Abstract

The established host-centric networking paradigm is challenged due to handicaps related with disconnected operation, mobility, and broken locator/identifier semantics. This paper soberly examines another topic of great interest: distributed information object resolution. After recapping the notion of an information object, we review object resolution in today’s Internet which is based on Uniform Resource Identifiers (URIs). We revisit the implications of DNS involvement in URI resolution and discuss how two different types of content distribution networks work with respect to name resolution. Then we evaluate proposals championing the replacement of DNS with alternatives based on distributed hash tables. We present the pros and cons and highlight the importance of latency in resolution. The paper positions these issues in the context of a Network of Information (NetInf) and concludes with open research topics in the area.

1. Introduction

A new networking paradigm called Network of Information, or NetInf, was recently introduced in the open literature [1]. In contrast with the dominant host-centric paradigm, based on perpetual connectivity and the end-to-end principle, NetInf adopts an information-centric approach, concurring with Jacobson and his colleagues [2], [3] on the need for new architectures focusing on content, not transport. Jacobson points out [2] that the Internet has for some time now been used for purposes not in its original design, which aimed at increasing resource sharing among participating hosts. This is why the TCP/IP paradigm is fundamentally concerned with reliability and consistency, having survivability and simplicity as its most important guidelines [4]. Nonetheless, an Internet based on client/server architectures is not very well-suited for trustworthy communication or information dissemination, and several add-ons were introduced.

In today’s Internet, information is segregated by its means of production and consumption. In NetInf, information takes center stage, irrespective of the means of production, consumption and transport. A key NetInf enabler is the overarching notion of information elements, two types of which are defined: data objects, which correspond to the bits and bytes of digitized content, and information objects (IOs). IOs are loosely defined in [1] through their main use, which is to identify other NetInf objects. Uniform Resource Identifiers [5] play a similar but more limited role today1.

This paper is organized as follows. Section 2 reviews the role of URIs, the Domain Name System (DNS), and content distribution networks (CDNs) in delivering information in today’s Internet. After recapping recent developments in DNS and CDNs, Section 3 evaluates proposals advocating the replacement of fundamental Internet machinery (such as the DNS) [6]–[8] with solutions based on distributed hash tables (DHTs), such as Chord [9], [10], and examines their suitability for NetInf. Section 4 identifies four aspects that will play a key role in future object resolution design-time choices. Finally, Section 5 summarizes the paper and outlines future research directions.

2. Information Object Resolution Today

This section explores the notion of IOs and relates it with URIs. An IO is loosely defined in [1] as an object that can “identify other objects irrespective of their particular digital representation.” The authors do not provide a concrete IO format, but do list some usage examples. For instance, an IO can refer to a given symphony encoded using different algorithms and configurations, such as, say, MP3 at 128 kb/s and 192 kb/s and AAC+ at 64 kb/s, all of which refer to the same content, thus being semantically the same. An IO about this particular symphony would point to all different encodings, and it is left to the NetInf client application (and other machinery) to determine which encoding to retrieve based on a set of criteria. Another example could be an academic paper, health report, or news site, which are made available using different formats: HTML for online browsing and PDF for hard copies (or to preserve a certain layout) both of which convey the same material. Although on a first reading this resembles with content negotiation in the current Web, it is actually more involved, as we will see later.

1. URIs are one of the three pillars that facilitated the widespread adoption of the WWW (the other two being HTML, for describing information/content, and HTTP, for transporting content). Since their introduction, however, they have been embraced and extended by other applications ranging from peer-to-peer (P2P) overlay networks, to VoIP and audio/video (A/V) streaming, to client-side and even serverless AJAX applications.
2.1. URI Resolution

URIs are clear-text, globally unique identifiers, acting as information pointers [5]. They are defined as a sequence of characters, not a sequence of octets and, in theory, there is no length restriction\(^2\). Typically, URIs are human-readable but do not have to be. URIs are flexible and extensible, allow for mash-ups, can be used to assemble documents dynamically, and there is extensive and tested code base for handling them in all programming languages. Then again, URIs are not self-certifiable, can be spoofed and, due to their readability, typos can lead to errors in accessing content.

URI resolution involves both local and remote entities (see Table 1). Consider the following hypothetical URI:

\[ U_1: \text{http://example.org/projects/?current#publications} \]

When retrieving the object pointed to by \( U_1 \), the following resolution steps take place, assuming all caches are empty.

**Step 1: Determine means of transport.** The browser determines it has to retrieve a page/object from the host with the name `example.org`. The URI scheme dictates the use of HTTP\(^4\) (explicitly in this case) and the establishment of an end-to-end connection to the respective default port (80, implicitly, as no port is given in the URI) of the host. The next step in URI resolution is to determine the IP address of `example.org`, which implies the invocation of DNS.

**Step 2: Determine peer endpoint.** The client name resolver, which is part of the application [12], contacts the local (preconfigured) name server asking for the IP address of `example.org`. The “local name server” may be running either on the same machine as the browser or on a different (but hopefully nearby, in network terms) host.

a. The local (recursive) name server asks a root name server for the IP address of `example.org`. The root name server does not return the requested IP address, but a list of name servers from the corresponding top-level domain (TLD), i.e., `org`.

b. The local name server contacts the TLD name server(s) asking for the IP address of `example.org`. They too do not return the requested IP address, but delegate the request pointing to a list of name servers for the second-level domain (SLD) `example.org`. They too do not return the requested IP address, but delegate the request pointing to a list of name servers for the second-level domain (SLD) `example.org`.

c. The local name server contacts the SLD name server for `example.org`, which resolves `example.org` to its IP address\(^6\). Once the hostname is resolved, the browser can continue to the next step.

**Step 3: Server-side object resolution and delivery.** The browser issues an HTTP GET request for the object pointed to by \( U_1 \). Upon receipt of this request the web server will:

a. map the WWW-visible path `/projects/` to a local file system resource\(^7\),

b. unfold “/projects/” to, say, “/projects/index.shtml” assuming that server-side includes (SSI) are used [13],

c. resolve the argument “current” within the given context and dynamically return the resulting content to the requesting client. The last URI part, “#publications”, is not meant to be resolved by the web server. It is an anchor point to be resolved in the following step.

**Step 4: Client-side object resolution and delivery.** The browser receives the (raw) content, renders it, and focuses on the #publications section. Further local processing may be required. Java and AJAX applications, for example, deliver the actual content after they run locally.

Note that I oversimplify the details in all steps listed above; see [4], [12], [14], [15] for more details. Here the emphasis is on highlighting aspects that often go unnoticed and acknowledge the diverse and distributed nature of this process as well as the ample opportunities for caching. For example, although the reader was originally interested in reviewing the publications list only, she decides to follow a link pointing to the list of project patents

\[ U_2: \text{http://example.org/projects/?current#patents} \]

which is in fact a distinct URI pointing to the information object “list of patents from the current projects”, but its resolution does not mandate following but Step 4 above.

On the other hand, a subsequent resolution of

\[ U_3: \text{http://example.org/projects/?past#deliverables} \]

requires going through Steps 3 and 4, though establishing an new TCP connection may not be necessary if HTTP 1.1 Keep-Alives are used.

2. In practice, modern browsers handle URI lengths of 100000 or more characters; servers impose stricter limits (approx. 10000 characters) [11].

3. Note the host-centric approach. URIs are agnostic to the content they are pointing to, and no assumptions about the information object should be made. Instead, a URI uniquely identifies a particular endpoint, on a global scale, were content is expected to reside. Content-specific information will be available once the web server at `example.org` is contacted.

4. ...which dictates the use of TCP, even if the request is simply meant to initiate A/V streaming, for example. If a different scheme is given, the segregation of information based on means of transport becomes evident.

5. On some platforms hostname lookup requests are blocking calls, unless proper care is taken when implementing the application.

6. The SLD name server can report other canonical names (CNAME) of the particular host, say, `www.example.org`, for future reference.

7. It may also need to map the request to one of several virtual hosts.

<table>
<thead>
<tr>
<th>Step</th>
<th>Entity</th>
<th>Processing</th>
<th>URI component</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Browser</td>
<td>Local</td>
<td>Scheme⇒protocol, port number</td>
</tr>
<tr>
<td>2</td>
<td>Name resolver</td>
<td>Local</td>
<td>Host name</td>
</tr>
<tr>
<td>2a</td>
<td>Root server</td>
<td>Remote</td>
<td>Host name</td>
</tr>
<tr>
<td>2b</td>
<td>TLD auth. server</td>
<td>Remote</td>
<td>Host name</td>
</tr>
<tr>
<td>2c</td>
<td>SLD auth. server</td>
<td>Remote</td>
<td>Host name</td>
</tr>
<tr>
<td>3a-c</td>
<td>Web server</td>
<td>Remote</td>
<td>Pathname</td>
</tr>
<tr>
<td>4</td>
<td>Browser</td>
<td>Local</td>
<td>Document anchors</td>
</tr>
</tbody>
</table>
Modern client-side technologies, including CSS, AJAX, Java, and Flash use URLs for serverless and asynchronous applications that go well beyond the typical client/server request/reply model. URLs proved indispensable in accessing information on the Internet. NetInf will need an equally powerful mechanism for accessing resources.

2.2. DNS: Then and Now

The Domain Name System (DNS) is a hierarchical distributed database [12], which is effectively employed by nearly all Internet applications. Its main task is to translate world-wide unique names into other identifiers [15]. Waltfish et al. noted that the “marriage-of-convenience” between DNS and WWW was constraining both infrastructures, concurring with earlier literature on the need to move away from host-based URIs; see [6] and the references therein. Before revisiting [6] in Section 3, it is important to point out that the state of DNS changed considerably since 2003.

Fig. 1 illustrates that the number of root name server sites alone has increased ten-fold during the last six years, according to data from www.root-servers.org. At the time of writing [6] (September 2003), root servers were located at 35 different sites; at the time of this writing (October 2008), root servers are located at nearly five times as many sites8. Fig. 2, based on data from www.root-servers.org on 2008-10-14, depicts the geographic distribution of root name server locations. Root servers are deployed in 57 countries in October 2008. Still, the geographic implications of root and TLD server locations will remain a contention issue [14].

According to VeriSign Domain Reports, 61 million domain names were registered by the end of 2003. In Q2 2008, the number of registered domain names had nearly tripled to 168 million. DNS is already handling name/identifier mappings in the order of hundreds of millions. DNS scaled well9, despite fears of imminent and extensive network performance degradation amid criticism about policies. Or, it might just be that the received criticism led to developments that otherwise would have never occurred.

Next we review the Akamai CDN, which capitalizes on DNS to deliver both complete web sites and specific objects, in particular those that are highly dynamic, focusing on the object resolution mechanisms employed. Then, we will consider object resolution in a popular P2P overlay network.

2.3. Object Resolution in the Akamai CDN

Akamai operates a CDN of over 34 000 servers (Q3 2008). The CDN comprises DNS servers, which play a crucial role in improving overall performance, and web servers which deliver the actual objects to requesting clients. Akamai makes extensive use of caching, has presence in some 70 countries and has deployed servers in nearly 950 networks10. These numbers are not accidental. Although the number of deployed servers has increased by a factor of 17 since 1999, the number of countries has remained in the range of 65-70 all along. The goal is clear: bring content (via replication, caching, and pre-fetching) closer to the users that matter and invest in infrastructure to deal with the rising demand.

---

8. Of course, a single site includes more than one physical server, although exact details are not publicly available. Overprovisioning is key and anycasting has been widely adopted. According to [16], more than 95% of root name servers use anycast and so do all large generic top-level domains (gTLD) name servers. Since February 2008, six root servers (A, F, H, J, K, M) introduced AAAA records, facilitating transition to IPv6.
9. VeriSign reportedly processed more than 48 billion DNS queries per day in Q2 2008, at peak. This is double the daily peak in the beginning of 2007, or more than one request per microsecond, at peak, solely at one operator. Karrenberg [15] similarly reports 10 billion queries, on average, every day on all root name servers in September 2007, when the monthly average load of all root name servers was more than “118 000 queries per second (90 000 in December 2004) with regular peaks at several times the average which are handled gracefully.”
10. The Akamai web site claims that 85% of web users are “a single network hop” from an Akamai server and estimates that its CDN handles 10-20% of all web traffic. This is not easy to verify. Akamai reports that its CDN handles over 300 billion requests per day. This is impressive though it is not clear what exactly constitutes a “request” (for example, a page inclusive of all embedded objects or each object delivered), and whether “web traffic” refers to requests, objects or bytes delivered.
The Akamai CDN uses URIs to identify information objects, following the same approach as the resolution described in §2.1, but employs more complicated server selection algorithms. Pan et al. [4] provide an illustrated survey of the Akamai CDN explaining the role of DNS in server selection. In typical (non-Akamai) web retrieval (refer to §2.1) up to six nodes are involved: the web client and server, as well as the local, root, TLD, and SLD name servers. There are up to five request/reply exchanges, including the start of data flow, and assuming no cache hits.

For Akamaized sites, according to [4], in addition, another three nodes are involved: two of them are part of the internal Akamai dual DNS ring, while a third delivers the Akamaized object(s). All in all, the number of request/reply pairs more than doubles to 11. Nevertheless, due to the extensive and widely dispersed infrastructure, Akamaized sites do tend to outperform traditional sites, on average.11

Pan et al. [4] point out that the main advantage of a DNS-based CDN is that it capitalizes on existing infrastructure and does not pose new requirements for end-user software. I add that commercial CDNs have also benefited from widespread infrastructure spending during the last decade. Besides the deployment advantages, DNS-based CDNs have operational and design pros; see Section 4. Although commercial CDNs may not be perfect, they do bode well with network operators and their policies and allow for governmental control. P2P-based CDNs do not exhibit such architectural traits.

2.4. Object Resolution in BitTorrent

BitTorrent networks [18] are also fundamentally based on URI resolution.12 The main architectural elements include the seed, the tracker, and the leech; see [8], [19], [20] for overviews. At the heart of this content distribution scheme are static files called torrents, which provide information about the file to be distributed, such as, its length and name, along with a hashed value (checksum of the file). Certain integrity checks ensure proper object distribution. Each .torrent file is in fact a metafile describing essential information about each object in the BitTorrent network.

Object resolution in BitTorrent has three levels of redirection (hostname to IP address, URI resolution to reach the tracker, dynamic resolution of the list of peers willing to seed for the particular .torrent) which include DNS and web-server-like object resolution. In contrast with the Akamai CDN, however, a BitTorrent network is almost self-managed, although end-user participation, along with their contributed resources in terms of storage, network bandwidth and processing is essential. Without them, no BitTorrent network would exist.

3. Information Object Resolution with DHT

Distributed hash tables (DHTs) have captured the imagination of the distributed computing world lately. In this section we will review proposals to replace (parts of) the URI resolution mechanisms with DHT-based architectures. The list of proposals examined below is not exhaustive, but it is representative as it includes pertinent seminal works.

3.1. On Replacing DNS with DHT

Cox et al. [7] explored the benefits (and handicaps) of replacing DNS with DHash, a P2P DHT built on top of Chord [9]. The benefits of replacing DNS with a DHT-based architecture include fault-tolerance, load-balancing, self-management, and smaller network operating expenditures (OPEX).13 The scheme proposed in [7], called DDNS, requires $O(\log_{2n})$ application-layer hops, where $n$ is the number of nodes on the Chord ring, assuming no caching, only for retrieving the server hostname in $U_1$ (refer to §2.1, Step 2). The remaining part of the URI would be resolved using Steps 3-4 as per usual. Cox et al. [7] evaluate DDNS assuming no node failures. They use data collected in 2000 and 2001 by Jung et al. [21] and simulate a rather small DDNS network comprising 1 000 nodes. A total of 260 000 resource record sets (RRSets) are inserted into the two examined systems (DNS vs. DDNS). Recall that today the DNS manages three orders of magnitude more domain names (168 million). Even for this small Chord-based network, with considerably fewer name/IP mappings in the form of RRRSets, the results are disappointing.14

The authors candidly conclude [7]:

It is easy to hide this problem [latency] in the big-$O$ notation, but if peer-to-peer systems are to support low-latency applications, we need to find ways to reduce the number of [remote procedure calls] per lookup.

11. Empirical evaluations showed that commercial CDNs cannot guarantee optimal server selection, and thus cannot minimize latencies under all scenarios (see [4] and the references therein). Johnson et al. [17] conclude that CDNs do add measurable value compared to random selection, by avoiding particularly “bad” server selections, but are far from perfect or optimal.

12. BitTorrent is a simple protocol that runs on top of HTTP and relies on URI resolution as well. After all, trackers are essentially web servers for ad-hoc content. In other words, the start of each and every download in a BitTorrent network goes through the normal URI resolution presented above to reach the tracker, which acts as a redirection and coordination point. In contrast with traditional web servers, the tracker does not typically store the requested object. Instead it returns a list of peers, which have (possibly only pieces of) the requested object. After that, the requesting peer contacts the peer offering the object directly. The tracker is informed about the progress of downloads as well as the availability of new pieces at different peers.

13. In contrast, BIND, the popular DNS implementation, requires careful configuration and high level of know-how. Moreover, with a DHT-based hostname resolution scheme, certain administrative problems are addressed.

14. With respect to hostname resolution latency, conventional DNS outperforms DDNS by a factor of ten. In the case of the simulated network, each hostname resolution request in DDNS requires nearly $\log_{10}1000 \approx 10$ hops to be resolved. For a DDNS with 1 million nodes, 20 hops are required. Today’s DNS needs typically less than three request/replies.
If DDNS where to serve all of today’s domain registrations, and each Chord node managed 100-1000 domain names, the Chord ring size would be in the order of 100,000 nodes. Each request served naively (i.e. without caching) would require more than 16 hops, which based on the results in [7], translates into a latency of over 1 s, on median.

Ramasubramanian and Sirer [22] propose the Cooperative Domain Name System (CoDoNS), which makes extensive use of proactive caching. Effectively, they revisit the computation/communication/storage (C2S) tradeoff and propose a solution that can deliver $O(1)$ resolution times. They use a 12-hour trace from 2000 which includes 281,943 total queries for 47,230 unique domain names to evaluate CoDoNS; see [22] for further details. They show that its hostname resolution latency is in the order of 1 ms, on median, and over 100 ms, on average. However, this comes at a high price: the average CoDoNS node is expected to provide 12.2 KB/s ($\approx 100$ kb/s) of bandwidth, and store approximately 10% of all domain names. How CoDoNS would be able to scale to three orders of magnitude increase in unique domain names is not addressed in [22].

3.2. On Semantic Free References

Walfish et al. [6] propose an even more ambitious architecture based on their version of semantic free references. Instead of replacing only the hostname part of a URI, semantic free references (SFRs) can replace any part of the URI; for details see [6]. SFRs stored in a global DHT can allow for both persistent object references and contention free references. SFRs do not have to be resolved using a DHT. Essentially they are flat, non-hierarchical strings of characters that follow the normal URI syntax, and are the result of a hashing function. It is the application of the hashing function that ensures the contention-free property.¹⁵

Trust and financing as well as scalable resolution are listed as distinct challenges that the proposed scheme would have to address in order to be deployed [6]. DNS is basically financed by the organization owning each domain. The common infrastructure includes the operation of root servers, but financing it is not a major concern in today’s Internet. In contrast, SFRs are delivered by random Chord nodes, and thus both trust and financing are not straightforward. One could consider the use of some kind of reciprocity rules, but several issues are left to be addressed in future work.

Walfish et al. [6] admit that the $O(\log_{2n})$ number of hops needed for each resolution could introduce clearly intolerable delays, as we saw earlier. Thus, the Web-over-SFR proposal makes use of aggressive caching in order to guarantee that, on average, requests are resolved in only 2-3 hops. Simulation results with a 1000-node Chord ring, show that lookup requests are resolved in 2-4 Chord hops [6]. Unlike [22], [6] does not report on the tradeoffs made in terms of storage and communication overhead. Based on [7], [21], this may translate into a resolution latency in the range 50-400 ms. Similarly with [22], Walfish et al. [6] do not actually explain how the proposed scheme would scale beyond the prototype level with respect to latency.¹⁶ That said, SFR does have properties that are important for a network of information. Namely that anyone can publish without owning any web server, object migration is straightforward, and the infrastructure is self-manageable.

3.3. P2P in a Network of Information

P2P technologies have taken networking by storm and I expect that they will play a vital role in realizing a network of information. Lu et al. [8] classify P2P overlay networks into structured and unstructured. The former distribute content at random peers, but at deterministically specified locations. In theory, resolution scales logarithmically with the number of peers. In practice, as we saw above, latency is a big issue. In their pure form, structured overlays are topology agnostic and do not take advantage of localities [10]. In a flat ring of $10^5$ nodes, $O(\log_{2n})$ translates into 30 hops/request at the overlay level, not at the network layer.

On the other hand, widely-deployed unstructured P2P technologies, such as BitTorrent, take an evolutionary approach capitalizing on existing infrastructure (DNS, end-host resources), and use URIs for name resolution. Lu et al. [8] state that they are self-scaling: as more nodes join the network, its value rises. In particular for BitTorrent, if content is available, it is guaranteed to be located in constant time. This is a significant advantage over structured P2P alternatives. If “rough consensus and running code” is still the measure of success then users have cast their votes.

Nevertheless, both structured and unstructured P2P overlays are based on host-centric approaches. For example, in structured approaches, information objects are expected to be located at a specific set of nodes on the ring. In this sense, content moves more freely in an unstructured overlay as it can be located anywhere in the network. Then again, in unstructured approaches, there is no cooperation with the underlying network. No hints are given, and the flash-crowed case of a router queuing 5000 packets carrying exactly the

¹⁵ In the proposed scheme, the object (called o-record) that an SFR refers to does not expire—ever. Moreover, SFR tags cannot be reassigned, thus allowing for permanent references, but o-records can be “emptied.” Presumably, a SFR to an empty o-record is the equivalent of a broken link in today’s WWW. However, in contrast with a file that does not exist, o-records will still claim space in the global SFR resolution machinery.

¹⁶ A prototype implementation based on DIHash/Chord is evaluated on PlanetLab and compared with that of [21]. Walfish et al. report that 98% of 15,000 lookup requests are resolved in two Chord hops. However, a close examination of the results shows that, on median, SFR is outperformed by DNS by almost a factor of five. SFR outperforms only 30% of the time, in the tail of the 2000 DNS data set distribution. In real-life experiments in September 2003 the researchers/users reported that “the perceived latency was generally indistinguishable from DNS,” although the prototype evaluation results were reported as suggestive only, not conclusive.
same payload [2] is not unlikely as all connections are end-to-end. All proposals discussed assume an IP-based Internet. New post-IP paradigms have yet to be considered, but should be part of forthcoming resolution scheme evaluations. For NetInf to materialize, it is necessary to come up with new solutions and better metrics to determine performance.

4. Discussion

DNS is effectively a read-only distributed data base for most of its users. Entries are made by authoritative users, and updates are done via zone transfers, with extensive caching involved. Anycasting has also proven an effective method to alleviate performance concerns. In traditional URI-based content distribution, object resolution does not actually depend on the number of nodes in the network: lookup times are deterministic, or \( O(1) \), and caching can be employed extensively. This is the level of performance that DHT-based resolution schemes have to meet. Indeed, DNS and CDNs managed to demonstrate an order of magnitude increase in reach and performance over the last decade. It is not unlikely that current evolutionary approaches will be able to repeat this feat. Thus, new proposals should deliver more than an order of magnitude increase in performance.

With respect to the suitability of structured identifiers, such as URIs, as opposed to flat identifiers such as hash values, for IO resolution in a future network of information, this review paper remains inconclusive. More empirical evidence, measurements, and simulation and modeling studies are necessary, in order to obtain an up-to-date understanding of the issues at hand. These are all items on a research agenda, which considers that four areas will play a key role in determining the value of novel approaches in name resolution for a future network of information: latency, mobility and multiaccess, C2S tradeoffs, and energy efficiency.

Latency. Object resolution queries themselves do not require much bandwidth, but delay is decisive. As network capacities increase, each millisecond counts. As storage becomes even more abundant and cheaper, time is the one commodity that remains at a premium. In other words, resolution latencies should be \( O(1) \), i.e. should not depend on the number of nodes in the network. Techniques such as anycasting, should allow the underlying network to locate the closest replica of a given object. Bringing content closer to the users has proven a successful strategy in the host-centric paradigm, as CDNs and the expansive deployment of root and TLD name servers in recent years in more countries and sites testify. In the information-centric paradigm, all object instances are the same, and the network should be able to locate the one that fits the purpose the best.

Mobility and Multiaccess. We have recently moved from a single (network) access/single (object) source (SASS) request/reply client/server paradigm, which still dominates Internet traffic with TCP end-to-end connections, to a single-access/multiple-source (SAMS) dissemination network paradigm, as in BitTorrent. Soon, devices will also adopt the multiple-access/single source (MASS) paradigm, as in multihoming, and finally we will head to multiple-access/multiple-source (MAMS) information-centric networks. MAMS requires IO resolution that is designed to take advantage of the capabilities of modern mobile multiaccess devices. This is an important line of future work [23].

The C2S tradeoff. Jacobson [2] points out that bits on the wire, fiber, ether or some storage device have exactly the same value. They are just that: bits. An information object resolution scheme should be able to capitalize on all available resources, and provide knobs and toggles that allow for the network and its users to make the computation/communication/storage (C2S) tradeoffs they want. Sometimes, local memory would be a plenty; other times network capacity will be overprovisioned. By centering on information one can make better decisions when resolving objects than when relying on an end-to-end approach.

Energy Efficiency. NetInf has the potential for being significantly more energy efficient than today’s Internet and IO resolution mechanisms will be essential in this. Take for instance the case of streaming Internet radio. Today, streaming a radio show from the station server(s) means that the (mobile) device network interface must remain on and active for the duration of the show. Contrast this with listening to a podcast of the show. The user first downloads and then listens to the episode and, during playout, the network interface can be inactive, which increases battery life significantly\(^{17}\). Taking this example one step further, and assuming that digital rights management is properly handled, radio streaming can be transformed taking energy consumption into consideration. With the ample memory capacities of modern mobile devices, songs, commercials, jingles, and so on, can all be stored locally and played according to the radio station play lists, while accounting for DRM and royalty payments\(^{18}\). Mixing live talk or commentary with locally stored content is technically feasible and should not pose a problem in the future Internet. IO resolution and storage will play a key role in an energy-efficient NetInf.

5. Summary and Future Directions

This paper reviewed a topic of great interest within the design space of a network of information, namely, distributed information object resolution. After summarizing the key elements of NetInf, the paper elaborated on the notion of information objects. We recapped the implications of DNS involvement in URI resolution, summed up two different

\(^{17}\) For example, the iPhone 3G can play audio tracks for up to 24 hours, according to the Apple web site, which is three times as much as its operational time when continuously connected to a Wi-Fi access point.

\(^{18}\) Time shifting is legal in most developed countries; with proper care, energy-efficient radio streaming can be practical in short order.
types of content distribution networks with respect to name resolution, and critically evaluated proposals championing the replacement of DNS with alternatives based on DHTs. After examining the advantages and disadvantages of each approach, we centered on the importance of latency in information object resolution and discussed the challenges that a NetInf IO resolution system must meet.

Items on our current and future research agenda include: (i) Collection of up-to-date empirical evidence from current networks: How valid are 10-year old data when evaluating IO resolution alternatives? (ii) evaluation of alternative proposals using large-scale simulations; (iii) study of the operational concerns in different resolution schemes; and (iv) development of mechanisms to address the tradeoffs arising.

Acknowledgments

This work has been supported by the IST 7th Framework Programme Integrated Project 4WARD, which is partially funded by the Commission of the European Union, and the Finnish Strategic Centres for Science, Technology and Innovation (ICT SHOK) “Future Internet” Programme. I thank Börje Ohlman and Jari Korva for their constructive comments and my colleagues in both projects for fruitful discussions. The views expressed in this paper are solely those of the author and do not necessarily represent the views of VTT or the respective projects and sponsors.

References


