Network Working Group J. Klensin
Internet-Draft February 15, 2004
Expires: August 15, 2004

A Search-based access model for the DNS
draft-klensin-dns-search-06.txt

Status of this Memo

Copyright Notice

Abstract

This memo discusses strategies for supporting "DNS searching" -- finding of names in the DNS, or references that will ultimately point to DNS names, by a mechanism layered above the DNS itself that permits fuzzy matching, selection that uses attributes or facets, and use of descriptive terms. Demand for these facilities appear to be increasing with growth in the Internet (and especially the web) and with requirements to move beyond the restricted subset of ASCII names that have been the traditional contents of DNS "Class=IN". This document proposes a layered system for access to DNS names in which the layers above the DNS involve search, rather than lookup (exactly known target), functions. It also discusses some of the issues and challenges in completing the design of, and deploying, such a system.

Table of Contents

1. Introduction and Executive Summary

The notion of "DNS searching" is somewhat of an oxymoron: the DNS is structured to only perform exact lookups of structured strings of labels. But, as discussed elsewhere, there is considerable demand for searching facilities -- partial and fuzzy matching, selection that uses attributes or facets, and searching using descriptive terms-- and that demand appears to be increasing with growth in the Internet (and especially the web) and with requirements to move beyond the restricted subset of ASCII names that have been the traditional contents of DNS Class=IN. This document proposes a three-component system for access to DNS names. In this approach, the DNS (more or less as historically implemented) comprises the base, with the other two components (or strategies) layered on top of it, but not necessarily on each other. These new approaches involve search, rather than the DNS's very restricted lookup (known-item search), functions. It also discusses some of the issues and challenges in completing the design of, and deploying, such a system.

These types of services are unnecessary as long as the problem is defined as "get non-ASCII identifiers into the DNS, but keep to a well-specified set of characters and usage so they retain strict identifier properties" --more or less the approach taken in IDNA [RFC3490]. Such approaches do not, as discussed in [RFC3467], solve the problem as perceived by many people. One non- technical way of looking at this is that the DNS is fundamentally downward-facing: it is designed to support references to network and host resources. Users want something upward-facing, i.e., that provides natural-language terminology and searching for resources of interest. And, as the IAB has pointed out [RFC2825], even if "fixing the DNS" did the job, that would be the easy part: the harder problem is considering and adjusting the applications and applications-level user interfaces.

It has been suggested that introducing a "directory" or "keywords" into, or above, the DNS could be used as a solution to the IDN problem and, often, several others. Probing statements about "directories" often quickly demonstrates that their advocates don't agree on what they mean. Similarly, many of those who advocate "keyword systems" use that term to describe something very different from the traditional use of keywords in information retrieval systems. This section outlines a three-strategy search/lookup model (adding two location strategies to the one provided by the DNS, i.e., constructing a two or three-layer model, rather than continuing with the single one we have today). These new strategies consist of two layers: the current DNS and a collection of alternate methods for searching for a network resource, identified by URIs, and using different additional information and searching methods. The key one

of these is a global faceted search-capable component using an extremely simple set of facets. The other proposal is for more context-specific or localized, but broader, search approaches. The overall document is intended as a strawman for criticism and development, rather than as a specific proposal. I.e., the details, especially for localized systems, are left for IETF WG or other efforts.

The document also introduces the concept of an expanded and powerful information cache, under user control, which should supercede the functions of now-traditional "address books for data", "bookmarks", lists of "favorites", and similar methods of retaining an association between a user-memorable name and a network resource. The additional power comes from retaining and utilizing, not only the name to resource binding, but also information needed to reconstruct a search and to determine when to do so.

As a terminology issue, the "layers" and components described here are probably best thought of as elements of the applications layer, with actual user-facing applications lying yet above them. The term "search layer" has been used below to refer to both the faceted and localized components where it appears to be needed for clarity or emphasis. "Sublayer" and "level" are sometimes used interchangeably with terms for those components: suggestions for better terminology would be welcomed.

For the two "above DNS" DNS, international ("universal") character sets and scripts are assumed and part of this initial design. Since actual or applications-applied DNS restrictions are not being inherited upward into these components, coding can be chosen for maximum utility and balance among language groups. E.g., native UCS-4 could be used as an alternative to a secondary encoding form such as UTF-8 or an ASCII-compatible ("ACE") recoding such as that contemplated by [RFC3490]. And, of course, coding systems for [ISO10646] characters are required only in on-the-wire transactions and operations closely connected to them; we would anticipate that localized search operations in areas that use other coded character sets might be carried out largely in those locally-chosen coding systems.

This document is intended to evolve into a framework and model for the layered search system, rather than a complete specification or even an approximation to one. It is complemented (for the global search system) by [Mealling-SLS], which discusses a CRNP-based [RFC2972] implementation model for that approach, by the more keyword-focused model of [Arrouye], by the caching data provided by Shi and Liu [SHI-CACHING] and, we hope, by other systems more tailored to specific languages or cultures. Additional documents are

expected to be developed that describe other aspects of all elements of this general proposal.

2. A multiple search-strategy environment

The material below suggests three or more components or alternatives for name lookup and search:

1. The DNS, with the existing lookup mechanisms and a single global name space in which names are unique.

   Names are placed in the DNS by those who wish to use those names themselves (e.g., for identifying hosts and resources within a home, an enterprise, or cooperating groups of organizations). The DNS was never designed for searching for, or querying of, an identifier by someone who does not already know what it is.

   A useful analogy has been drawn between DNS names and variable names in a programming language [Austein2001]. Expanding on it, locally-memorable, and hence internationalized, identifiers are important for network engineers and operators to use in referring to resources regardless of whether they are actually valuable for end users. Such engineers and operators are used to developing identifiers using highly-restricted character (and other) rules and to remembering the identifiers they create (or having other tools to deal with them).
2. A restricted, facet-based, global search system.

   This system still preserves a global name space, but name strings are not expected to be unique and the set of facet values for a given entity may not be (see Section 3.2).

   Names are placed into this type of system by those who want to be found, or want the names or resources to be found, by others. The assumptions are neither that those others will know exactly what name they are trying to access (where the DNS requires precise knowledge of names or very good guessing) or that names will be unique (where the DNS requires uniqueness). But the search activity is still based on names (and attributes), not topics.

   It may be useful to think of this layer as similar to "white pages" services. This comparison is discussed in more detail below.
3. Commercial, localized, and potentially topic-specific, search environments.

   These environments utilize multiple, localized, name spaces. These would typically be localized by language or (physical or

political) geography, but might be structured around, e.g., specific subject matter.

Names are placed into these systems by those who wish them to be found within the specific topic area context (or language or locality or combination of them). Because the environments are localized, different search terms and levels of granularity can be used in different search sites and name spaces.

It may be useful to think of this layer as similar to "yellow pages" services. Again, the comparison is discussed in more detail below.

4. Something else?

As discussed below, it is possible to think about the collection of automated content-based search systems that are available on the Internet -- e.g., the traditional "search engines" such as -- as components of the "search layer". The approaches described in more detail in this document utilize a "directory" approach, i.e., the information in them is put there precisely because some individual or enterprise would like to be found, and found using that information. The search engines, by contrast, rely on analysis of the content of online materials to index those materials.

The following may help illustrate the relationships being proposed here.

```
  |--------------------------------------------------------------|
  |               applications                                   |
  |--------------------------------------------------------------|
  |                           |   access cache    |              |
  | . . . . . . . .  . . . .  |-------------------| . . . . . .  |
  |     (directory-based approaches)    | (free-text approaches) |
  | Global faceted   | Topical, usually |       web search       |
  | name search      | local search     |                        |
  |--------------------------------------------------------------|
  |           Domain Name System                                 |
  |--------------------------------------------------------------|
  |  Network resources (by IP Address, Port, etc.)               |
  |--------------------------------------------------------------|
```

2.1 Base Layer Identifiers -- a lookup system and the DNS.

In this model, the DNS remains largely as is (see Section 3.4 and following) or, perhaps, a bit closer to its original purpose and assumptions than the direction in which it has evolved in recent

years. I.e., it is a distributed database, with precise lookups, whose lookup keys are identifiers for Internet hosts and other objects. We give up the notion that these identifiers should also serve as human-useful names or at least try to abandon that notion.

As an aside, note that some people have suggested that we should dehumanize DNS names entirely, e.g., prohibit the registration and use of any name that can be found in any dictionary for any language that can be represented in the DNS-acceptable character set. This proposal doesn't include that idea. But it is absent primarily because it does not appear that the transition process would be worth the time it would take to explore, rather than because it has no appeal.

The goal for this base component is relatively simple, unique, identifiers. It is probably desirable that these identifiers be able to have some human mnemonic value, but less important that they be tightly bound to real-world names and descriptions.

The inputs and outputs at this layer remain as they are in the DNS today. This proposal is complementary to those of [RFC3490] which accommodates non-hostname format names more or less directly in the DNS itself for circumstances in which they are deemed important for mnemomic or other purposes. "Hostname-format names" are those that are restricted to the ASCII-based "letter-digit- hyphen" (LDH) format traditionally used in Internet applications [HOSTNAME] and identified as prudent practice in section 2.3.1 of the DNS specification [RFC1035]).

2.2 A Globally-accessible Search Component for Names

A faceted search system with a small number of facets, built on top of the DNS.

Much of the current burden borne by the DNS would appear to be better focused on a search system that contains names and a small number of attributes represented in name facets. That DNS burden includes a wide range of non-identifier goals and constraints: names that a user can understand and find and that have significant mnemonic value, names with trademark implications, a wide variety of naming systems and, in general, helping people find the things for which they are looking. It is critical that the number of attributes be constrained to a minimal set, and that other attributes, especially those of special interest, be deferred to the third type of system, described below.

The term "attribute" is used here and below to identify the controlled vocabulary or rule-defined facets as distinct from the

free-form "name-string".

It is probably most useful to think about this layer in terms of a structured, multifacted, multihierarchical, thesaurus-like database with search capability (Cf. ISO IS 5127-1 and IS 5127-6 [ISO5127]), rather than as a "directory" in the sense of X.500 and its derivatives and antagonists.

2.2.1 The facets

A key question is what facets to use once the major commercial product requirements are removed (to the third search component, see below). It appears to me that, to satisfy to the critical name-uniqueness and real world pressures on the DNS, candidates for identifying facets might be

name-string. Characters from IS 10646, see below.

language. Presumably codes as specified in RFC 3066 (see Section 3.3.1)

geographical location. Country, and/or for some federal countries, country/ province ("state"). Granularity is important and there may be a case for an additional facet based in a coordinate system or for a two-level facet. See Section 3.3.2.

network location. If we can figure out what that means and how to express it in a canonical way.

industry category code. For companies, presumably derived from some existing official list such as the WIPO Nice Agreement list [WIPO-NICE]. The list would presumably require extension in some way to deal with non-commercial organizations and entities and to identify resources and services associated with people. See Section 3.3.3.

This typology gives the trademark view of the world somewhat more precedence in looking at name conflict issues than one might like in principle. But, in practice, one of the key issues we have encountered in trying to store "names", rather than identifiers, in the DNS is that the process unreasonably flattens the space, not only from a technical standpoint but from a usability one. That "Joe's Auto Repair" and "Joe's Pizza" can co-exist in the same geographical area without conflict or confusion and that "Joe's Pizza" in one area can co-exist with "Joe's Pizza" in another, again without conflict or confusion, are the consequence of the way we name and identify things in the real world. Most trademark rules are the consequence of those naming systems, not their cause and many perceived conflicts between the trademark system and DNS usage are the result of this flattening.

It is not intended that this level act as a white pages service for people. Doing so leads down several slippery slopes at once, including heightened privacy concerns and the associated requirement

to identify, authenticate, or authorize those submitting queries.

The general intent is that the list of facets be fixed by protocol and that possible values for each facet be controlled vocabularies, not necessarily (and probably not) controlled from the same source (see Section 3.2). We would hope to utilize existing terminology lists where possible. For a particular record (i.e., a name and its set of attributes), and especially if requirements for uniqueness can be bypassed or relaxed, the selection (from the controlled vocabularies) of particular facet values would be the responsibility of the entity registering the names. In other words, someone registering a "name" in this system would select values for each of the facets from the controlled vocabulary for that facet as part of the process of placing the name into a database.

It is important to note that the registration of that name would include all of the associated facets, although the vocabularies for all of the facets other than the "name-string" would be drawn from specific, external lists (controlled vocabularies or rules). It would not be desirable, and probably would not be feasible, for registrants to record their names in independent, facet-based, databases with one facet per database.

There is also no magic in the proposed system. Names are placed in the system with particular facet sets because a registrant wants them there. A registrant who wishes to have a given name-string associated with different facet values (e.g., to identify different locations or lines of business) will make multiple registrations.

While all faceted name strings would contain the same facets, there is no technical reason why one or more of these might not have a blank (or "missing") value, presumably causing a match to any search term for that facet. More important, searching for a name might omit one or more facets from the search, again matching any value that actually appeared in the database.

It should be clear that there is significantly more information (from the values of the facets) at this layer than there is in the DNS.

2.2.2 The name string

The names in this environment can reasonably be written in IS 10646 codes or some recoding of them. Since we would be starting more or less from scratch, we could select lengths and codings for maximum efficiency and utility, not to meet the constraints of existing software. In such a context, this author has a slight bias for direct UCS-4 coding. This is in preference to ASCII-compatible ("ACE") codes; compressed, null-octet-eliminating, systems such as

UTF-8; or surrogate introducers to hold things to 16 bits. The loss in transport efficiency is likely to be more than compensated for by gains in cleanliness and equal treatment of all scripts. And, if compression is needed, it is perhaps better to do it at the string level rather than the character one. The optimal coding may never be obvious: the "right" answer lies somewhat in the particular application and the eye of the beholder, and passions run very high on the subject. But that issue is separate from the main and important design arguments of this document.

The work done to define "stringprep" [RFC3454] and, later, the DNS profile in "nameprep" [RFC3491], developed for IDNA will be necessary to determine which names to actually store in the database. But the stakes are lower here than the "get it right or fail completely" constraint of the DNS lookup environment: one can imagine search mechanisms that would apply a more liberal set of matching rules (and/or localized and language-specific ones) than the rules used to encode names (much like recent applications protocols that explicitly distinguish between the formats one is permitted to send and those one is expected to accept (Cf. [RFC2822])).

At the same time, it would be sensible to permit short phrases as these "names", something which is not generally possible in the DNS (or in the IDNA standards). The necessity, in the DNS, to turn, e.g., "Lower Slobbovian University" into "LowerSlobbovianUniversity.edu", and then hope case will be preserved (or to use "lowerslobbovian.edu", or worse) is, ultimately, just another example of the unfortunate mismatch between the identifiers of the DNS and real-world naming systems. So we would assume that it is a design requirement to make it possible to use "Lower Slobbovian University" and "University of Lower Slobbovia" as stored names and, if both appear, to treat them as equivalent or approximately equivalent.

2.2.3 Case matching

In the system proposed here, case-matching should be treated as just another case of fuzzy searching and matching, not a relationship with unique status. As discussed below, in all cases, the user (or her agent) would provide a string, some subset of facets, and search-method specifications as input, and would receive a set of matching results, in the form in which they are stored in the database.

Case matching -- treating upper and lower case letters as identical -- is another historical DNS property that does not have a simple and unambiguous interpretation in the real world of non-ASCII character sets and a range of language applications. Some scripts contain glyph forms that clearly represent two cases, some scripts clearly do

not have case distinctions, and, as became clear during the development of IDNA, there are character-matching requirements in some languages (e.g., equality of simplified and traditional Chinese (e.g., see the work to handle preferred forms and variants for those characters [JET-Guideline], and below) for which the appropriateness of an analogy to case-matching has caused a considerable controversy, not least because of the apparent absence of a set of mapping tables that cover all of the possible character pairs. Even in scripts with clear case distinctions, it is common to find lower-case characters with no corresponding upper-case one (e.g., the German Eszet) or no upper-case character that maps uniquely to a lower case one (e.g., "A" in French may maps to either "a" and "a with accent"). Many of those who have examined the cases closely, including the working group that developed IDNA, have also discovered that, even for scripts with the presence of clear case distinctions, the matching rules sometimes differ by geographical locality.

It is not completely clear how case matching should best be handled. It does appear almost obvious at this time that the IDNA model is not adequate for many situations. That model essentially results in different rules being applied to different scripts: case matching in some situations, none in others, and some but not all characters in yet other cases. It may be the best compromise given the combination of the constraints of the DNS with the idiosyncracies of Unicode, but, with or without the DNS constraints, we should strive to treat all languages and scripts in as nearly an identical, or at least non-discriminatory, way as possible. (A discussion of a generalization of the JET Guidelines system for reducing the worst impacts of different treatment for different scripts in IDNA appears in [I-D.klensin-registration].)

While there are other options, it would appear to be better to handle case-matching on the server, as it is done in the DNS. As with other searching variants, it should be possible to return the form of the name as stored in the database while finding it using any of the user-acceptable variations (use of client-side string preparation for both the stored name and query formation, as a DNS internationalized on the IDNA model seems to require, loses information that some people consider important). Case-matching in the proposed faceted system could be applied (or not) as dictated either by a heuristic using the combination of the language facet and a query containing the preferred location-context of the user (see below). Or there could be an explicit query flag (or indicator carrying more than one bit of information). This author tends to prefer the latter because of a profound distrust of heuristics, but the question requires additional study.

2.2.4 More complex character matching

The case-matching strategy applies to more complex cases of character matching as well. If one can establish sufficient context, and specify the types of expanded matching to be used, and permit multiple variants to be returned to the application, then one could support matching of similar-appearing characters (e.g., Latin "A" and Greek Alpha), or Latin-derived and Cyrillic-derived scripts for Serbo-Croatian, or, perhaps most important, mapping between Traditional and Simplified Chinese (see Section 3.6.3).

2.2.5 Query formation and specification

As is common with systems of this type, we would anticipate the possibility of searching on any of the attributes and that searching on free-text strings would not be exact (i.e., near-match responses could be returned using any of several algorithms, with the user making choices). One could also imagine distance function calculations on appropriately structured restricted-vocabulary facets being implemented in some search engines. As is equally common, we should think about user interfaces that store both queries and response sets so that the responses could be used offline and refreshed when the client systems were attached to the Internet (see the discussion of caching in Section 3.6.2).

At the same time, we would assume that a search without at least some approximation to a name string would rarely be productive and would expect search systems to be optimized accordingly.

In summary, the goal at this layer is to provide tuples of human-recognizable (not just mnemonic) facets (names and attributes), but names that are relevant within the context set by the attributes, rather than a global system based on the names alone.

The input at this layer is a query consisting of search values for one or more of the facets, plus information to control the search. E.g., to the extent that designers of search protocols can provide the proper tools and terminology, one would expect the query to be accompanied by rule statements about how much "fuzziness" was permitted, how "distant" names might be from the chosen ones and still be selected, whether character set or language translation (or even phonetic recognition) was to be applied (and whether translation was to be restricted to a small group of languages or made more general) and so forth.

The outputs are still being discussed, but would appear to best be the full facet set of the matched tuple(s) (more than one such set if multiple tuples match) and one or more URIs [RFC2396] associated with

each tuple. These URIs, and the DNS entries which they contain and to which they refer, will generally have the same uniqueness properties of the DNS itself: while a query, or full set of matching facets, could match (and return) multiple DNS names, nothing would make the DNS names less unique than they are today (i.e., as the DNS requires).

An alternative to using URIs as the return information is to simply return fully-qualified DNS names, leaving ancillary information about location of particular information and the protocols to be used with them, outside the scope of this system. The author reluctantly abandoned that approach as it became increasingly clear in discussions that the additional information was necessary: users will search for information, or a resource and the mechanisms for using it, and are little-interested in what the network might, or might not, consider as a resource. Even if URIs are returned, one of many interesting questions appeared to be whether these search systems should pass through and return the DNS records themselves (labels, class, type, and target) or whether they should return names (labels) and let the applications do the DNS lookups. The latter now appears obvious, not only because it can be anticipated that some URIs will not directly utilize DNS names, but because the data expiration properties of DNS resource records may be very different from that of a data reference or search result that can point to and rely on the DNS name. The substitution of URIs for DNS names does, however, increase complexity and the risk of recursion problems, and that tradeoff should be understood and appreciated.

For the cases in which DNS-level information is the required response, a URI type might be created for that information and used to abstract the return information into something applications can then specify or decode as appropriate. As suggested above, use of such a URI would need to be carefully structured to avoid complex problems (e.g., recursion in either this system, the DNS, or both), but might be a reasonable approach.

Regardless of whether the output is a URI or a DNS name, DNS extensions such as IDNA will presumably be applied to them. I.e., the process of looking up DNS names that emerge from the these search components would presumably go through the extended process specified in IDNA or its descendants.

Experience with the DNS and other distributed databases also argues persuasively that these records are not forever. Unless there are no local copying and caching mechanisms (which seems unlikely and hard to enforce), some type of time to live (TTL) or other expiration or reverification mechanism will be needed.

2.2.6 Imprecise matching

In addition to the methods described elsewhere in this document, there is a long history in the information retrieval field of non-exact matching systems that could be applied in a searching system, even one based on an underlying database model of fixed facets and controlled vocabularies. Ones that might be applicable include spelling and sound-alike variations, synonyms and near-synonyms, opposite terms and category negations, and others.

2.2.7 Registration rules and query rules

As with the DNS, it may be more important to be conservative about what types of names are registered than to be restrictive about queries. At the same time, if there are well-known and easy to understand rules about registration restrictions (probaby implying that the same registration rules must be used globally), it should be possible to optimize query interfaces (corresponding to "resolvers" in traditional DNS terminology) to immediately return "invalid name" error messages rather than returning "not found" after a search.

One could, for example, easily imagine a query interface that would maintain a local (although periodically updated) table of ISO 3166-2 codes to perform validation against the major components of geographic names before initiating a search of a remote database.

Similarly, if a sublayer two database was created for a particular country and language, registrations in it would presumably be restricted to records for that country and language, and to name strings that conformed to validation rules developed for that country and language.

The category lists (i.e., restricted vocabularies) for each of the facets would presumably come from different, although standardized, databases, e.g., IS 3166-2 and UN/LOCODE for geographical information, RFC 3066 for language names [RFC3066], an extended version of the WIPO-NICE code set for industry codes. But the name databases themselves would contain a complete set of tuples for the facets (some, of course, might be missing or, more precisely, "let anything match").

2.2.8 Server location

The DNS avoids the problem of server location through its strict hierarchy model: hosts are preconfigured with hints as to how to find the root servers, and all other servers are found via delegation records. Since this model contemplates relatively independent sets of servers for individual records, under separate administration, the

convenience of the DNS model is not available. Instead, some mechanism for server location is needed. At least the following models seem plausible:

1. A common directory of servers, perhaps stored as bootstrap records in the DNS. The advantage of this approach is that it appears, on first glance, to be easy and obvious. Its disadvantages include an implicit requirement for agreement on how servers for selected and the list administered. Also, information about prioritization of servers and server search would need to be provided out of band in some way.
2. Each user would need to preconfigure a server, or have that information in a client default. That server would be responsible for determining which servers were to be searched if it did not have the needed records available, those would specify further servers, and so on. This approach would be straightforward and would solve the problem of prioritization. However, especially for searches for web resources, it would tend to automatically give the suppliers of dominant browsers control of server resources and search choices.
3. Some more automated, "rootless", procedure for service location might be used, as in [Mealling-SLS]. Such an approach holds significant promise, but has never been attempted at full Internet scale.

However, this issue may not be as important as would appear on first glance. Studies of user behavior indicates that they tend to be extremely impatient. If they have one preferred provider for commonly-used terms, it may be that, if that system does not find the information of interest, their course of action would be to abandon the search (or seek an entirely different method of searching), rather than to continue to other search services and databases. Conversely, it implies that a given search service will find it desirable to either be extremely comprehensive or to use multiple-server search in a way that is invisible to the user (and very fast).

2.3 The Third Component: Locality and/or content-domain-specific data and mechanisms

The problem with the faceted global search model is that there are a number of usability and marketplace pressures for naming systems that offer finer granularity and a better match for user needs. For many purposes, users want localized, not global, systems. This has been confirmed in those systems which have been included in experiments or partially deployed (see, e.g., [RFC2345] and the RealNames work and its variants), which require contextual localization, not a single global environment. There are many causes for this, but requirements for very specific searches that are geographic-area, topic-area, or language or culturally specific, tend to dominate the list.

The issue is perhaps illustrated by an example. Suppose the granularity of an entry in a faceted system is

{"Joe's", "UK", Restaurant,... }

Now, one might want to create a business around a restaurant directory for Bristol. The developers would probably want to construct a database that contained exact locations, type of food, menu information, prices, etc., and permit people to query it that way. That type of product bears a strong relationship to traditional yellow pages services: the best attributes to collect and the optimal way to organize them will differ by topic (e.g., for most people, "menu" has no obvious analogy in an automobile repair shop) and the business models are fairly established. Part of the history of those business models is the observation that, when there are competing yellow pages services (or guidebooks, or other, similar services), those who consistently make better (and "more accurate") choices of categories and keywords tend, other things being equal, to be judged "better" and to capture larger market share.

A related restaurant example may illustrate another important issue: if one looks for Chinese food in Phoenix, the search key "Chinese restaurant" may be appropriate and adequate: there are few of them, and they are not highly differentiated. In Washington, DC, enough restaurants specializing in Chinese regional cuisines are now operating that more precise categorization is needed. And, in Shanghai or Beijing, "Chinese restaurant" is presumably not a useful category at all: regional or other cooking styles would be key to finding something useful, rather than obtaining an extremely low-precision result.

One can imagine many different types of keyword and (yellow pages-like) directory services of this general character, using different types of protocol mechanisms as well as different types of database content and schema. But those services are nearly ideal candidates for competition: there is no requirement that either the providers or the services be global or unique or even highly standardized. Having all three search components bound to the same data sources and perhaps to the same information retrieval interfaces (see the developing DRIS work being done by Wang Liang and his colleagues [WangLiang2003]) --inheriting values from those databases or systems if one wants to think about it that way-- would provide a degree of consistency that might be very attractive to users, so there are clearly issues here that will need to be worked out in the marketplace.

Directories of these types are, of course, common and widespread outside the Internet. There is no shortage --some would say there is

a surplus-- of directories and guides to resources and services of particular types and in particular areas. Advertising or placement fees from resource owners support some of them. Others are supported by book sales or fees charged to users, and still others by a combination of approaches. Most of these directories and guides publish year after year and seem profitable.

Inputs for these locally-based systems will differ by service: one can imagine free-text interfaces and menus (but see Section 2.4) as well as systems that more closely resemble faceted search terms.

Outputs will normally be globally-valid names: URIs, DNS names, or sets of facets for the global search system. However, since users are unlikely to distinguish between global, faceted, systems and these local ones (or even between them and the DNS or other search mechanisms), it is important that these queries and results be locally cacheable on the same basis as global queries and results to preserve name and reference portability.

Summary: Just as the monohierarchical identifier-lookup system at the basic DNS level should be supplemented by a multilingual, multifaceted, multihierarchy search system on a global basis, that global system should be supplemented by a collection of localized, subject- and topic-specific systems. These localized systems need not be centrally coordinated in any way, although enough similarity of function and interface to permit a common local caching environment would almost certainly make them more consistent for users and easier to market.

2.4 The Non-directory Search Component: free-text searching applications

The approaches described above omit one set of techniques used today: "web searches" on full text or its equivalent. These systems have an important role (and, similar to the localized and topical searches described here, there seems no particular advantage to trying to standardize them worldwide). But their disadvantage, if seen as a DNS surrogate or replacement, is that they have difficulty distinguishing between the name of something, a pointer to it, and a reference to, or discussion of, it or how it works. The other systems discussed in this document are all "directories" in the sense that someone must make an explicit decision to put an entry in a database; they are not full text searching systems or analogues of them.

If, for example, one is looking for a web site for a company, a search of the local "yellow pages" would presumably find that site (assuming the company wanted to be found). Global faceted search (or even the DNS) might find it with some guessing, but this fourth level

would (as web search engines do today) probably not reliably distinguish the company's site from sites that reference the company or its products.

Locally-specialized search produces information that is explicitly bound to the query, i.e., what one is looking for, while a search engine returns values that also include sites where the subject of the query might have been mentioned.

2.5 Database and searching differentiation

In both of the directory-based search components, but especially for global faceted search, we assume that "compiling databases" (i.e., registry and, if appropriate, registrar functions) and "designing and building search functions and providing search services" are separate. It would be necessary to have database interfaces be sufficiently general and well-specified that referrals were possible and different search services could rest on top of them, but we would expect some search services to be much more extensive than others and for their vendors to seek increased compensation for those more extensive servces. In many cases, the market would eventually sort out the optimal combinations of capabilities and costs.

Ultimately, the term "fuzzy search", used extensively in this document and elsewhere, is handwaving. Whether heuristic or deterministic, one must devise, for each facet, systems for determining whether matches have occurred and, for inexact matches, whether the combination of query term and database entity are "close enough" together to be candidates for being returned as responses. We can imagine phonetic matching as well as character-string matching, application of contextual rules as well as simple character-pair rules for matching of Traditional and Simplified Chinese, and similar rules for matching of Kanji and kana strings. And we would presume that users, or their agents, would be able to control such decisions by choice of search providers, configuration, or choices on a per-search basis.

3. Context and Model Details: Global faceted search

3.1 The data search and access model

It is interesting that recent IETF "directory" work has focused on accessing mechanisms without worrying intensely about the underlying database content, maintenance, and update issues. Those latter issues seem to be the harder ones, i.e., the difference between LDAP [RFC2251] and CNRP [RFC3467] may make less difference than how we structure, maintain, match, and distribute the relevant data.

Of course, that does not suggest that work on accessing mechanisms is not important or that it isn't required. And, to deploy the model suggested above, we will need to deal with a pair of uncomfortable problems:

- CNRP looks interesting, but has not been widely implemented or deployed in production.
- LDAP is widely deployed, but primarily in implementations that contain sufficient extensions and special features to be non-interoperable. Effective referral mechanisms have also not be clearly standardized in LDAP, and this might provide a barrier.

Some readers of early drafts of this document have also suggested that the history of LDAP points to local extensions that will result in inconsistent search behavior, while CNRP may be better specified (or at least closer to a clean slate).

If we are going to choose -- and global faceted search certainly implies a choice -- we need to figure out how to do that.

3.2 Uniqueness of name structures in global faceted search

There are cases to be made both for and against uniqueness of names (more precisely, of the combination of the name-string facet and all of the other facets) at this sublayer, and even a partial middle ground, in which names are unique within a registry namespace, but there are mechanisms for identifying such spaces so that the names are unique across the Internet. The community should address the tradeoffs because no position is ideal; summaries of the extreme positions are below. In none of these cases is it necessary, or even desirable, that the name-string itself (without the additional "attribute" facet values) be unique.

3.2.1 The case for unique names

The IAB's discussion of DNS root uniqueness [RFC2826] demonstrates that DNS names must be unique, i.e., that there must not be alternate or surrogate root structures if the Internet is to survive as a seamless whole and be universally addressable and accessible. Even with imprecise matching, part of that argument may apply with the faceted search component, especially if it is the first level at which names and phrases in natural languages (hence including "multilingual" names), rather than constrained identifiers, appear. The mathematical arguments aside, the main argument for uniqueness is that a given combination of name- string and facets will then yield exactly one logical host (or web page, or equivalent, an approach called "direct navigation" in some of the so-called keyword proposals, see [Arrouye]). If this is not the case, it seems inevitable that users will be faced with choices they need to resolve

even when they have an exact match for a full set of facets.

Users clearly prefer such approaches. The often-cited user ideal is to be able to enter a very short and simple string and have exactly the desired resource appear, more or less on a "do (or find) what I mean" basis. Unfortunately, reality has a tendency to intrude on such systems, as discussed below and in [RFC3467].

If the name structures stored in the databases of a global system (faceted or otherwise) still must be unique, some mechanism for registries or structuring of names will be necessary to avoid conflicts. The problem is somewhat easier than the ones encountered by ICANN and its associated groups because the very structuring of the names and attributes creates opportunities for dividing up responsibilities, but the registration problems exist nonetheless and would need to be resolved. Consequently, the requirement for uniqueness is best avoided if that is possible.

3.2.2 Non-unique names

Conversely, one could have multiple appearances of the same set of facets (including the name-string), such that an exact match could still yield multiple "hits". This would have the advantage of eliminating all requirements for monopoly registries or [other] technical mechanisms for guaranteeing that name conflicts did not occur. The disadvantage is that it would force more user choices or heuristics, and at least some errors in which the wrong host or site was identified would be almost inevitable. Issues of non-uniqueness, and having to seek additional information to differentiate among choices, are typical in normal life (see the discussion of "caching", below); it is unlikely that we can make the Internet different.

As extensive use of intelligent per-user (or per-group) local caches or directories ("bookmarks", "favorites", etc.) evolve, which we consider almost certain, they might also make the difficulties with non-uniqueness insignificant. This would be especially likely if these directories contained not only a keyword and (DNS name or URI) target, but also a stored form of the search used so that local data could be recalculated and replenished. See Section 3.5 and Section 3.6 for some related discussion.

3.2.3 A middle ground approach: artificial uniqueness

A proposal was made in the initial version of [Mealling-SLS] that an additional facet could be added to represent the registry which records the names. If this were done, names could be kept unique within registries and would be globally unique as long as the registry-identifying facet had a unique value for each registry.

There would be no need to restrict the number of registries in this model or resolve naming disputes among them -- each one could have a unique, randomly-generated and assigned identifier-- so the approach could provide some degree of technical uniqueness while still preserving most of the benefits of the non-unique approach.

That model could, of course, be deployed at a "registrar" level instead, just by changing the assignment of the identifier facet from value-per-registry to value-per-registrar. Other variations are, of course, possible.

Whether it is desirable or not is an open question, since it inherently turns each registry (or registrar) into an arbiter of priorities or rights to names, or requires that some outside agency perform that function. There seem to be strong arguments for avoiding those alternatives.

3.3 Sources for controlled-vocabulary facets ("attributes")

We anticipate that most of the facets other than the name-string itself will have values chosen from controlled vocabularies. I.e., the user-registrants will be able to select whatever values seem to match their needs, but only from pre-defined lists of possible values. These are not intended as free-text entities; to make them free text would push the second-sublayer system toward the lowered precision of Internet search engines and other free-text search environments. The facet values that are not populated from controlled vocabularies will be determined by deterministic and unambiguous rules. For example if one of these attributes is a geographic location that uses a coordiate scheme, the definition of the coordinate scheme should be sufficient to yield a predicatable and exact value.

The question, then, is how to establish the vocabulary lists and write the definining rules.

It has been something of an Internet tradition, building on Jon Postel's principles for registration and registries, to try to avoid having IETF or IANA become embroiled in controversies about names, their ownership, propriety of using them, and so on. The use of IS 3166-1 alpha-2 codes as the basis for "country code" top-level domain names (see [RFC1591]) is just one instance of the application of this principle. Following this tradition, facets should be chosen, in part, on the basis of availability of pre-existing, well-known lists of names and authorities or, at worst, the ability to identify relatively non-controversial authorities who can quickly establish such lists.

Some specific possibilities are discussed in the subsections that follow.

3.3.1 Discussion of language identification

The IETF already has a standard for identifying languages and dialects, documented in [RFC3066] and based on ISO Standard 636. It appears that it would be usable here, with minimum fuzziness associated with an exact match of all subtags and a higher degree of fuzziness permitting matching different (national or dialect) variations on the same language.

3.3.2 Discussion of geographical identification

For larger countries, and areas with many semi-independent administrative districts, identification of the country may not provide sufficiently precise resolution. On the other hand, it is desirable to have a scheme that is hierarchical or that otherwise readily permits search expansion. Conceptually, the coding should be something like

country / administrative-district / city or town

Fortunately, such a system exists as a generalization of one that is in common use in the Internet. ISO 3166-2 provides a model, and list of values, for representing countries and administrative districts, and is designed to be compatible with the UN/LOCODE list when those further subdivisions are provided and satisfactory from a national point of view. Since ISO 3166 is probably even more satisfactory for this purpose than it is for its use in defining ccTLD names, it should probably be used (with the UN/LOCODE where appropriate) unless something clearly better can be found. For example, a complete coding using this approach would be something like "DE-BW-DESTR" for Stuttgart.

The corresponding matching rules seem obvious, but, to review them:

- o If the stored record contains all three elements, then a query of (and fuzziness=exact) should imply that

   "country" matches everything "country and subdivision" matches all cities in that subdivision, but does not match other subdivisions "country, subdivision, city" matches only that exact stored record.
- o ...???...

The "fuzziness" indicator should be fairly clear here, e.g., 0=exact, 10=match next level ("country, subdivision, city" matches the whole subdivision), 99=all levels ("country, subdivision, city" matches the

whole country), and intermediate values might match adjacent cities or subdivisions using some reasonable distance or adjacency function.

3.3.3 Discussion of Industry Types

The WIPO codes, discussed above, are suitable for companies and industries of the types that normally use the trademark system. They are not a good model for identifying individuals, nor are they suitable for most non-profit organizations, governments, and NGOs. The facet described as "industry type" here will need to be organized so as to identify and accommodate separate classification systems for different types of entities.

3.4 Deployment against the existing DNS base

As with the "new class" approach to DNS changes [NEWCLASS], the approach outlined here does not require any changes to the existing installed DNS base. But, like all solutions to the multilingual name issues, it requires changes to all relevant applications. The notion of moving from lookup to searching does imply that we will need not merely to change the code that calls the name resolution system, but also to rethink the UIs of those applications.

3.5 Thoughts about user interfaces (UIs)

There are many possible models for user interfaces to be used with a system of the type proposed here. The IETF should, as usual, remain agnostic about them. At the same time, some notions about possible user interfaces are important to demonstrate that the concepts are practical and to inform the design of protocol interfaces. So, with the understanding that other approaches are possible, and may be preferable:

As discussions on both DNS "searching" and under the somewhat misnamed topic of "multilingual names", and the general model presented here, have evolved, it has become apparent to some observers that these approaches would be best realized in conjunction with user-specific directories or memory with refresh capability, whether modeled on a local directory, or cache, or history file, or something else. It has been surmised that the behavior of typical users is to spend most of their time using or referencing known services and hosts (whether web sites, hosts used in email addresses, or other services) and much less time "searching" for unknown resources. If this is actually the case, then a typical reference should involve a DNS "name to address" lookup only, even though it would be desirable for the DNS name to not be visible to that user. The user might reasonably see his or her original collection of search terms, or a name assigned to that search or its results, but

actual searching would take place only as a first-time activity or in the process or refreshing the search and results (at user request or, perhaps, automatically). An approach for handling these issues appears in Section 3.6.2.

3.6 Implementation models

While this document is not an implementation specification, nor is it intended to substitute for one, some remarks about implementation issues may be helpful in understanding the concepts that appear elsewhere.

3.6.1 Calling and returning values

While it is not the only way to do it -- and others may be more efficient -- it is useful for this section to consider the database associated with a given server/provider as consisting of a collection of records, each of which consists of a key tuple consisting of one value for each of the facets, a URI, a text field of supplemental (or "comment") information, and an expiration date and/or TTL.

The textual field is expected to be useful to the user (or an agent acting for her) in differentiating between URIs or other target information. Some structuring of it, and/or tagging of the information it contains, would probably be desirable, as would some upper bound on length. A potentially long and highly structured logical content for this field could be provided by the use of a reference via a URI, or the primary URI could provide a pointer to such information that indirects to the actual content being sought.

A query to the server consists of a set of (facet, fuzziness indicator) pairs, one pair for each defined facet. A null value is defined for each facet; use of that value implies that any value for that facet in the database would be considered as matching, regardless of the fuzziness indicator value. A given server may choose to reject a query that specifies null values for one or more facets.

Unless all values for the permitted level of fuzziness are set to disable fuzzy searching and matching, the list of facet values in the query need not correspond exactly to any record in the database for records to be successfully returned.

The server can then return:

o A "not found" response, indicating that no records were found that matched the facets within the specified fuzziness criteria.

o A response that the query is not acceptable, e.g., because of null values for specific facets. This response would indicate the

facet(s) impacted.

- A normal reply. That reply would contain one or more records, each consisting of
  * The facet values as stored in the database
  * The URI as stored in the database
  * The textual comment value, if any
  * A TTL for the response.

This provides information about what was actually matched, suitable for caching and repeating of the query as needed as well as user selection from the records and/or immediate use of the URL.

3.6.2 The cache model

The model proposed here is ultimately one of "searching" -- finding a resource in a way that may involve some interaction between the various databases and the user. It differs from the "search engine" approach because the databases are intentionally populated (not unlike the DNS, although syntax and semantics are different) and the searches are highly structured. With the "search engines", the databases are largely populated by automated processes that walk through the Internet (or the web) picking up data of possible interest. And the queries use, at most, boolean conditions, not a structured, faceted, architecture. The obvious advantage of a search technique of this sort is that it should have good odds of permitting owners or operators of resources who wish those resources to be found to have them efficiently found by those who are looking for them. But it is unlikely that the process can be made as fast or resource-minimizing as a DNS lookup where the fully-qualified domain name is already known.

At the same time, the nature of the systems being proposed makes it likely that a user will think of a particular resource (or its location) in terms of the terminology used to find it. And, especially if the user needs to be involved in the search and resolution process to identify a resource (or resources) of interest from among those that match a set of facets, users will not tolerate having to go through the process on every reference. The combination of these factors makes it almost essential that it be possible to maintain a local table of references and targets (in DNS or URI terms) so that additional references soon after the first can be used by protocols to directly access recently-used targets without having to go through the search and resolution process each time.

The process of locating telephone numbers may constitute a useful analogy here. Few would find it conceivable to approach the telephone system the way the DNS is often approached: to start guessing and dialing numbers, without prior hints, as the way to find the correct number for a particular enterprise or individual.

Instead, people use a number of resources that lie outside the telephone number system itself to locate a number. These resources might include:

- o White pages directories. These support fairly exact search, but with some cross-references, added semantics or qualifiers, and the ability to take advantage of being able to see adjacent listings.
- o Calling a telephone operator and asking. This is typically an indirect method of conducting a white pages search, but may involve additional mechanisms.
- o Yellow pages directories. These are organized by topic, with the topics varying by locality. One first finds the relevant topic, then the entity of interest.
- o Other sources, such as advertisements, articles, or references.
- o The "ask a friend" technique: find someone who knows the relevant number and ask.

In each case, most users will find a way to remember the number if it will be used again. They are entered into personal address books, recorded in intelligent telephones, or remembered in "recently called" lists. These mechanisms are similar to the "cache" being posited here. And it should be noted that the mechanisms used are quite similar to those typically used for email addresses. "Guessing" may be appropriate for a number or email address that is already known, at least to some degree of certainty, but is not generally workable if the target value is unknown.

Even "soon" may be quite different from our experience with the DNS. With the DNS, the resources bound to names are typically IP addresses or something similar, and these change frequently enough, often on short notice, that "time to live" (TTL) values are typically short, rarely more than a day. By contrast, the faceted name resulting from a global search, or the information from a localized one, are likely to represent a fairly long-lived relationship with the DNS names and URLs that they identify. Clearly, it should be convenient for the user to reinitiate the search when that is obviously needed (e.g., if a target goes bad) but TTLs for ordinary refreshing of a search will, in most cases, be set closer to the order of weeks than to minutes or hours, with the DNS taking responsibility for the volatile portion of the references, as it does today.

A facility similar to this is provided for web browsers by the common "bookmarks" or "favorites" functions, but those support only a mapping between a user-specified name and a URI. If the URI goes bad, or the information used to determine it becomes obsolete, there is no mechanism for repeating the search other than the user's memory of what the name might have meant. Caching operations powerful enough to prevent unnecessary searches or user intervention in this environment require much greater functionality.

The facility needed here (extended "bookmarks" or "favorites" if it is useful to think of them that way) can be thought of as a local cache of queries and responses with sufficient information to both immediately locate a target associated with the user's perception of what was looked for and of "refreshing" the search if circumstances changed or values timed out. The idea of a "search" here should be very general, and might even extend to a reference such as "I asked my friend the following question" (we would not expect references of that type to be automatically repeated, but the source information is useful). In the presence of a particular query, a client system would presumably check for a matching cache entry. If one was not found, the specified search would be performed, yielding values that might require user intervention for selection. Once selected, the search, the full set of facets returned, the URIs, and any TTL information would be stored (possibly using a user-supplied name or tag) and the resource accessed via the appropriate DNS name or URI. If the search or tag was found in the cache, checks would be made for the values being current and then the DNS name or URI used directly, without going back through the search procedure.

One useful consequence of this is that the number of queries to the database will be roughly proportionate to the number of new inquiries by the user (increased by the impact of TTL timeout and user-initiated repeated search). That number should be much smaller, and hence imply significantly less load, than per- reference DNS queries.

3.6.3 An example: Looking at Chinese Traditional-Simplified Mappings

One of the problems that the IDN WG has been unable to solve in a satisfactory way is the requirement that strings written with Traditional Chinese ("TC") characters match those written with the corresponding Simplified Chinese ("SC") ones. The relationships among these characters have been variously described as similar to font differences that are not properly reflected in the IS 10646 coding and structure and as similar to case mapping in alphabetic scripts that support case. Although both are thought-provoking, there are significant weaknesses in both analogies. But the problem is sufficiently important that the working group has received requests to delay DNS-level internationalization implementation of all (or selected subsets of) Chinese characters.

Fortunately, mapping between TC and SC is fairly easily handled at sublayer two of the system proposed here. Details and variations still need to be worked out and a specific proposal chosen and refinded, but it appears that something similar to the following outline would be one option:

Unlike the DNS, the sublayer two system will have the critical language identification information available. This eliminates the problems associated with distinguishing Chinese character usage from uses of similar characters (at the same IS10646 code points) in Japanese and Korean. Assuming that the language is Chinese, "fuzziness" could be used to determine the precision of matching required. E.g., "no fuzziness" might be construed as "exact match", i.e., no attempt at TC-SC matching. A low (but non-zero) fuzziness value might permit unambiguous single- character (i.e., "one to one") matching between TC and SC characters, but no other variations. And a higher degree of fuzziness might match more extensively, including cases in multiple characters of context or user selection from a menu or pick list were needed to determine a correct match.

[[Note in draft: Is distinguishing between these two cases actually helpful? I would think it would be useful in the design of good-quality user interfaces]]

If it were worthwhile, other variations on a sublayer two system could be used to handle different character input models as server functions. For example, use of a different language subtype (or a heuristic on the name string) could permit phonetic input (presumably Pinyin, but, if anyone wanted it, a different subtype could permit use of alternate systems such as Wade-Giles) even though the names in the database were stored in Chinese characters. Use of phonetic input of course absolutely requires matching of TC and SC characters.

An interesting approach, using language-specific variant tables, appears in [JET-Guideline]. That approach is designed to work in conjunction with IDNA and involves client-based variant tables. With obvious modifications to bring the tables onto the server and to treat some of the variants as different degrees of approximations, it might be even more effective in the environment proposed here.

3.6.4 An example: Distance functions and Latin-based alphabets

The discussions of case mapping for scripts in which the rules are subtle or culturally dependent has restarted the argument in some quarters as to whether the case-mapping rule of the DNS was wise. The alternate position is that users are better off with a single form of writing an identifier and that they will then "get used to getting it right". The use of fuzziness with such scripts might permit this issue to be left to the user or interface designer, e.g., no fuzziness would imply no case matching, somewhat more fuzziness would permit case matching in those cases where the rules were exact and one-to-one, and additional fuzziness would permit matching, e.g., with and without diacritical marks or across character variants. The presence of language information makes these approaches much more

workable than they would be with the DNS, even with a more complex canonicalization process than is now anticipated in "nameprep".

3.7 Older applications

To fully realize the benefits of internationalized naming requires changing all relevant applications to understand the new method, whatever it is. Even the "internationalize the DNS" proposals are subject to this principle. Older applications will see distorted and unfriendly names under some systems, and no names at all under others (some approaches might cause implementations of some applications to fail entirely).

The environment contemplated here is a "no international names in old applications", i.e., "no new names without upgrading", one -- applications that have not been upgraded will not see internationalized names or other natural-language phrases, nor coded surrogates for them.

The advantages of a "no names without upgrading" approach are that it avoids confusion and the risk, however slight, of catastrophe. As with the original host table to DNS conversion, they provide an incentive to convert old applications to make newer naming styles, and newer names, visible. None of these transitions are ever easy, but it may be worth going through this one to get things right, rather than investing a large fraction of the pain to get a solution that doesn't quite do the job.

4. Context and model details: Localized and Topical Searching

...to be supplied??...

5. Comparisons to existing and proposed technology

5.1 The IDN Strawman

After the IETF IDN working group came into being, it rapidly excluded all models not based on the assumption that internationalized name referencing issues and requirements -- including the requirements, not heretofore satified even for ASCII-based names, to be able to search for things using the DNS-- could be achieved by placing non-ASCII identifiers into the DNS itself, in some coded form. These identifiers were commonly described as "multilingual names". That terminology further complicated the work program and consensus-seeking process in that working group.

Many of the problems associated with trying to overload the DNS in this way have been described in [RFC3467]. That document, and the

experience from which it is drawn, predict that the products of the IDN WG effort, specifically the IDNA standard, will ultimately fail if they are evaluated in the context of applications that require sensitivity to the characteristics of particular languages, rather than just an expanded set of characters to be used in identifiers. As implied in the "DNS role" [RFC3467] document, consideration of language-related issues and their appropriate handling was one of the primary motivations for the model developed here.

However, at least from the viewpoint of this author, one important question remains: assuming that the IDN WG's work is carefully confined to characters and identifiers, does the value of local-language identifiers justify putting non-ASCII strings into the DNS even if end users never see them? We argue in section 2.1 that it is not necessary and poses some risks. However, the "variables in programming languages" analogy and the "local directory or cache" approach, both outlined above, suggest that such names would be extremely useful and fairly safe if the limits of code-point-level matching and identifier-only use are taken narrowly and observed conservatively [ICANN-Permitted]. And, if one believes the model outlined here, or any competing "keyword" model (see next section), will achieve wide deployment and use, the needs and perspectives of such systems should condition the evaluation of IDN WG-produced alternatives. So there is a serious and complex set of engineering (and, realistically, political) tradeoffs to be evaluated in making the decision as to whether wide deployment of some version of the IDN work is appropriate.

5.2 "Keyword" systems

In the Internet object-referencing context, the term "keyword system" has been used to refer to many different things. Many would fit nicely into the localized search environment, but most of the existing proposals put them directly on top of the DNS, or skip the DNS entirely and go directly to IP addresses. The difficulty with these systems is that they either must be localized (e.g., a different system or database for each language, country, or smaller locality) or they don't scale well. Arguably, some don't scale well even if localized. In particular, they eventually suffer from either the "all the good names are taken" problem (of which the DNS is frequently accused) or they are very vulnerable to poor retrieval precision properties as the number of names (or keyword combinations) in the name space grows large. I.e., like so many other ideas for the global Internet, they work in constrained environments, but cannot adjust well to large scale.

Adapting bibliographic styles of keyword systems to operate locally as part the models proposed here would appear to be the best way

forward for such systems. It has been observed that what most users really want most of the time is localization, and locally-oriented keyword systems could satisfy much of that requirement. Keyword systems would also be strengthened by being placed on a base of use and language-sensitive naming and searching, and a very strong local cache, rather than on the low- context, monohierarchical, DNS.

Other types of keyword systems, including the one described by Arrouye and Popp [Arrouye], are really special cases of the global faceted search service. They rely on careful selection of names (and, consequently, resolution of "best fit" and "rights") to achieve uniqueness and, hence what they describe as "direct navigation" (see elsewhere in this document). Similar systems might utilize a set of keywords combined into a phrase that can be interpreted, possibly with permutation rules, in a search service. In the interest of simplification and presenting simple names to users, these systems are likely to omit most or all of the non- name string facets from user-visible search interfaces. Some further analysis, as to whether what is optimally desirable is a set of unordered keywords, or an ordered phrase that might contain such keywords, seems called for. Different answers could, of course, be implemented in different components of this model.

5.3 Client-side and server-side solutions

IDNA, and other key approaches considered during the IDN WG's deliberations, are essentially client implementations, applied to names before they are placed in the DNS and to queries before they are passed to the DNS. This contrasts with the existing use and protocols of DNS in which, e.g., string matching is done on the server. Ignoring speed of deployment (which can be argued either way), the advantage of client-side implementations is that they don't require changes to the DNS fabric itself (and therefore minimize the risk of damaging existing applications that rely on that fabric). Because the mechanisms discussed here do not rely on the DNS for any searching or matching activities, and are completely new, server-side implementations are again feasible: applications will require modification to access these services (just as they would to support a client-side implementation), but older, unmodified, applications will not touch them at all.

Server-side implementations have several advantages over client- side ones. If something complicated is being done, it is often possible to apply more computer resources, or larger tables, on a server, and to update those resources and tables more easily if needed. And server-side implementations tend to yield more uniformity of behavior relative to having a potentially wide mix of client implementations.

However, integrity protection procedures that depend on similar computations on client and server, such as those that rely on digital signatures computed over the data, may not eliminate the requirement for client-side computations. See Section 10

6. Comments on business models

Historically, the IETF has had even less desire to involve itself with business models than it has with user interfaces (see Section 3.5). But the approach outlined here, and the protocol and operational proposals that will derive from it, face a particular challenge: the DNS works well for its intended purpose (something we don't intend to change) and arguably works at least tolerably for some purposes, including as a search engine, for which it was not intended. Many of us see its quality and capabilities, when used as a search (or, more accurately, "guessing") engine deteriorating. Collapse, if it occurs, is still in the future if it occurs at all, although recent trends seem to point to less dependence on the DNS before that system passes its critical point (see [RFC3467] for more discussion on this topic). There are also considerable vested interests -- both economic and policy control-- associated with the current DNS structure and arrangements.

he ability to produce and deploy a different model, especially one that requires new work in several areas, against that backdrop will be challenging at best. Unless there are clear business models for doing so, the odds of success are quite low. So this section outlines some of the business issues and models not covered elsewhere in this document. As with the user interface discussion, it is not intended to be definitive: some of these models may fail and others may be more attractive. But it is intended to provide a sufficient demonstration of concept to perhaps permit the technical ideas to be taken seriously.

We observe that a telephone system analogy may be helpful. With the telephone system, there are registries, described as national numbering databases, that record which numbers are in use and by whom. There are white pages services which, given locale and some other information (e.g., whether business or residential in some areas) and a near or exact match to a name, provide name to number lookup. And there are yellow pages services, with precise categories and organization differing somewhat from one location to another. Organizations make money at all three levels, but the greatest aggregate income occurs with the yellow pages services.

At each of sublayers two and three, there are multiple services. Some of these would probably need to be operated as public goods, spreading costs over the producers of other services. Others would

presumably be directly profitable.

6.1 Faceted global searching

6.1.1 Facet listings and identification

For the attribute facets that rely on controlled vocabularies, some organizational structure would be required to oversee those vocabularies. As suggested elsewhere, the ideal would be to use pre-existing organizations and pre-existing lists (the WIPO classification of goods and services [WIPO-NICE] is an example of such a list, as would be the IS 3166-1 list traditionally used for country code domain names. Where such lists did not exist, it would be necessary to build arrangements for them. The maintenance of such vocabularies would, from a global Internet standpoint, be a public good.

6.1.2 Registration and searching

Actual registrations would be required for names and their attributes with, as mentioned above, multiple registrations when an individual, organization, or business wished to be registered with more than one attribute set. The economic model would presumably parallel the current registrar and registry business, with a charge for registration (since there is no intrinsic requirement for a single registry, registry services might well be competitive, eliminating the need for models that separate registries and registrars. However, lookup and search activities would be more flexible than the DNS, with extended services, including character set transposition, language translation, and potentially more extensive search variations being potential areas on which providers could compete, using fee for service or subscription models to support costs.

6.2 Localized and topical databases and searching

As mentioned above, yellow pages and publication of directories and guidebooks are traditionally where the money has been made. The analogies apply: one could imagine charging for entering information into the databases, or for searching, or for information delivered, or all three of these. And all have been used for papers and related databases.

7. Glossary

...??See Placeholder...

ACE ...To be supplied...
Encoding form ...To be supplied...
Facet ...To be supplied...
ISO10646 ...To be supplied...
Keyword (see Section 5.2)
Multilingual name (see Section 5.1)
UCS-4 ...To be supplied...
Unicode ...To be supplied...

8. Summary

The solution to the "multilingual DNS" problem, and to a series of other limitations of the DNS relative to today's expectations for naming and searching, lies in solutions targeted to those problems, rather than superimposing additional mechanisms on the DNS in ways that, those who advocate them hope, will not cause problems with older programs and unconverted infrastructure. Inserting new search layers avoids those risks and permits a clean solution that is adapted to the problems, rather than the limitations imposed by existing properties of the DNS.

9. IANA Considerations and related topics

At search layer two, it is difficult to think about how the system might function successfully without controlled vocabularies for each of the non-name facets. As discussed in section 2.2, we have already established one such registry (bound to an ISO standard), and mechanisms for utilizing it, with RFC 3066. The Madrid agreement and its predecessors [WIPO-NICE] provide classifications for types of businesses, but we would need to extend the registry for names that are not business-related. The two locational attributes are somewhat vague at this point, but controlled vocabularies would presumably be needed, and should, if possible, be drawn from stable, non-IETF, work (e.g., IS 3166-1 and 3166-2 might provide a foundation, and possibly a complete list, for the location vocabulary). Curiously, there is no technical reason why the name-strings themselves must be unique: that is one of the attractions of a model like this over attempting to overload the DNS. If conflicts or confusion occur, those are standard civil (marketplace or trademark) issues that can be resolved in their own environments, rather than posing special Internet problems.

10. Security Considerations

Additional layers of naming, searching, and databases imply addition of opportunities for compromising those databases and mechanisms. Part of the challenge with the model implied here is to determine how to secure and authenticate those databases and access (especially

modify access) to them. The good news is that, since the functions are new, we should be able to design security mechanisms in, rather than --as with the DNS-- have to try to graft them on to a structure not designed for them.

A particular issue is integrity protection of responses and possibly queries, analogous to the capabilities DNSSec is expected to provide for the DNS [DNSSEC??]. It would be desirable to avoid having to make potentially-complex signature computation on both clients and servers (as in DNS). One approach would be to authenticate the source and verify transmission integrity, but that may or may not be sufficient.

11. Acknowledgments

This document, and some related notes, are the result of thinking that has come together and evolved since before the issue of internationalized access to domain names came onto the IETF's radar. Discussions with a number of people have led to refinements in the approach or the text, even though some of them might not recognize their contributions or agree with the conclusions I have drawn from them (indeed, some of those discussions were rooted in challenges to the general ideas expressed here). Particularly important suggestions have come from, or arisen out of conversations with, Ran Atkinson, Harald Alvestrand, Rob Austein, Fred Baker, Christine Borgman, Eric Brunner-Williams, Randy Bush, Tim Casey, Vint Cerf, Kilnam Chon, Dave Crocker, Leslie Daigle, Patrik Faltstrom, Michael Froomkin, Francis Gurry, Marti Hearst, Paul Hoffman, Kenny Huang, Marylee Jenkins, Dongman Lee, Xiaodong Lee, Karen Liu, Mao Wei, Michael Mealling, Erik Nordmark, Gary Oglesby, Mike Padlipsky, Qian Huilin, James Seng, Theresa Swinehart, Tan Tin Wee, Len Tower, and Zita Wenzel, as well as some memorable long-ago conversations with Jon Postel and J.C.R. Licklider.

The first version of this Internet Draft was posted in May 2001, after fairly extensive public discussion of the underlying issues and required technology during the preceeding nine months.

12. Changes between versions

12.1 Major changes between version 05 and 06

This version continued to tune the document, completing and clarifying the transition between strict layering and alternative components above the DNS. Some other issues have been clarified and more details filled in.

12.2 Major changes between version 04 and 05

After considerable discussion (little of it, unfortunately, involving the IRNSS list) about return values and "layering", this version considerably restructures the model of both. In the return value case, the output of faceted search is now specified as containing one or more URIs, not just DNS name(s). There is some discussion of the motivation for this in the text, but the key issue is that users are almost certain to search for resources that make sense to them: the distinction between a DNS name and a URI is too subtle and the latter does not adequately locate user-visible resources.

The layering issue is more significant, at least in terms of this work and how it has been presented. Our original assumption was that the localized and topical searches would rest on top of the faceted ones, producing faceted values as outputs. After more discussion and examination of likely cases, it has become clear that the two are better thought of as independent and complementary models, with each other and with web searches -- all still layered on top of the DNS. The text has been changed to reflect this, but might not yet be completely consistent with it. Pointers to omissions would be appreciated.

13. Placeholders

For some reason, new ideas or approaches, or ways of presenting or clarifying existing ones, seem to arise immediately before a version of this document is submitted for posting. It has often been impossible to properly incorporate these. The following are pending, and will be picked up in the next revision:

1. A new section Section 3.3.3 on "Discussion of Industry Types" that will introduce a better model (and less handwaving) for handing industry type codes where they are appropriate and structuring data for that facet for other types of names. Version 05 contains the beginnings of that discussion.
2. The discussion of Localized and Topical Searching, for which Section 4 is a placeholder, must still be completed.
3. Section 3.6.3 should be reviewed with people who actually understand the language and issues and then rewritten (this task may have been superceded by the ongoing work with [JET-Guideline]).
4. Section 3.6.4, which is now just an outline, needs to be filled in.
5. Completion of the glossary, which seems to be necessary for readers who have not been immersed in, e.g., the discussions of the IDN WG. This work should be reviewed in the light of the definitions in [RFC3536] and elsewhere.

6. The material on calling and return values (Section 4), as originally prepared for draft version 02, was not coherent and has been replaced. The material there on the textual ("comment") field should be carefully reviewed -- it may not be right. (From 04 -- no comments yet received.)
7. The new section Section 2.2.6 may need to be expanded and discussed further.
8. In versions prior to -04, Section 3 was largely a discussion of what was then called search layer two (i.e., global faceted search) issues, with a few asides about search layer three (i.e., localized search). With version 04, that distinction has been made explicit and a new section four inserted as a placeholder for a similar discussion about search layer three. That section should be supplied for version 05.

Most of the references in this document are to examples of approaches to the systems outlined here, or provide additional information about the context of some of the suggestions, or are included to give credit for particular ideas or to better identify earlier and approaches. None of those references are normative in the protocol sense typically used in the IETF.

Normative References

[RFC0882] Mockapetris, P., "Domain names: Concepts and facilities", RFC 882, November 1983.

[RFC0883] Mockapetris, P., "Domain names: Implementation specification", RFC 883, November 1983.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2026] Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

[RFC2826] Internet Architecture Board, "IAB Technical Comment on the Unique DNS Root", RFC 2826, May 2000.

[RFC3066] Alvestrand, H., "Tags for the Identification of Languages", BCP 47, RFC 3066, January 2001.

Informative References

[Arrouye] Tan, T., Lee, X. and Y. Arrouye, "Keywords Systems - Definition and Requirements", draft-arrouye-keywords-reqs-01 (work in progress), February 2002.

[Austein2001]
Austein, R., "Private communication", 2002.

[HOSTNAME]
Harrenstien, K., Stahl, M. and E. Feinler, "Hostname Server", RFC 953, October 1985.

Also Braden, R., ed. "Requirements for Internet Hosts - Application and Support", RFC 1123, October 1989.

[I-D.klensin-registration]
Klensin, J., "Registration of Internationalized Domain Names: Overview and Method", draft-klensin-reg-guidelines-02.txt (work in progress), February 2004.

[ICANN-Permitted]
ICANN IDN Committee, "Briefing Paper on IDN Permissible Code Point Problems", February 2002.

Also see "Internationalized Domain Names (IDN) Committee: Final Report to the ICANN Board", http://www.icann.org/committees/idn/final-report-27jun02.htm, 27 June 2002.

[ISO10646]
International Organization for Standardization, "Information Technology - Universal Multiple-octet coded Character Set (UCS) - Part 1: Architecture and Basic Multilingual Plane", ISO Standard 10646-1, May 1993.

[ISO5127] International Organization for Standardization, "Information and documentation -- Vocabulary", ISO Standard 5127, 2001.

[JET-Guideline]
Seng, J., "Internationalized Domain Names Registration and Administration Guideline for Chinese, Japanese and Korean", draft-jseng-idn-admin-05 (work in progress), October 2003.

[Mealling-SLS]
Mealling, M. and L. Daigle, "Service Lookup System (SLS)", draft-mealling-sls-02 (work in progress), July 2002.

[NEWCLASS]
Klensin, J., "Internationalizing the DNS -- A New Class", draft-klensin-i18n-newclass-02 (work in progress), June 2002.

[RFC1591] Postel, J., "Domain Name System Structure and Delegation", RFC 1591, March 1994.

[RFC1625] St. Pierre, M., Fullton, J., Gamiel, K., Goldman, J., Kahle, B., Kunze, J., Morris, H. and F. Schiettecatte, "WAIS over Z39.50-1988", RFC 1625, June 1994.

[RFC2251] Wahl, M., Howes, T. and S. Kille, "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997.

[RFC2345] Klensin, J., Jr, T. and G. Oglesby, "Domain Names and Company Name Retrieval", RFC 2345, May 1998.

[RFC2396] Berners-Lee, T., Fielding, R. and L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, August 1998.

[RFC2772] Rockell, R. and B. Fink, "6Bone Backbone Routing Guidelines", RFC 2772, February 2000.

[RFC2822] Resnick, P., "Internet Message Format", RFC 2822, April 2001.

[RFC2825] IAB and L. Daigle, "A Tangled Web: Issues of I18N, Domain Names, and the Other Internet protocols", RFC 2825, May 2000.

[RFC2972] Popp, N., Mealling, M., Masinter, L. and K. Sollins, "Context and Goals for Common Name Resolution", RFC 2972, October 2000.

[RFC3454] Hoffman, P. and M. Blanchet, "Preparation of Internationalized Strings ("stringprep")", RFC 3454, December 2002.

[RFC3467] Klensin, J., "Role of the Domain Name System (DNS)", RFC 3467, February 2003.

[RFC3490] Faltstrom, P., Hoffman, P. and A. Costello, "Internationalizing Domain Names in Applications (IDNA)", RFC 3490, March 2003.

[RFC3491] Hoffman, P. and M. Blanchet, "Nameprep: A Stringprep Profile for Internationalized Domain Names (IDN)", RFC 3491, March 2003.

[RFC3536] Hoffman, P., "Terminology Used in Internationalization in the IETF", RFC 3536, May 2003.

[SHI-CACHING]
Shi, X. and K. LIU, "Caching Mechanisms in Layered DNS Search Services", draft-xhshi-dns-search-caching-00 (work in progress), October 2002.

[WIPO-NICE]
World Intellectual Property Organization, "", June 1957., "Nice Agreement concerning the International Classification of Goods and Services for the Purposes of the Registration of Marks", June 1957.

[WangLiang2003]
Wang, L., Guo, Y. and F. Fang, "Information Retrieval Protocol for Digital Resources", draft-liang-irpdl-03.txt (work in progress), November 2003.

[XDLee-cnnamestr]
Lee, X. and Y. WANG, "Chinese Name String in Search-based access model for the DNS", draft-xdlee-cnnamestr-01 (work in progress), November 2002.

[Z39-50]   American National Standards Institute, "Information Retrieval: Application Service Definition and Protocol Specification", ANSI Z39.50, ISO Standard 23950, 1995.

Author's Address

John C Klensin
1770 Massachusetts Ave, #322
Cambridge, MA 02140
USA

Phone: +1 617 491 5735
EMail: john-ietf@jck.com

Intellectual Property Statement

Full Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.