when the dictionary was uploaded.

8. Select the type of dictionary in the ‘Tipo’ (Type) combo box. There are two different types, general or unit. A unit dictionary has been created and will be used by a specific unit, generally for a specific survey. It may contain some specifications that make it not usable for any survey or other purpose, so only the “owner” unit can access it. On the other hand, a general dictionary can be used by any user of the software.

8. Select the unit that created the dictionary in the ‘Unidad’ (Unit) combo box. Any user of the system is assigned to a unit, so the units already in the system are available in this combo box.

9. Select the reference classification family in the ‘Clasificación’ (Classification) combo box. Initially there are two classifications available, occupations and economic activities. However, new classification families can be added by accessing directly the proper system table (see the maintenance manual).

10. Click on the ‘Nuevo’ (New) button.

The uploading process starts, and when it finish, a warning e-mail will be sent to the administrator.

The ‘Limpiar’ (Clean) button deletes the information in every box and cancels the uploading.

For deleting an existing dictionary, some steps must be undertaken:

1. Locate the dictionary to be deleted in the list of dictionaries. To this end, several filters can be set: by type of dictionary, classification and/or unit. The filter is set by selecting the preferred value in the corresponding combo box.

2. Once the dictionary has been located, click on the delete symbol (Trash). The system will ask for a confirmation.

- Management of coherence tables

The management of coherence tables, as it happens with the management of dictionaries, allows to upload a new coherence table or to delete an existing one, but it is not possible to modify the already available ones.
In order to add a new coherence table, the administrator must follow the steps below:

1. Fill the box ‘F. Coherencia’ with the full path of the file containing the Coherence table, which has to follow the structure presented in 3.2.1 Automatic coding. To help locating the corresponding file, the user can press the button ‘Examinar’ (Explore) to open an explorer and search it through the tree of folders.

2. Fill the text box ‘Nom. Tabla’ (Table name) with the name assigned to the coherence table. The name should follow a specific structure defined by the administrator.

3. Fill the text box ‘Desc. Tabla’ (Table description) with a wider description of the coherence table.

4. Select the type of dictionary in the ‘Tipo’ (Type) combo box. There are two different types, general or unit. A unit dictionary has been created and will be used by a specific unit, generally for a specific survey. It may contain some specifications that make it not usable for any survey or other purpose, so only the “owner” unit can access it. On the other hand, a general dictionary can be used by any user of the software.

5. Select the reference classification family in the ‘Clasificación’ (Classification) combo box.

6. Select the level of the classification the coherence table is intended for in the ‘Niv. Cod’ (Coding level) combo box.

7. Click on the ‘Nuevo’ (New) button.

The uploading process starts, and when it finishes, a warning e-mail will be sent to the administrator.

The ‘Limpiar’ (Clean) button deletes the information in every box and cancels the uploading.

For deleting an existing coherence table, some steps must be undertaken:
1. Locate the coherence table to be deleted in the list of coherence tables.

2. Once it has been located, click on the delete symbol (IBAction). The system will ask for a confirmation.

3.2.3. Management of users and units

Only users with an administrator role can access this interface.

The interface is divided in two different areas, one intended for the management of users and the other one for managing units, and these areas are in turn divided in two halves, one for uploading new items and another for deleting existing ones.

- Management of users

The management of users considers three different actions: add a new user, delete an existing one or modify its features.
In order to **create a new user** the administrator must:

1. Fill the text box ‘Nombre’ (Name) with the user’s first name.

2. Fill the text box ‘Primer Apellido’ (First Surname) with the user’s surname. There are countries, like Spain, in which two surnames are used. That is why there are two text boxes for surnames.

3. Fill the text box ‘Segundo Apellido’ (Second Surname) with the user’s second surname. His box is labeled (*) because it is optional.

4. Fill the text box ‘Email’ with the user’s e-mail address. All the notifications related to processes launched by the user will be sent to this e-mail address.

5. Fill the text box ‘Login’ with the user’s identification.

6. Fill the ‘Contraseña’ (Password) with the user’s access password.

7. Select the corresponding unit in the ‘Unidad’ (Unit) combo box. Any user of the system has to be assigned to a unit. The combo box offers the units already uploaded in the system. Should the user be assigned to a new unit, it has to be created prior to the user.

8. Select the role assigned to the user in the ‘Rol’ (Role) combo box. There are two different roles: user and administrator. The user can only access the automating coding interface, and can launch automatic coding processes. The administrator can, in addition, manage dictionaries, coherence tables, users and units by accessing the corresponding interfaces.

The ‘Limpiar’ (Clean) button deletes the information in every box.

In order to **modify the features of an existing user**, the step to follow are:

1. Locate the user whose features are going to be changed in the list of users. To this end, several filters can be set by typing the filter values in the same box used to introduce the information for a new user and clicking on the ‘Buscar’ (Search)
button. Users in the list can be filtered by name, first and second surname, e-mail address, unit and/or role, but not by login nor password.

2. Once the user has been located, click on the modify symbol (MOD). The system will load the user’s features in the boxes.

3. Modify the values to be changed.

4. Click on the ‘Guardar’ (Save) button.

For **deleting an existing user**, some steps must be undertaken:

1. Locate the user to be deleted in the list of users.

2. Once it has been located, click on the delete symbol (DEL). The system will ask for a confirmation.

**Management of units**

The management of units considers three different actions: add a new one and delete or modify an existing one.

In order to **create a new unit** the administrator must:

1. Fill the text box ‘Código’ (Code) with a 4-letter code which will identify the unit.

2. Fill the text box ‘Nombre’ (Name) with the unit short name.

3. Fill the text box ‘Descripción’ (Description) with a wider description of the unit.

4. Click on the ‘Nuevo’ (New) button.

The user can also **modify the features of an existing unit**:

1. Locate the unit to be modified in the list of units. To this end, several filters can be set by typing the filter values in the corresponding box and clicking on the ‘Buscar’ (Search) button.
2. Once the unit has been located, click on the modify symbol (-pencil). The system will load the unit features in the boxes.

3. Modify the values to be changed.

4. Click on the ‘Guardar’ (Save) button.

For deleting an existing unit, some steps must be undertaken:

1. Locate the unit to be deleted in the list of units.

2. Once it has been located, click on the delete symbol (trashbin). The system will ask for a confirmation.

3.3. Web service

In order to make the most of the automatic coding algorithm, a web service has been developed so that it can be used in the data collection, when it is carried out by electronic means or using computer assistance (CAPI, CAWI or CATI).

As the text is automatically coded just when it is captured, two are the main advantages:

1. It guarantees that every collected text has enough information to be automatically coded. If a text results not-coded, it is immediately known, and more information can be asked from the respondent.

2. It also allows to keep record of both the text and the resulting code, instead of keeping only the text, which will need to be coded in a later stage of the process.

In order to test the web service, a web interface simulating an electronic questionnaire was developed.
All the information needed is gathered in this interface: the classification family, the dictionary to be used, the coherence table, the auxiliary information, the classification level and the text to be coded.

In the test interface, the respondent has to make all the specifications itself. However, regarding a real questionnaire, the classification family, the dictionary and the coherence table would be parameters included in the programming of the collecting tool and absolutely transparent to the respondent. When the respondent introduces a text for answering a question, the program would collect the auxiliary information from the already given answers to previous questions in the questionnaire, and it would send all the parameters by the web services.

All this information would be treated following the automatic coding algorithm in the server, and a response would be sent back with the information included in the advanced exit file (see 3.2.1 Automatic coding).

When the collecting tool gets this response, it would have been programmed to do some specific actions, depending on the result of the algorithm. The test interface shows the following:

<table>
<thead>
<tr>
<th>Código</th>
<th>Literal</th>
<th>Salida</th>
</tr>
</thead>
<tbody>
<tr>
<td>622</td>
<td>VENDEDOR EXTINTORES</td>
<td></td>
</tr>
<tr>
<td>444</td>
<td>VENDEDOR DE LITRAMS</td>
<td></td>
</tr>
<tr>
<td>641</td>
<td>VENDEDOR DE REFRESCIOS EN GRANDA</td>
<td></td>
</tr>
<tr>
<td>549</td>
<td>VENDEDOR AL MISTRADOR DE COMIDAS PREPARADAS</td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>VENDEDOR DE HILAJERIA</td>
<td></td>
</tr>
<tr>
<td>263</td>
<td>VENDEDOR DE INMAJEDES</td>
<td></td>
</tr>
</tbody>
</table>

But a real questionnaire could act as follows:

- If the text has been coded or, in other words, the value of ‘Salida’ (Exit) is D1 or whatever starting by C (C1 to C8), the code of the best candidate is assigned, and the respondent goes ahead to the next question.

- If it is not (other values in ‘Salida’), the programme can pop up a message asking for a more descriptive text or show all the candidate codes so that the respondent selects one of them, for instance.
4. A real-life experiment: recoding the 2010 LFS quarterly samples to ISCO08 using the automatic coding tool

4.1. Stating the problem

Every time a new classification is implemented, there are some problems to be tackled. One of them is the link between statistical series in terms of the different versions of the classification. Therefore, the implementation of the CNO11 (Clasificación Nacional de Ocupaciones 2011, the national version of ISCO-08) will entail the need of linking the series in terms of ISCO88 national version, i.e. CNO94, and the series in CNO11. To this end, a period of double-coding is needed.

Since the LFS is using the CNO11 from January 1st, 2011, the INE department in charge of the LFS decided to double-code the sample corresponding to the four 2010 quarters. Considering that the Regulation adopting the CNO11 was passed on November 26th, 2010, and it was not publish in the official journal until December 17th, it was decided to re-code the already collected samples in terms of CNO11.

In December 2010 three quarterly samples were already available, so the first stage would consist in re-coding the main occupation for all the occupied people in these three samples. Later stages would deal with the fourth quarter, as well as with the second occupation for those having more than one job and the last occupation for the unemployed.

4.2. Re-coding process

Regarding the first three 2010 quarters, the number of occupied people whose main occupation needed to be re-coded was slightly over 65,000 per quarterly sample.

A first step in the re-coding process was based on the use of the correspondence table between CNO11 and CNO94. Those CNO94 codes that correspond to a single CNO11 code were directly transformed. Besides, the correspondences for the remaining CNO94 codes, which were related to more than one CNO11 code, were analyzed in order to define a table assigning a CNO11 code considering the value of some other information in the LFS file (economic activity, employment status and much less often, field of education). The results of this first step are shown in the following table:

<table>
<thead>
<tr>
<th>Occupied people</th>
<th>RECORDS RE-CODED</th>
<th>NON RE-CODED RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1to1</td>
<td>Using additional information</td>
</tr>
<tr>
<td>Q1</td>
<td>65.992</td>
<td>26.793</td>
</tr>
<tr>
<td>TOTAL</td>
<td>196.357</td>
<td>79.122</td>
</tr>
</tbody>
</table>
The next step consisted in the use of the **automatic coding tool** for the non-recoded records. Only those records including a text describing the occupation were suitable to be automatically coded, so those that did not include it were directly sent to the coding queue. A total of 72,758 texts were submitted to the automatic coding process. Prior to launching the process, a coherence table was constructed. The auxiliary information considered in that coherence table was the CNO94 code assigned in the LFS file. CNO11 codes with a correspondence with the original CN094 code were overweight. The value given to the algorithm parameters were the default ones. Regarding the results, and being the aim of the re-coding to have a double-coded file in order to link time series in terms of two different classification, it was deemed that only those texts re-coded with a CNO11 code fulfilling the correspondence table will be accepted. Those texts that were codified, but the resulting CNO11 code was not generating a pair CNO94-CNO11 in the correspondence table were sent to the coding queue. The following table shows a numeric summary of the results:

<table>
<thead>
<tr>
<th>TOTAL TEXTS</th>
<th>RE-CODED TEXTS, PAIR IN THE CORRESPONDENCE TABLE</th>
<th>NON RE-CODED OR RE-CODED BUT NOT FULFILLING THE CORRESPONDENCE TABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MATCHING WITH A DICTIONARY TEXT</td>
<td>IN OTHER STEPS</td>
</tr>
<tr>
<td>Q1</td>
<td>24,275</td>
<td>7,754</td>
</tr>
<tr>
<td>Q2</td>
<td>23,271</td>
<td>7,668</td>
</tr>
<tr>
<td>Q3</td>
<td>24,212</td>
<td>7,355</td>
</tr>
<tr>
<td>TOTAL</td>
<td>72,758</td>
<td>22,777</td>
</tr>
</tbody>
</table>

After this first attempt with the automatic coding tool, an analysis of the dictionary was carried out, adding new texts and synonyms. The most relevant problem was a systematic error affecting the texts originally classified in CNO94 with codes 601 and 602, which are the national equivalents to ISCO88 611 Market gardeners and crop growers. In CNO11 these codes correspond to 611 Crop growers and 612 Gardeners, horticultural and nursery growers. The error was due to a shortage of texts associated to 612 in the CNO11 dictionary. Besides, the texts to be coded underwent an improvement treatment, basically by amending some common misspellings. Once those improvements were implemented, the non re-coded texts (and those texts re-coded with a CNO11 not fulfilling the correspondence table) were submitted to a **second re-coding attempt using the automatic coding tool**.
The analysis of the 5,533 re-coded texts but not fulfilling the correspondence table is worth it, since they are identifying either records badly coded in the original file or errors in the automatic coding process or most likely, in the dictionary.

A summary of the most relevant figures of the whole automatic coding process (regarding both the first and the improved attempts) are included in the next table:

<table>
<thead>
<tr>
<th></th>
<th>RE-CODED TEXTS</th>
<th>NON RE-CODED TEXTS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PAIR IN THE CORRESPONDENCE TABLE</td>
<td>NOT Fulfilling the CORRESPONDENCE TABLE</td>
</tr>
<tr>
<td>TOTAL TEXTS</td>
<td>TOTAL</td>
<td>%</td>
</tr>
<tr>
<td>Q1</td>
<td>13.905</td>
<td>707</td>
</tr>
<tr>
<td>Q2</td>
<td>13.901</td>
<td>718</td>
</tr>
<tr>
<td>Q3</td>
<td>14.111</td>
<td>767</td>
</tr>
<tr>
<td>TOTAL</td>
<td>41.917</td>
<td>2.192</td>
</tr>
</tbody>
</table>

This result could have been highly improved by an adjustment of the algorithm parameters through a trial-and-error process, but the results were good enough to make this improvement unneeded.

The remaining 34,192 records were re-coded by means of a longitudinal study. When the CNO94 code and the economic activity were stable from a quarter to the next one, if it had been re-coded in one of the quarter samples, the CNO11 was assigned to the other one. Finally, a manual coding was conducted.

The fourth quarter sample as well as the second occupation and the last occupation for unemployed were re-coded following the same process. The availability of the automatic coding tool allowed re-coding all these variables for the four quarterly samples at the end of February.
4.3. Assessment of the results

As shown in the last table on the previous page, a total of 72,758 texts (none of them coded with a CNO-94 code with an univocal correspondence with CNO2009) underwent the automatic coding process, and 38,566 of them were coded. That means that a 53% of the texts were given a single code.

In order to assess these results properly, some relevant considerations have to be taking into account:

1. The results obtained strongly rely on how the variable is asked in the questionnaire, which affects the quality of the texts to be coded:

   - Interviewer vs. respondent

   The collected texts can be very different depending on who has provided the text. For instance, when re-coding the 2010 LFS, it was clearly noticed that the questionnaires were filled by survey interviewers, who were used to CNO-94. This caused two different effects on the texts. First, the wording of the texts was quite similar to the one used in the classification titles and it usually included abbreviations with a clear meaning for the interviewer, but not for other people (the interviewer itself was the one who had to code the text in a later stage). The second effect is due to the fact that the texts were to be coded in terms of CNO-94, not in terms of CNO-11. Therefore, the interviewers quite usually tried to use texts that perfectly define a CNO-94 category more than an actual occupation. As a consequence, these texts did not provide any additional information to that already provided by the CNO-94 code in the LFS file. All this affected the quality of the texts to be coded. When the text is provided directly by the respondents, all those effects are avoided: they are not used to any classification, so they do not copy the wording neither try to describe categories, and in addition, they do not use abbreviations (except perhaps some very common ones). On the contrary, texts provided by non-experts are more likely to include not enough information for coding at a given level of the classification (sub-specification).

   - Free text vs. list of categories + free text

   The texts to be coded can also be affected by the way they are collected in the questionnaire. In the 2001 Census in Spain, the questions about occupation and economic activity included a quite long list of categories. Every category was identified by an alphanumeric key, which could be transformed in a classification code using auxiliary information from the Census questionnaire. When the respondent was not completely happy with the selected key or even did not find the occupation in the list, he or she had to provide a free descriptive text. This type of question has three major advantages. Firstly, the category titles in the list influence the vocabulary and type of descriptions
used, so it allows leading the respondents towards similar descriptions to those in the dictionary. Besides, the reading of the list helps the respondents to understand what exactly they are asked about (for instance, confusion between occupation and economic activity variables was noticed in the responses to the last Census). Finally, the list makes the respondents aware of the type of categories in which their occupation has to be classified.

2. The availability of additional information and the use of a good coherence table are also highly relevant.

In the real-life experience of re-coding the 2010 LFS samples, the questionnaire included a lot of relevant information useful when determining the classification code. The LFS file, in addition to the CNO94 code and the occupation description, included the economic activity code, the employment status and the field of education. All this information was used in transforming some CNO94 codes into CNO11 codes. However, it was not used in the coherence table, which only considered CNO94 as auxiliary information. The construction of a coherence table including all the available information could have lead to a higher percentage of coded texts.

3. The results would have been much better by deeply analyzing the most common non-coded texts. This would have allowed detecting if these texts were not coded due to a wrong selection of candidates from the dictionary or to some specific algorithm parameters. In the first case, the dictionary could have been improved adding new texts to it (indeed, it was slightly improved between the 2 automatic coding attempts). The second problem could have being solved by adjusting the parameters to the final dictionary and the actual set of texts to be coded. This should be done by means of a ‘trial-and error’ process.

From the previous paragraphs it is clear that there was still room for improvement by making the most of all the auxiliary information through a new coherence table, adjusting the algorithm parameters and improving the dictionary, all three possibilities available in the automatic coding tool. However, although we were aware of this, we decided not to get involved in such improvements because there were two other advantages that make that effort not worth it:

- On the one hand, we were trying to code four consecutive samples. Since 5/6 of the sample remained from a quarter to the next one, we were confident that the longitudinal treatment would solve quite a lot of the non-coded texts.

- On the other hand, when the automatic coding process leads to a ‘non-coded’ result, it provides a set of candidate codes for the non-coded text (except when the text has too many words not included in the dictionary). This makes much easier the manual coding process of the remaining
texts, because in a vast majority of cases, the right code is one of the candidate ones, and quite often, the best or the second best one.

The first advantage is only valid in the very specific case explained, but the second one is associated to the automatic coding tool itself, and therefore, valid for any process.

Regarding these circumstances, the result of the process should not be evaluated just by considering the percentage of coded texts, which can be deemed quite high, and even more when there is such a big room for improvement. It should also take into account the simplification of the manual coding process due to the set of candidate codes assigned to every non-coded text.

The most meaningful and noteworthy result of the use of the automatic coding tool in the 2010 LFS re-coding is the fact that it allowed a single (and non full-time devoted to this task) person to re-code three different variables (main occupation, second occupation and last occupation of non-occupied workers) for the four 2010 quarterly samples, i.e. 330,856 records, in less than six weeks.

5. Availability of the automatic coding tool for other organizations and countries: Adaptation to other languages

The automatic coding tool has been developed using free software, and it will be available free of charge for any EU country or organization. To this end, a CD-ROM is available.

5.1. What is in the CD-ROM?

The CD-ROM includes all the software needed to deploy the tool in the application server. Also included in the CD-ROM are the three manuals already mentioned in this document:

- The installation manual is intended to make easy the installation of the software. It lists all the steps to follow when starting up the system. Instructions are specified for Tomcat 6. If it is going to be installed in a different application server, the staff in charge of IT systems will must adapt the instructions to the specific deployment application server.

- The maintenance manual includes all the items that can be configured: application deployment, databases, adaptation to other languages, file cleaning periodicity, location of the configuration files, as well as the messages and e-mail files, etc.
• The user’s manual is intended to provide the user with some guidelines on the good use of the system.

Dictionaries are not included since they are strongly linked to the classification and the type of texts to be coded, so it is not advisable to translate a Spanish dictionary to other languages. Besides, the way in which the synonyms are defined determines the synonym table to be specifically constructed for every different classification, and of course, every different language.

5.2. What additional works are required for the adaptation to other languages?

Three are the main tasks, apart from the software deployment, to undertake in order to use the tool in a different language:

5.2.1. Adaptation of the interface to the required language (screens, help files, etc.)

In order to make the tool understandable for the users in other language, all the screens, help files, messages and any other text used to give information to the user should be translated into the new language. Besides, the empty word table should also be translated and/or adapted to the new language. As explained above, the words included in that table are basically articles, conjunctions and prepositions, and the relevance of the semantic content of them can vary from a language to another, specially regarding prepositions.

The maintenance manual includes an specific chapter (4. Internationalization) devoted to explain the steps to follow for translating all this information. It can look difficult to understand, but it includes all the information needed and it should be easily understandable for the IT experts engaged in the tool deployment.

5.2.2. Preparation of the dictionary tables

There are some very important barriers that did not allow the use of a translated dictionary for coding in a different country:

• It has already made clear that dictionaries have to be drafted for a specific classification. Indeed, a change in the structure of categories could even make a dictionary inadequate to code the same variable, and this is what happens with different national versions of an international classification.
• The semantic content of a translated word is not always the same in the original and translated language. This is especially significant regarding the table of synonyms. The clearest example, but not the only one, is that of a polysemous word (a word with two or more meanings).

• It has been noticed that descriptions vary depending on who write down the descriptive text. For instance, when interviewers are involved, like in LFS, they tend to adapt the descriptions to their convenience, mainly using classification titles. When the text is given directly by respondents, they use a different kind of descriptions, strongly linked to the jargon of their specific industry or occupation. If these differences have been noticed at national level, it is clear that we can expect them to be even more serious when changing languages and national conditions.

Bearing this in mind, one of the most important works to adapt the tool to a new language is the construction of one or more dictionaries in the language of the country. To this end, a logical way of doing so would include the following steps:

• Collect real texts from the survey to be coded, or from other similar surveys (addressed to the same respondents, coding the variable at the same classification level, etc.). Usually these texts are already coded, since they are extracted from files corresponding to previous survey waves.

• Complete the coverage of the dictionary by adding ‘artificial’ texts. In this process special care has to be taken, since an excessive number of such texts could distort the weight of the words, and therefore, the tool performance.

• Analysis of the texts to check the coding and identify the most common texts and other that must be included in the dictionary as well as potential synonyms.

• Construct the dictionary tables with the table structures explain above, and load them following the instructions in the user’s manual.

• Check the coverage of the dictionary by means of a ‘try-and-error’ process. This ‘try-and-error’ process should also included the analysis of the algorithm values.

5.2.3. Adjustment of the algorithm parameters

The values given to the algorithm parameters were calculated to give good results when coding occupations with a specific dictionary in Spanish. These values could also provide satisfactory results when using other dictionaries, even in a different language. However, this cannot be guaranteed. Notwithstanding, it has already been
explained that the results can be improved by adjusting the algorithm values. This adjustment has to be carried out by means of a ‘trial-and-error’ process.

5.2.4. Checking the dictionary coverage and adjusting the algorithm parameters: Trial-and-error process

The ‘trial-and-error’ process should start by constructing a control set of texts. This control set should include a wide variety of preferably real texts, which were supposed to cover all the potential types of descriptions. If this is achieved, a really big control set is not needed.

In the first trial, the use of the default values in the algorithm is advisable. The control set of texts has to undergo the process and the results have to be analysed in order to identify:

1. Which coded texts are given the right code and which ones are not.

2. Regarding the non-coded texts, as well as the wrongly coded ones, it has to be determined if the error is due to the selection of candidate texts (generally associated to problems with dictionary coverage), or to the value given to a specific parameter in the algorithm.

Once the causes have been identified, they have to be corrected by modifying the dictionary or the algorithm parameters. Then a new test has to be taken, including not only the wrongly or non coded texts, but also the rightly coded ones, in order to confirm that the changes made are not affecting them.

5.3. Conclusions

The experience shows that once the dictionary, the algorithm and the way of collecting the texts have been adjusted; the benefits in terms of workload reduction are relevant, durable and exportable to similar surveys.

One of the most profitable efforts is that of defining a good table of synonyms. The use of synonyms has some advantages over the use of n-grams. Synonyms with the same root are well handled by n-grams (for instance ‘baker’, ‘bakery’ and ‘bakeries’), but synonyms with different roots are lost when using n-grams (e.g. ‘sale’, ‘trade’ and ‘commerce’). Besides, the use of n-grams can lead to wrongly associate words (for example, ‘translator’, ‘transmitter’, ‘transcription’ or ‘transport’).
6. Bibliography

- CNO-94, Clasificación Nacional de Ocupaciones, 1994, INE
- CNO-11, Clasificación Nacional de Ocupaciones, 2011, INE
- CNAE-09, Clasificación Nacional de Ocupaciones, 2009, INE.