

Leasachadh Corpas na Gàidhlig (LEACAG)

# **Final Report: Siostam briathrachais**

**Prepared for Bòrd na Gàidhlig  
(Research Project no. CR15-02)**

**By SOILLSE researchers/developers –**

Mark McConville, Stephen Barrett (University of Glasgow)

**January 2019**

# GEÀRR-CHUNNTAS

Ann an aithisg seo, bheir sinn cunntas air siostam-briathrachais paidhleat LEACAG. Is e stòr-dàta agus eadar-aghaidh air-loidhne a tha seo a tha a' coileanadh riatanas bho Bhòrd na Gàidhlig a bha anns an speac-rannsachaidh airson pròiseact CR15-02:

- “àite air-loidhne airson briathrachas ùr na Gàidhlig a cho-òrdanachadh, a dhearbhadh is a sgaoileadh”

A' leantainn amasan a' Bhùird, tha na feartan a leanas aig an t-siostam seo:

- “Coimheadar ri neo-chunbhalachd ann am briathrachas ùr tro àite eadar-ghnìomhach air-loidhne airson ùrachadh fìor-ama le luchd-inntrigidh ùghdarraichte. Thèid na riochdan as annsa le BSC a chomharrachadh anns a' ghoireas phoblach seo tro creative commons.”

Bha siostam-briathrachais LEACAG air a chur air bhog le luchd-rannsachaidh agus luchd-leasachaidh shiostaman aig Oilthigh Ghlaschu eadar Giblean 2016 agus Faoilleach 2018.

Is e siostam Lexicopia a tha ann an cuspair **Caibideil 1**, a tha ag obair mar theicneòlas *cùl-raoin* ('back-end') airson siostam-briathrachais LEACAG. Tha sinn a' mìneachadh carson a chruthaich sinn an siostam seo, agus tha sinn a' solarachadh stiùireadh teicnigeach mun t-siostam. Tha sinn air a bhith a' leasachadh Lexicopia aig Oilthigh Ghlaschu on bhliadhna 2013, agus is e siostam-leasachaidh faclaireachd air-loidhne a tha ann, a tha ùr-ghnàthach, fosgailte agus ioma-chànanach, agus fo bhuidhne adhartasan nuadh ann am faclaireachd agus coimpiutaireachd ('Lexicography 2.0', 'born-digital lexicography'). Tha Lexicopia stèidhichte air siostam stiùireadh fhaidhlichean a tha sìmplidh ach so-leudachaidh, agus air modail-dàta a tha a' cleachdadh XML ('eXtensible Markup Language'). Tha an siostam a' tasgadh toraidhean-lorg ('search results') airson siùbhlachd agus so-chleachdadh nas fheàrr. Tha Lexicopia a' solarachadh API ('application programming interface') coitcheann a dh'fhaodas luchd-leasachaidh a chleachdadh airson eadar-aghaidhean sònraichte a chruthachadh. Is urrainn do luchd-cleachdaidh na h-eadar-aghaidhean seo a chur gu feum airson a bhith ag obair leis a' phrìomh stòr-dàta ann an diofar dhòighean – fiosrachadh faclaireachd fhaighinn air ais agus a thaisbeanadh; fiosrachadh a chur a-steach agus ùghdarrachadh.

Is e stiùireadh teicnigeach airson eadar-aghaidh siostam-briathrachais LEACAG ('front-end') a tha ann an cuspair **Caibideil 2**. Tha an eadar-aghaidh seo mothachail ('responsive') agus furasta a chleachdadh, agus tha i a' toirt taic do dhearbhadh luchd-cleachdaidh ('user authentication'). Tha fiosrachadh ann mu dheidhinn structair an t-siostaim, mar a tha e ag obair, mar a tha na diofar phàirtean ag obair còmhla, agus mar a tha iad ag obair cuide ri Lexicopia. Faodaidh luchd-cleachdaidh clàradh airson an t-siostaim ma tha cunntas Google aca (le API Google Authentication). Tha stòr-dàta ann as urrainn do luchd-rianachd a chleachdadh gus luchd-pàirteachaidh a rianachadh agus gus sùil a chumail orra. Tha an stòr-dàta seo ag aithneachadh trì ìrean neach-cleachdaidh: luchd-cleachdaidh coitcheann, le comas fiosrachaidh mu fhaclan Gàidhlig a lorg; com-pàirtichean, le comas fiosrachaidh ùr a mholadh, agus luchd-deasachaidh, le comas fiosrachaidh ùghdarrachadh.

Ann an **Caibideil 3**, tha stiùireadh neo-theicnigeach ann airson luchd-cleachdaidh coitcheann an t-siostaim, a' gabhail a-steach Com-pàirtichean agus Luchd-deasachaidh.

Ann an **Caibideil 4**, tha sinn a' dèiligeadh ris na h-oidhirpean a tha fhathast romhainn ann a bhith a' leasachadh an t-siostaim paidhleat a tha seo, agus tha sinn a' dèanamh molaidhean airson an ath cheum ann an leasachadh corpais na Gàidhlig. Tha sinn a' dèiligeadh ris a' chùl-raon shòisio-chànanach, gu h-àraidh cùl-raon a' phròiseict *Diùth is Inneach*, a chaidh a

choileanadh ann an 2014, agus a dhearbhaich gu bheil luchd-cleachdaidh na Gàidhlig feumach air 'one-stop-shop' – aon àite air-loidhne airson goireasan cànanach Gàidhlig, a' gabhail a-steach briathrachas. Tha e air a bhith caran furasta siostam *meadhanaichte* ('centralised') a chur air bhog a tha a' coileanadh prìomh riathanasan a' phròiseict. Ge-tà, tha e air a bhith nas doirbhe còdan-obrach a stèidheachadh airson an t-siostaim, mar eisimpleir, stiùireadh do bhuidhnean Gàidhlig air mar as urrainn dhaibh a chleachdadh nan obair làitheil. Mar sin, tha sinn a' moladh modail *feadaraichte* ('federated') airson an ath cheum dhen phròiseact. Leis an t-seòrsa siostaim seo, b' urrainn do bhuidhnean an siostam a chleachdadh ann an dòighean nas sùbailte, agus bhiodh cothrom ann airson ceanglaichean nas dlùithe a dhèanamh eadar siostam-briathrachais LEACAG agus iomairtean faclair eachd Gàidhlig eile le maoinachadh bhon riaghaltas, mar eisimpleir Faclair LearnGaelic no Faclair na Gàidhlig.

# EXECUTIVE SUMMARY

This document reports on the development of the pilot LEACAG terminology system – an online database and interface, which aims to satisfy the following requirement from Bòrd na Gàidhlig in the specification for research project CR15-02:

- “an online space for managing, evaluating and disseminating new Gaelic terminology”.

This system has the following characteristics, again as required by the Bòrd:

- “Inconsistencies in new terminology will be tackled through an interactive online space allowing real-time updating by approved contributors. The forms preferred by BSC [i.e. CCC] will be indicated in this public resource made available by creative commons.”

Work on the LEACAG terminology system was undertaken by researchers and developers at Glasgow University between April 2016 and January 2018.

**Chapter 1** provides the rationale and technical documentation for the Lexicopia system, which provides the back-end for the LEACAG terminology system. Lexicopia has been developed at Glasgow University since 2013, and is a state-of-the-art, open-source, multilingual, online lexicographic development platform, influenced by some of the leading ideas of 21st century, ‘born digital’ lexicography (‘Lexicography 2.0’). Lexicopia is built around a simple but extensible file system and XML-based data model, and makes extensive use of search result cache-ing to improve performance and thus user experience. Lexicopia provides a generic API allowing developers to write bespoke interfaces to interact with the central database in different ways – retrieving and displaying information, adding information and ‘authorising’ information.

**Chapter 2** provides technical documentation for the LEACAG terminology system front-end, which should be understood as a responsive, user-friendly, bespoke interface to the Gaelic Lexicopia database, with additional support for user authentication. We discuss the structure, functionality and implementation of the different elements of the LEACAG terminology system interface and the model of interaction between this interface and the Lexicopia back-end API. As part of the development of the system, we built a user login system based on the Google Authentication API. User activity is managed and monitored by a custom-built database, which recognises three ‘levels’ of user – Users can search for information about Gaelic terms; Contributors can add information about Gaelic terms; and Editors can authorise information about Gaelic terms.

**Chapter 3** contains non-technical guidance for general users of the system, whether Users, Contributors or Editors.

**Chapter 4** includes a discussion of the some of the challenges that remain for developing this pilot online Gaelic terminology system, and some ideas which might usefully be addressed in the next phase of Gaelic corpus development. We discuss the sociolinguistic background to the development of the system, in particular the 2013 Dìùth is Inneach project, which first demonstrated the recognised need for a ‘one-stop-shop’ for Gaelic language resources, including terminology. While it has been a reasonably straightforward process to develop a ‘centralised’ online system which satisfies the basic criteria for the project, what has proved more challenging is establishing protocols for using this system in practice, in particular guidance for Gaelic development agencies as to how to integrate use of the system into their daily work practices. We thus conclude by proposing a more ‘federated’ model for the next phase of the project, which would allow Gaelic development agencies more flexible ways of interacting with the system, and would also facilitate a closer

relationship between the LEACAG terminology system and other publicly-funded Gaelic lexicographic initiatives, in particular the LearnGaelic dictionary and Faclair na Gàidhlig.

# CONTENTS

1	THE LEXICOPIA SYSTEM	1
1.1	Lexicopia and lexicography 2.0	2
1.2	The Lexicopia data model	4
1.3	The Lexicopia cache	7
1.4	Interacting with the Lexicopia database	11
2	IMPLEMENTATION OF THE LEACAG INTERFACE	19
2.1	Structure of the LEACAG homepage	20
2.2	Interacting with the LEACAG homepage	30
2.3	The LEACAG user database	35
3	USING THE LEACAG INTERFACE	37
4	FUTURE CHALLENGES	44
4.1	Background	44
4.2	Developing the LEACAG system	46
4.3	Future challenges	51
4.4	Links with other dictionary projects	54
5	REFERENCES	56

# 1. THE LEXICOPIA SYSTEM

This chapter discusses the back-end to the LEACAG terminology system – the Lexicopia lexicographic system developed at the University of Glasgow since 2013. The intended audience, at least for sections 1.2 through 1.4, is systems developers who need to get the LEACAG front-end installed on a new server, or who have been tasked with updating the system and need to know how the front-end interfaces with the back-end.

**Section 1.1** discusses the motivations behind the Lexicopia system in particular the concept of Lexicography 2.0.

**Section 1.2** presents the basic Lexicopia data model, i.e. the general structure of the file system and the lexical entry files.

**Section 1.3** discusses the Lexicopia caching system, designed to speed up performance of the system.

**Section 1.4** describes the many ways in which users can interact with the Lexicopia system, e.g. to look up entries, to add information to entries, etc.

## 1.1. Lexicopia and Lexicography 2.0

The Lexicopia system (McConville 2014, 2015ab) is a general infrastructure for multilingual lexicographic development developed at Glasgow University since 2013, inspired by the conclusions of the *Dlùth is Inneach* project (Bell et al 2014), as well as by the growing body of research into what has become known as ‘Lexicography 2.0’ (Zimmer 2014) – the study of how the traditional dictionary and thesaurus can be re-imagined and re-engineered for the internet age.

Work on the Lexicopia system has been driven by the following five principles of modern lexicography development:

1. **Dictionaries should be cognitively realistic.** The design of new dictionaries (and other lexical resources) should reflect what we know about the structure of the ‘mental lexicon’, the way that knowledge of words and phrases is stored and organised in our brains. In other words, 21st century lexicographers need to take into account the results of fifty years of research into psycholinguistics and cognitive science.
2. **The mental lexicon is not a list.** A traditional printed dictionary is essentially a list of words. However, we know from the work of psycholinguists and cognitive scientists that the mental lexicon is not organised as a list, but rather as a multidimensional, associative network, often termed a ‘wordnet’ (Miller 1995). In a wordnet, lexical entries are linked to each other according to a range of different syntactic and semantic relations: hyponymy (*oak – tree – plant*), meronymy (*leaf – tree – forest – timber, cow – beef*), antonymy (*happy – sad, war – peace*). If a modern dictionary should reflect the mental lexicon, and the mental lexicon is a network, then this means that modern dictionaries should follow the network model as well.
3. **The mental lexicon contains more than just words.** Again, the traditional printed dictionary is organised around the basic principle that each lexical entry is a word. However, something else we have learned from psycholinguistics and cognitive science is that humans store information about all manner of linguistic units in their mental lexicons (Jackendoff 1997) – prefixes and suffixes (e.g. *un-*, *post-*, *-ness*, *-ful*), compounds (e.g. *forest fire, civil war, happy-clappy*), phrasal verbs (e.g. *push on, hand in, give in to*), idioms (e.g. *kick the bucket, set the heather on fire*), proverbs (e.g. *a stitch in time saves nine, blood is thicker than water*), quotations (e.g. *we will fight them on the beaches, Brexit means Brexit*), and so on. Again, if modern dictionaries are to reflect the mental lexicon, and the mental lexicon is a network of all kinds of words and phrases, then our dictionaries should reflect this as far as is technically possible.
4. **The technology exists to create network-based dictionaries.** Both the Internet hardware and the World Wide Web software are based on the idea of a distributed network of resources of all manner of different types – computers, servers, printers, documents, images, videos, sound files. The ‘Internet of Things’ is taking this idea one step further by connecting up domestic appliances and other systems to each other using internet technologies. With this in mind, it seems like a relatively small step to implement a network-based model of lexicographic information which can be a practical alternative to traditional dictionaries and online lexical databases.
5. **Dictionaries should be explorable.** One of the great advantages of traditional printed dictionaries over standard online lexical databases is that dictionaries can be browsed, explored, thumbed through, opened at random, whereas databases can usually only be *queried* (i.e. ‘What does the word *serendipity* mean?’). Interacting with an online dictionary should not just be about looking words up one at a time. Rather you should be able to start out by looking something up and then find yourself drawn into the



vocabulary, distracted by the richness of the language. It shouldn't just tell you what a word means, but let you figure this out for yourself by exploring how the word relates to other words in the semantic domain. When you first open up a dictionary, whether physical or virtual, you generally have a question that you want an answer to – a known unknown. A good dictionary should do more than simply convert that known unknown into a known known. It should have the potential to divert the user down lexical garden paths into clusters of unknown unknowns and unknown knowns. They should come across words and patterns that they didn't know they didn't know, and also ones that they didn't know they did know.

Two other important principles have informed the design of the Lexicopia system over the years:

- As far as possible, the system should be constructed around standard, open-source technologies. From the start, we took this to mean that we would use the core **XML** (Extensible Markup Language) standards for representing structured data (like dictionary entries), agreed by the W3C (World Wide Web Consortium), the international organisation responsible for developing open standards to ensure the long-term growth of the Web.
- Lexicopia should be built around the industry-standard product development model used by the big dictionary publishing companies (Collins, MacMillan, etc.), which makes a key distinction between: (a) the company's core lexical database; and (b) the many derived dictionaries created by editors from this core database (Atkins & Rundell 2008). In this sense, when an editorial team is tasked with creating a new dictionary, their main job is to develop policies and procedures first of all for **selecting** data from the core database, and secondly for **displaying** that data in a user-appropriate manner. Thus, one lexical database can support many dictionaries of different kinds (concise dictionaries, bilingual dictionaries, learner's dictionaries).

## 1.2. The Lexicopia data model

### 1.2.1. File structure and naming

The basic file structure of the Lexicopia system is as follows:

- br
  - lexemes
    - gwez.xml
    - ti.xml
    - ...
- cy
  - lexemes
    - coeden.xml
    - tŷ.xml
    - ...
- gd
  - lexemes
    - craobh.xml
    - taigh.xml
    - ...
- ...

In other words, every language in the system has its own top-level directory where all data for that language is stored. Languages are identified by their standard ISO 639 two/three digit names, e.g. `br` for Breton, `cy` for Welsh, `gd` for Scottish Gaelic.

Each language directory contains a sub-directory called `lexemes`, and this is the core lexical database for the language, where all of the lexicographic information compiled by lexicographers is stored. For example, for every word or phrase in the Gaelic lexicon, there will be an XML file in `gd/lexemes` containing all of the information about that word or phrase in Lexicopia standard format – all of the data concerning the word *craobh* will be found in `gd/lexemes/craobh.xml`, for *taigh* in `gd/lexemes/taigh.xml`, and so on.

One of the core design features of Lexicopia is that lexical headwords and filenames should be identical as far as possible:

- Accented characters should be preserved in filenames, e.g. `mòr.xml`, `cangarù.xml`.
- Uppercase characters should also be preserved, e.g. `Dùn_Èideann.xml`, `Seumas.xml`.
- Spaces should be converted into underscores, e.g. `taigh_beag.xml`, `craobh_challtainn.xml`.
- Homographs are represented using suffixes. For example, the Gaelic word *breac* is ambiguous between an adjective ('speckled') and a noun ('trout'). Appropriate filenames would be `breac-adj.xml` and `breac-n.xml`.

One important technical consideration arises from these naming conventions – any lexical database needs to be hosted on a server running a case-sensitive, case-preserving filesystem (e.g. a standard Linux server).

## 1.2.2. XML file structure

The XML files for each word and phrase in the language follow a common XML schema:<sup>1</sup>

lexeme

- part\* @ref
- form+
  - orth #PCDATA
- trans+ #PCDATA
- note\* #PCDATA

An alternative, equivalent representation for this schema (less concise but which might be more familiar to some readers) is as follows:

```
<lexeme>
  <part ref="..." />
  <part ref="..." />
  . . .
  <form>
    <orth>. . .</orth>
  </form>
  <form>
    <orth>. . .</orth>
  </form>
  . . .
  <trans>. . .</trans>
  <trans>. . .</trans>
  . . .
  <note>. . .</note>
  <note>. . .</note>
  . . .
</lexeme>
```

The root element of a Lexicopia XML entry file is always `lexeme`. A `lexeme` element has the following components:

- **Zero or more `part` elements, linking to other lexemes which are contained within the current lexeme.** For example: (a) the lexical entry for *craobh challtainn* ‘hazel tree’ contains `part` elements referring to both *craobh* ‘tree’ and *challtainn* ‘hazel’; and (b) *taigh beag* ‘toilet’ contains links to both *taigh* ‘house’ and *beag* ‘small’. The `part` elements illustrate a first aspect of the ‘network-like’ nature of Lexicopia data – entries are connected to each other using syntagmatic relations, allowing users to browse directly between components and compounds and vice versa.
- **One or more `form` elements, each representing one particular inflectional form of the lexeme, with the first one representing the ‘citation’ form (or ‘headword’ form).** Each form element contains exactly one `orth` element, containing the actual text of the orthographic representation. For instance, the lexeme *craobh* ‘tree’, will have at least the

---

<sup>1</sup> Note that `#PCDATA` stands for ‘parsed character data’ which in this context just means unstructured Unicode text strings.

following `form|orth` elements: *craobh*, *craoibh*, *craoibhe*, *craobhan*. In time, we envisage adding phonetic and grammatical information to `form` elements as well.

- **One or more `trans` elements, each representing an English translational equivalent for the lexeme (or some other language, e.g. French for Breton lexemes).** For example, the Gaelic lexeme *comhairle* has at least two `trans` elements: *advice*, *council*. With regard to translational equivalents, the following liberal approach is recommended for Gaelic Lexicopia as a whole – if any Gaelic-to-English or Gaelic-to-English dictionary or glossary lists English term Y as equivalent to Gaelic term X, then Y should be listed as a `trans` element in entry X.
- **Zero or more `note` elements, each containing some kind of lexicographic note about the lexeme.**

For example, here is the (abbreviated) lexical entry for *craobh challtainn* contained in the file `gd/lexemes/craobh_challtainn.xml`:

```
<lexeme>
  <part ref="craobh"/>
  <part ref="calltainn"/>
  <form>
    <orth>craobh challtainn</orth>
  </form>
  <form>
    <orth>craoibh challtainn</orth>
  </form>
  <form>
    <orth>craoibhe calltainn</orth>
  </form>
  <form>
    <orth>craobhan calltainn</orth>
  </form>
  <trans>hazel tree</trans>
  <note>Feminine compound noun.</note>
  <note>Listed in Colin Mark's dictionary.</note>
</lexeme>
```

There are two other features of the source XML files that can be mentioned at this point:

- A `trans` element can have an optional `index` attribute, with two possible values: (a) `index="no"`, means that this English translational equivalent should not be included in an index but only displayed in the entry; and (b) `index="only"` means that this English translation should be included in an index but not displayed. If we take the example of *Riaghaltas na h-Alba* this allows a lexicographer to mark different translations for display in entries (i.e. 'the Scottish Government') and for indexing (i.e. 'Scottish Government'). If there is no `index` attribute (the default position), then the translational equivalent will be both indexed and displayed.
- Any element can have an optional `resp` element, indicating the editors or authorities which have explicitly taken responsibility for the existence of that element. For instance, if Stòrlann have explicitly authorised that *mearcair* is their recognised Gaelic term for the element 'mercury (Me)' to be used in high school chemistry lessons, then the relevant `lexeme/trans` element can be specified as `resp="Stòrlann"`.

## 1.3. The Lexicopia cache

The lexicographic data stored in `gd/lexemes` is technically and theoretically sufficient to construct a user interface to the dictionary around. However, when we built the first version of the interface we immediately ran into problems of efficiency – as soon as the lexicon grew to over a thousand entries, searches for words and phrases began to take more than 10 seconds to run, which we deemed to be much too long, especially since we envisage the lexicon containing many tens of thousands of entries. In order to provide a sustainable solution to this problem, we added explicit cache-ing functionality to the system. There are three types of cache that any user interface can have access to to speed up searches:

- a target language index
- an English language index
- a pre-generated HTML file for every lexical entry.

This, the basic directory structure for each language in the system is actually as follows:

```
▸ gd
  ▸ lexemes
    ▸ craobh.xml
    ▸ taigh.xml
    ▸ ...
  ▸ cache
    ▸ targetIndex.json
    ▸ englishIndex.json
    ▸ html
      ▸ craobh.html
      ▸ taigh.html
      ▸ ...
```

The following sections will discuss each of these caches in turn.

### 1.3.1. The target index cache

The JSON file `gd/cache/targetIndex.json` is automatically generated from the source lexical entries in `gd/lexemes` using a custom PHP script `generateTargetIndex.php`. The JSON file has the following basic format:

```
{"targetIndex": [
  {
    "target": "comhairle",
    "id": "comhairle"
  },
  {
    "target": "craobh",
    "id": "craobh"
  },
  . . .
]}
```

In other words, it is a list of pairs of:

- the orthographic citation form of the lexeme
- the lexeme's unique identifier, taken from its filename.

The purpose of this cache file is to make searching for Gaelic words and phrases more efficient to implement in a user interface. Rather than having to trawl through every XML file every time the user presses a key, the interface can just query the cached index.

The PHP script is run from the command line using a single parameter (the relevant ISO 639 language code), as in:

```
> php generateTargetIndex.php gd
```

The script runs through the lexical entry files in `gd/lexemes` one at a time, and creates a PHP object for every form of the lexeme (including inflected forms), including: the form, the filename, and the English translations. These entries are then sorted alphabetically (ignoring case and accents), converted to JSON, and written to the output file.

On the current Lexicopia installation on the `dasg.ac.uk` server, this script has been set to run automatically each day, at 2am GMT, in order to ensure that the cache is as up-to-date as possible.

### 1.3.2. The English index cache

The JSON file `gd/cache/englishIndex.json` is automatically generated from the source lexical entries in `gd/lexemes` using a custom PHP script `generateEnglishIndex.php`. The JSON file has the following format:

```
{"englishIndex": [
  {
    "en": "mercury",
    "targets": [
      {"id": "mearcair", "form": "mearcair"},
      {"id": "airgead_beò", "form": "airgead beò"},
      . . .
    ]
  },
  {
    "en": "advice",
    "targets": [
      {"id": "comhairle", "form": "comhairle"},
      . . .
    ]
  },
  . . .
]}
```

In other words, it is essentially a list of pairs of:

- an English term
- the IDs of all the Gaelic lexemes which have it as a translational equivalent.

The purpose of this cache file is to make searching for English words and phrases more efficient to implement in a user interface. Rather than having to trawl through every XML file every time the user presses a key, the interface can just query the cached index.

The PHP script is run from the command line using a single parameter (the relevant ISO 639 language code), as in:

```
> php generateEnglishIndex.php gd
```

The script runs through the lexical entry files in `gd/lexemes` one at a time, pulling out every English translation (not explicitly listed as `index="no"`). The resulting array is then sorted alphabetically and turned into a set by removing duplicates. The script then goes through each of these English terms in turn, trawling through the lexical files again for all the Gaelic lexemes that are associated with that English translation. A PHP object is created for the English term, including: the term itself, and the IDs of all the Gaelic lexemes that are associated with it. Finally, these entries are sorted, converted to JSON, and written to the output file.

On the current Lexicopia installation on the `dasg.ac.uk` server, this script has been set to run automatically each day, at 2am GMT, in order to ensure that the cache is as up-to-date as possible.

### 1.3.3. The HTML cache

The HTML files in `gd/cache/html` are automatically generated from the source lexical entries in `gd/lexemes` using a custom PHP script `generateHTML.php`. There should be one such HTML file for every XML lexical entry file in `gd/lexemes`. These files are formatted caches listing: (a) the other lexemes which are **parts** of the lexeme; and (b) the other lexemes which **contain** the lexeme. These files are used to speed up the display of lexical entries in web interfaces (see section 2.4 for details).

For example here are the (abbreviated) contents of `gd/cache/html/Gàidhlig.html`:

```
<dt>Components:</dt>
<dd>
  <a class="lexicopiaLink" data-id="Gàidheal"
    title="Gael, Highlander">Gàidheal</a>
</dd>
<dt>Compounds:</dt>
<dd>
  <a class="lexicopiaLink" data-id="bàrdachd_Ghàidhlig"
    title="Gaelic poetry">bàrdachd Ghàidhlig</a>
</dd>
<dd>
  <a class="lexicopiaLink" data-id="leabhraichean_Gàidhlig"
    title="Gaelic books">leabhraichean Gàidhlig</a>
</dd>
```

This HTML source will then be displayed by web browsing software as something like:

Components:

[Gàidheal](#)

Compounds:

[bàrdachd Ghàidhlig](#)

[leabhraichean Gàidhlig](#)

Note that the blue underlined text chunks are hyperlinks, which when clicked will take the user to the relevant lexical entry.

The PHP script is run from the command line using a single parameter (the relevant ISO 639 language code), as in:

```
> php generateHTML.php gd
```

It runs through the lexical entry files in `gd/lexemes` one at a time, and for every `gd/lexemes/*.xml` it creates a file `gd/cache/html/*.html`. First of all, every component of the word is written to the file as a `lexicopiaLink`. Then the script searches through the entire lexicon for all the lexemes containing the current lexeme as a component, and then writes each of these to the file as a `lexicopiaLink`.

Again, on the current Lexicopia installation on the `dasg.ac.uk` server, this script has been set to run automatically each day, at 2am GMT, in order to ensure that the cache is as up-to-date as possible.



## 1.4. Interacting with the Lexicopia database

There are currently six supported ways in which a user interface can be programmed to interact with the Lexicopia database:

- retrieve and display a lexical entry
- add a new lexical entry
- add a new comment to an existing lexical entry
- add a new English translational equivalent to a lexical entry
- add a new orthographic form to a lexical entry
- authorising an English translational equivalent

Each of these is discussed in one of the following sections.

### 1.4.1. Retrieving and displaying entries

A user interface can retrieve a lexical entry by calling a custom-designed PHP script – `generateLexicalEntry.php`. When calling this script two URL parameters must be provided via an AJAX call:

- `lang` – the relevant ISO 639 language code
- `id` – the relevant lexical ID

For instance, the entry for *craobh* in the Gaelic lexicon can be retrieved as follows:

```
generateLexicalEntry.php?lang=gd&id=craobh
```

This PHP script generates a formatted HTML version of a lexical entry on the fly from the relevant XML file in the lexical database.

It generates HTML in the following format, based on the entry for *craobh challtainn* in section 1.2.2:

```

<div class="lexicopiaEntry">
  <h1>
    <span class="lexicopiaHeadWord">craobh challtainn</span>
    <span class="lexicopiaEnglish">hazel tree</span>
  </h1>
  <dl>
    <dt>Forms:</dt>
    <dd>craobh challtainn</dd>
    <dd>craoibh challtainn</dd>
    <dd>craoibhe calltainn</dd>
    <dd>craobhan calltainn</dd>
    *** CACHED HTML from gd/cache/html/craobh_challtainn.html ***
  </dl>
  <ul>
    <li>Feminine compound noun.</li>
    <li>Listed in Colin Mark's dictionary.</li>
  </ul>
</div>

```

When viewed in a web browser, this HTML will appear as follows (assuming the standard stylesheet `lexicopiaEntries.css` is being used):

## **craobh challtainn** hazel tree

### Forms:

craobh challtainn  
 craoibh challtainn  
 craoibhe calltainn  
 craobhan calltainn

### Components:

[craobh](#)  
[calltainn](#)

- Feminine compound noun.
- Listed in Colin Mark's dictionary.

The way that the PHP script generated this entry is worth mentioning, since it involves an efficient interaction between on-the-fly processing and the Lexicopia cache:

- The headword, English translations, inflectional forms and notes are generated directly from the XML file, on-the-fly whenever the script is called.
- Components and compounds are loaded from a cached HTML file, if one exists.

After trying out a range of different approaches to this, we decided that the approach discussed here allows the best combination of database synchronisation and efficiency. In other words, the entry displayed will always be fully up-to-date with the lexical database,

except for the cached components and compounds, which will be updated every day, as discussed in section 1.3.3.

## 1.4.2. Adding a new entry

The Lexicopia system provides a static PHP class, `NewEntry`, to enable a user interface to allow users to add new entries to the database. This class has a single public static function, `addEntry`, that can be called externally using the following instruction:

```
NewEntry::addEntry($_POST, $path);
```

The second input parameter here, `$path`, specifies the location of the Lexicopia system on the relevant web server. Note that the value of `$path` must end in a / (or the appropriate operating system symbol to separate directory names).

The first input parameter `$_POST` is a collection of POST data in the following format:

```
"lang" => ". . ."  
"en" => ". . ."  
"target" => ". . ."  
"related" => ". . ."  
"notes" => ". . ."  
"userEmail" => ". . ."
```

These fields denote the following things:

- `lang` – the ISO 639 language code, e.g. `gd`
- `en` – the proposed English term, e.g. `ambitious`
- `target` – the proposed target language term, e.g. `mianmhor`
- `related` – related inflectional forms, e.g. `mianmhoir`, `mianmhoire`
- `notes` – any useful lexicographic notes provided by the contributor, e.g. `adjective`
- `userEmail` – the contributor's email address.

When a new entry is being processed, the following things happen:

Firstly, a new lexical ID is created based on the target value specified by the contributor. Two things are important here:

1. The ID must be clear that it has been recently contributed and not yet verified by an editor. For this reason, every contributed ID is prefixed with the string `qqq-` (denoting that it is somehow to be considered 'quarantined').
2. The ID must be unique within the system, and must not have been used for any previous lexical entry. For this reason, every contributed ID is suffixed with the timestamp representing the time at which it was submitted (as the number of seconds since the 'Unix Epoch' – January 1 1970 00:00:00 GMT).

So for example, a new lexical ID for *mianmhor* might take the form:

```
qqq-mianmhor-1513988040
```

Secondly, a new XML file is added to the relevant database, for example:

gd/lexemes/qqq-miannmhor-1513988040.xml

This file will have the following standard format, following the schema discussed in section 1.2.2:

```
<lexeme>
  <form>
    <orth>miannmhor</orth>
  </form>
  <trans resp="#####@gmail.com">ambitious</trans>
  <note>Related forms: miannmhoir, miannmhoire</note>
  <note>Note: adjective</note>
  <note>Contributed by user #####@gmail.com at 1513988040.</note>
</lexeme>
```

Thirdly, a new entry is added to the cached target language index file, e.g. gd/cache/targetIndex.json:

```
{
  "target": "miannmhor",
  "id": "qqq-miannmhor-1513988040"
}
```

Fourthly, if the English term en already exists in the relevant cached English index file (e.g. gd/cache/englishIndex.json), then this entry is supplemented with the new Gaelic term:

```
{
  "en": "ambitious",
  "targets": [
    {"id": "àrd-amasach", "form": "àrd-amasach"},
    {"id": "qqq-miannmhor-1513988040", "form": "miannmhor"}
  ]
}
```

Otherwise a whole new entry is added:

```
{
  "en": "ambitious",
  "targets": [
    {"id": "qqq-miannmhor-1513988040", "form": "miannmhor"}
  ]
}
```

### 1.4.3. Adding a new comment to an existing entry

The Lexicopia system provides a static PHP class, `Comment`, to enable a user interface to allow users to add new lexicographic comments to entries in the database. This class has a single public static function, `addComment`, that can be called externally using the following instruction:

```
Comment::addComment($_POST, $path);
```

The second input parameter here, `$path`, specifies the location of the Lexicopia system on the relevant web server. Note that the value of `$path` must end in a / (or the appropriate operating system symbol to separate directory names).

The first input parameter `$_POST` is a collection of POST data in the following format:

```
"lang" => ". . ."
"id" => ". . ."
"comment" => ". . ."
"userEmail" => ". . ."
```

These fields denote the following things:

- `lang` – the ISO 639 language code, e.g. `gd`
- `id` – the lexical ID of the entry the comment is being added to
- `comment` – the content of the comment itself
- `userEmail` – the contributor's email address.

When a new comment is being processed, a new `note` element is appended to the relevant XML entry file, including the email of the contributor who added it, and the timestamp, for example:

```
<note>Masculine noun [#####@gmail.com:1513988040]</note>
```

### 1.4.4. Adding a new English equivalent to an entry

The Lexicopia system provides a static PHP class, `EnTrans`, to enable a user interface to allow users to add new English translational equivalents to entries in the database. This class has a public static function, `addEnTrans`, that can be called externally using the following instruction:

```
EnTrans::addEnTrans($_POST, $path);
```

The second input parameter here, `$path`, specifies the location of the Lexicopia system on the relevant web server. Note that the value of `$path` must end in a / (or the appropriate operating system symbol to separate directory names).

The first input parameter `$_POST` is a collection of POST data in the following format:

```
"lang" => "...
"id" => "...
"trans" => "...
"form" => "..."
```

```
"userEmail" => "..."
```

These fields denote the following things:

- `lang` – the ISO 639 language code, e.g. `gd`
- `id` – the lexical ID of the entry the comment is being added to
- `trans` – the English translational equivalent itself
- `form` – the target language headword form
- `userEmail` – the contributor's email address.

When a new English equivalent is being processed, a new `trans` element is appended to the relevant XML entry file, including the email of the contributor who added it as a `resp` attribute (see 1.2.2), for example:

```
<trans resp="#####@gmail.com">ambitious</trans>
```

Secondly, the new English equivalent is added to the cached English index file `englishIndex.json` file:

- If there is already an entry for the English term, then a new `id` is added as follows:

```
{
  "en": "ambitious",
  "targets": [
    {"id": "àrd-amasach", "form": "àrd-amasach"},
    {"id": "miannmhor", "form": "miannmhor"}
  ]
}
```

- If there is no entry for the English term, then a new one is added as follows:

```
{
  "en": "ambitious",
  "targets": [
    {"id": "miannmhor", "form": "miannmhor"}
  ]
}
```

### 1.4.5. Adding a new orthographic form to an entry

The Lexicopia system provides a static PHP class, `FormOrth`, to enable a user interface to allow users to add new orthographic forms to entries in the database. This class has a single public static function, `addFormOrth`, that can be called externally using the following instruction:

```
FormOrth::addFormOrth($_POST, $path);
```

The second input parameter here, `$path`, specifies the location of the Lexicopia system on the relevant web server. Note that the value of `$path` must end in a `/` (or the appropriate operating system symbol to separate directory names).

The first input parameter `$_POST` is a collection of POST data in the following format:

```
"lang" => ". . ."  
"id" => ". . ."  
"target" => ". . ."  
"userEmail" => ". . ."
```

These fields denote the following things:

- `lang` – the ISO 639 language code, e.g. `gd`
- `id` – the lexical ID of the entry the comment is being added to
- `target` – the orthographic form itself
- `userEmail` – the contributor's email address.

When a new comment is being processed, a new `form` element is appended to the relevant XML entry file, containing a new `orth` element, the latter including the email of the contributor who added it as a `resp` attribute (see 1.2.2), for example:

```
<form resp="#####@gmail.com">  
  <orth>miannmhoire</orth>  
</form>
```

Secondly, the new form is added to the cached target index file `targetIndex.json` file:

```
{  
  "target": "miannmhoire",  
  "id": "miannmhor"  
}
```

### 1.4.6. Authorising an English equivalent

The static PHP class `EnTrans`, discussed in 1.4.4 has another public static function, `authEnTrans`, which enables an external interface user to officially authorise an English equivalent in an entry. This function can be called externally using the following instruction:

```
EnTrans::authEnTrans($_POST, $path);
```

The second input parameter here, `$path`, specifies the location of the Lexicopia system on the relevant web server. Note that the value of `$path` must end in a `/` (or the appropriate operating system symbol to separate directory names).

The first input parameter `$_POST` is a collection of POST data in the following format:

```
"lang" => ". . ."  
"id" => ". . ."  
"trans" => ". . ."  
"userEmail" => ". . ."
```

These fields denote the following things:

- `lang` – the ISO 639 language code, e.g. `gd`
- `id` – the lexical ID of the entry the comment is being added to
- `trans` – the English translational equivalent itself

- `userEmail` – the editor's email address.

When an instruction to authorise an English equivalent is being processed, the relevant `trans` element in the correct XML entry file is located, and the email of the contributor who added it, and the timestamp, as a `resp` attribute, for example:

```
<trans resp="#####@gmail.com">ambitious</trans>
```



## 2. IMPLEMENTATION OF THE LEACAG INTERFACE

This chapter discusses the design and structure of the front-end to the LEACAG terminology system. The intended audience is systems developers who need to get the LEACAG system installed on a new server, or who have been tasked with updating the system and need to know how the front-end works. To a large extent, this chapter is technical documentation for the system, and hence is necessarily somewhat repetitive in places.

**Section 2.1** describes the basic structure of the LEACAG homepage, and its most important elements.

**Section 2.2** describes the many ways in which a user can interact with the LEACAG homepage, and how these link with the Lexicopia interaction discussed in section 1.4 above.

**Section 2.3** discusses the structure of the LEACAG user database, which keeps track of logged-in users and their activity.

## 2.1. Structure of the LEACAG homepage

The homepage for the LEACAG terminology system is located at:

<http://dasg.ac.uk/leacag>

When viewed in a web browser, this page should look something like this screenshot:

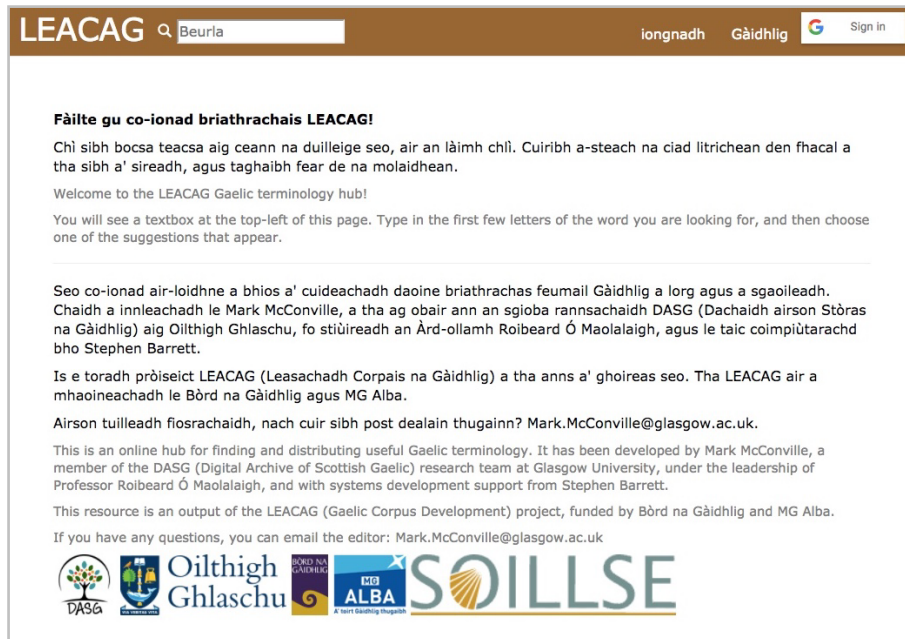


Figure 2.1: The LEACAG homepage

This page was created using the Bootstrap API for responsive, mobile-first projects,<sup>2</sup> and as such the page contents will re-scale, re-size and reorganise when viewed on different kinds of screen. For example, when viewed on an iPhone, this page will look more like this:

<sup>2</sup> <https://getbootstrap.com>

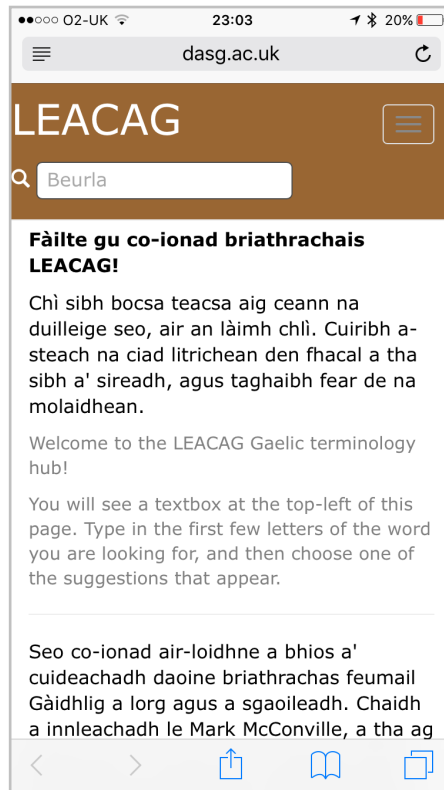


Figure 2.2: The LEACAG homepage on an iPhone

The homepage consists of two parts:

- the navigation bar along the top (brown background)
- the content pane underneath (white background).

### 2.1.1. The navigation bar

The most important parts of the navigation bar are on the left (immediately after the LEACAG logo):

`input#englishSearchField`

This search box is visible (and focussed) by default (with the word 'Beurla' as default background text). Users can enter English terms here that they wish to find Gaelic equivalents for.



Figure 2.3: `input#englishSearchField`

### `input#gaelicSearchField`

This search box is hidden by default, and only gets displayed when `li#enToGdToggle` has been clicked (see below). When visible (at the top left, with the word 'Gàidhlig' as default background text), users can enter Gaelic terms they wish to look up in the dictionary.

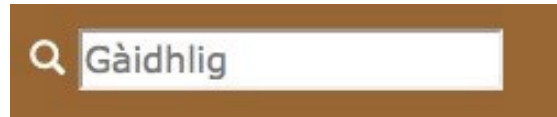


Figure 2.4: `input#gaelicSearchField`

Note that these two search fields are never visible at the same time. Rather, the user can toggle between them using the buttons described below.

On the right of the navigation bar is a series of links, in white text, which are underlined when the user hovers the cursor over them:

### `li#randomEntryLink`

This link is always visible, labelled 'iongnadh' (see Figure 2.1). When the user clicks on it, a random entry is selected from the lexical database and displayed in the content pane.

### `li#enToGdToggle`

This link is visible by default, labelled 'Gàidhlig' (see Figure 2.1). When the user clicks on it, `input#englishSearchField` is replaced with `input#gaelicSearchField` on the left of the navigation bar, and `li#enToGdToggle` is replaced with `li#gdToEnToggle` on the right.

### `li#gdToEnToggle`

This link is hidden by default, labelled 'Beurla'. When it has been made visible and the user clicks on it, `input#gaelicSearchField` is replaced with `input#englishSearchField` on the left of the navigation bar, and `li#gdToEnToggle` is replaced with `li#enToGdToggle` on the right.

In other words, the user can use the two toggle buttons to alternate between the two different states of the navigation bar, i.e. between English search and Gaelic search.



Figure 2.5: The two navigation bar states

## li#newEntryLink

This link is hidden by default, labelled 'moladh', and is only visible if the user has signed in using a Google account and this Google account is recognised by the system as belonging to an accredited Contributor. When clicked, it allows the Contributor to add a new entry to the lexical database.



Figure 2.6: The navigation bar for a signed-in user

At the very right hand side is a button which allows the user to sign in or out of the system using their Google account. If the user is not logged in, the button reads 'Sign In', otherwise it reads 'Sign Out'.

## 2.1.2. The content pane

The two most important parts of the rest of the page are as follows:

### div#homePageText

This section of text is visible by default, displaying some description of the resource, as well as logos at the bottom.

**Fàilte gu co-ionad briathrachais LEACAG!**

Chì sibh bocsa teacsa aig ceann na duilleige seo, air an làimh chli. Cuiribh a-steach na ciad litrichean den fhacal a tha sibh a' sireadh, agus taghaibh fear de na molaidhean.

Welcome to the LEACAG Gaelic terminology hub!

You will see a textbox at the top-left of this page. Type in the first few letters of the word you are looking for, and then choose one of the suggestions that appear.

---

Seo co-ionad air-loidhne a bhios a' cuideachadh daoine briathrachas feumail Gàidhlig a lorg agus a sgoileadh. Chaidh a innleachadh le Mark McConville, a tha ag obair ann an sgioba rannsachaidh DASG (Dachaidh airson Stòras na Gàidhlig) aig Oilthigh Ghlaschu, fo stiùireadh an Àrd-ollamh Roibeard Ó Maolalaigh, agus le taic coimpiùtarachd bho Stephen Barrett.

Is e toradh pròiseict LEACAG (Leasachadh Corpais na Gàidhlig) a tha anns a' ghoireas seo. Tha LEACAG air a mhaoinèachadh le Bòrd na Gàidhlig agus MG Alba.

Airson tuilleadh fiosrachaidh, nach cuir sibh post dealain thugainn? [Mark.McConville@glasgow.ac.uk](mailto:Mark.McConville@glasgow.ac.uk).

This is an online hub for finding and distributing useful Gaelic terminology. It has been developed by Mark McConville, a member of the DASG (Digital Archive of Scottish Gaelic) research team at Glasgow University, under the leadership of Professor Roibeard Ó Maolalaigh, and with systems development support from Stephen Barrett.

This resource is an output of the LEACAG (Gaelic Corpus Development) project, funded by Bòrd na Gàidhlig and MG Alba.

If you have any questions, you can email the editor: [Mark.McConville@glasgow.ac.uk](mailto:Mark.McConville@glasgow.ac.uk)



Figure 2.7: div#homePageText

## div#lexicalText

This section of text is empty by default. When a lexical entry is loaded from the database, it is displayed here. This section alternates with `div#homePageText`, i.e. either one of them is always visible, but never are both visible at the same time.

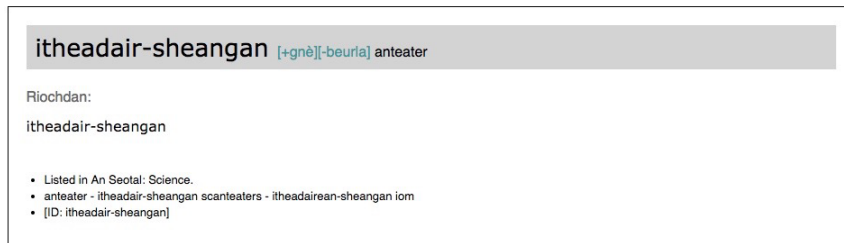


Figure 2.8: An example of `div#lexicalText`

There are a few other important parts of the content pane that are hidden by default and only displayed under certain conditions:

## ul#suggestionsDropDown

This drop-down list is both empty and hidden by default. When three or more characters are entered in either `input#englishSearchField` or `input#gaelicSearchField`, it is populated by every entry in the database which starts with the relevant string, and then made visible, so that the user can select one entry for display, or continue to enter more letters.



Figure 2.9: `ul#suggestionsDropDown`

### **span#noResultsMessage**

This text ('Chan eil toradh ann don cheist seo') is hidden by default. It is only displayed when the characters entered in either `input#englishSearchField` or `input#gaelicSearchField` fail to match any entry in the database.



Figure 2.10: `span#noResultsMessage`

### **div#gaelicEquivalentsList**

This is hidden by default. It is only made visible if the user searches for an English term for which the database contains more than one Gaelic equivalent (e.g. 'mercury'), in which case it lists each of these for the user to click on (after the preamble 'Faclan Gàidhlig airson ...').



Figure 2.11: `div#gaelicEquivalentsList`

### **div.loggedInStatusMessage**

This is hidden if the user is not signed in, but if she is signed in, it displays the text 'Air a chlàradh a-steach mar ...' plus the user's name (and possibly her role, if she is an admin).



Figure 2.12: `div.loggedInStatusMessage`

In addition, there are four links at the bottom of the page that are hidden by default and only displayed if: (a) `div#lexicalText` is currently displaying a lexical entry; (b) the user is logged in; and (c) the user has the requisite authority (i.e. Contributor or Editor):

### **a#addCommentLink**

This link, labelled ‘Cuir iomradh ris a’ bhriathar seo’, is only displayed if the system recognises the user’s account as an accredited Contributor or Editor. When the link is clicked, `form#addCommentForm` is made visible, allowing the user to add a lexicographic comment to the entry currently being displayed.

### **a#addEnglishLink**

This link, labelled ‘Cuir Beurla ris a’ bhriathar seo’, is only displayed if the system recognises the user’s account as an accredited Contributor or Editor. When the link is clicked, `form#addEnglishForm` is made visible, allowing the user to add a new English equivalent term to the entry currently being displayed.

### **a#addFormOrthLink**

This link, labelled ‘Cuir foirm ris a’ bhriathar seo’, is only displayed if the system recognises the user’s account as an accredited Contributor or Editor. When the link is clicked, `form#addFormOrthForm` is made visible, allowing the user to add a new orthographic form to the entry currently being displayed.

### **a#authEnglishLink**

This link, labelled ‘Ùghdarraich Beurla anns a’ bhriathar seo’, is only displayed if the system recognises the user’s account as an accredited Editor. When the link is clicked, `form#authEnglishForm` is made visible, allowing the user to authorise an English equivalent term in the entry currently being displayed.

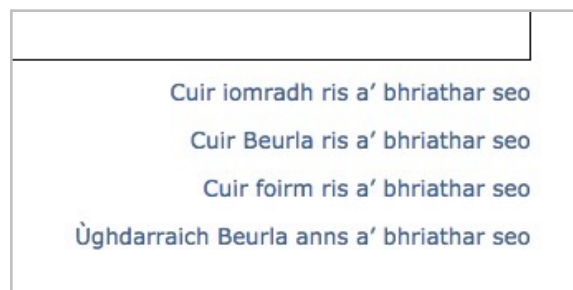


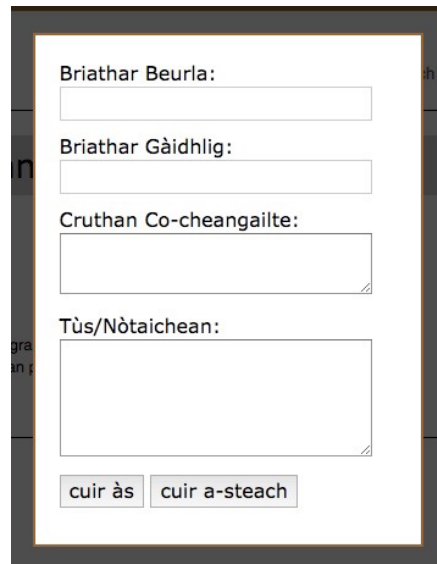
Figure 2.13: Contributor and Editor links



There are also five forms for users to send information to the server. These are all hidden by default, and are only displayed as a result of an approved user selecting some course of action:

#### **form#newEntryForm**

This form is hidden by default. It is only displayed if the user clicks `li#newEntryLink`. Contributors can use this form to add a new entry to the database.



The screenshot shows a form with the following elements:

- Briathar Beurla:** A single-line text input field.
- Briathar Gàidhlig:** A single-line text input field.
- Cruthan Co-cheangailte:** A multi-line text input field.
- Tùs/Nòtaichean:** A large multi-line text input field.
- Two buttons at the bottom: **cuir às** and **cuir a-steach**.

Figure 2.14: form#newEntryForm

#### **form#addCommentForm**

This form is hidden by default, and is only made visible when `a#addCommentLink` is clicked. Contributors can use this form to add a comment to the current lexical entry being displayed.



The screenshot shows a form with the following elements:

- Iomradh:** A single-line text input field.
- Two buttons at the bottom: **cuir às** and **cuir a-steach**.

Figure 2.15: form#addCommentForm

### **form#addEnglishForm**

This form is hidden by default, and is only made visible when `a#addEnglishLink` is clicked. Contributors can use this form to add an English equivalent term to the current lexical entry being displayed.

A screenshot of a web form titled "Beurla:". It features a large empty text input field with a small cursor icon at the bottom right. Below the input field are two buttons: "cuir às" and "cuir a-steach". The form is displayed within a dark grey frame, and the word "Cuir" is partially visible at the bottom right corner of the frame.

Figure 2.16: `form#addEnglishForm`

### **form#addFormOrthForm**

This form is hidden by default, and is only made visible when `a#addFormOrthLink` is clicked. Contributors can use this form to add an additional orthographic form to the current lexical entry being displayed.

A screenshot of a web form titled "Foirm:". It features a large empty text input field with a small cursor icon at the bottom right. Below the input field are two buttons: "cuir às" and "cuir a-steach". The form is displayed within a dark grey frame, and the word "Cuir" is partially visible at the bottom right corner of the frame.

Figure 2.17: `form#addFormOrthForm`

### **form#authEnglishForm**

This form is hidden by default, and is only made visible when `a#authEnglishLink` is clicked. Editors can use this form to authorise an English equivalent term in the current lexical entry being displayed.

A screenshot of a web form titled "Beurla ùghdarraichte:". It features a large empty text input field. Below the input field are two buttons: "cuir às" and "cuir a-steach". The entire form is enclosed in a dark grey border. A small "Cuir" label is visible in the bottom right corner of the border.

Figure 2.18: `form#authEnglishForm`

### **div#submitThanksPopUp**

This text ('Mòran taing!' plus a 'dùin' button) is hidden by default. It is only displayed when **one of** `form#newEntryForm`, `form#addCommentForm`, `form#addEnglishForm`, `form#addFormOrthForm` **or** `form#authEnglishForm` **has been submitted.**

A screenshot of a pop-up message box. It contains the text "Mòran taing!" in a large, bold font. Below the text is a single button labeled "dùin". The message box is enclosed in a dark grey border.

Figure 2.19: `div#submitThanksPopUp`

## 2.2. Interacting with the LEACAG homepage

The user can interact with the LEACAG homepage in the following ways (depending on the level of their accreditation):

- clicking on the LEACAG logo
- clicking on `li#enToGdToggle`
- clicking on `li#gdToEnToggle`
- clicking on `li#randomEntryLink`
- entering a character into `input#englishSearchField`
- entering a character into `input#gaelicSearchField`
- clicking on an item in `ul#suggestionsDropDown`
- clicking on `li#newEntryLink`
- submitting `form#newEntryForm`
- clicking on `a#addCommentLink`
- submitting `form#addCommentForm`
- clicking on `a#addEnglishLink`
- submitting `form#addEnglishForm`
- clicking on `a#addFormOrthLink`
- submitting `form#addFormOrthForm`
- clicking on `a#authEnglishLink`
- submitting `form#authEnglishForm`

Each of these interactions is dealt with in the following sections.

### 2.2.1. Clicking on the LEACAG logo

When the user clicks on the LEACAG logo at the top-left of the homepage, the page is reloaded in its default state, as in Figure 2.1 (though signed-in users are not signed out). In other words, the content pane is rest to its default state.

### 2.2.2. Clicking on `li#enToGdToggle`

When the user clicks on the button labelled 'Gàidhlig' on the right of the navigation bar (i.e. `li#enToGdToggle`, see Figure 2.5), a JQuery event handler is called, which:

- hides the English search box `input#englishSearchField` and the Gaelic search toggle button `li#enToGdToggle`

- shows the Gaelic search box `input#gaelicSearchField` and the English search toggle button `li#gdToEnToggle`, and focuses the former
- resets the content pane to its default state (i.e. Figure 2.7)

### 2.2.3. Clicking on `li#gdToEnToggle`

When the user clicks on the button labelled 'Beurla' on the right of the navigation bar (i.e. `li#gdToEnToggle`, see Figure 2.5), a JQuery event handler is called, which:

- hides the Gaelic search box `input#gaelicSearchField` and the English search toggle button `li#gdToEnToggle`
- shows the English search box `input#englishSearchField` and the Gaelic search toggle button `li#enToGdToggle`, and focuses the former
- resets the content pane to its default state.

### 2.2.4. Clicking on `li#randomEntryLink`

When the user clicks on the button labelled 'iongnadh' on the right of the navigation bar (i.e. `li#randomEntryLink`, see Figure 2.6), a JQuery event handler is called, which:

- resets the content pane to its default state
- makes an AJAX call to a PHP script, which chooses a random entry from the cached Lexicopia Gaelic index `gd/cache/targetIndex.json` (cf. 1.3.1), and then returns it in the JSON format `{"id": "..."}`
- calls the JavaScript function `updateContent(id)` with this ID, retrieving and displaying the relevant lexical entry in the content pane (i.e. `div#lexicalText`), and displaying the various Contributor and Editor links at the bottom where appropriate (i.e. `a#addCommentLink`, `a#addEnglishLink`, `a#addFormOrthLink`, `a#authEnglishLink`).

### 2.2.5. Entering a character into `input#englishSearchField`

When the user enters a character into the English search box `input#englishSearchField`, a JQuery event handler is called, which waits until three characters have been entered and then makes an AJAX call to a PHP script. This script finds every entry in the Lexicopia English cache file `gd/cache/englishIndex.json` (cf. 1.3.2) which starts with the relevant characters, and then returns them as an array of JSON objects `{"id": "...", "value": "...", "label": "...", "item": "..."}`, where:

- `id` is the first target, e.g. `{"id": "mearcair", "form": "mearcair"}`
- `value` and `label` are the English term, e.g. `mercury`
- `item` is the whole entry, e.g. `{"en": "mercury", "targets": [{"id": "mearcair", "form": "mearcair"}, {"id": "airgead_beò", "form": "airgead beò"}]}`

This information is then used by the JQuery UI autocomplete event handler to display the search results in `ul#suggestionsDropDown`, as in Figure 2.9. If no search results are returned, then the relevant message (i.e. `span#noResultsMessage`) is displayed as in Figure 2.10.

## 2.2.6. Entering a character into `input#gaelicSearchField`

When the user enters a character into the Gaelic search box `input#gaelicSearchField`, a JQuery event handler is called, which waits until three characters have been entered and then makes an AJAX call to a PHP script. This script finds every entry in the Lexicopia Gaelic cache file `gd/cache/targetIndex.json` (cf. 1.3.1) which starts with the relevant characters, and then returns them as an array of JSON objects `{"id": "...", "value": "...", "label": "...", "item": "..."}` , where:

- `id` is the lexical ID, e.g. `airgead_beò`
- `value` and `label` are the Gaelic term, e.g. `airgead beò`
- `item` is the whole entry, e.g. `{"target":"airgead beò", "id":"airgead_beò", "en":"mercury"}`

This information is then used by the JQuery UI autocomplete event handler to display the search results in `ul#suggestionsDropDown`, as in Figure 2.9. If no search results are returned, then the relevant message (i.e. `span#noResultsMessage`) is displayed as in Figure 2.10.

## 2.2.7. Clicking on an item in `ul#suggestionsDropDown`

When the user clicks on one of the suggested items in the search results drop-down list `ul#suggestionsDropDown` (see Figure 2.9), the JQuery UI autocomplete event handler calls the JavaScript function `chooseSelectedTerm`. This function works differently depending on whether a Gaelic or an English search is being conducted.

If a Gaelic search is being conducted (via `input#gaelicSearchField`), the JavaScript function `updateContent(id)` is called with the appropriate lexical ID, retrieving and displaying the relevant lexical entry in the content pane (i.e. `div#lexicalText`), and displaying the various Contributor and Editor links at the bottom where appropriate (i.e. `a#addCommentLink`, `a#addEnglishLink`, `a#addFormOrthLink`, `a#authEnglishLink`).

If an English search is being conducted (via `input#gaelicSearchField`), there are two possible courses of action:

- If there is only one search result, the JavaScript function `updateContent(id)` is called with the appropriate lexical ID, as described above.
- If there is more than one search result, then these are displayed in the Gaelic equivalents list (i.e. `div#gaelicEquivalentsList`), as in Figure 2.11. When one of these equivalents is clicked, the associated event handler calls `updateContent(id)` with the appropriate lexical ID, as described above.

### 2.2.8. Clicking on `li#newEntryLink`

When a Contributor clicks on the button labelled ‘moladh’ on the right of the navigation bar (i.e. `li#newEntryLink`, see Figure 2.6), a JQuery event handler is called which displays the form for adding new terms to the system, i.e. `form#newEntryForm` (see Figure 2.14).

### 2.2.9. Submitting `form#newEntryForm`

Once a Contributor has completed the form for adding new entries (`form#newEntryForm`, Figure 2.14) and clicks ‘cuir a-steach’, the relevant event handler clears and hides the form, displays the ‘thank you message’ (`div#submitThanksPopUp`, Figure 2.19), and calls a PHP script to process the form data. This PHP script calls the Lexicopia PHP function `NewEntry::addEntry`, discussed in 1.4.2, adding the new term to the lexical database and to the cached indexes as well.

### 2.2.10. Clicking on `a#addCommentLink`

When a Contributor clicks on the link labelled ‘Cuir iomradh ris a’ bhriathar seo’ at the bottom of a lexical entry (i.e. `a#addCommentLink`, Figure 2.13), a JQuery event handler is called which displays the form for adding comments to entries, i.e. `form#addCommentForm` (Figure 2.15).

### 2.2.11. Submitting `form#addCommentForm`

Once a Contributor has completed the form for adding comments to an entry (`form#addCommentForm`, Figure 2.15) and clicks ‘cuir a-steach’, the relevant event handler clears and hides the form, displays the ‘thank you message’ (`div#submitThanksPopUp`, Figure 2.19), and calls a PHP script to process the form data. This PHP script calls the Lexicopia PHP function `Comment::addComment`, discussed in 1.4.3, adding the comment to the relevant entry in the lexical database.

### 2.2.12. Clicking on `a#addEnglishLink`

When a Contributor clicks on the link labelled ‘Cuir Beurla ris a’ bhriathar seo’ at the bottom of a lexical entry (i.e. `a#addEnglishLink`, Figure 2.13), a JQuery event handler is called which displays the form for adding new English equivalents to entries, i.e. `form#addEnglishForm` (Figure 2.16).

### 2.2.13. Submitting `form#addEnglishForm`

Once a Contributor has completed the form for adding new English equivalents to an entry (`form#addEnglishForm`, Figure 2.16) and clicks ‘cuir a-steach’, the relevant event handler clears and hides the form, displays the ‘thank you message’ (`div#submitThanksPopUp`, Figure 2.19), and calls a PHP script to process the form data. This PHP script calls the Lexicopia PHP function `EnTrans::addEnTrans`, discussed in 1.4.4, adding the English translation to the relevant entry in the lexical database.

### **2.2.14. Clicking on `a#addFormGraphLink`**

When a Contributor clicks on the link labelled ‘Cuir Beurla ris a’ bhriathar seo’ at the bottom of a lexical entry (i.e. `a#addFormOrthLink`, Figure 2.13), a JQuery event handler is called which displays the form for adding new orthographic forms to entries, i.e. `form#addFormOrthForm` (Figure 2.17).

### **2.2.15. Submitting `form#addFormGraphForm`**

Once a Contributor has completed the form for adding new orthographic forms to an entry (`form#addFormOrthForm`, Figure 2.17) and clicks ‘cuir a-steach’, the relevant event handler clears and hides the form, displays the ‘thank you message’ (`div#submitThanksPopUp`, Figure 2.19), and calls a PHP script to process the form data. This PHP script calls the Lexicopia PHP function `FormOrth::addFormOrth`, discussed in 1.4.5, adding the new orthographic form to the relevant entry in the lexical database.

### **2.2.16. Clicking on `a#authEnglishLink`**

When an Editor clicks on the link labelled ‘Ùghdarrach Beurla anns a’ bhriathar seo’ at the bottom of a lexical entry (i.e. `a#authEnglishLink`, see Figure 2.13), a JQuery event handler is called which displays the form for authorising English equivalents to entries, i.e. `form#authEnglishForm` (Figure 2.18).

### **2.2.17. Submitting `form#authEnglishForm`**

Once an Editor has completed the form for authorising English equivalents to an entry (`form#authEnglishForm`, see Figure 2.18) and clicks ‘cuir a-steach’, the relevant event handler clears and hides the form, displays the ‘thank you message’ (`div#submitThanksPopUp`, Figure 2.19), and calls a PHP script to process the form data. This PHP script calls the Lexicopia PHP function `EnTrans::authEnTrans`, discussed in 1.4.6, authorising the English translation to the relevant entry in the lexical database.



## 2.3. The LEACAG user database

The LEACAG system includes a MySQL database which keeps track of registered users and their activity. This database can be used by the Administrators to generate reports about activity within the system within particular time periods, as well as to control accreditation of Contributors and Editors.

The database has three tables:

- `leacag_user`
- `leacag_userActivity`
- `leacag_formSubmission`

Each of these tables will be discussed in the following sections.

### 2.3.1. `leacag_user`

The `leacag_user` table records five pieces of information about every user who has logged in to the system using their Google accounts:

- `email` – the email address associated with the Google account
- `name` – the name associated with the Google account
- `accessLevel` – the accreditation level of the user, i.e. Basic User (1), Contributor (2), Editor (3), Administrator (5)
- `firstLogin` – the date and time the user first logged in to the system
- `lastLogin` – the date and time the user last logged in to the system

When a user first logs into the system using their Google account, a new entry is made in the `leacag_user` table. Thereafter, every time the user logs in, the `lastLogin` field is updated. When deciding whether to display content restricted to Contributors and Editors, the LEACAG system consults this table to ascertain whether the user has the authority to see the content.

The Administrators have the ability to promote Basic Users and Contributors and to demote Contributors and Editors.

### 2.3.2. `leacag_userActivity`

The `leacag_userActivity` table logs all searches conducted by the LEACAG system. Each time a user selects a search result from the drop-down list (`ul#suggestionsDropDown`), a new entry is added to this table with the following information:

- `googleId` – the Google ID number of the user conducting the search
- `email` – the email address associated with this Google ID

- `searchTerm` – the term searched for
- `failed` – whether the search failed or not (`success = 0`, `failure = 1`)
- `language` – whether the search was for an English term (`en`) or a Gaelic one (`gd`)
- `timestamp` – the date and time of the search

It is particularly interesting to keep track of the failed searches, in order to ascertain demand for particular English terms in Gaelic.

### **2.3.3. `leacag_formSubmission`**

The `leacag_formSubmission` table logs the activity of Contributors and Editors. Every time one of the forms is submitted (i.e. `#newEntryForm`, `#addCommentForm`, `#addEnglishForm`, `#addFormOrthForm`, `#authEnglishForm`), an entry is added to this table with the following information:

- `email` – the email address of the Contributor or Editor submitting the form
- `timestamp` – the date and time the form was submitted
- `type` – the type of form being submitted (i.e. `newEntry`, `addComment`, `addEnglish`, `addFormOrth`, `authEnglish`)
- `content` – the content of the form being submitted

# 3. USING THE LEACAG INTERFACE

This chapter contains some basic user guidance for the LEACAG terminology interface, and is thus intended for a general audience.

The LEACAG terminology hub is accessible at the following URL:

<http://dasg.ac.uk/leacag>

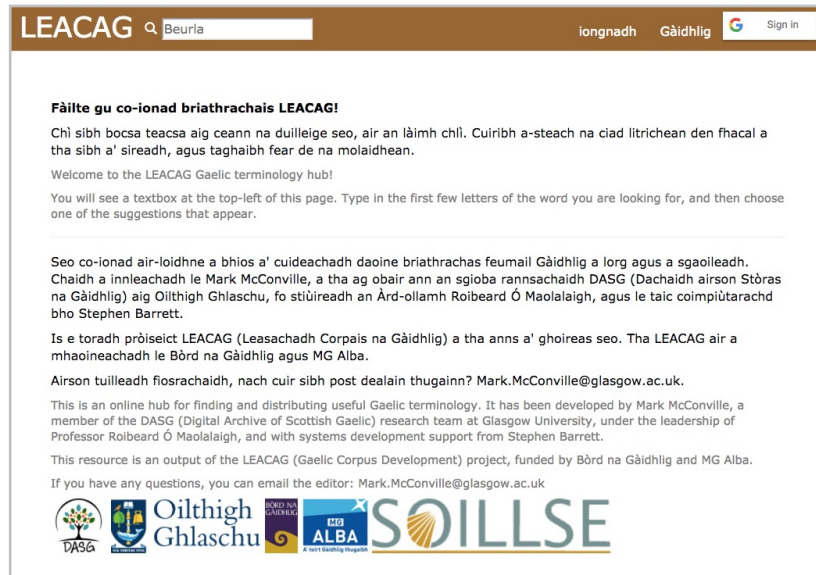


Figure 3.1: The LEACAG terminology homepage

## 3.1. Signing in via Google

At the top right of the LEACAG homepage, you should see a Google button saying "Sign In". Anyone who has a Google account can sign in to the LEACAG system. If you are already signed in to your Google account in another browser tab you may well have been signed in to the LEACAG system automatically and the button will say "Sign Out".

If you don't have a Google account, you can register for one here:

<https://accounts.google.com/SignUp?hl=en>

## 3.2. Searching for an English term

All users of the LEACAG system, whether signed in or not, can search for an English term.

At the top left of the LEACAG homepage, you should see a search box the background text 'Beurla'. You can use this search box to search for English terms in the LEACAG lexical database. For instance, if you are looking for the English term 'sodium chloride', just type the first three letters 'sod' into this search box and then select the relevant term from the drop-down list that then appears. The lexical entry for the Gaelic term 'sodium clòraid' should then appear in the main part of the page.

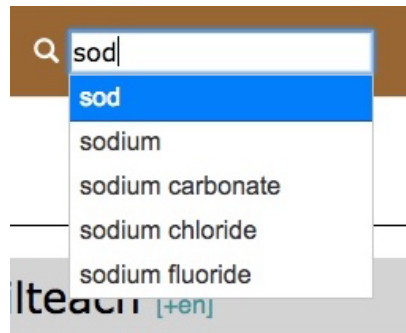


Figure 3.2: Conducting an English search

Sometimes a given English term will have more than one Gaelic equivalent, for example 'mercury'. In this case, a list of all the relevant Gaelic terms will be displayed, for the user to select in turn.



Figure 3.3: Multiple search results for an English term

If there is no suitable Gaelic term in the database, an error message is displayed.



Figure 3.4: Search failure error message

### 3.3. Searching for a Gaelic term

All users of the LEACAG system, whether logged in or not, can search for a Gaelic term.

At the top right of the LEACAG homepage, you should see a button labelled 'Gàidhlig'. This button is used to toggle between English and Gaelic search modes. If you click on this button, you should see that the background text in the search box changes from 'Beurla' to 'Gàidhlig' signalling that it is now expecting Gaelic terms to be searched for.

For instance, if you want to find out whether the Gaelic term 'làmh-choille' is in the lexical database. Just type the first few letters 'lamh' into the Gaelic search box (you can ignore accents) and all Gaelic terms starting with these four letters will appear in a drop down list. You can then select the relevant one in order to display the lexical entry.

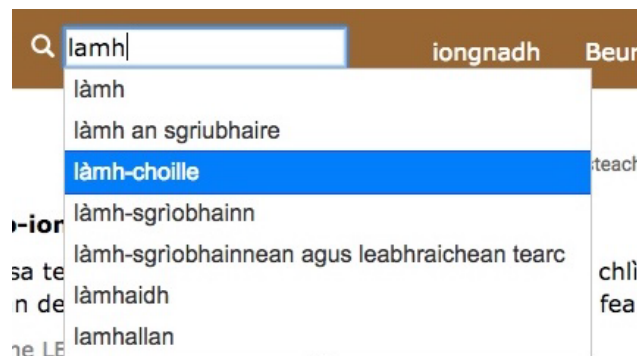


Figure 3.5: Conducting a Gaelic search

If there is no suitable Gaelic term in the database, the same error message is displayed as with the English search.

### 3.4. Looking up a random entry

All users of the LEACAG system, whether logged in or not, can look up a random Gaelic term.

At the top right of the LEACAG homepage, you should see a button labelled 'iongnadh'. Every time you click this button, a random entry from the lexical database is displayed.

### 3.5. Contributing a new entry – Contributors

Certain users of the LEACAG system can be officially authorised as ‘Contributors’, meaning that they can add information to the lexical database, in particular new lexical entries.

If you are a Contributor, you will be able to see an extra button up at the top right of the LEACAG homepage, labelled ‘moladh’. If you click this button a web form will appear, enabling you to add a new entry to the database. (You should have checked beforehand that the Gaelic term you want to add is not already in the database.)

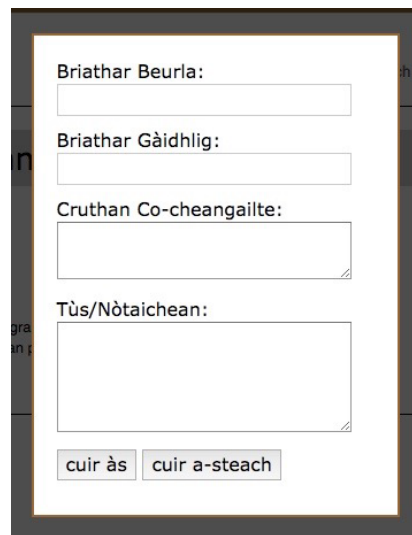
The image shows a web form for adding a new entry. It contains four text input fields: 'Briathar Beurla:', 'Briathar Gàidhlig:', 'Cruthan Co-cheangailte:', and 'Tùs/Nòtaichean:'. At the bottom, there are two buttons: 'cuir às' and 'cuir a-steach'.

Figure 3.6: Adding a new entry

You can enter the following information in this form:

- the English term, e.g. ‘airgun’
- the proposed Gaelic term, e.g. ‘gunna adhair’
- any morphologically related forms, e.g. ‘gunnaichean adhair (plural)’
- the source where you found the Gaelic term, e.g. ‘Colin Mark’s dictionary, p. 350’, along with any other relevant notes about the term, e.g. ‘masculine gender’.

Once you have completed the form, you can click the ‘Cuir a-steach’ button to add the new term to the lexical database.

### 3.6. Contributing a comment to an existing entry

Contributors can also add lexicographic comments to existing lexical entries.

If you are a Contributor, and you are looking at a lexical entry in the system, you will be able to see a link at the bottom right of the LEACAG homepage, labelled 'Cuir iomradh ris a' bhriathar seo'. If you click on this link, a web form will appear, enabling you to leave a comment (e.g. 'This word is often heard in Lewis', or 'I heard this expression on Radio nan Gàidheal this morning').

A screenshot of a web form titled "Iomradh:". The form has a white background and a dark border. At the top, the title "Iomradh:" is displayed in a bold, black font. Below the title is a large, empty text input field with a thin border and a small cursor icon at the bottom right. Underneath the input field are two buttons: "cuir às" on the left and "cuir a-steach" on the right, both in a light gray box with black text. In the bottom right corner of the form, the text "Cuir i" is partially visible.

Figure 3.7: Adding a comment to an entry

Once you have completed the form, you can click the 'Cuir a-steach' button to add the new comment to the relevant entry in the lexical database.

### 3.7. Contributing a new English translation to an existing entry

Contributors can also add English translations to existing lexical entries.

If you are a Contributor, and you are looking at a lexical entry in the system, you will be able to see a link at the bottom right of the LEACAG homepage, labelled 'Cuir Beurla ris a' bhriathar seo'. If you click on this link, a web form will appear, enabling you to add a new English translational equivalent for a Gaelic term.

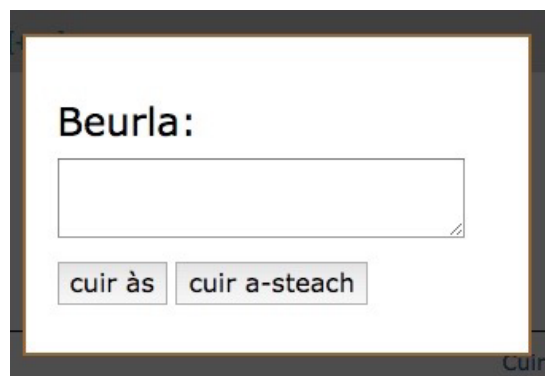
A screenshot of a web form titled "Beurla:". The form has a white background and a dark border. At the top, the title "Beurla:" is displayed in a bold, black font. Below the title is a large, empty text input field with a thin border and a small cursor icon at the bottom right. Underneath the input field are two buttons: "cuir às" on the left and "cuir a-steach" on the right, both in a light gray box with black text. In the bottom right corner of the form, the text "Cuir i" is partially visible.

Figure 3.8: Adding a new translation

Once you have completed the form, you can click the 'Cuir a-steach' button to add the new English translation to the relevant entry in the lexical database.

### 3.8. Contributing a new orthographic form to an existing entry

Contributors can also add orthographic forms to existing lexical entries.

If you are a Contributor, and you are looking at a lexical entry in the system, you will be able to see a link at the bottom right of the LEACAG homepage, labelled 'Cuir foirm ris a' bhriathar seo'. If you click on this link, a web form will appear, enabling you to add a new orthographic form. Examples of this might be adding plural forms, or genitive singular forms.

The image shows a screenshot of a web form. At the top, the word 'Foirm:' is displayed in a bold, black font. Below this is a rectangular text input field with a thin border. Underneath the input field are two buttons: 'cuir às' on the left and 'cuir a-steach' on the right. The buttons have a light grey background and a thin border. The entire form is enclosed in a dark grey border. In the bottom right corner of the form area, the text 'Cuir k' is partially visible.

Figure 3.9: Adding a new form


Once you have completed the form, you can click the 'Cuir a-steach' button to add the new orthographic form to the relevant entry in the lexical database.



### 3.9. Authorising an English translation in an existing entry – Editors

Certain users of the LEACAG system can be officially authorised as ‘Editors’, meaning that they can authorise information in the lexical database, in particular English translations of Gaelic terms.

If you are an Editor, and you are looking at a lexical entry in the system, you will be able to see a link at the bottom right of the LEACAG homepage, labelled ‘Ùghdarraich Beurla anns a’ bhriathar seo’. If you click on this link, a web form will appear, enabling you to authorise



The image shows a screenshot of a web form. At the top, the text 'Beurla ùghdarraichte:' is displayed. Below this is a rectangular text input field. Underneath the input field are two buttons: 'cuir às' on the left and 'cuir a-steach' on the right. The entire form is enclosed in a dark border.

English translational equivalent for a Gaelic term.

Figure 3.10: Authorising an English translation

Once you have completed the form, you can click the ‘Cuir a-steach’ button to authorise the English translation in the relevant entry in the lexical database.

## 4. FUTURE CHALLENGES

The aim of this chapter is to raise some issues that remain to be resolved along with a proposed direction for future development. We also look at potential links and relations with both the LearnGaelic and Faclair na Gàidhlig dictionary projects.

### 4.1. Background

There are three aspects to the background of this part of the project that are worth considering: (a) the 2014 *Dlùth is Inneach* report; (b) the originally specified aims of the LEACAG project; and (c) the fragmented state of Gaelic lexicographic infrastructure as things stand.

#### 4.1.1. *Dlùth is Inneach* (CR12-03)

The 2013 *Dlùth is Inneach* public consultation project (Bell et al 2014) reached the following conclusions regarding Gaelic lexicographic and terminological resources:

- “People feel that existing provision of Gaelic language resources is fragmented and lacks coordination. A clear desire was expressed for an **online ‘one-stop-shop’** for Gaelic corpus resources: a single website where people can go to get instant, authoritative, trustworthy, detailed advice on advanced lexical and grammatical usage.”
- “People want **more informative dictionaries**, with much more systematic explanations of distinctions relating to shades of meaning, context, dialects, colloquial versus formal usage, and contemporary versus archaic usage.”
- “There is a need for **greater consistency in new Gaelic terminology**. Where there is obvious, reasonable demand for a new item of Gaelic terminology, professional users have a clear preference for there to be a single agreed Gaelic term rather than a range of competing synonyms created by different organisations.”

#### 4.1.2. LEACAG (CR15-02)

The *Dlùth is Inneach* conclusions were part of the process that fed into the creation of CCC and the decision to fund the current project.

The originally specified aims of LEACAG include the following:

- “Development of an **online space for managing, evaluating and disseminating new Gaelic terminology**.”
- “Inconsistencies in new terminology will be tackled through an interactive online space allowing real-time **updating by approved contributors**. The forms preferred by BSC [i.e. CCC] will be indicated in this public resource made available by creative commons.”

### 4.1.3. Overview of the current Gaelic lexicographic infrastructure

Gaelic users are certainly justified in considering the current Gaelic lexicographic infrastructure as 'fragmented' and 'uncoordinated'.

Firstly, there are a range of public online databases maintained by agencies with responsibility for some domain of Gaelic development:

- Stòrlann (An Seotal)
- Ainmean-Àite na h-Alba
- Scottish Natural Heritage (Faclan Nàdair)
- Historic Environment Scotland (Gaelic Thesaurus)
- Sabhal Mòr Ostaig (Stòr-dàta Briathrachais Gàidhlig)

Secondly, some agencies don't have databases but rather publish their glossaries online as static PDFs:

- Education Scotland (Gaelic terminology summary)
- Bòrd na Gàidhlig (Faclair Rianachd Phoblaich)
- MNE Media (Guidelines for terminology in Sports programmes)

Thirdly, there is also a range of important private initiatives collated or managed by individual lexicographers:

- Michael Bauer's *Am Faclair Beag*
- Roy Wentworth's *Faclair Bun-tomhasach, Faclair Cànanach*
- etc. etc.

Finally, there is a range of dictionaries available only in printed format, in particular Boyd Robertson and Ian MacDonald's *Teach Yourself Gaelic Dictionary*, Colin Mark's *Gaelic–English Dictionary*, etc.

There is no means of central co-ordination for these resources. In order to find all information about a particular word, a user would have to consult each of these resources in turn. In addition, there is no means for overall quality control, for example to authorise particular items of official terminology as having cross-agency validity (should this be considered desirable).

## 4.2. Developing the LEACAG system

### 4.2.1. The ‘one-stop-shop’ infrastructure

In its very simplest conception, an online one-stop-shop for Gaelic lexicography would involve two webpages:

- **English search:** A webpage where users can enter an English term into a search box, and as a result every Gaelic equivalent will be displayed, along with metadata about where the Gaelic term comes from, which agencies recommend it, whether it is to be regarded as official cross-agency usage (‘kite-marked’), and so on.
- **Gaelic search:** A webpage where users can enter a Gaelic term into a search box, and as a result every piece of lexicographic information about that term will be displayed, along with metadata about where each piece of information comes from, whether that piece of information is to be considered ‘official’, and so on.

Gaelic users can then use these two webpages to find all the relevant information about the words they are looking for, including metadata about the source and authority of this information, and thereby make informed linguistic choices. Searches could even be filtered to only return information from pre-selected sources, e.g. CCC-approved terminology.

In order to function properly, these two search pages will need to be linked up to the different lexicographic databases managed by the different agencies in some way, as in Figure 4.1.

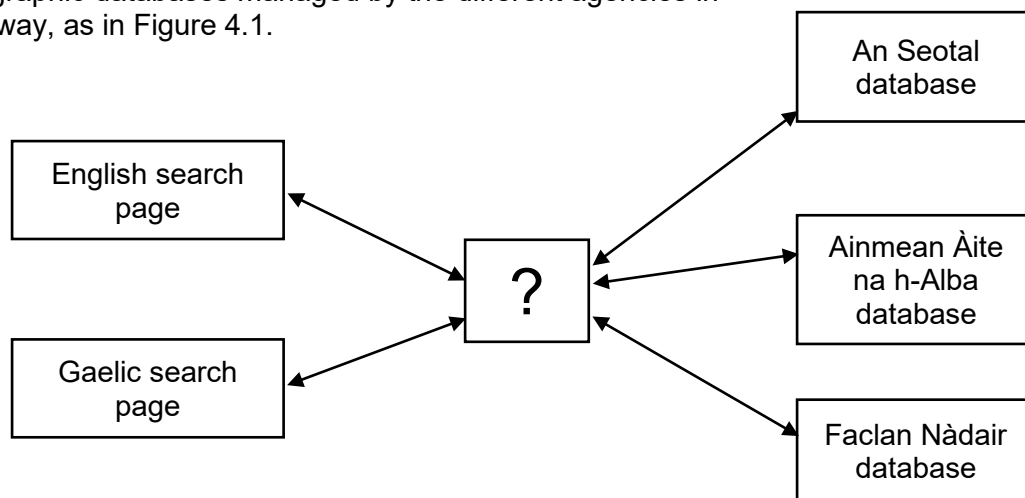


Figure 4.1: One-stop-shop basic model

The box containing the question mark in Figure 4.1 represents the technical problem that we needed to solve in order to make the one-stop-shop model work. How do we write a search facility that can query any number of different lexical databases, each of which has been designed by different people, at different times, for different purposes, and with different schemata for structuring lexicographic information?

## 4.2.2. A central lexicographic database for Gaelic

The simplest way of solving this problem is as follows:

- Create a new central lexicographic database for Gaelic.
- Copy all of the other local databases into this central database.

This model is represented schematically in Figure 4.2.

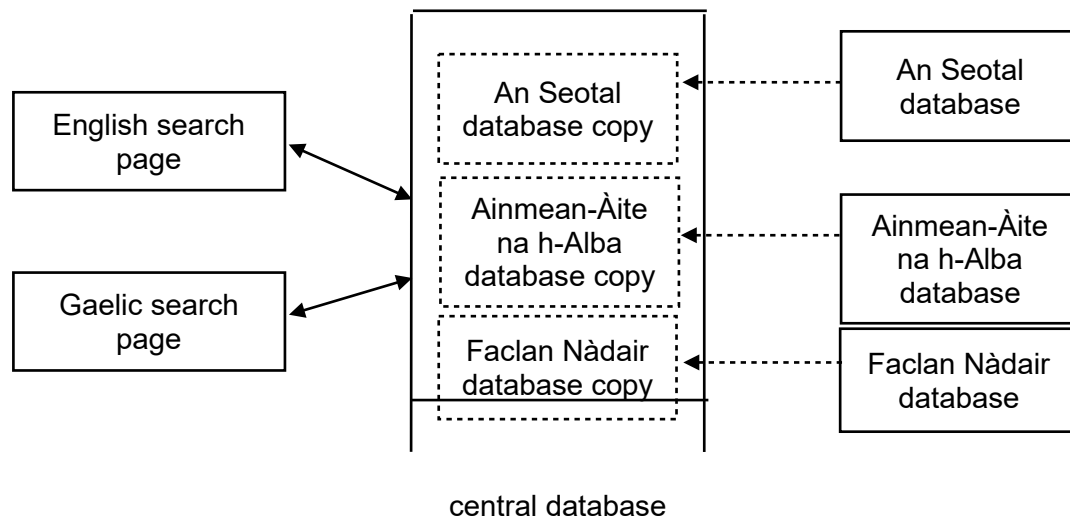


Figure 4.2: Central lexicographic database model

In order to get this model working, each agency would simply need to make their local database available to the central database developers in some kind of standard export format, for example as a spreadsheet. The developers would then write an automatic procedure to import each local database into the common format required by the central database.

## 4.2.3. The problem of synchronisation

The main problem with the central database model represented in Figure 4.2 is the following:

- How do central database developers ensure that the central database remains in sync with the local databases?

Since the central database contains a static copy of each local database, imported at a particular time, whenever a change is made to the local database the central database will immediately be out of date.

There are three conceivable solutions to this problem:

- Every time a change is made to a local database, a new copy of the database is immediately created and sent to the central database.
- Every week (or day, or month) a new copy of each local database is created and sent to the central database.
- Each local database is copied into the central database one time only, and after that no more local changes are made.

Of these three options, the last one is by far the most straightforward to implement, and is also the safest, since every time the import happens there is a risk of data contamination. This leads to a radically centralised database model for Gaelic lexicography, as represented in Figure 4.3.

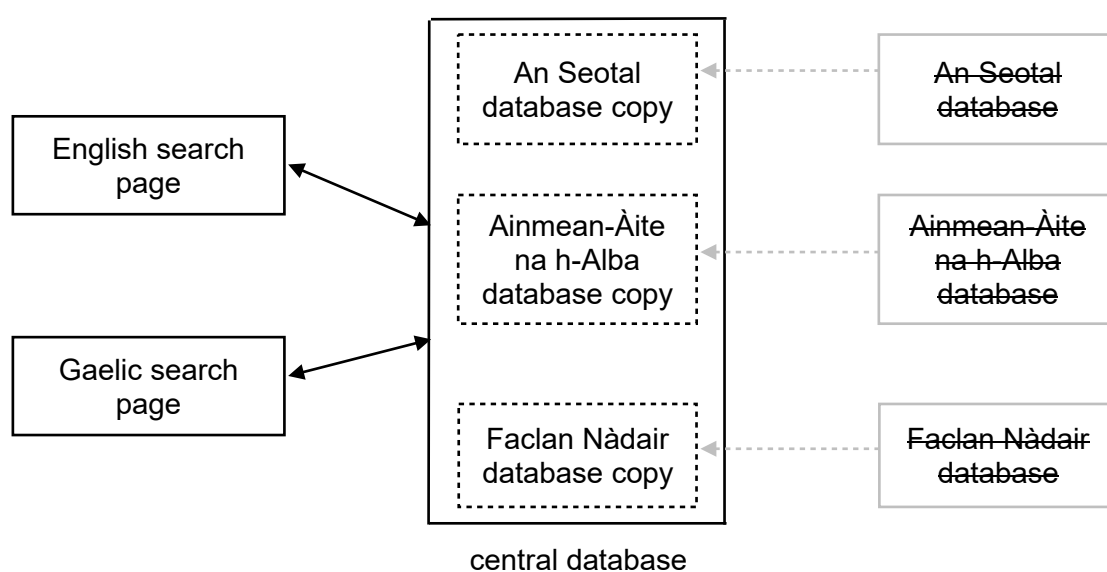


Figure 4.3: A radically centralised lexicographic database model

#### 4.2.4. Benefits of the radically centralised model

The radically lexicalised model represented in Figure 4.3 has significant advantages for Gaelic lexicographic development as a whole:

- It involves a minimal, one-off cost for Gaelic agencies. All they have to do is convert their local lexicographic database into a spreadsheet and send it to the central database developers, and then they never need to worry about this ever again.
- Individual Gaelic agencies will no longer need to maintain their own lexicographic and server infrastructure, since the central database will do this for them, thus significantly reducing maintenance costs and eliminating redundancy.

## 4.2.5. Challenges for the radically centralised model

However, the radically centralised model itself raises some interesting questions:

- Who will have the authority to add new information to the central database?
- Who (if anyone) will have the authority to delete information from the central database?
- Who will have the authority to officially 'kite-mark' existing information in the central database?

The original aims of the project mentioned 'updating by approved contributors', but:

- What does 'updating' mean here? Adding? Deleting? Kite-marking?
- Who are the 'approved contributors'?

## 4.2.6. The LEACAG system

In our proposal for the CR15-02 tender, we essentially proposed a radically centralised model, which has now been implemented, based on our existing, open source Lexicopia system:

- There is a central lexicographic database which currently includes static copies of An Seotal (from 2014) and Faclan Nàdair (from 2016), as well as our own lexical database developed since 2011.
- There is an English search page and a Gaelic search page.

In addition, we have implemented a user authentication system recognising three classes of user (via Google Sign-in):

- **searchers** – 'non-approved' users who can search the central database but cannot 'update' it in any way.
- **contributors** – 'approved' users who can add but not delete information.
- **editors** – 'approved' users who can both add and delete information, and also kite-mark information as being editorially approved in some way.

The full model is represented in Figure 4.4.

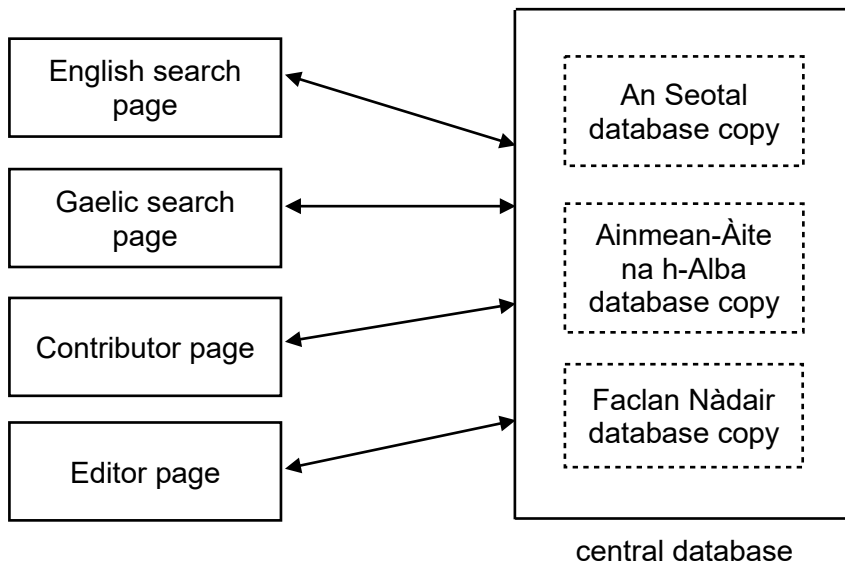


Figure 4.4: LEACAG interface and database model



## 4.3. Future challenges

The system as described above has been constructed and made available online. However, there are a few unresolved challenges that remain to be addressed.

### 4.3.1. Contributors and editors

As yet, no decisions have been made about who can be contributors and who can be editors:

- The *Dlùth is Inneach* project identified the existence of a large number of Gaelic professionals who are keen to be actively involved in the lexicographic process in some way. Thus it would make sense to have a fairly large number of contributors and a liberal policy for accrediting them. For instance, every Gaelic agency could nominate one or more contributors to add information to the central database as and when they coin a new Gaelic term.
- The original CR15-02 project specification implied that preferred terminology would be approved by CCC and hence that there would presumably be an editor delegated by CCC.
- However, the system also has flexibility for multiple sub-editors. For instance, each agency could nominate its own sub-editor with the authority to add or remove an agency-specific 'kitemark' for particular entries (or parts of entries). Each agency could then choose to run its own branded search interface to the central database, which will only return entries and information kite-marked by its own sub-editor.

### 4.3.2. Copyright of lexicographic material

The original CR15-02 project specification states that the resource should be:

- “a public resource made available by creative commons”.

This phrase is somewhat ambiguous, since it fails to specify exactly what kind of Creative Commons (CC) licence should apply, for example:

- CC-BY – Anyone can do anything with the data, as long as they say where it came from.
- CC-NC – Anyone can do anything with the data, as long as it is for non-commercial purposes.

However, the implication appears to be that the data will be made freely available to individuals and organisations to re-use as they wish, either commercially or not. What is unclear is whether or not we are legally allowed to import Gaelic terminology that has been collated or created by other organisations, and then republish this under a CC licence.

The legal situation relating to lexicographic data is unclear, but the basic facts appear to be as follows:

- Words are not subject to copyright.
- Dictionary entries are subject to copyright.

As an example, take the following entries from An Seotal and Colin Mark's dictionary:

- **iar-leasachan** suffix *n(m)*, **iar-leasachain** – suffix *poss c*, **iar-leasachain** – suffices *pl*, Cuspair: Cànan
- **iar-leasachan** *nm* / **i.-mhìr** *nf* suffix

On the one hand, we know we can include the following information in the central database:

- There is a Gaelic masculine noun *iar-leasachan* meaning ‘suffix’.
- The plural and genitive singular forms are both *iar-leasachain*.
- This word is listed in both An Seotal and Colin Mark’s dictionary.

On the other hand, we know that we cannot include either entry verbatim in the central database, without gaining permission from Stòrlann or Routledge.

However, between these two extremes there is plenty of uncertainty. For instance, An Seotal’s Terms & Conditions state that users are not allowed to “compile or create derivative works” from their lexicographic data.

There are two ways in which we can completely cover ourselves legally:

- We can ask the agency to make their lexicographic data available under an explicit CC license.
- We can ask the agency for permission from the agency to include the data within the central lexicographic database, with the understanding that this will then be made available for everyone to use.

Since we are reliant on the agencies making the data available to us in a suitable format, one of these approaches is probably implicit in any case, since by making the data available to us they are agreeing to do so. However, it would be a good idea to make this clear via some kind of legal agreement.

In addition, it will be necessary to clarify at this stage which organisation is ultimately responsible for the central lexicographic database and hence is asking for the legal permission.

### 4.3.3. A federated lexicographic database for Gaelic

A further challenge to the radically centralised model derives from the fact that not all agencies might be willing to give up complete control of their lexicographic data to a central body. They might decide that they are happy to share some of their data but not all of it. This takes us back to the problem of synchronisation – How can we ensure that the local and central databases be updated together, without too much duplication of effort?

With this in mind, there is an alternative (or more accurately a supplement) to the radically centralised model in Figure 4.3 that we might consider implementing in some subsequent phase of LEACAG.

Over the last five years or so, researchers active in the European lexicographic community have been developing an EU-funded model for ‘federated’ lexicographic databases, allowing multilingual lexicographic data to be shared among different organisations, without need for a centralised repository. This model was launched officially in September 2017 at the eLex conference in the Netherlands, along with the various open technical standards that make it possible.

Rather than the central database model in Figure 4.3, this federated model would look more like Figure 4.5.

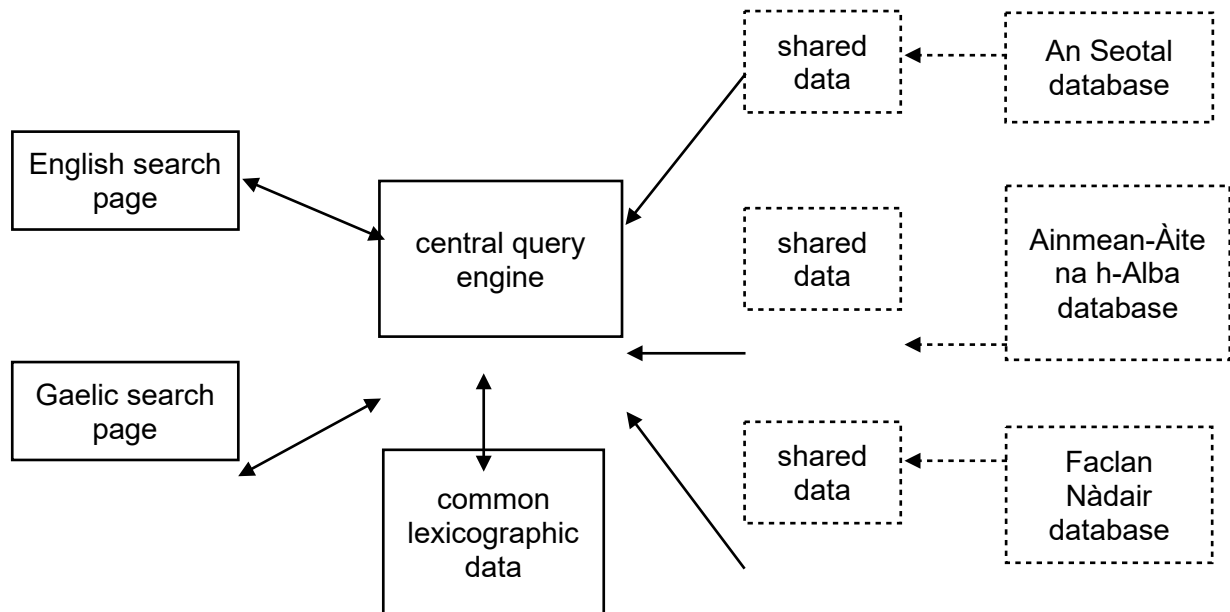


Figure 4.5: A federated lexicographic database model for Gaelic

The most important aspect of this model is that an agency does not have to give up control of its lexicographic database to the central repository. Data is not copied to a central server, but is rather accessed remotely over the Internet.

All that an agency needs to do in this case is to agree to publish some portion of the data in its database in a particular standard file format (RDF-OntoLex-LEMON), and then to update that file regularly to keep the whole system as synchronised as possible. By doing so, the agency will be publishing its data under an implicit Creative Commons licence. Apart from this, the agency can keep on working as it already does with its local systems and database.

From this perspective, the federated model is working like a Google for Gaelic lexicography – not all the data is stored on a central server, but it does contain links to places where all the other data can be found. Thus, the whole system works as a single database, despite the data being stored on different parts of the Internet.

Within the Lexicopia project on which the system that we are developing is based, we have already started the process of migrating behind the scenes to the federated database model. When this is done, agencies that maintain Gaelic lexicographic databases will have a choice of ways in which they can integrate with the central system:

- Move their databases into the common lexicographic data area, and then register as subeditors with the system, as originally intended.
- Maintain control of their data and agree to share it publicly in the agreed standard format and update it regularly.

## 4.4. Links with other dictionary projects

### 4.4.1. The LearnGaelic dictionary

At the very start of the LEACAG project, an additional element was added to the list of specified aims:

- that the online lexicographic system would be able to be used to update and extend the LearnGaelic dictionary.

As things currently stand, MG ALBA's LearnGaelic dictionary has no actual editor, who decides what should be included as an entry and what the content of entries should be. For the first few years of its existence, this was the responsibility of an external lexicographic consultancy, but this relationship has now lapsed.

Using the LEACAG system (at least as originally construed) as a content management system for the LearnGaelic dictionary appears to have the following implications:

- that CCC (or some such committee) would take on the role of Editor for the LearnGaelic dictionary, or at least would be responsible for delegating this task to some person or group.
- that the existing LearnGaelic dictionary database would be copied over into the central lexical database so as to make this possible.
- that the existing LearnGaelic dictionary web interface would be reconfigured to search the central lexicographic database rather than the current local one with an explicit filter for CCC-approved terminology only.

All of these things are possible within the scope of the current system, although there remains some confusion about whether the agreement under which the LearnGaelic dictionary data was 'purchased' actually allows MG ALBA to make it available through Creative Commons.

### 4.4.2. Faclair na Gàidhlig

Faclair na Gàidhlig is a long-term inter-university project to create a comprehensive historical dictionary of Scottish Gaelic, similar to the Oxford English Dictionary or the Scottish National Dictionary. The project is mainly funded by the Scottish Funding Council. The last 15 years have focused on preparatory issues like building the corpus of texts that lexicographers will use, but dictionary compilation is due to get underway over the next couple of years.

To a certain extent the aims of the Faclair na Gàidhlig and LearnGaelic dictionaries are complementary:

- Faclair na Gàidhlig will be mainly aimed at academics interested in the history of Gaelic.
- The LearnGaelic dictionary aims to be a practical dictionary of the modern language for use by learners, users and language professionals.

However, there is one important area of overlap. The Faclair na Gàidhlig business plan has always assumed that they would start off by creating a range of small dictionaries for

learners of Gaelic, similar to the concise dictionaries published by Scots Language Dictionaries, as a potential extra revenue stream to fund the historical dictionary.

One possibility for an accommodation between these projects would be to bring Faclair na Gàidhlig into the federated lexicographic database model in some way, thus allowing the projects to share certain types of information, e.g. a common list of Gaelic dictionary headwords, inflectional forms, senses, etc.

# REFERENCES

- Atkins, Sue & Michael Rundell (2008): *The Oxford Guide to Practical Lexicography*. Oxford University Press.
- Bell, Susan, Mark McConville, Wilson McLeod & Roibeard Ó Maolalaigh (2014): *Dlùth is Inneach – Linguistic and Institutional Foundations for Gaelic Corpus Planning*, Soillse / Bòrd na Gàidhlig.
- Jackendoff, Ray (1997): *The Architecture of the Language Faculty*, MIT Press.
- McConville, Mark (2014): 'Lexicopia / GD – Gaelic Language Policy 2.0'. Paper presented at *Rannsachadh na Gàidhlig 8*, Edinburgh, June 2014.
- McConville, Mark (2015a): 'Lexicopia'. Paper presented at the *Faclair na Gàidhlig Technical Workshop*, March 2015.
- McConville, Mark (2015b): 'Representing constructions in an electronic dictionary of Scottish Gaelic'. Paper presented at the *15th International Congress of Celtic Studies*, Glasgow, July 2015.
- Miller, George A. (1995): WordNet: 'A Lexical Database for English'. *Communications of the ACM* 38(11): 39–41.
- Zimmer, Ben (2014): 'Lexicography 2.0 – Reimagining Dictionaries for the Digital Age'. *Dictionaries: Journal of the Dictionary Society of North America* 35: 275–286.