LINK PATTERNS IN THE WORLD WIDE WEB

Tawfiq Khalil¹ and Ching-Seh (Mike) Wu², Ph.D.

¹Department of Computer Science and Engineering, Oakland University, Rochester, Michigan, USA
²Department of Computer Science and Engineering, Oakland University, Rochester, Michigan, USA

ABSTRACT

In this paper we classify the different link patterns during the evolution of the World Wide Web and the methods used to support data modeling and navigation, we identify six patterns based on our observation and analysis and examine the core technology supporting them (especially the link mechanism) and how data is structured within them. First, we review the document web paradigm known popularly as Web 1.0, which has primarily focused on linking documents. We overview the common algorithms used to link people to documents and service providers. Then we review Web 2.0, which has primarily focused on linking Web Services (known as mashups) and people (known as the Social Web). Finally, we review two more recent patterns: one for linking objects to create a global object web and one for linking data to create a global data space. As part of our review, we identify some of the challenges and opportunities presented by each pattern.

Keywords: Big data, classic web, global document space, graph databases, global object space, global data space, Linked Data, NoSQL, semantic web, social web, relational databases, Web 1.0, Web 2.0, Web 3.0.

I. INTRODUCTION

The World Wide Web (WWW) has enjoyed phenomenal growth and has received wide global adoption without regard to factors such as the age, ethnicity and location of its users. The web’s user base has grown continually since its inception and is expected to comprise nearly 3 billion users by 2015 [1]. Many features have contributed to the Web’s success, such as its ease of use, the near-ubiquity of its access, and the wealth of valuable content it contains. The ability to link to content and to navigate from one web page to
another is a core functionality of the original architecture of the classic web. The web’s evolution since its inception can be characterized according to the nature of what it has linked. The web has grown from a document repository to an application platform allowing users to conduct transactions such as buying books, paying bills, taking online classes, and connecting to each other. As a result, the general function of the web has evolved from merely linking documents to linking people to service providers, linking services to other services, and linking people to people. Researchers have made many attempts to build a more intelligent web, and in this paper we will focus on linking objects and linking data.

The web presents many opportunities and challenges to the research community, including how to better interlink the classic web and obtain knowledge from the information available on the web. Web mining techniques can be classified according to three categories: web structure, web content and web usage mining [2]. These techniques are characterized based on the data (i.e., structure and linkage) used for the mining process. Web structure mining focuses on the structure of the links between the documents. Web content mining extracts the content of the document as an input to the mining process. Web usage mining uses the user’s interactions on the web as an input to the mining process. Web 2.0 has contributed to the increase of the size of data available on the web and necessitate research for structuring and linking data differently and more intelligently.

This paper is organized into four parts. Section II reviews the data structure and linkage in classic web. It identifies two link patterns: among documents and between people and documents (including service providers). Section III focuses on Web 2.0 and the explosion of data leading to Big Data. It identifies two link patterns: linking services and linking people. Section IV reviews the Web of Object architecture and how objects are interlinked. Finally, section V will review Linked Data and its technology stack.

II. CLASSIC WEB (WEB 1.0)

2.1. Fundamental Architecture

Three fundamental standards comprise the infrastructure of the World Wide Web:

- **Uniform Resource Identifiers (URI):** Globally unique identifiers for resources on the internet [3]
- **Hypertext Transfer Protocol (HTTP):** A standard internet protocol for accessing resources on the internet [4].
- **Hypertext Markup Language (HTML):** A standard language for rendering information on the browser (content format) [5].

![Figure 1. Web of documents high-level architecture](image)
Fig. 1 depicts an internet user is using a browser to request a static page on the internet by specifying the protocol (HTTP) and the location (URL) for the resource. The request is routed over the internet until it reaches the server hosting the resource. The server responds to the user with the resource and the browser is using the HTML representation of the resource to render the resource information.

Most of the current applications on the internet follow a three-tier architecture to provide a dynamic and interactive user experience (see Fig. 2):

- **Presentation Layer:** This layer is responsible for presenting information to the user and invoking the subsequent Business Logic Layer to perform requested tasks. Users interact with this layer directly.

- **Business Logic Layer (BLL):** This layer contains business rules for the application and invokes the subsequent Data Access Layer to obtain the requested data.

- **Data Access Layer (DAL):** This layer contains classes responsible for connecting and executing commands on the database based on the BLL request. It is beyond the scope of this paper to present an overview of all the various web application architectures.

2.2. Data Structure

The standard data structure used to represent data on the internet is HTML. However, it is only focused on the content and format of a hypertext page and does not have any metadata that describes the content of the page.

```html
<!DOCTYPE html>
<html>
<title>
Sample Page for Link Patterns in the World Wide Web
</title>
<body>

<p><b>This is bold text.</b></p>
<p><u>This is underlined text</u></p>

</body>
</html>
```

Figure 2. Three-tier web application architecture

Figure 3. Sample HTML code
2.3. Data Linkage

2.3.1. Pattern 1: Linking Documents and Anchors Within the Document

The HTML specification provides an explicit linking method (hypertext links, or hyperlinks) that can connect the document to other documents (see Fig. 3, line 1) or link text in the document to another section of the current document (see Fig. 3, lines 2–3). The HTML <a> tag stands for “anchor” and its attribute href stands for “hypertext reference” [6].

```html
1. <p><a href="http://secs.oakland.edu">Link to Oakland University</a></p>
2. <p><a href="#TOC">Click here for TOC!</a></p>
3. <p><a name="TOC">Table of Contents</a></p>
```

**Figure 4.** Example of HTML hypertext links

2.3.2. Pattern 2: Linking People to Documents and Services

In this pattern, linkage is not explicit as in Pattern 1. Search engines were created to fulfill that need. An online user who does not have the URL (Uniform Resource Locator) for the document or service can use a search engine to find documents related to his or her keywords. Search engines have rapidly become the gateway to the web and represent the most visited sites on the internet. Search engines use web mining techniques to identify matches based on the user’s criteria. Data is needed in order to apply these techniques [7]. The decentralized nature of the web environment means that it must be “crawled” and the resultant information must be centralized in a search index.

In Fig. 5, a web crawler (also known as a spider or robot) selects a set of URLs from previous crawls as a starting point, fetches information about those pages, and subsequently follows the links on those pages to identify additional pages to index. During this process, new information is inserted into the search index and changes are either updated or deleted. The query interface is responsible for executing queries against the search index and returning results to the user based on their criteria.

**Figure 5.** Search engine high-level architecture
Hyperlink Induced Topic Search (HITS) was developed at the IBM Almaden Research Center and uses the hyperlink structure of web pages to infer notions of authority. When a page has a hyperlink to another page, it implicitly endorses it. HITS defines two types of pages: hubs, which provide a collection of links to authorities, and authorities, which provide the best source of information about a subject. The central concept of this algorithm is that there is a mutually reinforcing relationship between authorities and hubs (i.e., a good hub is a page that points to many good authorities and vice versa). However, the web does not always conform to this model: many pages could point to other pages for reasons such as paid advertisement and may not necessarily endorse them.

PageRank was developed at Stanford University by Larry Page and Sergey Brin, the co-creators of Google. PageRank is similar to HITS in its method for finding authoritative pages. The key difference is that not all links (votes) are considered equal in status. Highly linked pages have more importance than scarcely linked pages. Backlinks (incoming links) from high-ranked pages count more than links from lower ones. To calculate the PageRank of a page (or node), it is mandatory to calculate the PageRank of all pages (nodes) pointing to it [12].

There is a great interest among the research community in determining how to optimize search engine algorithms and to improve search results by taking into account the content of a page (such as its title, headings, and tags) as well as user behavior (such as clicks and co-visited pages) [8–11]. The search engine must also be capable of identifying and blocking the efforts of spammers to spuriously increase page ranking using methods such as doorway pages (pages full of keywords related to the site), pages dedicated to creating links to a specific target page, and cloaking (pages that deliver different content to web crawlers than that seen by regular users).

2.4. Evaluation

The goal of the World Wide Web is to provide a decentralized and dynamic environment for interlinked, heterogeneous documents. Linking documents is, of course, a built-in feature of HTML. Search engines satisfy the need for linking people to documents. Web crawlers follow hyperlinks to create search indexes and thereby centralize information about the decentralized web using web mining algorithms.

Many challenges face search engines. Web documents are written for many different locales in many different languages, dialects, and styles. The number of documents on the web is always on the rise. Efforts by spammers to improve their visibility and ranking within search results are continuously evolving. There is also no guarantee that a site will be indexed in the crawling process by a certain time or at all (unless there are many other external links pointing to the page). Crawlers may not be able to follow links that use Java events such as onclick. Crawlers will also fail to index documents that are accessible only by a search box (dynamically generated content) or that require authentication. In addition, the web is constantly moving target for indexing: it is a dynamic environment and its content is frequently added, updated, dynamically generated, and deleted.

3. WEB 2.0

3.1. Fundamental Architecture

The following are the key technologies underlying Web 2.0:

- XHTML and CSS as presentation standards
- The Document Object Model (DOM) for dynamic display
XML and XSLT for data interchange and transformation
XMLHttpRequest for asynchronous data retrieval
Web Services:
- XML-RPC, SOAP, and REST for communication protocols
- WSDL for describing interfaces
- UDDI for discovering web services
AJAX (Asynchronous JavaScript and XML) enables applications to update portions of a page without the need to reload the entire page and prevents the user from having to wait for the page updates.
Smartphones for easing communication and collaboration in the Social Web

AJAX is considered an essential component of Web 2.0 applications [13], [14].

Web 2.0 was not clearly defined until Tim O'Reilly prescribed the concepts and principles of Web 2.0 [13].

3.2. Data Structure
XML is a hierarchical representation of data that enables machines and applications to easily interchange, transform, and parse data. Other formats have also been used such as JSON which uses key-value pairs to represent data.
XML has played a large role in the evolution of Web 2.0. Elements of XML were combined with HTML to create XHTML in order to enrich web documents and make HTML more accessible, machine and device independent, and well-formed. The Simple Object Access Protocol (XML over HTTP) has also been used for web services communication. It is also part AJAX which is a key component of Web 2.0 that enriches internet applications.
Syndication technologies such as RSS and Atom are used in Web 2.0 to notify subscribers (users and applications) about the existence of updated content. They are also built on XML (see Fig.s 7 and 8).
Figure 7. An example of RSS code

Figure 8. Rendering of the RSS code presented in Fig. 7

Link Patterns in the World Wide Web
Paper on Link Patterns in the web
RSS Version: 2.0
Language: en-us
Classic Web: Information on classic web
Web 2.0: Information on web 2.0

3.3. Data Linkage

3.3.1. Pattern 3: Mashups, Orchestration, and Choreography

In this pattern, applications and web services consume and remix information from heterogeneous sources. Sharing information between web applications and services can enrich the user experience and contributes to a more dynamic and powerful web.

Fig. 9 illustrates a web application utilizing Yahoo’s map service to present directions to the user and also using an RSS news feed to present news to the user related to his or her interests, etc.
Web Services are software components accessible on the web and exposing their operations to others for use. Linking web services is the foundation for SOA (Service Oriented Architecture). There are two models for linking web services, as illustrated in Fig. 10. In the orchestration model, participating web services are controlled by a central web service that manages the other services’ engagement. Meanwhile, in the choreography model, each web service must know when to become active and with which service it needs to interact.

![Image of two models for SOA: web service orchestration and choreography](image)

**Figure 10.** An illustration of two models for SOA: web service orchestration and choreography

### 3.3.2. Pattern 4: Linking People

With the advent of Web 2.0, people became an integral part of web applications. Online users became contributors rather than passive consumers. Users gained the ability to tag, comment, review, publish, vote on, and express approval for (i.e., “like”) content. They also gained the ability to link to their friends and create a community of friends or a network of professionals. Web 2.0 users can also contribute to the news, sending photos and comments when news channels are not available. For these reasons, Web 2.0 has been described as having an “Architecture of Participation” [13].

Despite this, data published on the web is still not structured in a semantic way that represents its relationship to other entities. The architecture of data stored in the web application must be able to accommodate a large and dynamic volume of information, including relationships among entities. In addition, it must return information about linked entities in real time.

Traditionally, many web applications used relational databases to manage information [16], [17]. Relational databases rely on tables to manage similar data into rows and columns. Rows are uniquely identified inside the table via a primary key. It is a good practice to normalize tables to reduce data redundancy and increase data integrity. Foreign keys are used to link tables (entities) to each other. Normally, tables are joined in the query when selecting data that is available in multiple tables by utilizing the foreign key relationship. For performance reasons, database architects typically denormalize tables to reduce joins and improve performance, which subsequently causes data redundancy and may negatively affect data integrity.

Figure 11 depicts a small entity relationship model that has information about users, their friends, and the orders they made. Creating queries to find information from this model can quickly become very complex. For example, take a query intended to find friends of Tawfiq who bought the same item as he did. The more joins we have, the more performance will suffer. It would be more challenging to find friends of a friend of a friend three or four
levels deep who bought the same product. Relational databases have been widely used and successfully accommodated to business needs; unfortunately, they lack efficiency in addressing strongly interlinked datasets.

Figure 11. The entity relationship model

The objective of non-relational databases (also known as NoSQL) is to avoid the potential processing overhead and complexity of relational databases by allowing redundancy and by relaxing the constraints imposed on relational databases. NoSQL is more suitable and prevalent in cloud environments because of its ability to horizontally scale (via sharding) more than relational databases. Most NoSQL databases can be classified according to their underlying storage mechanism: key-value (e.g., Amazon Dynamo), document (e.g., MongoDB, RavenDB, CouchDB), column family (e.g., Apache HBase, Google BigTable), and graph store (e.g., neo4j, HyperGraphDB).

The first three NoSQL models listed above store data in a disconnected manner which necessitates inserting an identifier (such as a foreign key) into the other document, value, or column. This is not enforced by the store and could lead to dangling pointers. On the other hand, graph databases naturally support network topologies such as social networks. Fig. 12 depicts the entity relationship in Fig. 11 using a graph model and it shows how easy it is to follow the path to answer a query seeking friends of Tawfiq who bought the same item as he did.

Figure 12. The property graph model
In a graph database model, real instances are depicted but it does not contain a general entity relationship model like relational database models do. The graph model consists of nodes (i.e., entities), relationships (directed edges or connections that have a label and are not generic), and properties (in the form of a key-value pair) for both nodes and relationships. Adding properties to the relationship provides a great value by representing metadata (such as the strength of the connection) between entities. Another advantage of the graph model is that it is schema free. New nodes and relationships can be added without the need for migration or downtime as is the case with relational databases.

3.4. Evaluation

Although Web 1.0 has enjoyed many successes and made access to information nearly seamless, it has provided only read access to users and has prevented them from contributing to its knowledge base. Researchers have tried to compensate the lack human feedback by developing algorithms looking at links between pages, content, and user behavior.

Web 2.0 embraced the human intelligence and changed the role of online users from passive consumers to contributors. It created an attractive, easy to use, and powerful platform for collaboration. This platform is used not only for people to contribute to pages (with reviews, tags, comments, and other content) but to also link services and people.

Developers are able to quickly build powerful applications by calling ready-made web services instead of building them from scratch. Researchers have developed various languages (such as WSFL, WSCI, as BPEL) for web service composition to build a business processes. Manual composition of web services is time consuming and not scalable. This has led researchers to automate the composition process based on functionality and QoS attributes (such as availability, reliability, security, and cost) for the service selection process [20]–[24]. As a testament to the importance of this topic, a Google Scholar search for “dynamic web service composition” yields more than 500,000 articles and research papers. However, OWL-S is a service ontology that enables dynamic web service discovery, composition, and monitoring. It has three parts: the service profile, the process model, and the grounding [25]. A key observation is that the service profile (which includes QoS information) is provided by the service publisher and consumers cannot contribute to it. This is reminiscent of the read-only restriction that is characteristic of Web 1.0. Researchers have tried to address this issue by proposing new frameworks that include the application of social networks to web service composition [26]–[30].

Web 2.0 has also created a convenient and intelligent environment for people to connect to each other. People are increasingly using social networks to connect with friends and others from various backgrounds. Web applications are charged with managing information in an optimized way and applying analytical and inferential algorithms in order to make intelligent recommendations based on the network topology. Traditional relational databases fall short due to their rigid constraints and their inability to scale horizontally. However, social networks are a natural fit for the graph model. The ability to model social network data in a graph provides a great advantage due to the several hundred years of mathematical and scientific studies made on graphs. Breadth-first (one layer at a time) and depth-first (one path at a time) search algorithms can be used for traversing the graph. Dijkstra’s algorithm (using a breadth-first search) can be used to find the shortest path between two nodes in the graph. There are also many graph theory techniques that can be applied to the graph for analysis and inference. Triadic closure (if node A is connected to node B and also connected to node C, there is a possibility that B has some relation to C) and
structural balance principles can be applied to induce relationships and make recommendations [31].

Social networks have created many opportunities for researchers. A key area of developing research is sentiment analysis and opinion mining. Analyzing text (such as reviews, comments, messages, etc.) to determine whether it reflects a positive or negative attitude is a critical step in the sentiment analysis process [32], [33]. Another prominent research area is community detection based on the analysis of linked people interests, “likes,” and induced opinion [34]–[36].

4. FOREST: WEB OF INTERACTING OBJECTS

4.1. Fundamental Architecture

Functional Observer REST (FOREST) is a resource-oriented architecture proposed by Duncan Cragg [37]. In this architecture, domains or applications can share information via interlinked objects, whether locally or across the network. An object is identified by a URL that includes the object’s globally unique ID. An object’s state is determined by its own state and the state of other objects that it observes without the need for calling the other objects’ methods (Functional Observer Pattern). REST over HTTP (using GET for poll and POST for push) is used for object state transfer (see Fig. 13). FOREST objects can be written in traditional languages such as Java, C#, Python, or naturally in declarative languages (such as Prolog or Clojure, or Erlang). It is necessary to use HTTP headers ETag (digest of the object content) and max-age (how long to cache the object) to control the client-side cache for future GET requests in the initial post.

4.2. Data Structure

FOREST objects can be serialized or represented in XML, XHTML, or JSON

4.3. Data Linkage

4.3.1. Pattern 5: Linking Objects

Hyperdata is data represented in objects that are linked up into a global object Web by using URL and object unique id as a linkage method
4.4. Evaluation

Functional Observer REST (FOREST) is a resource-oriented framework for implementing domain and application logic by creating functional dependencies among linked objects. An object is identified via a unique identifier and its states are evaluated according to its current state along with the states of the objects’ that it observes. However, a given object cannot tell by whom it is observed. The interactions and the state dependencies are based on the application logic and are not globally realized. That is, objects are not semantically described but work together according to the application constraints. Overall, the framework provides interoperability (objects can be serialized to XML, JSON, or XHTML), scalability (objects can be distributed and linked), and evolvability (observed objects’ state can be pushed and pulled).

5. LINKED DATA- SEMANTIC WEB (WEB 3.0)

Many people use the terms Linked Data, Semantic Web, and Web 3.0 interchangeably. It is critical for our discussion to clarify the distinction among them. The next version of the World Wide Web (Web 3.0) focuses on supplementing raw data on the web with metadata and links to make it understandable by machines for automation, discovery, and integration. Semantic Web employs a top-down technology stack (RDF, OWL, SPARQL, and RIF) to support this goal. Linked Data is a bottom-up approach that uses Semantic Web technologies to enact Web 3.0.

There are other approaches for enacting Web 3.0 without the use of Semantic Web technologies. Microformats and microdata are examples. Microformats provide standard class definitions to represent commonly used objects on the web in HTML (objects such as people, blog posts, products, and reviews) . This allows web crawlers and APIs to easily understand the content of the site. It has also rel attribute that provides a relationship meaning for the hyperlink. For example, rel="home" in <a href="URL" rel="home">Home</a> gives the hyperlink a meaning that the link is relative to the homepage [40].

Similarly, schema.org provides schemas (i.e., itemtype which has properties similar to microformats classes and properties). Microdata format (such as itemscope, which includes the itemtype and itemprop) is used to provide metadata to HTML content [42].

Linked Data is similar to Microdata and Microformats in its method for enacting Web 3.0. However, it uses a Semantic Web technology stack (i.e. OWL) for the vocabulary and RDF for the data model. In addition, vocabulary in Linked Data (classes in microformats and itemtype in microdata) is not limited to a certain organization for updates. Finally, another key difference is that the described data item in microformats and microdata does not have a unique identifier as it does in Linked Data (URI). For these reasons, our focus in this section will be on Linked Data.

5.1. Fundamental Architecture

- **Uniform Resource Identifier (URI)** is a globally unique identifier used as a name for the described entity
- **Hypertext Transfer Protocol (HTTP)** is a standard internet protocol to for accessing resources on the internet.
- **Resource Description Framework (RDF)** is a graph-based data model for describing things (entities) in the form of triples: subjects (nodes), predicates (edges as a relations), and objects (nodes can be a literal value or a referring to a subject of another triple). For example, in Fig. 12 one of the triples in the graph has Tawfiq as the subject, ordered as a
predicate, and order id:12 as an object. In Fig. 14, the RDF model describes two persons (Tawfiq and Mike) and establishes a connection (“knows”) between them using FOAF (Friend-of-A-Friend) vocabulary.

```
1. <rdf:RDF>
2.  
3.  
4.  
5.   <foaf:Person rdf:ID="me"> 
6.     <foaf:name>Tawfiq Khalil</foaf:name> 
7.     <foaf:givenName>Tawfiq</foaf:givenName> 
8.     <foaf:familyName>Khalil</foaf:familyName> 
9.     <foaf:mbox rdf:resource="mailto:tawfiq.khalil@hp.com"/> 
10.   <foaf:schoolHomepage rdf:resource="http://www.oakland.edu"/>
11. </foaf:Person>
12. </rdf:RDF>
```

Figure 14. Example of RDF Model

5.2. Data Structure

RDF can be serialized in RDF/XML, RDFa (RDF embedded in HTML), Turtle, N-Triples, and RDF/JSON.

5.3. Data Linkage

5.3.1. Pattern 6: Linking Data

There are three types of RDF links. Vocabulary links point to the definition of the vocabulary used to represent the data. Relationship links point to related entities in other data sources. Identity links are pointers to other descriptions about the same entity from different data sources. For example, if we are describing a person, the vocabulary link could use the FOAF (i.e. OWL DL) vocabulary to describe the person and his social connections (Fig. 14, line 4), the relationship link would be a link to his school or a friend (Fig. 14, line 11), and the identity link can provide links to different representations about the person using owl#sameAs to indicate that the URIs are representing (have different views of) the same entity from different sources (Fig. 14, line 16).

5.4. Evaluation

The intent of Web 3.0 is to make data understandable by machines to improve reuse and integration. Linked data is based on two major Semantic Web technologies: RDF as a graph-based data model and OWL (Web Ontology Language) as a vocabulary for describing classes and properties including the relationships among classes, equality, cardinality, and enumerated classes.

Linked Data has received worldwide adoption and from various domains. Linking Open Data Cloud (LOD Cloud) group catalogs datasets that are presented on the Web as Linked Data has become an immense repository for publishers. As of September 2011 there were more than 290 datasets available which include more than 30 billion triples published by various domains [43]. Search engines such as Falcons and SWSE enable users to search for Linked Data using keywords. In addition Sindice, Swoogle, Watson, and CKAN provide APIs for applications to look up Linked Data.
While access to Linked Data provides great opportunities for publishers and consumers, they are also faced with many challenges. The human interaction with Linked Data is not as user friendly and intuitive as Web 1.0. HTML presents data in a friendly manner for users to view, however, data formatting is missing in Linked Data as it is intended for machines to understand. In addition, applications face many challenges in order to search, integrate, and present the data in a unified view. There are thousands of ontologies (third party and user-generated) used to describe the data. Data fusion requires data integration and aggregation from different sources written in different languages thus requires data cleansing (including deduplication and removal of irrelevant or untrustworthy data) and mapping of the schemas used to describe the data. Many researchers have tried to address the issues of data quality and trustworthiness by using cross-checking, voting, and machine learning techniques [44]. Inference techniques can also improve quality by discovering new relationships and automatically analyzing the content of data to discover inconsistencies during the integration process.

Link maintenance presents another challenge since RDF contains links to other data sources that could be deleted at any time, causing the links to be dangling. Frameworks to validate links on a regular basis or use syndication technology to communicate changes are proposed means to address this issue. Another interesting research area is related to automatically interlinking data based on its similarity.

Another challenge in Linked Data is related to the core technology that it uses. OWL has significant expressivity limitations. OWL 2.0 was introduced to resolve some of the shortcomings of OWL 1.1 such as expressing qualified cardinality restrictions, using keys to uniquely identify data as well as the partOf relations (asymmetric, reflexive, and disjoint). In addition, OWL is written in RDF (triples) which makes it complex to express relations such as “class X is the union of class Y and Z.” Essentially, OWL is a description language based on first order logic and it is unable to describe integrity constraints or perform closed-world querying [46]. The Rule Interchange Format (RIF) was introduced on top of OWL to include rules using Logic Programming. However, a true rule specification in logic programming is fundamentally incompatible with OWL [47].

6. CONCLUSION

The World Wide Web has been adopted by billions of users globally for its wealth of information and its ease of use. Information on the web has been linked in many ways since its inception to optimize automation, discovery, and reuse. The more relationships and links that are applied to the data, the better knowledge can be induced from it. It is important to recognize the different link patterns in the web to identify some of the opportunities and challenges in each pattern and to be able to recommend new patterns to better serve online users and service providers. It is evident that each iteration of the web’s evolution—from linking documents to documents, to linking people to documents, to linking services to services, to linking people to people, to linking objects to objects, and finally to linking data to data—has increased the value of the network and has made it an increasingly rich and valuable platform. These efforts and their results are inspiring researchers to work on the challenges, recommend new patterns in the web, or to apply these patterns to real life physical objects as we see in Internet of Things (IoT) initiative. This paper is also used as a foundation for our research for a new web pattern to optimize the use of data on the web and overcome some of the shortcomings in the current methods by focusing on better methods for publishing data and improve linkage.
### TABLE I. LINK PATTERNS EVALUATION RESULTS

<table>
<thead>
<tr>
<th>Link Patterns</th>
<th>Linking Documents</th>
<th>Linking People to Documents</th>
<th>Linking Services</th>
<th>Linking People</th>
<th>Linking Objects</th>
<th>Linking Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Version</td>
<td>Web 1.0</td>
<td>Web 1.0</td>
<td>Web 2.0</td>
<td>Web 2.0</td>
<td>Web 2.0</td>
<td>Web 3.0</td>
</tr>
<tr>
<td>Link Type</td>
<td>Explicit</td>
<td>Implicit</td>
<td>Implicit</td>
<td>Implicit</td>
<td>Explicit</td>
<td>Explicit</td>
</tr>
<tr>
<td>User Access</td>
<td>Read Only</td>
<td>Read Only</td>
<td>Read/Write</td>
<td>Read/Write</td>
<td>N/A Only</td>
<td>Read &amp; Application Level</td>
</tr>
<tr>
<td>Link Mechanism</td>
<td>Hyperlinks</td>
<td>Search engines</td>
<td>UDDI Search and auto discovery</td>
<td>Request/accept a connection</td>
<td>Object GUID</td>
<td>URI</td>
</tr>
<tr>
<td>Impact</td>
<td>Global Document Space</td>
<td>Ease of access to information and service providers</td>
<td>Service Mashups</td>
<td>Social Network</td>
<td>Global Object Space</td>
<td>Global Data Space</td>
</tr>
<tr>
<td>Shortcomings/Challenges</td>
<td>• No typed links • No collaboration capabilities. • Not understandable by machines.</td>
<td>• Crawlers are used to centralize and index documents instead of realtime or near-realtime lookup. • Search algorithms mainly rely on links which may be used for other purposes. • Web usage and content mining are used to optimize results. • Web content mining must accommodate different languages and locales. • Spurious efforts to increase visibility (spam) must be blocked. • Users are not able to provide feedback on results.</td>
<td>• Optimize dynamic service composition using semantic and social web techniques. • Quality of Service verification.</td>
<td>• Published data is not represented semantically • Sentiment analysis, opinion mining and community detection. • Optimize data management for intelligent recommendations based on the network topology.</td>
<td>• No typed links between objects. • Object interactions and state dependencies are constrained by domain and application logic. • No metadata is used to describe the object.</td>
<td>• Limited expressivity in OWL and RDF • Link maintenance • Data integration and aggregation • Lack of support for integrity constraints and closed-world querying • Automatically interlink similar data</td>
</tr>
</tbody>
</table>
REFERENCES


