DaXtra Competency

Description of profile competency element

APAC Enquiries: +852 3695 5133
APAC Email: asia@daxtra.com
APAC Support: +852 3695 5133
APAC Email: support@daxtra.com

APAC Head Office:
Unit 1205 OfficePlus,
93-103 Wing Lok Street,
Sheung Wan, Hong Kong

EMEA Enquiries: +44 20 7801 6323
EMEA Email: busdev@daxtra.com
EMEA Support: +44 131 653 1260
EMEA Email: support@daxtra.com

EMEA Head Office:
Top Floor South, Harbour Point,
Newhailes Road, Musselburgh,
EH21 6QD, United Kingdom

USA Enquiries: +1 804 767-1351
USA Email: usa@daxtra.com
USA Support: +1 804 767-1351
USA Email: helpdesk@daxtra.com

USA Head Office:
3310 W. Clay Street,
Richmond,
VA 23230, United States

Table of Contents

CV Parsing: Output Profiles

When a CV/Resume is parsed by DaXtra, the result is an output profile, in either XML or JSON; this profile contains all the information we have extracted from the CV. There are many features in the profile, including personal details, work histories, education histories and so on. The extracted data is normalised and placed into structured elements.

DaXtra has an extensive skills taxonomy (DaXtra Skills Taxonomy or DST) combined with a powerful parsing engine, and we extract many skills, job titles and qualifications from CVs; these are placed in the **Competency** elements of the profile. In this document we will focus on the contents of **Competency** element, and how you can use the extracted information to best effect.

Below are two examples of default outputs, in XML and JSON, of a **Competency** element for the job title 'Registered Nurse', for the CV fragment:

> 6/6/2010 – 09/12/2013, Registered Nurse, Intensive Care Unit

In XML an output **Competency** element looks like this:

```
<Competency oids='22_1 30_4_1' description='Held Position' auth='no' name='Registered Nurse'>
	<TaxonomyId id='medicine' idOwner='DAXTRA ' description='medicine'/>
	<CompetencyEvidence name='Extent of Experience' typeDescription='Extent of Experience'
type='MONTH' lastUsed='2013'>
		<NumericValue description='MONTH'>92</NumericValue>
	</CompetencyEvidence>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>Registered Nurse</StringValue>
	</CompetencyEvidence>
	<CompetencyWeight type='skillLevel'>
		<NumericValue minValue='0' maxValue='100'>39</NumericValue>
	</CompetencyWeight>
	<CompetencyWeight type='skillProficiency'>
		<StringValue minValue='BASIC' maxValue='EXCELLENT'>BASIC</StringValue>
	</CompetencyWeight>
</Competency>
```

and in JSON:

```
"Competency" : [
	{
		"skillLevel" : 39,
		"skillName" : "Registered Nurse",
		"description" : "Held Position",
		"auth" : false,
		"skillUsed" : {
			"value" : 92,
			"type" : "Months"
		},
		"TaxonomyId" : {
			"idOwner" : "DAXTRA",
			"description" : "medicine"
		},
		"lastUsed" : "2013",
		"skillProficiency" : "BASIC",
		"skillAliasArray" : [
			"Registered Nurse"
		]
	},
```

The **Competency** element will be explained in detail in 'The Competency Element' section.

Default skills vs Customer skills

During CV/Resume extraction DaXtra parser will identify skills, qualifications and job titles using both our DaXtra Skills Taxonomy (DST) and the linguistic context. When a skill is identified through the DST it is marked as 'authorised' and is normalised to its main (canonical) form, for instance, B2B => 'Business to Business' or 'Exchange' => 'Microsoft Exchange Server'.

Default output

Normalised (canonical) Competency names

Skills and qualifications in DaXtra-generated profiles are represented as **Competency** elements. If such a **Competency** was matched though the DaXtra Skills Taxonomy (DST), the skill is marked as 'authorised': in XML the **auth** attribute is set to 'yes' (or 'true' in JSON) and the XML **name** attribute (**skillName** in JSON) is normalised to its main (canonical) form. For example, if in a resume we encounter a string "B2B" in a suitable context this will create a **Competency** node with the **name** 'Business to Business' and the **auth** attribute set to 'yes'; this is triggered because the DST contains 'B2B' alongside other aliases for the 'Business to Business' skill.

The actual string "B2B" which was matched by DaXtra parser in a resume will then be put into an **Alias** sub-element of the **Competency** element. The canonical form 'Business to Business' will be put in the first **Alias** subnode as well as in the **Competency** node **name** attribute.

We have to preserve the canonical form in the first **Alias** sub-element because the **Competency** element **name** attribute might later be remapped to a customer's preferred form of a skill, as we will show in detail in the section **Custom output** below.

Here is an example for the string "B2B" which DaXtra identified as a skill and normalised to 'Business to Business' (only the relevant section is shown):

XML form

```
<Competency oids='6 ' description='Skill' auth='yes' name='Business to Business'>
        <CompetencyEvidence typeDescription='Alias' name='Alias'>
                <StringValue>Business to Business</StringValue>
        </CompetencyEvidence>
        <CompetencyEvidence typeDescription='Alias' name='Alias'>
                <StringValue>B2B</StringValue>
        </CompetencyEvidence>
```

JSON form

```
"Competency" : [
        {
                "auth" : true,
                "skillName" : "Business to Business",
                "description" : "Skill",
                "skillAliasArray" : [
                        "Business to Business",
                        "B2B",
                ]
        },
```

As noted above DaXtra parser can also identify skills without relying on the DST and instead just using the linguistic context. In this case the **name** attribute of the **Competency** node will be set to the actual string which was matched in the text and the **auth** attribute in XML is set to 'no' ('false' in JSON) or omitted.

Custom output

If a customer needs to match their own list of skills and/or skill codes, then after the initial default skill extraction user-specific skill mapping can be applied.

Customers can develop and maintain their own skills taxonomy (called a Master Skills File or MSF) using tools provided by DaXtra, or DaXtra can create an MSF from the customer's skills list. The MSF is used to produce custom profiles containing Competencies which reflect your skills list, no matter what your skills are named (including names in languages other than English), and no matter what phrases you want to map to these skills, again including non-English phrases.

If the DaXtra parser finds a match for a skill (normalised or non-normalised) in your Master Skills File, the **Competency name** attribute will now be set to your skill name, and there will also be an **id** attribute which will match to your database code.

The following transformations will be applied to **Competency** element:

- the **Competency** element **name** attribute will be changed to your form of a skill name (this is why the normalised skill name was preserved in the first **Alias** subnode);

- there will also be a **CompetencyId** sub-element with an **id** attribute which will match to your database code, (through these ids skills can be automatically populated in your database).

Sample Master Skills File

See below some sample nodes from a short Master Skills File:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<AllSkills>
	<entry>
		<category>skill</category>
		<type>Skill</type>
		<code>240</code>
		<descr>Biz2Biz</descr>
		<parent>MySkills</parent>
		<auth type="match">Business to Business</auth>
	</entry>
	<entry>
		<category>skill</category>
		<type>Skill</type>
		<code>123</code>
		<descr>Digital Marketing</descr>
		<parent>MySkills</parent>
		<alias locale="en" cs="n" ambig="n">Digital</alias>
	</entry>
</AllSkills>
```

This file contains two skills ('Biz2Biz' and 'Digital Marketing'), with codes and mappings.

Mapping from DaXtra extracted skills (auth)

When mapping to a customer **code**/**descr** from DaXtra extracted skills, you need to specify in the MSF a mapping using an **auth** element. The **auth** element means, "if a skill with that **auth** name has been identified, remap it to the content of the MSF **code** and **descr**."

For example, if your MSF contains a skill with a **descr** (name) 'Biz2Biz', with the **code** '240', (as in the example above) and the DaXtra system identifies 'Business to Business', we would now use this to produce the following output (only relevant fields shown):

XML form

```
<Competency oids='6 ' description='Skill' auth='yes' name='Biz2Biz'>
        <CompetencyId id='240' idOwner='me'/>
                <CompetencyEvidence typeDescription='Alias' name='Alias'>
                <StringValue>Business to Business</StringValue>
        </CompetencyEvidence>
                <CompetencyEvidence typeDescription='Alias' name='Alias'>
                <StringValue>B2B</StringValue>
        </CompetencyEvidence>
```

JSON form

```
"Competency" : [
        {
                "auth" : true,
                "CompetencyId" : {
                        idOwner" : "me",
                        "id" : "240"
                },
                "skillName" : "Biz2Biz",
                "description" : "Skill",
                "skillAliasArray" : [
                        "Business to Business",
                        "B2B"
                ]
        },
```

Note that the **Competency** element now contains a different **name** attribute / **skillName** sub-element ('Biz2Biz' rather than 'Business to Business'), an **id** attribute / sub-element (here '240') and an **idOwner** attribute / sub-element (the customer account). The original authorised name ('Business to Business') is listed as the first **Alias** so it can be recovered if necessary.

Identifying Skills which were not extracted by DaXtra (alias)

Although the DST has extensive coverage of many industry sectors, it is possible that some skills have not yet been included. Through the use of the MSF it is possible to augment the DaXtra system with new skills. In this case you will need to create an **entry** node in the MSF with skill **alias** element(s) set to how this skill will appear in a CV. For instance, an **entry** in your MSF might be:

```
        <entry>
                <category>skill</category>
                <type>Skill</type>
                <code>123</code>
                <descr>Digital Marketing</descr>
                <parent>MySkills</parent>
                <alias locale="en" cs="n" ambig="n">Digital</alias>
        </entry>
```

This will ask the DaXtra system to match every word “Digital” as a skill and normalise it to ‘Digital Marketing’, and assign the user **code** 123. However, please note that this might not be a good idea because the word “Digital” might be used in a different context; when creating **alias** mappings one needs to be careful. There are attributes (locale, cs and ambig) which help to restrict matching to more specific contexts:

- locale – specifies the language of the resume; “en” means match only in English text and not in other languages
- cs – case sensitive; if set to “y” the case must match to the case in the **alias** ‘Digital’
- ambig – specifies whether the word is highly ambiguous; when set to “y” the **alias** will be matched only in the proximity of other known skills

Skills grouping

Using the MSF it is possible to aggregate multiple individually-matched skills into a single collective name (**descr**). For instance, if in your application you would like to record multiple Microsoft certifications as a single skill ‘Microsoft Certified’, you can list all these certifications as **auth** elements in an **entry** ‘Microsoft Certified’ in your MSF. If DaXtra does not extract a particular certification, e.g. MCAF (Microsoft Certified Azure Fundamentals) you can list it as an **alias** element in the same **entry**:

```
<entry>
    <category>qualification</category>
    <type>Degree/Qualification > Professional</type>
    <code>127</code>
    <descr>Microsoft Certified</descr>
    <parent>MySkills</parent>
    <auth type="match">Microsoft Official Trainer</auth>
    <auth type="match">Microsoft Certified Professional</auth>
    <auth type="match">Microsoft Certified Network Engineer</auth>
    <alias locale="en" cs="y" ambig="n">MCAF</alias>
</entry>
```

Now, let’s say that in a resume we find the sentence, “I have an MCNE qualification.”

Firstly, “MCNE” will be parsed by DaXtra through the DST as ‘Microsoft Certified Network Engineer’, as explained above. Then through the application of your MSF, ‘Microsoft Certified Network Engineer’ will be mapped to the **descr** ‘Microsoft Certified’ (represented in the XML profile as the **name** attribute), with the **code** 127, (represented in the profile as the **CompetencyId** sub-element **id** attribute). This will produce the following **Competency** element in the profile:

XML form

```
<Competency oids="77 " description="Degree/Qualification > Professional" auth="yes"
name="Microsoft Certified">
    <CompetencyId id="127" idOwner="me"/>
    <CompetencyEvidence typeDescription="Alias" name="Alias">
        <StringValue>Microsoft Certified Network Engineer</StringValue>
    </CompetencyEvidence>
    <CompetencyEvidence typeDescription="Alias" name="Alias">
        <StringValue>MCNE</StringValue>
    </CompetencyEvidence>
<Competency>
```

Note that the **name** initially normalised through the DST will be listed as the first **Alias** and the actual text match (here ‘MCNE’) will be listed below.

Let’s look at another example, with a resume including the phrase, “I also have an MCAF.” Since ‘MCAF’ is not included in the DST it will not normally be extracted, but through the use of the MSF **alias** in this case it will be identified and extracted, and then mapped to ‘Microsoft Certified’.

XML form

```
<Competency oids="92 " description="Degree/Qualification > Professional"
name="Microsoft Certified">
        <CompetencyId id="127" idOwner="me"/>
        <CompetencyEvidence typeDescription="Alias" name="Alias">
                <StringValue>MCAF</StringValue>
        </CompetencyEvidence>
<Competency>
```

Skills expansion

Conversely you can use the MSF to create multiple Competencies from one single match.

For instance, for a sentence “I passed TOEFL exams” the DaXtra System will create a qualification ‘TOEFL’, but you might want to additionally create two skills named ‘TOEFL Certified’ and ‘Knowledge of English’. In order to do this, you will need to create two entries in your MSF:

```
        <entry>
                <category>qualification</category>
                <type>Degree/Qualification > Professional</type>
                <code>1123</code>
                <descr>TOEFL Certified</descr>
                <parent>MySkills</parent>
                <auth type="match">TOEFL</auth>
        </entry>

        <entry>
                <category>qualification</category>
                <type>Degree/Qualification > Professional</type>
                <code>1124</code>
                <descr>Knowledge of English</descr>
                <parent>MySkills</parent>
                <auth type="match">TOEFL</auth>
        </entry>
```

Now when DaXtra identifies the ‘TOEFL’ qualification you will have two **Competency** elements created in the profile:

XML form

```
<Competency oids="96 " description="Degree/Qualification > Professional" auth="yes"
name="TOEFL Certified">
        <CompetencyId id="1123" idOwner="me"/>
        <CompetencyEvidence typeDescription="Alias" name="Alias">
                <StringValue>TOEFL</StringValue>
        </CompetencyEvidence>
<Competency>

<Competency oids="96 " description="Degree/Qualification > Professional" auth="yes"
name="Knowledge of English">
        <CompetencyId id="1124" idOwner="me"/>
        <CompetencyEvidence typeDescription="Alias" name="Alias">
                <StringValue>TOEFL</StringValue>
        </CompetencyEvidence>
<Competency>
```

The Competency Element

The **Competency** element of the profile can represent one of three things: a skill of the candidate (Skill); a qualification obtained by the candidate (Degree/Qualification); or a position the candidate held during employment (Held Position). The **Competency** element has the following structure:

<Competency description='Skill > IT' auth='yes' name='Python Programming'>

First of all a **Competency** element has a **description** attribute, set to one of the following values:

Skill
Skill > IT
Skill > Language
Degree/Qualification
Degree/Qualification > Professional
Held Position
Industry
Location (* for skilling projects only)
Organization (* for skilling projects only)

Meaning: the type of the skill as determined by the parser

* skilling project introduced in the **Custom output** section: DaXtra can create a Master Skills File from the customer's own skills dictionary

The **name** attribute contains the name of a skill found and matched in the CV. This may come from dynamic parsing, from DaXtra's built-in dictionaries, or from a customer Master Skills File (MSF). In the first case the **name** attribute is the same as the string found in the CV; in the latter two cases the **name** attribute is normalised. For a DaXtra built-in skill the normalisation is the canonical form taken from the DaXtra taxonomy (DST); for a customer skills file, this attribute takes the name from the MSF.

The **auth** attribute is a boolean value and indicates whether the skill is found in DaXtra's built-in dictionaries. DaXtra maintains dictionaries of common skills – each of these "authorised" skills has a normalised form and one or more aliases that we regard as synonyms; the aliases are strings that used to match to CVs. Many of these authorised skills have aliases in more than one language.

Competency element: CompetencyId sub-element

The **Competency** elements can be mapped to the internal ids of a particular classification schema, for example, database codes; these ids are represented in the **CompetencyId** sub-element. For instance, if a company XXRecruit has a classification schema where 'Python' has the id 20789 in the database, this will be represented in the XML as follows:

<CompetencyId id='20789' idOwner='XXRecruit'/>

The **id** attribute is a string value (if spaces are needed, they are represented with a double underscore) and indicates the mapped internal ids that are defined in a customer Master Skills File.

The **idOwner** attribute is a string value (no spaces) and indicates the account used for parsing the CV.

Competency element: TaxonomyId sub-element

The **Competency** elements can be classified with a taxonomy; this is represented by the **TaxonomyId** sub-element. A taxonomy **id** attribute represents an industry/sector. It relates to the candidate's job titles, skills and qualifications, rather than the area in which a company operates, so an IT developer working for a bank will have the industry 'it' rather than 'banking-finance'.

<TaxonomyId id='it~progr-lang' idOwner='DAXTRA' description='it > progr-lang'/>

The **id** attribute is a string value and indicates the taxonomy name, without spaces. Sometimes there is a tag on the end, e.g. -mb, which indicates that the assignment to that industry is not certain.

The **idOwner** attribute is a string value (no spaces); taxonomies are built-in and hence display the account 'DAXTRA' rather than the customer's own account.

The **description** attribute is a string value and indicates the taxonomy name.

For **Competency[description=Skill*]**, possible values are:

accountancy
aerospace
agriculture
automotive
banking-finance
caretaking-cleaning
catering
construction
customer-service
edu
edu~higher
edu~preschool
edu~school
electronics
engineering
engineering~manufacturing
engineering~mechanical
fashion-art
food-drink
health-safety
health-safety~recycling
hr
hr~recruitment
insurance
it
it~ccplusplus
it~dotnet
it~java
legal
marine
marketing
media
medicine
medicine~nursing
medicine~pharmaceutical
mngm-consult
oil-gas
procurement
property
railway
retail
sales
science
science~library
secretarial
security
social-services
social-services~charity
social-services~forces
social-services~public-sector
social-services~youthwork
sports
telecomms
transport
travel

There may be a second part, separated by >, e.g.

it > progr-lang
medicine > infirmary > surgery

This second part is a specific area or type of skill within the industry, and is specific to that industry.

For **Competency[description=Degree/Qualification]** or **Competency[description= Degree/Qualification > Professional]**, other values are available:

high school or equivalent
certification
associates
vocational
some college
intermediategraduate
bachelors
masters
doctorate
certification
HND/HNC or equivalent
secondary
professional

Competency element: CompetencyEvidence sub-element

The **Competency** elements can have **CompetencyEvidence** sub-elements of three kinds:

CompetencyEvidence[typeDescription=Extent of Experience]
and
CompetencyEvidence[typeDescription=Application]
and
CompetencyEvidence[typeDescription=Alias]

CompetencyEvidence[typeDescription=Extent of Experience] represents the duration of a job, or of skill experience when it is explicitly mentioned in the text, e.g. "Java - 3 years experience", or a skill that is associated with a certain work history timeline.

This feature is marked as

```
<CompetencyEvidence name='Extent of Experience' typeDescription='Extent of Experience'
type='YEAR' lastUsed='2019' >
        <NumericValue description='YEAR'>3</NumericValue>
</CompetencyEvidence>

<CompetencyEvidence name='Extent of Experience' typeDescription='Extent of Experience'
type='MONTH' lastUsed='2015'>
        <NumericValue description='MONTH'>58</NumericValue>
</CompetencyEvidence>
```

The **type** attribute indicates the type of time measured; possible values are YEAR | MONTH | DAY.

The **lastUsed** attribute is the year in 4-digit format and indicates the date when the skill was last used - only for jobs (‘Held Position’), or skills (‘Skill’, ‘Skill > IT’) within a work history. For example, if a skill is taken from a work history record of “2010 to 2015”, the lastUsed year will be 2015. If the end date is “till present” then the year will be the last modified date of the document. If the last modified date is not available, the year will be set to the current year.

CompetencyEvidence[typeDescription=Application] represents the kind of application of the skill when it is explicitly mentioned in the text. e.g. "Oracle 8i - design and administration"

This feature is marked as

```
<CompetencyEvidence name='Application' typeDescription='Application' type='design'>
        <StringValue>design</StringValue>
</CompetencyEvidence>

<CompetencyEvidence name='Application' typeDescription='Application' type='administration'>
        <StringValue>administration</StringValue>
</CompetencyEvidence>
```

CompetencyEvidence[typeDescription=Alias] represents the string that was found in the CV, and which has been matched to this skill. The field value may have the same wording as the **Competency name** attribute, or it may be different, e.g. the **Competency name** attribute could be ‘User Interface Design’, while the alias which has triggered the match is “UI design”. This is because of one or more of the following:

a) the **Competency** has been given a **name** attribute taken from the customer MSF
b) the **Competency name** attribute is normalised by DaXtra’s built-in dictionaries
c) analysis by the parser has resulted in changes to the **Competency name** attribute

If some of the above apply then the first **Alias** is set to a skill name (normalised or non-normalised) and other aliases are listed below it, before mapping the skill name to the MSF.

This feature is marked as

```
<CompetencyEvidence typeDescription='Alias' name='Alias'>
        <StringValue>Python Programming</StringValue>
</CompetencyEvidence>

<CompetencyEvidence typeDescription='Alias' name='Alias'>
        <StringValue>Python</StringValue>
</CompetencyEvidence>
```

Or

```
<CompetencyEvidence typeDescription='Alias' name='Alias'>
        <StringValue>French</StringValue>
</CompetencyEvidence>

<CompetencyEvidence typeDescription='Alias' name='Alias'>
        <StringValue>Français</StringValue>
</CompetencyEvidence>

<CompetencyEvidence typeDescription='Alias' name='Alias'>
        <StringValue>Francés</StringValue>
</CompetencyEvidence>
```

Competency element: CompetencyWeight sub-element

The **Competency** elements can be scored among themselves. This indicates the strength of the skill, qualification or position, so they can be sorted accordingly. The weight is applicable only within the same kind of **Competency** i.e. you can compare weights among skills (Competency[type=Skill]), among qualifications (Competency [type=Degree/Qualification]), or among positions (Competency [type=Held Position]). Competency elements can have CompetencyWeight sub-elements of three kinds:

CompetencyWeight[type=skillLevel]
and
CompetencyWeight[type=skillProficiency]
and
CompetencyWeight[type=skillCount]

CompetencyWeight[type=skillLevel] represents the numeric weight assigned to a skill. This is an integer between 0 and 100 and is calculated by an algorithm which evaluates recency, number of mentions, and durations of work histories using this skill. It is not an absolute value and can only be used as a comparison with other skills in the same CV, rather than for comparisons across different CVs. Some customers use this measure to select a candidate's top skills, e.g. the 5 with the highest weight.

This feature is marked as:

```
<CompetencyWeight type='skillLevel'>
        <NumericValue minValue='0' maxValue='100'>99</NumericValue>
</CompetencyWeight>
```

The **NumericValue** is an integer type and possible values are 0-100.

CompetencyWeight[type=skillProficiency] represents the simplified weight of skill proficiency.

This feature is marked as

```
<CompetencyWeight type='skillProficiency'>
        <StringValue minValue='BASIC' maxValue='EXCELLENT'>EXCELLENT</StringValue>
</CompetencyWeight>
```

The **StringValue** is a string type and possible values are: BASIC | WORKING | GOOD | EXCELLENT. The values are derived from **skillLevel**.

- For **Competency[description=Skill > Language]**, we have the values BASIC | INTERMEDIATE | ADVANCED | EXCELLENT | NATIVE, e.g.:

```
<CompetencyWeight type='skillProficiency'>
        <StringValue minValue='BASIC' maxValue='EXCELLENT'>NATIVE</StringValue>
</CompetencyWeight>
```

- For **Competency[description=Degree/Qualification]** or **Competency[description= Degree/Qualification > Professional]**, we can also have values for

Bachelor degrees:

First_Class
Second_Class_Upper
Second_Class_Lower
Third_Class
Honours

Masters degrees:

Pass
Merit
Distinction

or free text extracted from the CV, e.g.:

```
<CompetencyWeight type='skillProficiency'>
        <StringValue>GPA: 3.4</StringValue>
</CompetencyWeight>
```

- For **Competency[description=Held Position]**, the relevance is estimated from the CV work histories.

Current position or within 1 year – EXCELLENT
Within 3 years – GOOD
Within 5 years – WORKING
Otherwise – BASIC

CompetencyWeight[type=skillCount]

This is for **Competency[description=Industry]** only, and reflects the number of timelines where DaXtra parser identifies a particular industry.

Special Competency examples

“A-Level” Competency Features

A Levels are represented as Competencies together with their subjects and marks: A,B,C,D are represented in the CompetencyWeight element as EXCELLENT, GOOD, INTEMEDIATE and BASIC.

So, “A Levels : Geography (A)” will be represented as:

```
<Competency description='Degree/Qualification' auth='yes' name='A Levels/Grades > A Levels > A Levels Geography'>
	<TaxonomyId id='high school or equivalent' idOwner='DAXTRA' description='high school or equivalent'/>
	<CompetencyEvidence name='Application' typeDescription='Application' type='Geography'>
		<StringValue>Geography</StringValue>
	</CompetencyEvidence>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>A Levels/Grades > A Levels > A Levels Geography</StringValue>
	</CompetencyEvidence>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>A Levels : Geography (A)</StringValue>
	</CompetencyEvidence>
	<CompetencyWeight type='skillLevel'>
		<NumericValue minValue='0' maxValue='100'>83</NumericValue>
	</CompetencyWeight>
	<CompetencyWeight type='skillProficiency'>
		<StringValue minValue='BASIC' maxValue='EXCELLENT'>EXCELLENT</StringValue>
	</CompetencyWeight>
</Competency>
```

Driving Licence

This is represented as **Competency[description=’Degree/Qualification > Professional’]**

```
<Competency description='Degree/Qualification > Professional' auth='yes' name='Driving Licence'>
	<TaxonomyId id='certification' idOwner='DAXTRA' description='certification'/>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>Driving Licence</StringValue>
	</CompetencyEvidence>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>Driving Licence: Full</StringValue>
	</CompetencyEvidence>
	<CompetencyWeight type='skillLevel'>
		<NumericValue minValue='0' maxValue='100'>82</NumericValue>
	</CompetencyWeight>
</Competency>
```

Security Clearance

This is represented as **Competency[description='Degree/Qualification > Professional']**

```
<Competency description='Degree/Qualification > Professional' auth='yes' name='Security Clearance
> Security Clearance BC'>
        <TaxonomyId id='certification' idOwner='DAXTRA' description='certification'/>
        <CompetencyEvidence typeDescription='Alias' name='Alias'>
                <StringValue>Security Clearance > Security Clearance BC</StringValue>
        </CompetencyEvidence>
        <CompetencyEvidence typeDescription='Alias' name='Alias'>
                <StringValue>BC Security Clearance</StringValue>
        </CompetencyEvidence>
        <CompetencyWeight type='skillLevel'>
                <NumericValue minValue='0' maxValue='100'>82</NumericValue>
        </CompetencyWeight>
</Competency>
```

For Security Clearance we can have the following levels:

Security Clearance
Security Non Clearance
Security Clearance CONFIDENTIAL
Security Clearance TSSI
Security Clearance SECRET
Security Clearance TOP SECRET
Security Clearance SCI
Security Clearance SSBI
Security Clearance SBI
Security Clearance POLY FS
Security Clearance POLY LS
Security Clearance POLY CI
Security Clearance PUBLIC TRUST
Security Clearance BC
Security Clearance SC
Security Clearance DV
Security Clearance CTC
Security Clearance CRB
Security Clearance DBS
Security Clearance NATO
Security Clearance EU

Language from certification

Language proficiency could also be derived from a language certification, for example:

TOEIC 880 => English Excellent

```
<Competency description='Skill > Language' auth='yes' name='English'>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>English</StringValue>
	</CompetencyEvidence>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>TOEIC 880</StringValue>
	</CompetencyEvidence>
	<CompetencyWeight type='skillLevel'>
		<NumericValue minValue='0' maxValue='100'>84</NumericValue>
	</CompetencyWeight>
	<CompetencyWeight type='skillProficiency'>
		<StringValue minValue='BASIC'
maxValue='EXCELLENT'>EXCELLENT</StringValue>
	</CompetencyWeight>
</Competency>
```

IELTS 6.5 => English Intermediate

```
<Competency description='Skill > Language' auth='yes' name='English'>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>English</StringValue>
	</CompetencyEvidence>
	<CompetencyEvidence typeDescription='Alias' name='Alias'>
		<StringValue>IELTS 6.5</StringValue>
	</CompetencyEvidence>
	<CompetencyWeight type='skillLevel'>
		<NumericValue minValue='0' maxValue='100'>48</NumericValue>
	</CompetencyWeight>
	<CompetencyWeight type='skillProficiency'>
		<StringValue minValue='BASIC'
maxValue='EXCELLENT'>INTERMEDIATE</StringValue>
	</CompetencyWeight>
</Competency>
```

Appendix A - XML/JSON Summary

XML						JSON			
XML element	**XML attributes/ sub-elements**	**XML attributes/ sub-elements**	**XML attributes**	**Example value**	**Example content**	**JSON object**	**JSON objects**	**JSON objects**	**Example value**
Competency	oids			22_1 30_4_1		Competency			
	description			Held Position			description		Held Position
	auth			no			auth		false
	name			Registered Nurse			skillName		Registered Nurse
	CompetencyId	idOwner		me			CompetencyId	idOwner	me
		id		JOB:123				id	JOB:123
	TaxonomyId	id		medicine			TaxonomyId		
		idOwner		DAXTRA				idOwner	DAXTRA
		description		medicine				description	medicine
	CompetencyEvidence	name		Extent of Experience			SkillUsed		
		typeDescription		Extent of Experience				value	92
		type		MONTH				type	Months
		lastUsed		2013			lastUsed		2013
		NumericValue	description	MONTH	92				
	CompetencyEvidence	typeDescription		Application			skillAliasArray		
		name		Application					
		type		Health					Health
		StringValue			Health				
	CompetencyEvidence	typeDescription		Alias					Registered Nurse
		name		Alias					
		StringValue			Registered Nurse				
	CompetencyWeight	type		skillLevel			skillLevel		
		NumericValue	minValue	0	39				39
			maxValue	100					
	CompetencyWeight	type		skillProficiency			skillProficiency		BASIC
		StringValue	minValue	BASIC	BASIC				
			maxValue	EXCELLENT					

Appendix B: Glossary

XML	JSON	Description
Competency	Competency	element/object containing skill/qualification/job title
oids		sequential number linking Competency node to information in CV
description (Competency)	description	type of Competency
auth	auth	denotes whether or not a skill is in DaXtra's in-built skill lists
name (Competency)	skillName	normalised name of parsed skill
CompetencyId	CompetencyId	sub-element/object containing custom id information
idOwner (CompetencyId)	idOwner (CompetencyId)	name of account used for parsing the CV
id (CompetencyId)	id	custom id linking skill to customer's database
TaxonomyId	TaxonomyId	
id (TaxonomyId)		name of Taxonomy
idOwner (TaxonomyId)	idOwner (TaxonomyId)	name of account owner of the Taxonomy (always DAXTRA)
description (TaxonomyId)	description (TaxonomyId)	name of Taxonomy
CompetencyEvidence (for name='Extent of Experience')	SkillUsed	this sub-element/object contains information on the length of experience
name='Extent of Experience'		see above
typeDescription='Extent of Experience'		see above
type (CompetencyEvidence, for name='Extent of Experience')	type	type of length of time, e.g. months, years
lastUsed		date the skill or job was last used
NumericValue (CompetencyEvidence, for name='Extent of Experience')	value	total length of time the skill was used for
	lastUsed	date the skill or job was last used
CompetencyEvidence (for name='Alias')		this sub-element includes details of strings found in the CV which match to this skill
name='Alias'		see above
typeDescription='Alias'		see above
StringValue (CompetencyEvidence, for name='Alias)	skillAliasArray	this sub-element/object contains a list of strings found in the CV which match to this skill
CompetencyEvidence (for name='Application')		this sub-element describes the kind of application of the skill
name='Application'		see above
typeDescription='Application'		see above
type (CompetencyEvidence, for name='Application')		the kind of application of the skill
StringValue (CompetencyEvidence, for name='Application')	skillAliasArray	the kind of application of the skill (in JSON this is a value in **skillAliasArray**
CompetencyWeight (for type='skillLevel')		this sub-element denotes a numeric weight assigned by algorithm for proficiency in this skill
type='skillLevel'		see above
NumericValue (CompetencyWeight)	skillLevel	a weight assigned by algorithm to proficiency in this skill
CompetencyWeight (for type='skillProficiency')		this sub-element denotes a normalised Competency weight taken from the numeric weight, or free text
type='skillProficiency'		see above
StringValue (CompetencyWeight)	skillProficiency	a weight assigned by algorithm to proficiency in this skill