AS OriginBox 1.6

User's Manual

June 2008

Table of Contents

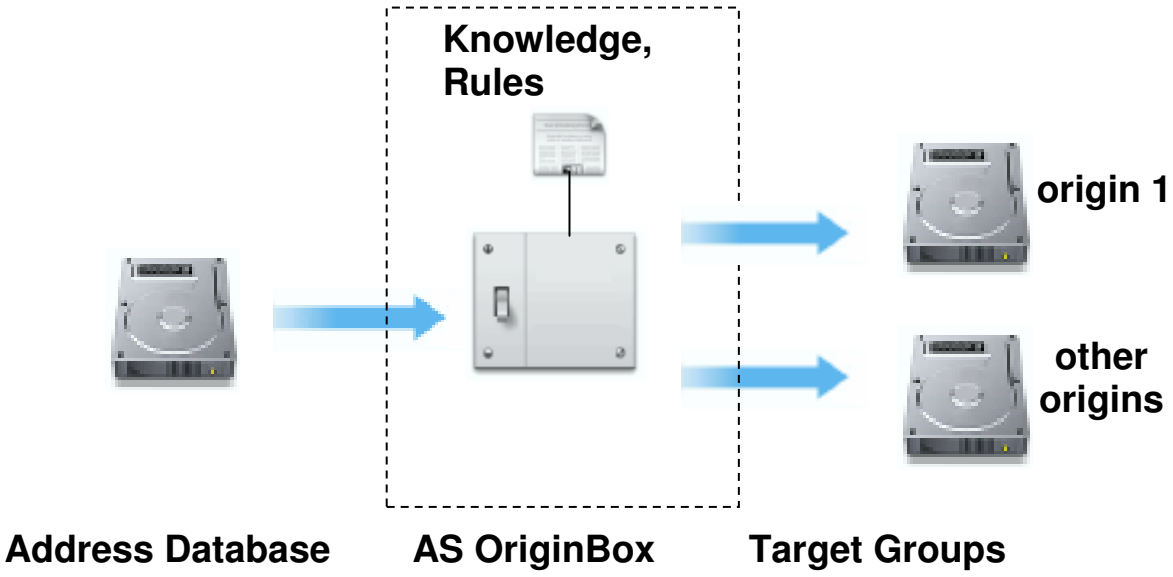
1 Introduction

The functions of the AS OriginBox library are designed to automatically determine the origin of a person´s name.

Based on a knowledge-based search concept the AS OriginBox analyzes the incoming name elements and finds out their possible origins. The specialty of this extraordinary determination is the knowledge- and rule based algorithm, which does not just consider special name endings, but knows where the origin of names is based.

Generally the AS OriginBox works in three main steps: First the entered name is transformed into a structured format, where a person´s first names and last names are separated from each other. Then the first names and last names are checked separately and as a result they receive all possible origin flags by retrieving this information from the knowledge base. This process also takes into account that various names are known in more than one culture or language and that sometimes name components have different meanings dependent on their origins. Finally the name elements (especially first names and last names) are combined to be verified in a certain rule base. The result of the whole process will be a clear code describing the origin of an individual´s complete name.

A typical example of the application of the AS OriginBox is its support with marketing actions by telephone companies, e.g. to offer special telephone tariffs to foreign fellow citizens for calls into their native countries. Before starting such a marketing campaign the appropriate target groups must be identified first. The example below separates origin 1 (e.g.Turkish names) from all other origins.



The AS OriginBox functions can be used in online as also in batch-processing environments.

At present the following language or cultural environments are implemented:

- ASIAN
- AFRICAN
- SPANISH
- EASTERN EUROPEAN
- SCANDINAVIAN
- ARABIAN
- BENELUX
- HEBREW
- INDIAN
- INDONESIAN
- ITALIAN
- GERMAN
- FRENCH
- BRITISH
- TURKISH
- GREEK
- RUSSIAN
- YUGOSLAVIAN
- POLISH
- CZECH
- HUNGARIAN
- ROMANIAN
- BULGARIAN
- ALBANIAN

2 AS OriginBox Components

The AS OriginBox software package consists of

- a CD-ROM
- the manual.

The CD-ROM usually contains the following files in the directory

<CD-DRIVE>:\ASBaseTools\OB\<operating system>:

where <operating system> is one of the following: hp-ux, linux, solaris, win32, aix.

Depending on the licensed operating system platform of the OriginBox the contents of the CD may differ. At present the OriginBox library is available for the following operating systems: Windows98/ME/NT/XP, Linux, Sun Solaris, HP-UX, IBM AIX.

- the OriginBox main library designed for:
  Windows: as_originlib.dll
  Sun Solaris: liborigin.so / liborigin64.so
  HP-UX: liborigin.sl
  Linux: liborigin.so
  IBM AIX: liborigin.a

- origintab.dat:
  Knowledge table including international names, prefixes, origins, etc.

- asrules.dat
  Rule table for the identification of name structures and combinations of origins of first and last names.

- Example executable program, which is an easy demo-application using the AS OriginBox routines from the OriginBox library.
  as_origin.exe for Windows platforms
  as_origin for all other platforms

  Functionality and implementation should be easier to understand after considering the source “as_origin.c”.

- as_origin.c
  Source code of above executable.
  If you use a Unix/Linux platform you will additionally find a makefile (makefile_asorigin) to create the as_origin executable.

- as_structures.h
  Header file for C-Programs which should be integrated for using the AS OriginBox. An example of its usage is shown in as_origin.c

  In this header file also some structure definitions for the AS ConvertBox, AS MatchBox and the AS PostBox are given, which can be ignored if the mentioned products are not applied.

- ob_testdata.txt
  Example of an exported address database for use with the demo-application “as_origin”.

3 Installation and First Steps

The installation of the AS OriginBox is quite easy. Just transfer the contents of the <CD-DRIVE>:\ASBaseTools\OB\<operating system> directory on the CD-ROM into an empty directory on the hard disk of your system. To ensure the **as_originlib** or **liborigin.so/liborigin.sl/liborigin.a** library will be found through the operating system by calling it from any directory, either set a "system path" onto this directory or move the library into a directory already having a "path-definition". For this task the support of your system administrator may be needed.

For checking whether everything works properly now start the enclosed demo-application "as_origin".

The following steps should be taken as described below:

1. Start the program as_origin in a DOS-Box (Windows) or terminal window (Unix/Linux). First some information about the status of the source code versions, which have been used to generate the OriginBox library, will be shown. Then a small menu will appear.

```
$Id: as_convertlib.c,v 1.34 2003/08/21 09:36:03 Standard Exp $
$Id: as_common.c,v 1.32 2003/08/15 10:45:00 Standard Exp $
$Id: as_interpret_name.c,v 1.34 2003/08/15 10:44:41 Standard Exp $
$Id: as_normalize_name.c,v 1.35 2003/08/19 16:01:16 Standard Exp $
$Id: as_wb.c,v 1.20 2003/08/18 15:41:30 Standard Exp $

Make your choice
[ 1] = call origin_name in batch mode
[ 2] = call origin_name in online mode
[ 3] = end
==➔
```

2. Now choose [1]: call origin_name in batch mode.

3. Enter "ob_testdata.txt" for the name of the address source file.

4. Enter the name of the result file i.e. "out.txt"

5. The next question

```
countrycode: (nl/de/ch/tr/ru/nn):
```

asks the countrycode for the input data with the following abbreviations and meanings

de	German
ch	Swiss
nl	Dutch
tr	Turkish
ru	Russian
nn	not named / all others

The first impression about this question may certainly lead to confusion, as it is in fact the origin, which is to be determined. This information is in fact used - only in cases of doubt, though – as a kind of pre-definition in order to be able to set a certain determination of the origin according to the corresponding context (address data base). This is illustrated by the following example. The name “Robert Martin” can originally belong e.g. to France, England and Germany. In these cases the parameter “countrycode” serves as the additional information about the expected result. In German address data bases this name will certainly be identified as a name of German origin.

Now the program will process all records from “ob_testdata.txt” and write the results to “out.txt”

If all the steps above are successfully executed the AS OriginBox will be installed correctly and will be ready for further usage.

4 Determination of Name Origins with the AS OriginBox

In order to guarantee the greatest possible flexibility of the AS OriginBox its functions can be called by using direct (symbolic) entry points within the library. By this way, a function call can be made both within a graphic interface and on command line level. Furthermore, the function call is the same for every platform.

4.1 Description of Input and Output Structures

The name elements that have to be checked and the accompanying parameters are transferred as encapsulated structures. These structures must first be completed with corresponding data or are sorted out after the function call.

4.1.1 Name Input Structure

The structure for passing a name into the OriginBox contains several fields of name components. It is possible to enter pre-formatted names here, or in case they are not pre-formatted, use the name-field for the complete name input. The sense of the pre-formatted input structure is the following: If a customer already possesses these details (and this information is correct), they will be taken into consideration during the AS OriginBox processing.

The following picture shows the structure definition for the input name of the OriginBox as determined in the header file “as_structures.h”. Therefore, all definitions are made in a C-like manner. Please note that all character strings that have to be passed in this structure are zero terminated. That means if you have a string of the length of 50 characters, you will have to allocate 50+1 bytes for this string. Furthermore, it is recommended to empty an input structure (setting it completely to 0x00 on all fields) before filling it with the data that have to be computed. Otherwise there is always a higher risk of input data from the last call disturbing the recent call or of previous contents not being completely overwritten. (The name of the structure is CONVERT_IN because we use the same structure as for the AS ConvertBox.)

```
struct CONVERT_IN
{
    unsigned long id;
    unsigned char gender[1+1];          /* gender 'm' = male, 'f' = female; unknown else */
    unsigned char titulation[50+1];     /* titulation e.g. Herr */
    unsigned char title[50+1];          /* title e.g. Doktor */
    unsigned char initial[50+1];        /* initials e.g. R.W. */
    unsigned char firstname[100+1];     /* firstname */
    unsigned char name_prefix[50+1];    /* is prefix e.g. von */
    unsigned char name[250+1];          /* companyname or lastname */
    unsigned char name_suffix[50+1];    /* is postfix e.g. junior */
    unsigned char name_addition[50+1];  /* additional name elements e.g. department 3 */
    unsigned char profession[50+1];
};
```

The following table describes the elements of the CONVERT_IN structure and what kind of input data is expected in each field:

Field	Type	Description
id	unsigned long	ID of a record. The contents of this field are passed through the function and directly to the output and is not obligatory. Nevertheless, it is always useful to specify the id because it can be helpful especially in batch processing tasks.
Gender	unsigned char[1+1]	If you have gender information about an individual name you can pass this information here. The ConvertBox will check this information and use it for determination of the gender in the output structure. The following gender information can be specified: ”m”: male ”f”: female unknkown else
titulation	unsigned char[50+1]	This field describes the titulation/personal address of the name like “Herr”, Frau”, “De Heer”, etc.
title	unsigned char[50+1]	Put in the title field, e.g “Doktor”, “Prof.”, “Ir”
initial	unsigned char[50+1]	Describes the initials of a name, like “ECJ” or “E.C.J.”
firstname	unsigned char[100+1]	Input field for first name
name_prefix	unsigned char[50+1]	Prefix-field, e.g. “van”, “de”, etc.
Name	unsigned char[250+1]	This field is used for : 1. The last name of an individual person 2. The complete name of an organization 3. The complete name of an individual person, if the names are not pre-formatted
name_suffix	unsigned char[50+1]	Suffix-field, e.g. “junior”, “sr.”, etc.
name_addition	unsigned char[50+1]	Additional information of a name, e.g. “department”
profession	unsigned char[50+1]	Input field for a person´s profession, such as “Frisör”, “Kapper”, etc.

4.1.2 Origin Output Structure

This structure describes the exported structure and contains all possible result elements.

```
Struct ORIGIN_OUT
{
    unsigned int id;
    unsigned char status[10+1];
    unsigned char origin[32+1];
    unsigned char origin_firstname[32+1];
    unsigned char origin_lastname[32+1];
    unsigned char origin_firstname_detail[65+1];
    unsigned char origin_lastname_detail[65+1];
    unsigned char origin_message[1000+1];
    unsigned char pattern_out[100+1];
};
```

The following table describes the elements of the ORIGIN_OUT structure and what kind of output data is returned from the OriginBox in each field:

Field	Type	Description
id	unsigned long	ID of a record as specified in the CONVERT_IN structure (see above)
status	unsigned char[10+1]	Result status of the OriginBox function call. The status will be explained in detail in chapter ***4.2.5 Using the Status Code for the Evaluation of the Results***
origin	unsigned char[32+1]	"Bit-coded" field for the complete name where the origin bits are set to "1". This result field will be explained in detail in chapter ***4.2.6 Description of the Origin Result Fields***
origin_firstname	unsigned char[32+1]	see origin
origin_lastname	unsigned char[32+1]	see origin
origin_firstname_detail	unsigned char[65+1]	Detailed information concerning the first two first names in case of more than one first name in the input. (See also description for field origin)
origin_lastname_detail	unsigned char[65+1]	Detailed information for the first two last names in case of more than one last name in the input. (See also description for field origin)
origin_message	unsigned char[1000+1]	Message text that describes the message in plain text
pattern_out	unsigned char[100+1]	Result of the identification in the rules table. This result field will be explained in detail in chapter ***4.2.7 Description of the pattern_out Result Field***

4.1.3 Configuration Structure

The third necessary structure is the configuration for the AS OriginBox. This structure contains the definition of knowledge and rules base as also the parameter for the countrycode and the used codepage. The structure definition in C-Syntax is the following:

```
struct OB_CONFIGURATION
{
    unsigned char knowledgetable[255+1];
    unsigned char ruletable[255+1];
    unsigned int  codepage;
    unsigned char countrycode[10+1];
};
```

The table below gives an overview about the elements of the OB_CONFIGURATION structure and the meaning of each configuration parameter. Below this overview some parameters are described in detail.

Configuration parameter overview:

Field	Type	Description
knowledgetable	unsigned char[255+1]	Name and directory of the AS knowledge table. Usually the name of the knowledge table for the AS OriginBox is “origintab.dat”. If the knowledge table is located in the directory where the OriginBox is started, no path information is necessary. Otherwise it is always appropriate to specify the complete path, i.e. “C:\as\release14\origintab.dat”. This parameter is obligatory.
Ruletable	unsigned char[255+1]	Name and directory of the AS rule table. Usually the name of the knowledge table is “asrules.dat”. This parameter is obligatory. For the use of a path definition please refer to the parameter “knowledgtable”.
Codepage	unsigned int	Specifies the codepage of the input data where the following settings are possible at the moment: 0: Windows/Ansi 1: DOS/OEM Other codepages are not supported at the moment but if necessary they can be implemented within short time.
Countrycode	unsigned char[10+1]	The countrycode for the input data has to be set. There are the following possible values at the moment: “de”: German/Germany “nl”: Dutch/Netherland/BeNeLux “ch”: Swiss/Switzerland “tr”: Turkish/Turkey “ru”: Russian/Russia “nn”: Not Named/unknown

As mentioned above some of the parameters have to be explained in a more detailed way to provide a complete comprehension how the OriginBox works and what influence on the result can be expected for each parameter.

4.1.3.1 Configuration Parameters

4.1.3.1.1 Knowledge Table

The knowledge table contains information about all kinds of names and name elements like first names, last names, prefixes, titles etc. Furthermore, there are also stored details regarding geographical items, professions and company names. This information is used while executing the AS OriginBox function(s). Once an OriginBox function has been called and the dynamic library has been loaded into the main memory the knowledge table is also loaded into the main memory of your computer with the first function call. It will stay here until the library is released from the memory. An internal flag is set indicating that the table has been loaded and that re-loading is not necessary.

Usually the original name “origintab.dat” of the knowledge table should not be changed so that this parameter should be almost fixed. However, it is possible to determine another name for the knowledge table, e.g. in case you want to test a new version of the knowledge vs. an older one. The following functionality has to be known, though: As described above the knowledge table is loaded into the memory with the first function call to the OriginBox. If you want the knowledge table to be changed this **cannot** be done by just changing the parameter “knowledgetable” and calling the function again. First the dynamic library has to

be released from the memory and afterwards be loaded again with the new value of the name of the knowledgetable. Please note that this procedure is very time-consuming and should not be carried out in normal production environments.

4.1.3.1.2 Rule Table

The rule table contains rules how to define the origin of names and name elements after being qualified by means of the knowledge table. The rule table functions in the same way as the knowledge table in the main memory (description regarding the knowledge table can be applied accordingly for the rule table as far as main memory is concerned). After being loaded into the memory it will stay here until the dynamic library is released from the memory.

4.1.3.1.3 Codepage

The determination of the codepage is necessary because the OriginBox internally always uses the same character encoding. This method enables the user to apply the same knowledge and rule table on every platform with every character set. On processing data with the OriginBox the input data are first encoded from the origin codepage to the internal character set and afterwards decoded to the codepage that is specified.

4.1.3.1.4 Countrycode

The first impression about this parameter seems to be paradox, because it is the origin that is to be determined. This information is actually used, however, exclusively in cases of doubt in order to be able to set a certain determination of the origin according to the corresponding context (address data base), which will be illustrated by the following example. The name “Robert Martin” can originally belong e.g. to France, England and Germany. In such cases the parameter “countrycode” serves as additional information about the expected result. In German address data bases this name will certainly be identified as a name of German origin.The current version supports the countrycodes: de, ch, nl, tr, ru, nn.

4.2 Implementation of the OriginBox

The origin of name is qualified by the AS OriginBox with a function **as_origin_name**, which can be called up both online and in batch processing. Therefore, first the input structure CB_CONVERT_IN is filled in with the relevant name elements and then the configuration structure OB_CONFIGURATION with the parameter set. Afterwards the call the function is called. A C example of the call is shown below. A complete integration example of the function call can be found in the C source code “as_origin.c”, which is part of the delivery on the CDROM.

4.2.1 Declaration

The function declaration for structuring names is shown below in C-Syntax.

```
Int as_origin_name (struct CONVERT_IN *name_in,
                    struct ORIGIN_OUT *origin_out,
                    struct OB_CONFIGURATION *cfg);
```

4.2.2 Return Code

After the execution of the function as_origin_name the output structure is filled in and the return code is computed.

Return Code	Description
= 0	Successful operation
!= 0	A fault has been found.

4.2.3 Parameter Description

Please refer to chapter ***4.1 Description of Input and Output Structures*** for the description of the structures used as parameters.

Parameter	Type	Mode	Description
name_in	*CONVERT_IN	Input	Pointer on the input structure with the name to be structured
name_out	*ORIGIN_OUT	Output	Pointer on the output structure with the conversion results of the name(s)
cfg	*OB_CONFIGURATION	Input	Pointer on the configuration structure that has to be used for the structuring process.

Please note that the parameters are all pointers to the defined structures. Please take care about the memory allocation before executing the call, otherwise a memory exception will occur.

4.2.4 C Example

```
...

    unsigned char buffer[250+1];
    int rc;

    struct CONVERT_IN name_in;
    struct ORIGIN_OUT origin_out;
    struct OB_CONFIGURATION cfg;

    unsigned char command[10];
    unsigned char countrycode[10];

    memset(&cfg, 0, sizeof(struct OB_CONFIGURATION));
    cfg.codepage = 0;
    strcpy(cfg.knowledgetable, "origintab.dat");
    strcpy(cfg.ruletable, "asrules.dat");

    strcpy(countrycode, "x");
    while (strstr("nl de ch nn", countrycode) == NULL)
    {
        printf("countrycode: (nl/de/ch): ");
        gets(countrycode);
    }
    strcpy(cfg.countrycode, countrycode);


    strcpy(command, « y ») ;
    while (tolower(command[0]) == 'y')
```

```
    {
        memset(&name_in, 0, sizeof(struct CONVERT_IN));
        memset(&origin_out, 0, sizeof(struct ORIGIN_OUT));

        printf("First Name: ");
        memset(buffer, 0, 250);
        gets(buffer);
        strcpy(name_in.firstname, buffer);

        printf("Last Name : ");
        memset(buffer, 0, 250);
        gets(buffer);
        strcpy(name_in.name, buffer);

        rc = as_origin_name(&name_in, &origin_out, &cfg);
        if (rc < 0)
        {
            printf("error in as_origin_name: return code: %d\n", rc);
        }

        printf("Returncode.......: [%d]\n", rc);
        printf("Status...........: [%s]\n", origin_out.status);
        printf("Origin...........: [%s]\n", origin_out.origin);
        printf("Origin Firstname.: [%s]\n", origin_out.origin_firstname);
        printf("Origin Lastname..: [%s]\n", origin_out.origin_lastname);
        printf("Firstname Details: [%s]\n", origin_out.origin_firstname_detail);
        printf("Lastname Details.: [%s]\n", origin_out.origin_lastname_detail);
        printf("Origin Message...: [%s]\n", origin_out.origin_message);
        printf("Pattern Out......: [%s]\n", origin_out.pattern_out);

        printf("\nnew conversion? (y/n) : « ) ;
        gets(command) ;
    }

...
```

4.2.5 Using the Status Code for the Evaluation of the Results

The Status Code of the output structure contains information on the internal evaluation of the names that have been processed. It can (and should) be used to decide whether a (re)structured name can be used as an either sure or unsure result. Some additional information about the quality of the result is given below the table. The status field has the following format:

Position	Validation Area	Description
1	I, O, U	**S**: The function call has been **s**uccessful (this does not mean that an origin could be determined) **W**: The function call could be processed, but the result is not reliable. (**w**arning) **E**: An internal error has occurred. No result is computed.
2	0..7	Global status of the origin determination. If the value is > 0, then one or more possible origins have been found. The lower the number the less complex the algorithm for finding this result (1=usually best quality) has been used. In case of a determination of a single possible origin, the value at position 8 is used to specify the reliability of the result. **0**: No rule for the given first name / last name combination could be found. A usual reason is that first name and last name have different origins but nothing in common, e.g. first name = British and Benelux; last name = German and French **1**: The complete name has a certain origin because a rule has been found. This is the best possible range concerning the quality of the determination. **2**: The complete name has a certain origin but only by explicitly considering the expected countrycode. Otherwise the complete name is ambiguous. For example, “Peter Martin” has more than one possible origins. Taking into consideration that a German name is expected (“countrycode” = German) the process will result into a German origin. **3**: Both, first name and last name have a certain origin each, but not in common (Example: first name is only German; last name is only French) **4**: First name and last name have more than one certain origin in common, and they all differ from the specified countrycode **5**: Either the first name or the last name is unknown, but the part of the name that is known is of one specific origin. This status can also be returned if the first name or the last name is missing in the input. **6**: Either the first name or the last name is unknown and the part of the name that is known is of more than one specific origin.
3	0..1	**0**: The first name (none of the words) has not been found in the knowledge with one or more possible origins. **1**: The first name has been found in the knowledge with one or more possible origins.
4	0..1	**0**: The last name (none of the words) has not been found in the knowledge with one or more possible origins. **1**: The last name has been found in the knowledge with one or more possible origins.
5	M, F	**M**: The name indicates a male person **F**: The name indicates a female person

6	0..3	**0**: Nothing of the first name has been used to determine the (possible) complete origin. Either the determination has been executed by a method not using the first names at all (see position no 2: Status 5 or 6) or (both) first names have not been found in the knowledge base. **1**: Only the first part of the firs name has been used to determine the (possible) complete origin. **2**: Only the second part of the firstname has been used to determine the (possible) complete origin. **3**: Both parts of the first name have been used to determine the (possible) complete origin.
7	0..3	**0**: Nothing of the last name has been used to determine the (possible) complete origin. Either the determination has been executed by a method not using the last names at all (see position no 2: Status 5 or 6) or (both) last names have not been found in the knowledge base. **1**: Only the first part of the last name has been used to determine the (possible) complete origin. **2**: Only the second part of the last name has been used to determine the (possible) complete origin. **3**: Both parts of the last name have been used to determine the (possible) complete origin.
8	0..4	If a certain origin is found, this value will specify the reliability of the result. **0**: Not a single origin! **1**: Best possible result: unique origin first name and origin last name **2**: Either first name or last name is unknown and the other part is of one single origin and not in the expected country **3**: Either first name or last name is unknown and the other part is of one single origin which is also expected in the “countrycode” **4**: Either first name or last name is unknown and the other part is of more than one single origin – including the expected origin.
9	0	Not yet used
10	0..1	This field indicates whether passed first name and last name components have internally been exchanged while being processed **0**: Not exchanged **1**: First name and last name components have been exchanged

Additional explanations about the quality of the identification:

Of course, the quality and the ratio of correctly identified origins depend to a great extent on the quality of the input. If you already have some kind of structured data or your input data is more or less “homogeneous” this will normally lead to better and more valid results than completely unstructured or varied, “heterogeneous” input data. With a data source consisting of various kinds of data it is obviously harder to find a good configuration matching all conditions.

Nevertheless, the following general recommendations can be made:

- The determination of origins is based on the first name, last name (including maiden name) and name prefix. Thus with well-structured data the best results will be achieved in the process by filling in this information into the corresponding fields of the input structure.

- Leave out all unnecessary information like titles, professions or name additions, as they may probably disturb the internal structuring process of identifying first names and last names. The input structure CONVERT_IN supports those fields, though. However, this is only for compatibility reasons with the AS ConvertBox.

4.2.6 Description of the Origin Result Fields

The origin result fields show the origins of the complete name, of the first name and the last name in a bit-coded manner. The field is represented as a char array of 16 bytes plus one string termination byte. Each position is coded to a certain origin, whereas a '0' at a position means that an origin is not set and a '1' stands for the origin of the according country being active. Therefore, a typical origin result field could, for example, look like this "0000100100110000000000000000000000". The positions are encoded as follows:

Position	Origin
1	Asian
2	African
3	Spanish
4	Eastern Europe
5	Scandinavian
6	Arabian
7	Benelux
8	Hebrew/Jewish
9	Indian
10	Indonesian
11	Italian
12	German
13	French
14	British
15	Turkish
16	Greek
17	Russian
18	Yugoslavian
19	Polish
20	Czech
21	Hungarian
22	Romanian
23	Bulgarian
24	Albanian
25	Not yet defined
26	Not yet defined
27	Not yet defined
28	Not yet defined
29	Not yet defined
30	Not yet defined
31	Not yet defined
32	Not yet defined

Example:
An origin field with the contents "00000000000**10**1000000000000000000", where Byte No. 12 and 14 are set to '1', represents the origins "German" and "British".

Please note that the field is not really bit-coded – it just looks like this. Instead it is byte-coded where each byte is a char ‘0’ or ‘1’.

The 4th position needs some additional explanation: In previous versions of the AS OriginBox there used to be only one common flag concerning all Eastern Europe countries. This means that e.g. Russian as also Polish, Hungarian (or any other Eastern Europe) origins were identified by this flag. With Release 1.4.1 of the AS OriginBox the Eastern Europe origins have been returned more detailed in flags 17 to 24. Nevertheless, the 4th position is also still set in cases an Eastern Europe origin is identified, whereas a more detailed determination is not possible at that state.

4.2.7 Description of the pattern_out Result Field

Usually the pattern_out result field contains the decimal value of the complete origin determination for a person´s full name. As the pattern_out is a character array the result is given as a zero terminated string. The value is set according to the position number of a Bit (Byte) set in the origin result field. Hence, when a name has been identified as German the contents of the pattern_out field will be “12”.

Sometimes a complete identification cannot be made because the first name is of one certain meaning and the last name is of another certain meaning, i.e. the first name is exclusively German and the last name is exclusively Italian. In these cases the pattern_out will be completed with the decimal number for fhe first name and the decimal number for the last name separated by a slash ‘/’ (12/11). In other cases first name and last name combinations are possible or might occur in two or more origins (cultural environments). Then the pattern_out is filled with both possible origins separated by a dash ‘-‘. The output “7-11” means that the complete name is possible in Benelux as well as in Italy.

Please note that the pattern_out field is to be considered more as a suggestion than a determined identification. In combination with the status field you should also take into account the reliability of this result. In some situations you may e.g. want to identify the Turkish women with a typically German last name (due to marriage). For selections like these the origin flags of first name and last name as also the status information have to be selected separately.

5 Appendix