Entity Layer Internet Architecture and Services

By Miroslav Bonchev Bonchev

https://mbbsoftware.com

Abstract

The Internet is currently almost fully centralized at the application level. The centralized model is not appropriate for distributed by nature services such as search engines, social websites, and many others, which leads to substantial socio-economic and environmental losses. The paper suggests the replacement of the current Application layer of the Internet with an Entity layer that ensures proper information ownership and execution of work. In the proposed model, websites continue to operate as they are. However, the current web-application (website) becomes only one of many sibling services working in parallel, which owners can add as they require. The connectivity and interoperation between these services enable direct connection between users and providers, and eliminates the middlemen, such as centralized search engines and social websites. The model provides a non-intrusive, non-interrupting solution able to gradually transform the Internet at the top utilization level from an improper and inefficient centralized to a proper and efficient distributed architecture. The model can also stimulate business, restore online freedoms, make browsing more efficient, and even have a positive environmental impact by eliminating inefficient centralized services all through market tools.

The author aims to create free and open-source implementations of as many as possible of the services described herein. Your support in this effort is most appreciated. Please feel welcome to support the project on its website at https://www.mbbsoftware.com/products/elias/default.aspx.

Update: December 1, 2020, rev. 11

First release: September 2017

Contents

1. Introduction – Problems with the Centralized Internet

A centralized model of data and work at the Application layer of the Internet is appropriate for representing entities such as companies and persons and their services. However, distributed by nature services such as search engines, social networks, and many others must be implemented in a distributed manner to be efficient. Nevertheless, the Internet is almost fully centralized at the application level as of the year 2020. Some examples include:

1. Centralized search engines.Some of the issues with centralized search engines include:

   1.1. **Bias** – centralized search engines are always biased as follows:
      a. there is only one first result;
      b. there is only one first page;
      c. there is only one thing seen first;
      d. there is only one first impression.

   Thus a person is always first impressed with whatever the search engine decides to. That instills bias in the person, especially if this is the first time they are introduced to a matter. It also directs where the user will spend time first, consider to purchase first, etc.

   Further, a centralized search engine lists all other results in order by its criteria, amplifying the effect. Thus centralized search engines are always biased by the belief system of their creators who decided what should be listed first and what last.

   1.2. **Censorship** – centralized search engines can decide to not display particular results, for any reason as they please.

   1.3. **Incorrectness** – centralized search engines scan for webpage changes periodically, hence not always have the latest data. They can also run out of storage, i.e. not have all data. They interpret the content of any page, in a way that does not necessarily match the way the creator of the page interprets it. Thus, they can miss, omit, or return incorrect results.

   1.4. **Concealed blackmail** – users and websites have no control over the order of the results. As a consequence of that and also because of 1.1., 1.2. and 1.3., the only way for a website to guarantee being displayed as a top result to any audience is to buy adverts from the search engine. Again, note that users cannot influence search results whatsoever either.

   1.5. **Content exploitation** – a centralized search engine does not give any value, whatsoever to the information it displays. It merely displays content picked up from third-party websites in an order it chooses. Thus, centralized search engines use other people's information for profit, without paying them for exploiting their content. If objected, consider how often people would use Google if it displays 100% paid adverts and zero native results!

1.6. **Illegitimate use of content** – most centralized search engines do not ask the owners of websites for permission to use their content for any purpose, including for search results and profit. Due to the current centralized model of the Internet at the application level, website owners have no choice but to silently agree with the exploitation of their content in the hopes that the search engine might send a user to visit their website.

1.7. **Indiscriminate use of content** – while it is possible to tell centralized search engines to not index certain web pages through the no-index directive, it is not necessary for a centralized search engine to comply with it, and it still may index the page, or use its content otherwise. Also, parts of webpages cannot be prohibited from indexing at all.

1.8. **Middleman self-insertion** – centralized search engines embed themselves as middlemen between websites and users. They exploit the websites as per 1.4, 1.5, and 1.6 while at the same time manipulate the users per 1.1, 1.2, and 1.3, even if the latter is not on purpose. Note that if the user knows the URL of a resource then they do not require a search engine. However, to embed themselves even more as a middleman, first Chrome by Google, modified the address bar of the browser to allows users to enter in it a search term in addition to a URL. Thus centralized search engines exploit the natural preference of humans to enter a word or a term, as opposed to a URL even if they know the latter. Thus people are encouraged to always execute a search as opposed to enter a URL, which makes people and websites increasingly more dependent and subjected to the centralized search engines.

1.9. **Market manipulation and skewing** – centralized search engines earn money by selling adverts. Having ads based business enables centralized search engines to produce and offer apparently “free” software, but paid via ad watching. Thus the actual price of an advert-paid product remains hidden from the user. However, manufacturers of properly priced software become significantly market-disadvantaged against the “free” search engine software. As a consequence, such manufacturers have to make their software also “free” and are de-facto forced to become customers of the advert distribution platforms of the centralized search engines. Further, are centralized search engines able to prove that they do not punish no-ads websites, and do not promote websites which display adverts purchased from them?!

1.10. **Suppression of small business** – to maintain the appearance of “fairness” as well as to maximize profits centralized search engines use advert bidding. The average Cost-Per-Click (Google Search Network, as of 2020) is reported to be from $1 to $2 per click, with significantly higher prices in certain industries. Paying for an advert click, of course, does not guarantee that the user will buy, or indeed even that the user is at all considering a purchase. Thus, for small businesses without large resources, it can be impossible to be found in centralized search engine results, be it in the “native” or advert results.

1.11. **Danger to society** – if a centralized search engine gains a monopoly, then it will be able to impact the entire civilization in every aspect of its life, including health, fashion, personal beliefs, etc. In addition to 1.9 and 1.10, some search engines attempt to manipulate also the business, economy, and society, blocking political websites, etc. Further, most centralized search engines make pornography and other society destroying activities completely available to anyone including children and adolescents – imagery, video, and links are made available to anyone. The rationale behind this may be that morally compromised people are more likely to prefer to "pay" by watching adverts rather than with money at hand.

1.12. **Inefficiency** – centralized search engines index the internet, which means that each of them de-facto makes a full or partial copy of it. Therefore massive amounts of computers were manufactured and continuously operate to maintain these sort of duplicate copies of the internet, using enormous amounts of energy, materials, and resources. According to statista.com, only Google used 10.6 terawatt-hours in 2018, which in terms of coal would be 2,175,601.4 tons of CO_2 emissions. The staggering energy inefficiency of the centralized search engines, due to the continuous indexing and making of duplicate copies of the internet, is even more shocking if one considers, that the energy and resources are always consumed even if there are no searches at all, or a result of a search is useless due to changes on the page which have not yet been indexed.

Centralized search engines unduly accumulate enormous power, yet they are completely unaccountable and present enormous liability and socially-economic and environmental losses to society. ELIAS offers a simple and effective solution.

2. Sales websites, such as Amazon, eBay, and others have become the de-facto places for online shopping and are another type of centralized search engines. Because these websites have taken such a central position, businesses have to open accounts with them and pay fees so that they can have online sales. Some of these websites also sell adverts to the sellers. Thus businesses must maintain their websites, plus multiple online-stores on the centralized sales websites, pay sales fees to them, and pay adverts to them to compete with the host and the other sellers there. ELIAS offers a simple and effective solution.

3. Broker websites, such as Booking.com, Expedia.com, and others, are another type of centralized search engine, which plug themselves between the service provider and the consumer. An interesting observation is that if one opens Google and search for say "Hotel in London" most results will be of broker websites and not websites of hotels in London. Thus with the help of the generic centralized search engines, users are forced to pay up to 35% fees to the broker websites. Thus, once again, businesses must maintain their website, as well as online-stores on every broker website, which brings burden and financial losses for them and consumers. ELIAS offers a simple and effective solution.

4. “Social” websites, such as Facebook, Twitter, and others. Even from the first sight, it is obvious that it is wrong for people to provide their personal information to third parties. Yet, “Social” websites are created to precisely collect personal data. The issues with such websites include:

   4.1. Collection of user data that allows for all kinds of misuse and abuses.

   4.2. Undue profit from the personal information of users, without sharing any profits with them.

   4.3. If a centralized “social” website gains a monopoly, then it will be able to unduly impact the entire civilization in every aspect of its life including personal beliefs.

   Centralized “social” websites unduly accumulate enormous power by collecting people’s personal information. ELIAS offers a simple and effective alternative solution.

5. Similar challenges with other online activities including online dating, job search, job offers, and many others are resolved by the ELIAS architecture.

6. Email has become a standard for online communication. Although in its current implementation it is in theory decentralized, it is constructed poorly and allows for all kinds of abuses, spam, misrepresentation, phishing, snooping, stealing of data, and so on. Email AS IS allows easy centralization, utilized by companies’ s.a. Yahoo and Google. Email “AS IS” is so dreadful that almost all business communication from unknown senders is considered spam or some type of a threat and is usually immediately deleted. The ELIAS architecture proposes another mail architecture to resolve these issues.

The problems arising from the centralization of distributed by nature services on the Internet are enormous. A simple analogy would be to attempt to set a pyramid upside-down. Although, in theory, it may be possible, in practice, this will be problematic. However, placed on its base, the system is stable and requires no additional work, balancing, and support.

The ELIAS Internet architecture presented in this proposal breaks the centralized model and provides a decentralized alternative. It gives people and businesses their online freedom, and all benefit from removing the middlemen with their centralized services.

2. Entity Layer Internet Architecture and Services

The standard OSI network model includes seven layers as follows:

7	Application
6	Presentation
5	Session
4	Transport
3	Network
2	Datalink
1	Physical

The OSI model defines the application layer as the user interface responsible for displaying received information to the user.[1]

We propose to replace the **Application** layer with an **Entity** layer. The **Entity** layer represents an organization/person and implements the services provided by their site. The **Entity** layer consists of a **Domain** and **Services** sublayer on top of it. The Domain sublayer represents web address and identifier of the organization. The Services sublayer includes the services provided by the site of the organization.

<table>
<tr><td rowspan="2">7</td><td rowspan="2">Entity</td><td>Services</td><td>query</td><td>search</td><td>societal</td><td>work</td><td>broadcast</td><td>sell</td><td>mail</td><td>website</td><td>…</td></tr>
<tr><td>Domain</td><td colspan="9">– a sub-layer representing the entity (organization or person) URL and all services above.</td></tr>
<tr><td>6</td><td colspan="11">Presentation</td></tr>
<tr><td>5</td><td colspan="11">Session</td></tr>
<tr><td>4</td><td colspan="11">Transport</td></tr>
<tr><td>3</td><td colspan="11">Network</td></tr>
<tr><td>2</td><td colspan="11">Datalink</td></tr>
<tr><td>1</td><td colspan="11">Physical</td></tr>
</table>

The services of the Entity layer have the following properties:

1. Services work exclusively with the data of the domain (site) on which they are installed.
2. Services can be called from other services working on the same, or from another domain.
3. Services can be called from users and apps e.g. fetching a webpage by a browser.
4. Services can be added and removed from a site by simply installing and uninstalling them.
5. There is a mandatory **query** service installed on each site, which enables calls to the services on the site by interface ID as opposed to the by service name.

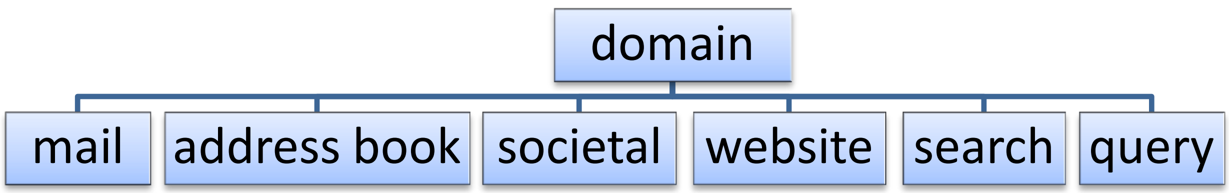
We will refer to the proposed model of the 7th OSI layer as the Entity layer Internet Architecture and Services (ELIAS) model or architecture.

A domain (site) can have any number of services installed on it, with only the “Query” service being mandatory. Example services include:

1. Query - required service that enables calls to services by interface ID.
2. Website - standard website application.
3. Search - the site search engine.

4. Sell - for sale of items (eBay and Amazon like)
5. Work - for work connections (Linked in like), job offers, and job seek (Job seek like).
6. Broadcast - for posting small public messages (Twitter like).
7. Societal - online connections and posting of data (Facebook like).
8. Mail - application enabling a new type of secure email service.
9. [many more] - such as online dating, and so on.

The diagram below shows an example domain which has several services installed on it. Note that the ELIAS architecture requires services to work with both their, and their siblings' data. For example, the search service will return results only for its sibling website, societal, etc. services, where it can be called from the user directly, the website search box, the search services of other domains, and external apps.



For the remaining of the document, when referring to a services we will understand ELIAS services.

2.1. Work

1. Services are installed and operate on domains (logically) and webservers (physically).
2. Services installed on a domain provide information for the services running on that domain only.
3. Services can call, and be called from, other services running on the same or other domains, as well as from desktop and mobile applications.
4. Different types of services can work in cooperation e.g. the mail and social services running on a domain can work with the address-book service on the same or another domain.
5. Services provide interfaces that are queried by other services for information or execution of work. Access rules apply depending on the nature of the service and its configuration.
6. Services can be installed and removed from a site at any time.
7. A mandatory Query service is installed on every domain to enable calls to services by their GUID.

2.2. Addressing

The following address schemas can be used:

1. **[service-name]-[service-GUID].elias.domain.tld**
2. **elias.domain.tld/[service-name]-[service-GUID]**
3. **domain.tld/elias/[service-name]-[service-GUID]**

These and other addressing schemas can be used by sites depending on their circumstances. Callers to services can determine which of the addressing mode is used by the particular site/domain and use that addressing mode.

- The **[service-name]** segment represents the service name, for example, mail, search, societal, broadcast, image, etc. The following service names are reserved:
  - **[empty]** - is reserved for user navigation of services.
  - **query** - is reserved for system use and address query.
  - **website** - is reserved as www.domain.tld and domain.tld alias.

  The names of all other services are important only if called directly. Service names should indicate their intention. When installed, services with non reserved name are added with their GUID in the address, to rule out name collisions and enable the adding of new and competing services without registration with a central service name authority.

4. **elias** – architecture identifier. This segment plays the role of a namespace allowing seamless addition of other 7th OSI model level architectures in the future.

The suggested addressing model enables a seamless transition from the current centralized to a decentralized model. It also enables easy integration with websites of any size and type of deployment, as well as easy extendibility.

2.3. Extendibility

Anyone can create a new implementation of an existing service for as long as it complies with public interfaces for the service. Anyone can invent entirely new services, for as long as they comply with the public interfaces of the query service.

2.4. User Access

Users can access services as follows:

1. From the service itself, e.g. a mail service where the user logs in their mail account.
2. From a sibling service, e.g. website search box uses its search sibling service to find data in it.
3. From another website, that requests this website's service, e.g. check for item for sell, work, etc.
4. From specialized software providing access to services, e.g. desktop app, browser extension, etc.

2.5. Transport

Services are built on top of the HTTP/S data transmission protocol.

2.6. Privacy and access

The amount of information that services share depends on them, for example, an address book service may hide or make public its contacts to various levels depending on its settings, and the requester.

3. Services

The ELIAS architecture allows for any kind of service including:

1. Query – reserved system service.
2. Website – the domain website service.
3. Home service – aggregator user interface services, e.g. social, work, etc.
4. Search engine – search the local site sibling services or makes Internet searches.
5. Social – societal (Facebook), broadcast (Twitter), video (YouTube), image (Instagram) like, etc.
6. Work – job search, professional connections, etc.
7. Sale – Amazon, eBay, eShop like
8. Messenger
9. Address book/connections
10. Advertiser – advertise anything
11. Spamless mail
12. Dating
13. Trust services
    13.1. Explicit – such as authenticated testimonials, documents, and files
    13.2. Implicit – using the elias.domains connections (e.g. MAC – Mutually Approved Contacts)
14. Connector services, and many more.

Services can work independently one from another, or in cooperation. For example, the spamless mail service may also use the explicit and implicit trust services to provide some suggested trust level to incoming mail from unknown (as well as from known) sources.

Search Engine (Fair)

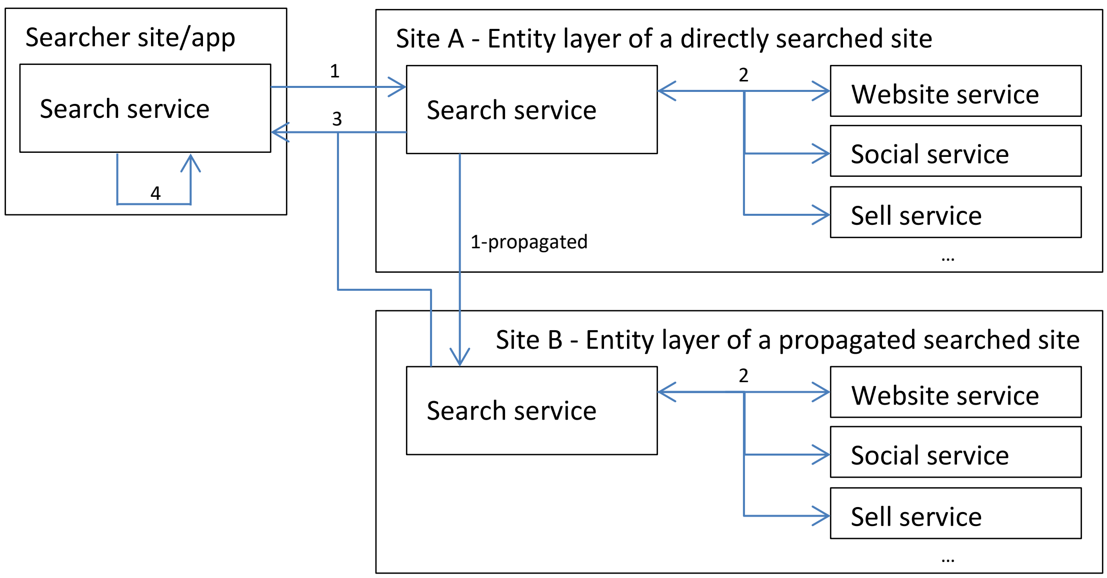
The (Fair) Search Engine is a methodology for internet search, in which search results are generated using the collective operation of many websites, without caching the internet. In contrast, centralized search engines collect data in advance, and provide results using the cached data. The Fair Search Engine consists of **a searcher**, which can be the search or another service of a site, or any app that makes search requests, and **the search services hosted on the queried domains**. The search service of a domain can search any permitting service on that domain including website, social, sales, etc.

Method of Work

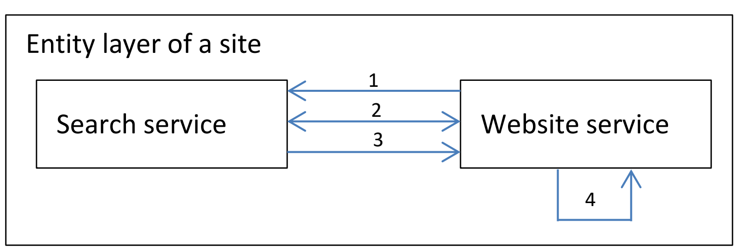
1. The searcher generates a search request based on the user request or other data and queries the search services of bookmarked for search domains.
2. The search services of the called domains execute the search on the required sibling service, e.g. website, social, broadcast, sell, etc. and return the results to the searcher.
3. Depending on the request, the search services of the queried sites may propagate the search request to their bookmarked for search domains, thus creating an exponential search.
4. The searcher displays or otherwise uses the results returned from all queried domains.
5. Cross websites' results are not accepted and are ignored – the searcher ensures compliance.

Thus, each domain provides its own search engine to service queries regarding the content of the site, i.e. its services. The collective work of the searcher and the search services of the queried websites constitute the **Fair Search Engine**. The following diagrams depict the Fair Search Engine mechanism:

- Step 1 indicates the search request. Step 1-propagated indicates any propagated search request.
- Step 2 indicates the search execution on each site by its search service.
- Step 3 indicates the return of search results.
- Step 4 indicates the display of search results, bookmarking of newly discovered sites, etc.



The following diagram shows the simpler case of search on the same domain. For example, a visitor browses a website and uses the search box to find information on it, or a user searches its social service to find their posts from the past, etc. In this case, the searcher is the website/social service of the site, which calls its sibling search engine to find information on it and display the results.



1. Search request
2. Search processing
   (Using any suitable backend technique)
3. Return of results
4. Display the results

The search service of the site searches its sibling website (caller) and returns its results to it.

The Fair Search Engine includes auxiliary features, including bookmarked domains for search initiation, search propagation, search discovery, search-service algorithm, search request and responses, and other elements, which can vary depending on implementation and version. However, they must ensure interoperability with the search services of the other sites (domains) on the network.

Searcher Request

The search requests provide the search query to the search services of the bookmarked for search domains, e.g. search.elias.domain.tld or query.elias.domain.tld. The search request implementation uses POST requests, although it is possible to use GET requests also. POST requests are preferred to enable:

1. Asynchronous return of results from directly and indirectly queried domains.
2. Homogeneous type of messages carrying the search results.
   While results for the initial search requests can be returned with the GET response data, it makes more sense to return results from propagated (indirect) requests to the searcher using POST. Thus for simplicity and homogeneity, we prefer using POST requests.

A search query consists of structured fields providing the details of the search request. For example:

1. Search request Version – GUID
2. Search request ID – GUID
3. Searcher URL e.g. search.elias.domain.tld
4. The search query, where
   a. Quotes "…" define a request for an exact match, e.g. "some text"
   b. AND operator defines a request for having both phrases on the webpage
   c. OR operator defines an inclusive OR for "phrase A" OR "phrase B"
   d. XOR and EOR, define a request for having one or the other, but not both phrases
   e. NOT "text" defines a request for not having the phrase
5. Meta fields, such as
   a. Services to search in – website, societal, broadcast, video, work, sell, news, etc. … [all].
   b. Content category – food, computers, tourism, etc.
   c. Media type – text, image, video, pdf, etc.
   d. Author – must/must not
   e. Date – must/must not be before, after, range
   f. Country – must/must not be
   g. Geographic area – must/must not be
   h. Propagation request, and others.

Search Results

The Search Results consists of structured groups of data e.g. REST, XML, etc. The search results are provided to the searcher using POST requests for system homogeneity as explained in the Search Request section. For example:

1. Response Version - GUID
2. Query GUID (same as search request ID)
3. Response GUID
4. Result 1
   a. Relative address (to comply with cross-site restriction rule)
   b. Title

    c. Sample Text
    d. Meta Data
5. Result 2 …

The searcher displays the search results to the user or uses them otherwise, upon receiving them.

Search Bookmarks

Each ELIAS search service keeps a list of domains bookmarked by the user for search initiation. Thus searches always start from the websites in the area of interest and preferences of the user. The searcher provides a mechanism to allow the user to easily bookmark domains for searches. Also, the searcher can provide various enhancements for contextualized bookmarking, such as enable grouping of bookmarks for “Social”, “Sales”, “News”, etc. searches; thematic searches, e.g. search “flowers”, levels, caches of old searches, and any other apparatus for search streamlining and request optimization.

Search Propagation

Depending on the request, the queried ELIAS search services may propagate the request to their bookmarked for search domains. In this case, each search service passes the search query down to its own bookmarked websites, thus exponentially propagating the request. Cyclical and duplicate searches will not occur since each search request has a GUID. Various mechanisms can be employed to set the number of propagation levels, such as:

- Providing a fixed depth of search query propagation;
- Callbacks to the searcher to check if the search is still valid, and so on.

The search propagation mechanism will advance the search until the user cancels the search or the set of search-bookmarked-connected domains is exhausted.

Search Initiation Domains Discovery

Newly installed searchers will have only a few, if any, bookmarked search initiation domains to produce satisfactory search results. Searchers can use various strategies to solve this problem. Searchers can discover search initiation domains by querying:

1. Community hosted domain-dictionaries that store the type of the content of sites, which the domain owners provide, e.g., food, travel, software, geographic location of the business, etc.
2. The top level TLD registrars for any domains (will be slow but comprehensive). TLD registrars can extend their records and provide content/metadata domain-dictionaries, although unlikely.
3. The social, broadcast, work, email, and other connections of the user for their search initiation domain bookmarks, if they would provide them.
4. Searchers can use visited but not bookmarked by the user domains for search initiation.
5. Searchers can have a certain number of built-in domains to initiate searches.
6. Searchers can utilize centralized search engines to discover new search initiation domains.

Searchers can use many of the above and other strategies to discover new search initiation domains.

Search Processing Algorithm

The Fair Search Engine is agnostic to the search processing algorithms executed on sites. It is up to the manufacturers of search services, and site owners to decide how the local search service algorithms will work, including the use or not of indexes, access methods, which pages and sections on them to index, how to interpret their content, content categorization e.g. flowers, food, etc. (in addition to the service categorization that occurs naturally for the ELIAS architecture), media type, and so on.

Searcher Callback

Before providing results, queried search services may call the searcher for additional information, such as the validity and state of the search query, or anything else.

Search Tuning

Search services can be optimized to provide an easy selection of thematic search bookmarks, category search bookmarks, levels of bookmarks, search meta properties, requests for use of specific search processing algorithms and parameters, propagation algorithms, various views for results, and so on.

Diversity

Any developer can create and offer search services. Owners of sites can install search services, as well as any other service, manufactured by any supplier.

Sufficiency

The Search Domains Discovery mechanism guarantees that good results will be found if they exist, even though that may take time when searches start with only a few or none at all suitable search initiation domains.

Fair Search Engine Conclusion

The Fair Search Engine overcomes the faults of the centralized search engines. The method produces:

- Optimal results for the user as searches are started from their bookmarked for search sites.
- Over time the search results will continuously improve as the user finds and bookmarks more search domains matching their preferences to start searches from.
- The search domain discovery guarantees that all valid results will be found. However, this may take time, especially if there are no suitable bookmarked domains to start the search.
- Fair searches are likely to be generally slower than centralized ones, however, while the user reviews the first result(s), more will arrive. Hense this characteristic will have little significance.

Centralized search engines will continue to exists. However, their role will significantly decrease.

Social Services

Social services provide the functionality of social websites, such as Facebook, Twitter, YouTube, Instagram, Linked-in, etc. The service allows for the owner of the domain to:

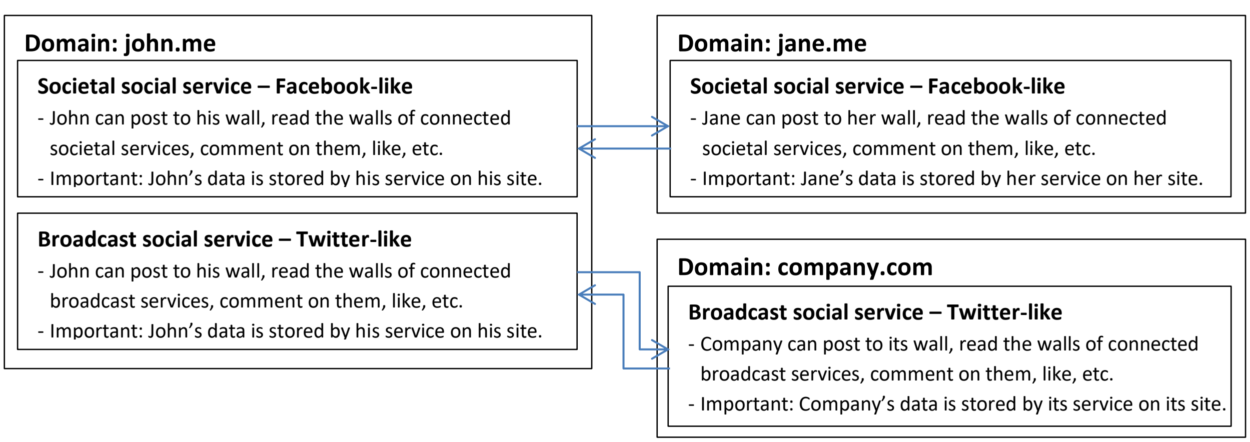
1. Make posts to their wall, and;
2. Connect to the respective social service of other domains;
3. Read their own and connected walls, post comments, likes, etc. to connected “social” services.

Social services can be tailored to be more or less like Facebook, Twitter, YouTube, Instagram, Linked-in, or any other social media website. The social services enable the same functionality. However, the difference is that the social service of a domain keeps its content and only enables displaying it by connected with it other domains’ social services.

Note that the ELIAS architecture allows for partial connections, as well as different levels of connections depending on the nature of the social service. For example, a video service may normally allow anyone to view its content, while a societal social service would normally allow only approved contacts to view its content. Some services may allow a partial view of their content and so on.

Example Implementation

In this example, we will consider a societal service implementing the functionality of a societal website, similar to Facebook, which is installed on a user domain john.me. The owner of the domain John, logs into their societal service and makes posts, upload images, etc. as he would normally do in his ordinary societal-website page. To connect with his friend’s Jane societal service, John clicks the connect button on his service and provides her domain jane.me. Alternatively, John can use his search engine to find her societal account. After the request to connect is made and Jane accepts it, their societal services become connected and able to see each other’s walls, comment, like, etc. Comments that Jane posts on John’s wall are stored by his societal service. Comments that John posts on Jane’s wall are stored by hers. John also has a broadcast social service installed on his domain, which is connected to the broadcast service of a firm company.com. The following diagram depicts this social service architecture:



Diversity

There can be a great number of diverse social services, such as:

- Societal - similar to Facebook
- Broadcast - similar to Twitter
- Work - similar to LinkedIn
- Job - for job offers, job search, freelancing, etc.
- Video - similar to YouTube
- Image - similar to Instagram
- Advertising - similar to Gumtree, etc.
- Date - Dating service
- [Many more]

Within these general types, services can differ to any extent, for example, video service A may allow anyone to view its content, while video service B may require a paid subscription, and video service C may allow both.

Hosting

Hosting companies can offer to host services to the general public as they do for websites, WordPress, blogs, apps, etc. Also, people can host their services on their own devices, such as old computers, Raspberry Pis, etc., including on their mobile phones.

Advertising

Social services can be enabled to display adverts that they fetch from connected adverts services. Any adverts displayed by social services will be seen by the owners and their friends, with revenue counting towards the wall owner. Part of the revenue can be used to pay for the site hosting.

Migration

Social services keep their data in suitably to allow their owners to change hosts with a “single-click”. Migration from the current centralized social websites, such as Facebook and Twitter, to the new ELIAS social services, can be made through apps running on the former platforms, which collect all user data and store it in a way that can be imported by the new ELIAS services.

Selling Services

Selling services enable people and companies to offer their goods and services in a way that allows buyers to find them in a simple and unified manner.

1. Sellers upload their stock onto their domain, as follows:

   a. Individuals with occasional items for sale would enter the data about the things to sell in their domain Sales service.
   b. Retailers with large stock catalogs upload their items to sell on their websites as usual, and/or to their Sales service similarly to individuals.

2. Buyers use either their domain Sales (or Search service) to find items for sale. The Sell service uses the Search service under the hood to find items for sale on the Internet.

The search service works in the standard for the Fair Search Engine way. The searcher makes the search request to the search services of the user's sales-bookmarked (or connected) domains and displays the results that the local search services found in their Sales and Website sibling services. Note that the prices of the items can be fixed, bidding, donation, or anything else – the Fair Search is agnostic to the actual information. The searcher can also query centralized sell websites such as Amazon and eBay.

The ELIAS architecture enables the owners of the information to keep it on their sites and the buyers to search in the e-stores of all their favorite online retailers simultaneously, even if they are thousands, in a simple and unified way. Note that ordering by criteria, e.g. by price, will be generally slower than in centralized sales websites.

Home service

The Home service is, in essence, a website, which provides the user with easy access to all their user interface enabled services, such as societal (Facebook like), broadcast (Tweeter like), Work (Linked like), Sell (Amazon/eBay like), Video (YouTube like), Image (Instagram like), etc. The Home service only allows users to access all their user interface enabled services from a central location.

Spamless Email

The current email system allows abuses such as spamming, phishing, and so on. The ELIAS architecture allows us to define new email systems in which such abuses are minimized. The new mailing services can work stand-alone or can work in cooperation with other services.

The spamless email service is designed to send and receive messages similar to the ordinary email, however, it minimizes spam, phishing, and malware to almost zero. Suppose that domain-A, domain-B, domain-C, domain-D, and domain-E have elias-mail service installed on them. Also, suppose that user-A is a mail user at domain-A, user-B at domain-B, user-C at domain-C, and user-D at domain-D.

The spamless email service will use the '#' symbol to identify the user and the service type to avoid confusion with the old email, e.g. user#domain.tld. Since ELIAS services use HTTP/S, all elias-mail data is transported via the message header parameters and body.

The algorithms below only outlines some general ideas of how a spamless and safe email can be achieved. Further research, experimentation, and testing with groups of users is required.

Mutually Approved Contacts

Mutually Approved Contacts (MAC) are peers who acknowledge each other's elias-mail identity. Suppose that user-C at domain-C and user-D at domain-D wish to become MAC with each other.

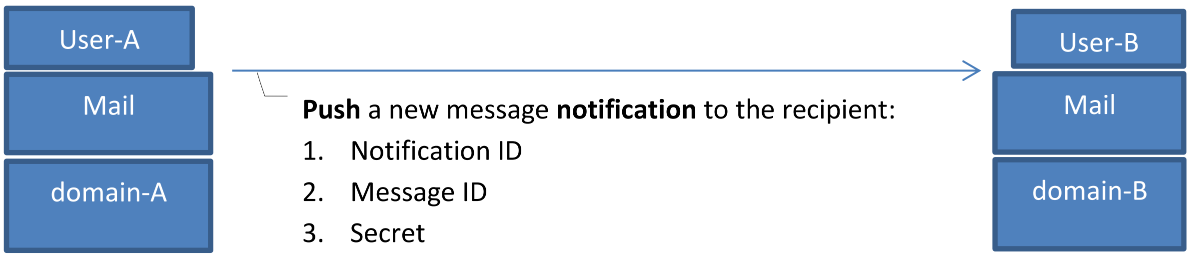
1. User-C logs into their elias-mail service or compatible elias-mail access software.
2. User-C sends a MAC request to the user-D. The request can be made as (a) an administrative request or (b) as a response to a message from user-D. The MAC request contains:
    - 2.1. The requester mail address (user-C#domain-C.tld);
    - 2.2. The public key of the requester mail server (mail.elias.domain-C.tld).
    - 2.3. Message-chain ID.
3. Mail.elias.domin-D.tld receives the request.
4. Mail.elias.domin-D.tld accepts or declines the request based on some criteria.
    - 4.1. The request is accepted for presentation to the user – GO TO NEXT.
    - 4.2. The request is auto-declined, e.g. the sender is a known spammer. Reply admin-decline – END.
5. User-D at domain-D logs into their elias-mail service or elias-mail access software.
6. User-D accepts or declines the MAC request.
    - 6.1. User-D accepts the request. The mail service there does the following:
        - 6.1.1. Generate a strong identification password for user-C.
        - 6.1.2. Store the user-C identification password securely. The mail service of user-D#domain-D.tld will expect the password in all messages from user-C#domain-C.tld. If not provided, the mail.elias.domain-D.tld service will treat the message as from an unknown sender.
        - 6.1.3. Encrypt the user-C password using its public key (provided with the request).
        - 6.1.4. Respond to user-C#domain-C.tld with a message containing:
            - 6.1.4.1. The encrypted user-D owned, user-C identification password.
            - 6.1.4.2. The user-D public key.
            - 6.1.4.3. Message-chain ID.
    - 6.2. User-D declines the request, the mail service responds with a decline message – END.
7. The requester (user-C) receives the response from the contact (user-D).
    - 7.1. If the contact (user-D) accepted the request:
        - 7.1.1. The requester (user-C) stores the contact (user-D) identification password, which it will use whenever it sends a message to user-D#domain-D.tld.
        - 7.1.2. Generate a strong user-D identification password.
        - 7.1.3. Store the user-D identification password securely. The mail service of user-C#domain-C.tld will expect the password in all messages from user-D#domain-D.tld. If not provided, the mail.elias.domain-C.tld service will treat the message as from an unknown sender.
        - 7.1.4. Respond to user-D#domain-D.tld with a message containing:
            - 7.1.4.1. The encrypted user-C owned, user-D identification password.
            - 7.1.4.2. The user-C MAC identification password of user-D#domain-D.tld.
            - 7.1.4.3. Message-chain ID.
    - 7.2. If the contact (user-D) declines the request, then display a message to the user.
8. Notify user-D for MAC process completion.

At this point, user-C and user-D are MAC (Mutually Approved Contacts) contacts.

Section I – Send Message

Suppose that user-A at domain-A wishes to send a mail message to a user-B at domain-B.

1. User-A logs in to their elias-mail service, or elias-mail access software.
2. User-A writes a message and sends it to user-B#domain-B.tld.
3. The mail service at domain-A sends a new message notification mail.elias.domain-B.tld:



The notification tells the recipient that there is a new message and contains:

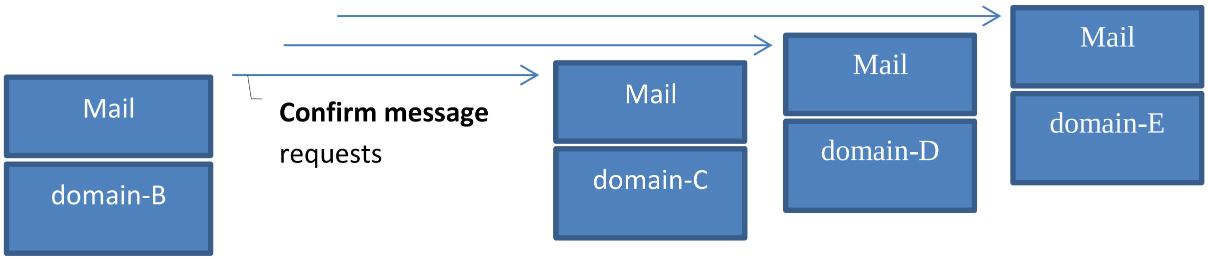
- Notification unique ID
- Notification timestamp
- Sender name
- Message unique ID
- Secret
- Notification signature
- MAC password (hash) if the user-A#domain-A.tld and user-B#domain-B.tld are MAC.
- Sender's Public signing key

Note that the notification does not include any details about the message, which allows the sender to modify the message until the moment the recipient fetches it.

4. The mail service at domain-B receives the new message notification. Is the message is from a MAC?
   4.1. Yes – send a new message acknowledgment to the sender. END.
   4.2. No
      4.2.1. Is the sender a known spammer?
         4.2.1.1. Yes – return message declined with reason to the sender (mail.elias.domain-A.tld).
         4.2.1.2. No – the recipient mail.m3m.domain-B.tld requests all its MACs to verify the existence of the declared mail message.

The recipient contacts only its MACs (as opposed to all known to it email addresses) to limits the number of messages that are sent, and thus also preventing DDoS-like attacks on mail servers.

Suppose that user-3B#domain-B.tld is MAC with users on services mail.elias.domain-C, mail.elias.domain-D and mail.elias.domain-E. If user-3B#domain-B.tld has no MACs, then they either accept the mail or ask non MAC domains for the favour to act as MAC in establishing the legitimacy of the message from user-A#domain-A.tld to user-B#domain-B.tld.

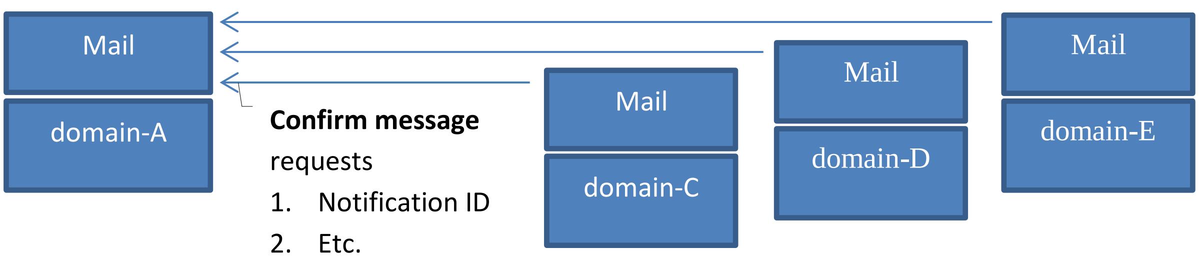


The new message confirmation and witnessing request contain:

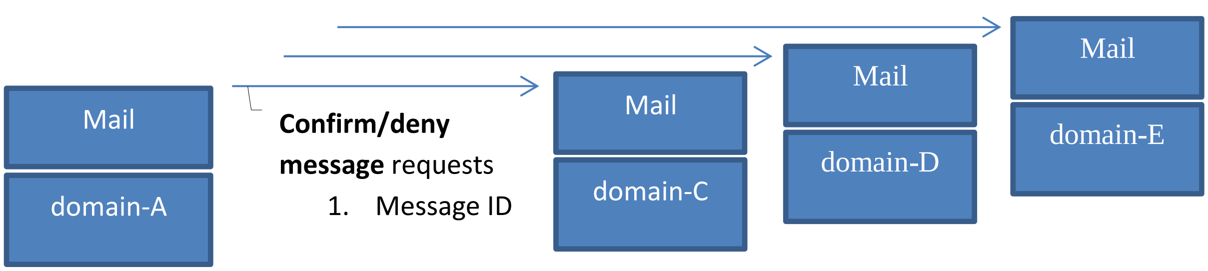
- The sender's email user-A#domain-A.tld
- Notification unique ID
- Notification timestamp
- Message unique ID
- Notification signature

Note: The verification can be made by one or several levels of depth of MACs.

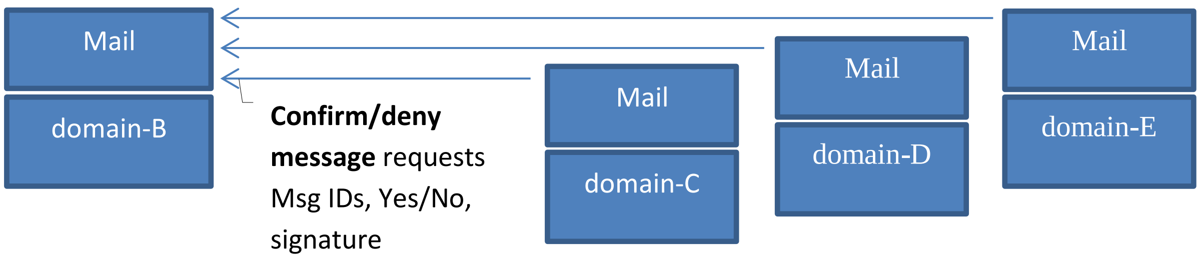
4.2.1.2.1. The witness mail services (mail.elias.domain-C/D/E.tld) call the sender (mail.elias.domain-A.tld) to confirm the message sent to mail.elias.domain-B.tld.



4.2.1.2.2. The sender (mail.elias.domain-A.tld) confirms or denies the existence of the message, with a digitally signed Yes/No response.

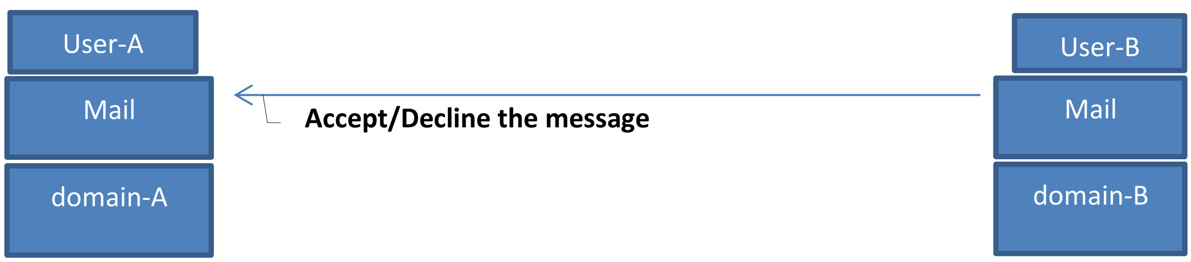


4.2.1.2.2.1. The witness mail services confirm or deny the message existence back to the recipient (mail.elias.domain-B.tld).



4.2.1.2.2.2. Remember the information for some time.

4.2.1.2.3. The recipient (mail.elias.domain-B.tld) responds to the sender (mail.elias.domain-A.tld).



4.2.1.2.3.1. If 100% return message confirmed then:

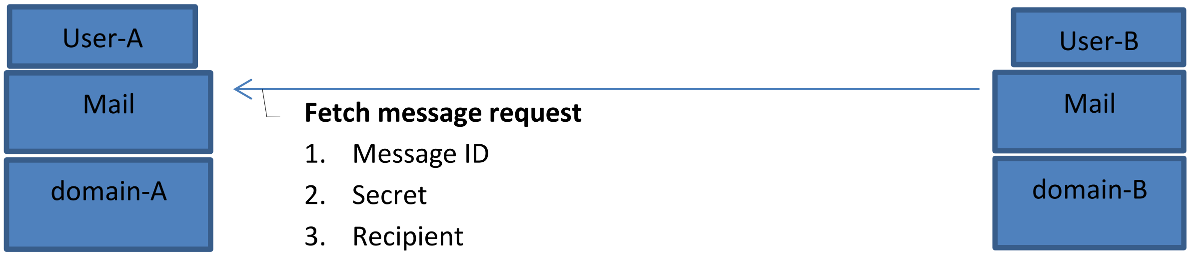
1. Send to sender (mail.elias.domain-A.tld) message accepted.
2. Store the message data for processing when the recipient user logs in
3. Record the responses of all contacts so lying contacts can be identified.
4. End of protocol section.

4.2.1.2.3.2. If < 100% return message not confirmed then:

0. Try a few times using various contacts as witnesses, if always < 100, then:
1. Send to sender (mail.elias.domain-A.tld) message declined with reason.
2. Record the responses of all contacts so lying contacts can be identified.
3. Sender mailbox: Tell the user about the failed sending.
4. END.

Section II – Receive Message

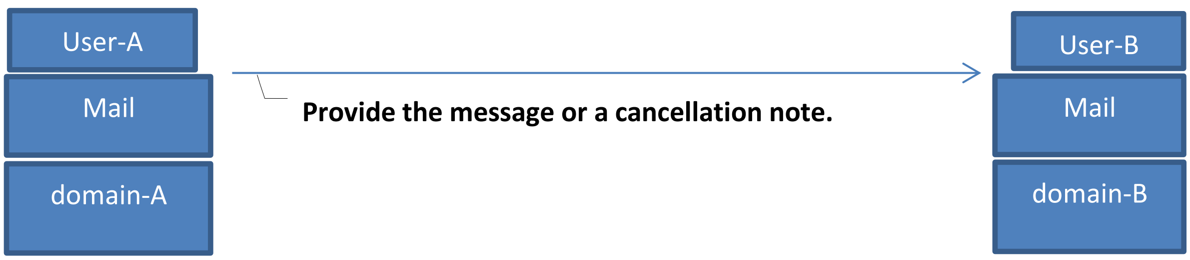
1. The user logs into their elias-mail account or uses elias-mail compatible client to log in.
2. The mail recipient (user-B#domain-B.tld) server goes through all confirmed message notifications and fetches the messages, in this example the message from user-A#domain-A.tld.



The fetching message requests the actual message and contains:

- Fetching ID
- Message ID
- Secret
- Recipient
- Notification ID

3. The sender (user-A#domain-A.tld) provides the actual message or a message cancellation note.



4. The recipient's inbox lists the valid messages.

5. The user reads the message.
   a. If the sender is a contact of the recipient or is a contact of any of the users of mail.elias.domain-B.tld then END.
   b. If the sender is not in the recipient contacts, then the message has a "Spam" checkbox and a "Blacklist sender as a spammer" button. If the user blacklists the sender then the mail server notifies all its contacts that a "sender/domain/IP address is a spammer". The notification can be sent recursive say 20 levels. END

The ELIAS architecture enables numerous algorithms that can be invented to confirm the message origin and sender. Such algorithms can also use additional ELIAS services to establish the sender identity and the legitimacy of the message, such as Address book, Social services, and connections, etc.

Section III – Antispam and anti-phishing

Contacts count spam and phishing notifications, say 5 alerts for a sender mark them as a spammer for some time, say 3 days. Contacts know that the accusation is a truthful opinion as they have a receipt that a message was sent. Thus the mechanism is able mostly root out spamming, phishing, etc.

Connector Services

The Connector services allow users to connect to providers of services. These services are akin to RSS feeds. However, a site would have a single RSS service that would handle all RSS feeds of the user. Indifference, the Connector services would be one for each linked website. The linked website would supply its connector service so that the user can connect to it from their Home service. For example, a newspaper can provide a Connector service called “BestNews”, which people can install on their sites. Then they can read the newspaper, or whatever the BestNews connector service allows them to do, from their Home service without navigating to the newspaper website.

Conclusion

The ELIAS Service Level Internet Architecture provides a simple and effective solution to transform the internet from a service-centralized system where monopolies abuse all other participants, into a service-distributed system where information and services are distributed in their natural configuration providing optimal benefits for all participants. The model removes middlemen by allowing the provides and users to connect directly.

The current websites of organizations and people will require very small or no changes at all. Anyone can create and offer new services. Websites will only need to install the services that they require: typically a search service and possibly social, sell, etc. on their existing webservers.

Upon adoption of the model, some services may change their outreach and characteristics. For example, societal (Facebook like) service may no longer be used by many organizations as new services may provide a better fit for their needs.

The model can stimulate business for hosting companies and manufacturers of physical and software personal web servers as well as software developers who create service implementations. The model can also increase the profitability of businesses, as they will be relieved from the expenses for inefficient ads, sales fees (Amazon, eBay), 30%+ broker websites fees, etc. Individuals can now keep their information private and are now able to benefit from adverts that they can include on their social pages. Job seekers and employers will be able to connect directly. Businesses can begin to trust that business correspondence received via elias-mail is generally not spam. Censorship becomes impossible. Socio-economic losses and negative environmental impact caused by centralized search engines and centralized services can decrease significantly, as they become mostly obsolete and abandoned. To mention only a few of the positive developments that the model can bring.

The ELIAS model does not preclude the service-centralized entities from existence. However, it is the opinion of the author that those will decline as users and businesses adopt the proposed or similar decentralized model.

Closing Notes

This paper is only an initial blueprint of ideas on how to free the internet from undue monopolies. As with any technical proposal, subsequent development is likely to require changes due to unseen or incorrectly inferred relationships between entities and requirements. However, the core principle that information is always stored and is processed on its owner's facility, and is exchanged with participating parties through compatible interfaces is fundamental and must remain in any implementation.

References

[1] "What Is The OSI Model?". CloudFlare. 2019. Retrieved November 4, 2019.

Other notes on adverts

As an entirely different but related subject, it seems an interesting question if payment by seeing adverts is legal. It certainly is immoral in the author's view. There seem to be several relevant points.

1. The user, i.e. the buyer, clearly does not know the "de-facto" price that they will pay for using the service in terms of monetary equivalent of the time spent to see the adverts. Thus, the question: Is it legal to provide a service with an undefined price? Is this not misleading and hence illegal?

2. The user does not know for what they bargain for as adverts can have content offensive to the user, which they would not consider seeing otherwise. So, is it legal to sell "unseen" goods, which cannot be returned and refunded?

3. Actor A pays actor B to take the time of person C, in whatever way, and promulgate the information of A to C, so that C is compelled to buy form A. Is this legal? Does it not sound like a schema by A and B to defraud C, especially as C believes that they use a FREE service of B?

If adverts are legal, then shouldn't there be some protection for users that limits their exposure to adverts. For example, there can be a legal limit on the number of adverts displayed on a page or the number of adverts per visit.

Or, if there are sufficient grounds, then outlaw adverts on non-personal websites. That is, to make online services require payment and not favor to use them, by law. So Facebook and Google will have to charge their users directly. Although ELIAS provides a complete solution to the problem with centralized services, outlawing adverts can help to maintain fairness in a monetary sense.