TECHNICAL TOOL SPECIFICATION

A. Details

Institution: The University of Sheffield

Contact: Michael Pidd, HRI Digital Manager <m.pidd@sheffield.ac.uk>

Address: Humanities Research Institute, University of Sheffield, 34 Gell Street, Sheffield S7 3QY

Name of the Tool: *[to be decided]*

Summary

This specification describes the business requirements, technical requirements and proposed architecture for a Tool that is intended to address the problem of *digital orphans*.

Digital orphans are online assets (in this case, research resources) that are deemed to be undiscoverable, unused, unknown or forgotten by the wider research community because they are invisible or inaccessible to the normal mechanisms of discovery, such as search engines, subject catalogues, aggregation sites and other subject-specific websites. The invisibility of online resources can be due to a combination of factors such as poor technical design, poor presentation of content, poor marketing and an absence of individual and/or institutional support.

The Tool proposed in this specification is intended to address these problems by being capable of developing a discovery-friendly version of a resource's textual content at the record level. This discovery-friendly version of a resource's content, presented as a set of optimised data records, will then mediate between the resource and discovery services. The Tool will achieve this in two ways: a Crawler will retrieve a copy of the resource's textual content, including data contained in databases; and an Analyser will generate discovery-friendly records from the content using Natural Language Processing techniques.

B. Scope

1. This technical specification seeks to address the problem of *digital orphans*. Digital orphans are online assets (in this case, research resources) that are deemed to be undiscoverable, unused, unknown or forgotten by the wider research community because they are invisible or inaccessible to the normal mechanisms of discovery, such as search engines, subject catalogues, aggregation sites and other subject-specific websites.

2. For the purposes of this specification, the online research resources that are of interest to us are typically created and hosted by organisations such as universities, museums and archives. They are often intended to serve as research, teaching and learning infrastructure or bring specialised knowledge to a wider audience. However, this Technical Specification is applicable to any website.

3. Given that discovery services such as search engines are the main way in which new users discover an online resource, a resource's failure to expose itself to these services effectively will significantly reduce new traffic. In the case of search engines, the resource is likely to be poorly indexed and given a low (if any) ranking in search results, whilst subject catalogues and aggregation sites are unlikely to accession the resource at all. Even when a resource has been 'discovered' by users, they will often continue using discovery services in order to directly locate content if the site is badly designed (eg. if it has a confusing structure, an unintuitive interface, broken links and unmemorable URLs).

4. When a resource becomes a digital orphan it is problematic because it is no longer fulfilling the purpose for which it was created, it is limiting access to knowledge and it is giving a poor return for the (often considerable) time and financial investment. Digital orphans are therefore a concern to the wider research community because they potentially endanger future investment.

5. However, this is more than a search engine optimisation problem. A failure to make a resource's underlying data available in an easily re-usable way will limit interest from aggregation sites which typically require access to a physical copy of the data. For example, aggregators such as http://www.nines.org, http://www.18thconnect.org and http://www.europeana.eu require content to be made explicitly available to them in RDF and related formats.

6. The invisibility of online resources can be due to a combination of factors such as poor technical design, poor presentation of content, poor marketing and an absence of individual and/or institutional support. For example, many resources are designed without adherence to principles of best practice in search engine optimisation, resulting in low ranking by search engines. Further, funding models mean that content creators are often required to move on to other projects, thereby neglecting to maintain, improve and continue marketing the online resources which they have created.

7. Retrospectively implementing features that improve discoverability is a particular problem because of funding models: new content creators can be educated to design with discoverability in mind but there are many hundreds, if not thousands, of research resources that already exist for which the implementation of discoverability features is not feasible due to time, resource and willingness.

8. A chief characteristic of poor discoverability, borne out by research undertaken by Sero Consulting and Owen Stephens Consulting as part of Jisc's *Spotlight on the Digital*, is the tendency for content creators

to consider their resource at the collection level only, when in fact the content that will be of most interest to users and discovery services resides at the record level. A great deal of thought might go in to the discoverability of the resource as a whole, but this often conveys little meaningful information (eg. what does the site title “Connected Histories” really tell us?) As such, many of the characteristics of poor discoverability occur at the level of the individual record - the level with most discovery potential!

9. The Tool proposed in this specification is intended to address these problems by being capable of developing a discovery-friendly version of a resource’s textual content at the record level. This discovery-friendly version of a resource’s content, presented as a set of optimised data records, will then mediate between the resource and discovery services.

C. Business Requirements

10. The Tool’s primary aim should be to make the resource discoverable to search engines such as Google. That is, the resource should comply with the majority of commonly held rules and recommendations for search engine optimisation.

11. The Tool’s secondary aim should be to make the resource discoverable to aggregation services such as http://www.nines.org and subject-specific websites or directory services such as JISC Content. That is, it should be easy for the third party to a) point to the resource’s content using hyperlinks, b) acquire citation information for the resource’s content, c) and/or acquire a copy of the resource’s content for indexing and re-use. Note that establishing, granting or enforcing permissions to re-use a resource’s content are outside the scope of the Tool.

12. The Tool should be a tool, not a service. In other words, it should be self-contained, performing a series of processes without the need for external dependencies for its operation, apart from that considered to be part of its operating environment (eg. server or desktop software). It is possible that a user might wish to deploy the Tool as part of a service, but the Tool should not be designed with a specific service environment in mind.

13. The Tool must undertakes a series of data processes. It must not concern itself with permissions, policies, personnel or larger infrastructure etc. Non-process issues about how the Tool and its results are to be used should remain the responsibility of individual content providers (henceforth referred to as “Owners”).

14. The Tool should be lightweight and self-contained for ease of maintenance and scalability (eg. so that it can be deployed locally by an individual Owner or bundled into a larger, cloud based service).

15. The Tool must be easy to use, requiring only minimal setup and minimal technical understanding of its operation.

16. The output of the Tool (hereafter referred to as “discovery-friendly records”) must be easy to deploy and require no further intervention by another tool, service or personnel in order to fulfil the Tool’s discoverability aims.

17. There should be no requirement for the Owner to physically modify the resource in order to operate the Tool and deploy the discovery-friendly records.

18. The Tool should have no licensing or operating costs associated with its implementation and sustainability. Costs associated with its operating environment (eg. server infrastructure, service implementation, personnel etc) are beyond the scope of the Tool itself.

19. There must be a number of pre-conditions for an online resource to be accessible to the Tool. These pre-conditions, if not present, also might be the reason for the resource's poor discoverability. The pre-conditions are as follows:

    19.1. If the resource has a robots.txt file governing access by automated means, the directives should be sufficient to enable the Tool to crawl all the content that the Owner considers to be accessible to the public.

    19.2. Content must be accessible without requiring a username, password, captcha or any other authentication feature. Any content that requires a user to register and login will not be accessible to the Tool, even if the Owner considers this content to be publicly accessible.

D. General Technical Requirements

20. The Tool should be concerned with extracting and processing natural language and should not need the content of a resource to conform to any specific data structure in order to make it relevant to all online resources (such as a specific XML or database schema), irrespective of individual data structures and architecture.

21. The Tool must use open standards, its programming code and functions should be adequately documented and it should be made available in an open source repository for further development by third parties.

22. The Tool must undertake two processes: it must be capable of crawling the content of an online resource and then analysing the crawled content from a discovery perspective. The output of these processes should be a set of discovery-friendly records which can be deployed alongside the original resource for use by discovery services.

23. The Tool must be able to undertake these processes with minimal human intervention or guidance.

24. It must be easy to re-run the Tool on the same site, reflecting the need to update the site's discoverability as-and-when the site's content is updated.

25. The following sections address the two technical processes to be performed by the Tool: crawling and analysing.

E. Technical Requirements for the Crawler

26. The Owner must be able to point the Crawler at their resource (because the resource is technically undiscoverable!) and it must be able to undertake a complete crawl of the site by a) following hyperlinks and b) submitting queries to databases (for crawling a site's underlying database, see points 36 - 39).

27. The Owner should not need to have server access in order to run the Crawler, although s/he will need server access if the discovery-friendly records are to be uploaded to the same directory as the resource (see point 61). This is intended to accommodate Owners who might not have permission to deposit the output of the Tool (discovery-friendly records) in the root directory of their resource. For example, the resource might be contained in a third-party data management system such as OCLC's CONTENTdm which prohibits direct access to the server.

28. Given that the Owner should not need to have server access in order to run the Crawler, this also means that the Tool could be used by somebody other than the Owner. It will not be the responsibility of the Tool to determine whether or not it has the Owner's permission to crawl a resource, although the Tool

must obey robots.txt directives when present and it must require the user to accept Terms & Conditions concerning the legal and responsible use of third-party data prior to the crawl being undertaken.

29. The Crawler must identify any broken links in the resource and report these to the Owner. The Tool will not be able to fix broken links, since this requires manual intervention and is outside the scope of the Tool. The Tool will not be required to validate HTML or identify missing assets – such functionality is out of scope.

30. The Crawler must be able to acquire and return a copy of all textual content associated with a resource, at both collection and record levels, irrespective of the site's data structure and encoding formats (eg. a specific XML or database schema).

31. The Crawler must crawl HTML-rendered output (i.e. the HTML pages that are displayed to the end-user) and not the underlying data.

32. In addition to HTML-rendered output, the Crawler should ideally retrieve data from plain text, Microsoft Word, RTF and PDF output formats.

33. The Crawler must be able to return crawled text as natural language, without structure. This is because subsequent analysis of the content and representation in a discovery-friendly format should not be influenced by structures or formats imposed by the Owner.

34. The crawled text should be returned to the Tool's operating environment without any analysis having been conducted by the Crawler itself. This is so that the processor overhead on crawling can be minimised. The analysis of crawled content should be undertaken by the Analyser as a subsequent process.

35. The Crawler will need to identify itself to the resource's server using an appropriate http userAgent string.

36. The Crawler will need to crawl the deep web. *Deep web* is a term used when referring to content that is held in databases and which require users to submit search terms in order to retrieve it. In other words, there is no explicit, physical link to the content; it has to be retrieved by users guessing what keywords might be contained in the content, as is the case with web search engines.

37. Given that our aim is to improve the discoverability of online research resources, the deep web is likely to characterise the majority of our data. Therefore, retrieving this data should be a key priority of the Tool.

38. Where the deep web is concerned, the Crawler will need to submit search requests to the site in the same way that a user would and seek to do so in a way that retrieves all records contained in the underlying database. This is because the Crawler will be retrieving HTML-rendered output rather than the underlying data.

39. The Crawler will need to use a combination of natural language rules and dictionaries in order to retrieve deep web data. It will also need to verify that it has retrieved all possible content and avoid returning duplicates.

40. The Crawler will need to avoid spider traps. *Spider traps* is a term used to refer to content that is generated algorithmically, such as an online calendar, thereby creating infinite content.

41. The Crawler will not be required to retrieve any content generated by Javascript. This is considered to be an ongoing problem for conventional web crawlers such as Heritrix (developed by the Internet Archive) and, as such, solving this problem is not within the scope of the Tool.

42. The Crawler will need to return a record of all page locations and their URLs so that the discovery-friendly records generated by the Tool will be able to point discovery services to the actual user-viewable content.

43. The Crawler will need to avoid areas that reside behind login screens and (in some cases) subscription gateways. Access to restricted content is beyond the scope of the Tool and it will be the responsibility of the Owner to provide the Tool with free access to content.

44. The Crawler must comply with the instructions of any robots.txt files. It is accepted that restrictions laid down by a robots.txt file reflect the Owner's discovery intentions (eg. making certain areas of data off limits and restricting the frequency of return visits). However, the Crawler should generate a warning if the rules are considered to be too restrictive (the sort of restrictions that are probably contributing to the resource's poor discoverability!)

45. The Crawler must crawl within the resource's domain only; that is, beneath the top-level URL supplied by the Owner. It must not follow links that direct it outside the root directory of the resource.

46. Existing open source crawlers such as the Internet Archive's Heritrix and Apache's Nutch should be considered as the basis for the Crawler.

F. Technical Requirements for the Analyser

47. The Analyser will analyse the data retrieved by the Crawler and generate discovery-friendly records.

48. The Analyser should generate one discovery-record per collection record, especially given that it is at the record level that discoverability is often overlooked.

49. The discovery-friendly records should assist search engine indexing by making available a more 'indexable' version of the site's content. Search engines should be directed to index the discovery-friendly records in addition to the resource itself by having the Analyser generate an XML sitemap that includes the location of the discovery-friendly records. The discovery-friendly records might wish to make use of the *rel="canonical"* tag to inform discovery services about the relationship and status between the discovery-friendly records and the actual content.

50. Ideally the Analyser will produce a solution whereby discovery services index the discovery-friendly records but direct or recommend users to the actual content.

51. The discovery-friendly records should assist aggregation and re-use by third party services by making available the site's content in a more accessible format. That is, third party services should be able to acquire a copy of the resource's underlying data in a suitably structured format, such as RDF, simply by knowing that discovery-friendly records are present in the root directory of the resource.

52. The discovery-friendly records should assist inbound linking (from external websites) and citation by making available more intuitive, meaningful URLs for the content. For example, the original URL http://www.myresource.org/search.php?kw=spade can be presented as the short URL http://www.myresource.org/spade. Both the original and the short URLs should be recorded in the discovery-friendly record metadata.

53. In practice, the operation of transforming the incoming short URLs to the original URL could be undertaken using REWRITE statements in a .htaccess file. The discovery-friendly record would use the short URL in the *rel="canonical"* tag whilst the .htaccess file would rewrite the short URL as the original URL. The Analyser could create the .htaccess file alongside the discovery-friendly records and instruct the Owner to place both the records and the .htaccess file in the root directory. The Analyser will need

to determine whether or not the server has 'Allow Overrides' enabled for its web directory before making the .htaccess feature available to the Owner.

54. The Analyser is expected to generate the following discovery-friendly data for each individual record using the content crawled from the original resource:

    54.1. A short, meaningful URL as the location/filename of the discovery-friendly record.

    54.2. A record title based on the Analyser's understanding of the content.

    54.3. Keyword metadata based on the Analyser's understanding of the content.

    54.4. The textual content of the record, presented without structure or formatting.

    54.5. Deduced metadata (see points 55, 56 and 57 below).

    54.6. The date when the discovery-friendly record was generated.

    54.7. The short or true URL of the original content as a canonical reference (depending whether or not the Tool is able to create a .htaccess file (see point 53 above).

55. The Analyser will generate *deduced metadata*. This will be the most experimental aspect of the Tool. Deduced metadata will be content that does not actually exist on the real site. It is new data – synonyms - that is generated using a thesaurus, based on a further analysis of the keywords extracted from the content. For example, if a record in the online resource contains the keyword *spade*, the Analyser might generate further keywords such as *shovel* and *trowel*, although analysis will need to determine whether, for example, *spade* is being used as a noun or a verb (in which case synonyms might be *dig* and *delve*). The purpose of the deduced metadata is to increase the discoverability of the resource by improving the possibility that semantically similar keyword searches submitted to discovery services (eg. search engines) will return the resource. The deduced metadata must be present in the discovery-friendly records only, not the actual website, and clearly identified as non-original content.

56. We are not aware of any examples in which deduced metadata is generated from the content of a website to aid discovery, although there are parallels with the practice of adding non-relevant paragraphs of text to a webpage purely to improve search engine ranking, often to mislead search engines and users (for example, a gambling site trying to promote itself to users who are searching for something entirely different). Deduced metadata is different to this because it is related to the content of the original webpage, having been 'deduced' from its content. The process of generating deduced metadata is not challenging, although there is a processing overhead which might influence the resource requirements of the tool.

57. When generating deduced metadata, the Analyser will draw upon a pre-loaded thesaurus of modern English synonyms (British, American and International English). The starting point for this would need to be an open source, downloadable database of synonyms, such as WordNet (http://wordnet.princeton.edu) which is used for the thesaurus in OpenOffice. In the future, it might be possible for the Owner to add their own thesaurus of synonyms for foreign languages and older variants of English. The Tool will establish a technical file format for creating additional thesauri.

58. The discovery-friendly records should reflect discovery needs rather than the needs of the site's branding/identity. For example, the formation of a record's title should reflect an analysis of the record's content and an estimation of the most 'discovery worthy' elements of the content, rather than the name of the site. This might mean that the record's title is not considered to be grammatically correct – it might

be a series of nouns.

59. The Analyser will create the discovery-friendly records and save them locally.

60. The discovery-friendly records will need to sit in the root directory of a web resource and interface between discovery services and the real resource. They could be organised and referenced using an XML sitemap.

61. It must be the Owner's responsibility to physically place the discovery-friendly records on the server of their web resource via an FTP upload program or similar. However, it could be possible for the Owner to place the discovery-friendly records on a different server (eg. as part of an alternative, separate website) and the records will still point to the original resource.

62. It should be easy to overwrite the discovery-friendly records with new versions, reflecting the need to update the records in response to updates to the site itself.

63. It is usual for aggregation services and other third-party sites to arrange access to an online resource's content through the Owner, unless a public mechanism for accessing the data is available, such as an API or download link. However, the discovery-friendly records created by the Tool are intended to be publicly accessible and so third-party services should only need to know the location of the discovery-friendly records. The Owner should be given guidance about how s/he might wish to draw attention to the presence of the discovery-friendly records, such as a visible link to each discovery-friendly record on each page of content in the actual resource.

G. Proposed Architecture

64. The Tool (both the Crawler and the Analyser) might be written using the Java programming language and capable of being compiled to run locally as a command-line executable or on a web server.

65. A basic HTML user interface should be available to permit the Owner to submit the URL of his/her resource.

66. Data retrieved by the Crawler, prior to analysis, will be stored as flat, plain-text records that are lightly structured, possibly using the RDF XML format to differentiate the different types of data (eg. the original URL and the textual content). The question as to where this temporary, retrieved data is stored prior to analysis (should it be stored local to the Owner or at the Tool's location, if the two are different?) cannot be answered at this stage.

67. The discovery-friendly records will be organised and made visible to discovery services (particularly search engines) using the XML Sitemap standard. A sitemap normally records the location of all HTML files. In this instance the sitemap will record the location of all discovery-friendly records, including the records of deep web content (not normally included in a conventional sitemap).

68. The discovery-friendly records might use *rel="canonical"* referencing to declare their status in relation to the original (canonical) content.

69. The discovery-friendly records might use emerging RDF implementations such as Google RDFa when encoding data objects in order to improve discovery and data extraction by discovery services (see https://support.google.com/webmasters/answer/146898?hl=en).

70. The process of using the Tool is anticipated to be as follows:

    70.1. The Owner provides the Tool with the top-level URL of the resource.

70.2. The Tool Crawler visits the resource and reads the robots.txt file, if present.

70.3. If the robots.txt commands are too restrictive then the Tool informs the Owner, otherwise the Tool obeys the commands by turning them into internal rules.

70.4. The Tool also retrieves information about the resource's server environment, specifically whether or not the 'Allow Overrides' directive is enabled.

70.5. If the 'Allow Overrides' directive is enabled, the Tool will offer the Owner the option of generating a .htaccess file for short URL forwarding.

70.6. The Tool Crawler then proceeds to crawl the site, retrieving textual data.

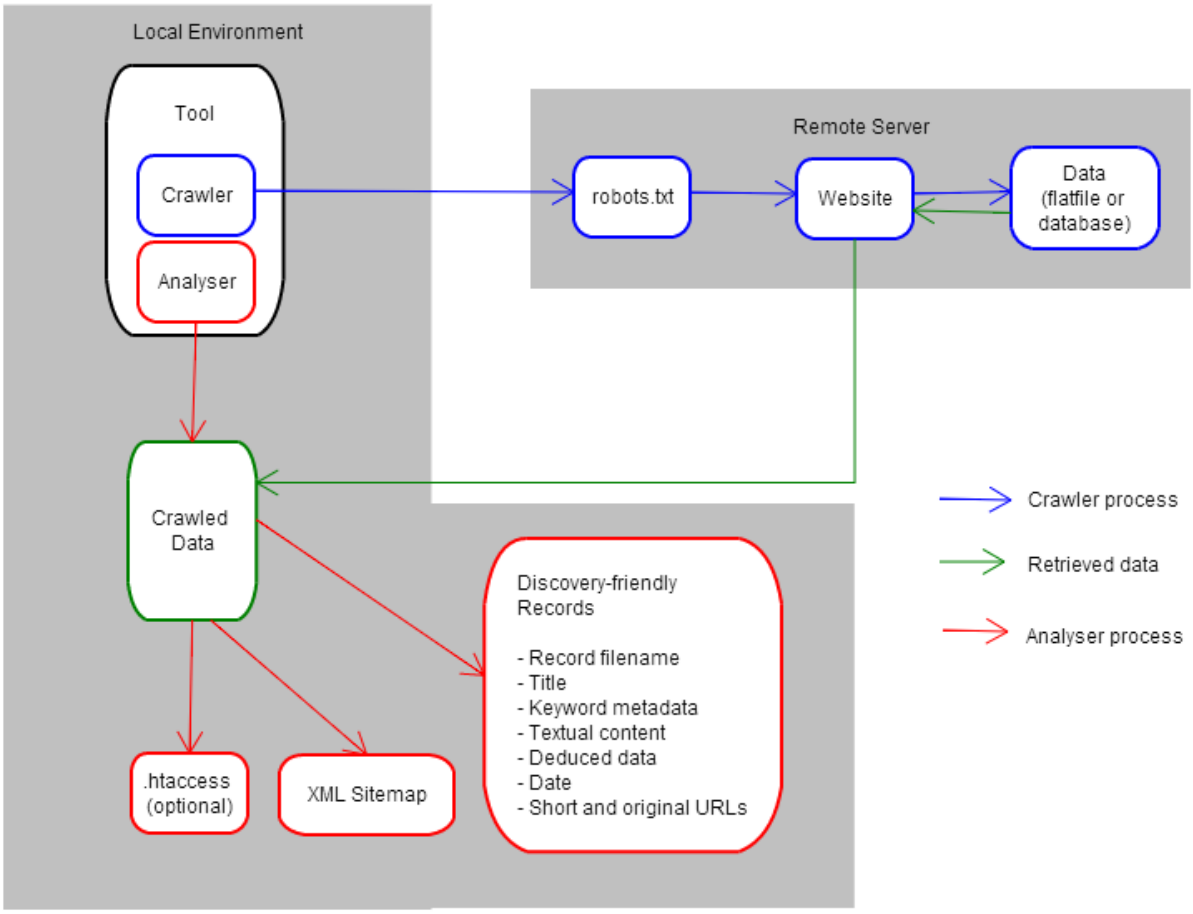
70.7. Any broken links are reported to the Owner.

70.8. When crawling is finished, the Tool Analyser will analyse the retrieved data and generate discovery-friendly records (including a .htaccess file, if requested).

70.9. The Tool Analyser will then generate a complete XML sitemap of the discovery-friendly records.

70.10. The Owner is asked to download the completed records and XML sitemap and instructed to place them in the root directory of the original online resource.

71. The following diagram illustrates the Tool's Crawler and Analyser processes.



72. The following diagram illustrates how the discovery-friendly records, when deployed, will interface between discovery services and the online resource.