I. J. Computer Network and Information Security, **2023, 3, 47-64**
Published Online on June 8, 2023 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2023.03.04

A lightweight Data Exchange Format for Mobile Transactions

M. C. Mohammed Shameer*
Department of Computer Science, Farook College (Autonomous), Calicut, Kerala, India
E-mail: shameer@farookcollege.ac.in
ORCID iD: https://orcid.org/0000-0002-9430-2944
*Corresponding Author

P. P. Abdul Haleem
Department of Computer Science, Farook College (Autonomous), Calicut, Kerala, India
E-mail: abdulhaleem@farookcollege.ac.in
ORCID iD: https://orcid.org/0000-0002-8172-8700

Yazik K. Puthenpediyakkal
Senior Software Engineer, Zalando SE, Wilhelm-Guddorf-Str 12A, 10365, Berlin, Germany
E-mail: yazik.puthenpediyakkal@web.de
ORCID iD: https://orcid.org/0009-0003-9276-8702

Received: 22 December 2021; Revised: 14 May 2022; Accepted: 19 September 2022; Published: 08 June 2023

Abstract: XML and JSON are commonly used data exchange formats that are widely in use in wireless networking environments. The verbose and redundant nature of XML documents incurs huge transportation overheads in data communications. JSON is a data format that reduces the document size; but its scope is confined to text and numeric data. Also due to the reasons such as lack of schema and limited interoperability features, JSON is more suitable for web based applications, compared to wireless or mobile environments. Since the literature reports serious concerns about the performance of existing data exchange formats in resource constraint networks, there is scope for a lightweight data exchange mechanism. This paper introduces a new lightweight, schema aware data exchange format for data representation and interchange. The proposed format, called LXML, is schema aware and non-binary format based on the XML standards and has the potential to be an alternative format for XML and JSON in a wireless environment. Experimental findings indicate that LXML is a less verbose and efficient data exchange format and its performance is found to be better than the existing non binary data exchange formats.

Index Terms: Mobile Computing, Data Exchange Schemes, XML, JSON and LXML.

1. Introduction

The applications running on handheld devices such as tablets or smart phones have a clear separation between the front end, the business logic and the back end. In the majority of enterprise applications, this back end data service is provided by the Customer Relationship Management (CRM) systems or a server. The major challenge in developing services to such devices is the transfer of data between the front end and back end. Interoperability is a major concern in dealing with distributed mobile applications that exchanges information between devices and other systems.

Mobile and related communication technology is pervasive and is growing at a rapid pace. To yield its optimum benefits, it requires modern support infrastructure, support services and specialized task force to manage the distributed environment. Statistics reveal that the total number of smart phone users has crossed over 3 billion in 2020 and the global mobile workforce is expected to reach 1.87 billion by 2022 [1]. Unfortunately, supporting these huge requirements for wireless communications, existing network infrastructure and technologies are being reused. Increasing connectivity and improving the efficiency of the existing systems, protocols and standards is expected to improve the situation**.**

Since the infrastructure in wireless environments is found to be inadequate to support its growing need, there is a scope for improving the efficiency of the existing technologies. Considering the exchange of the data, the common standards used by organizations are XML and JSON. For accessing web services, the Simple Object Access Protocol

(SOAP) and Web Service Description Language (WSDL) use XML for expressing its parameters [2]. Similarly, XML RPC, an XML based information interchange protocol mostly relies on XML for data exchange [2]. Since XML and JSON raise many serious concerns in data transmissions [3], an improved format can play an important role in data communication especially in distributed mobile environments.

2. Resource Constraint Networks (RCN)

Advances in the fields of communication and technologies facilitated the growth of mobile devices and allied services; these developments opened a new realm of computing. Due to the inadequacy of infrastructure facilities to cope with the exponential growth in the mobile computing arena, conventional networking resources have been utilized.

The wireless mobile devices and wireless network have many limitations. The screen size and resolution of a mobile device is much smaller when compared with personal computers. Such devices require a data representation with fewer character sets to suit the mobile environment. Limitation in storage capacity and caching is another constraint in the mobile environment. To improve the responsiveness of applications that depend on online/offline transactions, storage and caching is essential [4,5].

Another important constraint with mobile wireless environments is the limited capacity of battery power. In mobile applications, a huge amount of battery is consumed while transmitting, receiving and processing online data through transactions [6]. Applications that sink data from server or cloud require syncing the data periodically through a process known as polling. Polling heavily consumes battery power [7]. Wireless communication protocols such as WLAN and Bluetooth together with mobile data usage through network providers also considerably increases battery power requirements [7].

The lower processing speed in mobile devices is also a concern in a mobile environment. Processing huge volumes of data constantly reduces the device response time [8].

Increased dependency to traditional network infrastructure for wireless medium, causes frequent failures and latencies [4]. Since mobile devices demand instant responses for their queries and updates, even latencies that may be tolerated by a user in wired devices, are unacceptable in mobile devices. Lack of reliability in wireless networks and latencies should be reduced. The unreliable connection leads to retransmission of data. In a medium with low bandwidth, frequent retransmission of data adversely affects the user's comfort and increases the delay in processing to a large extent.

Chances of security breach are another major concern in the mobile environment. Nowadays XML based web services are commonly used for data transmission. Here security is implemented mostly using secure socket layer (SSL) and encryptions. But such a level of security is not sufficient in a wireless environment which is an entirely different platform with a large number of devices [9,10]. Security should be implemented at element levels rather than at document level.

Unlike the traditional environment, here the battery power is consumed for operating the device. The power consumption depends more on the amount of data transmitted than that consumed by CPU cycles [8].

Enhancing wireless mobile device applications with a data exchange format that can help in reducing the amount of polling, minimizing the amount of data transmitted, reducing processing time, designing models that can tolerate delays, and resolving network related issues can improve the performance of wireless mobile applications to a large extent [4].

Affordability of smart phones is still another issue that is left unaddressed. Even though the number of global smart phone users was 3.6 billion [11-13], only one-third (35.13%) of the world population afford a smart phone. While 81% of Americans own a smart phone, the smart phone penetration ratio in Bangladesh is 5.4%. Also, statistics reveal that only 4% of the adult population in Ethiopia and Uganda own smart phones [11].

Studies reveal that 70% of the total web traffic happens using mobile phones [14]. While accessing social media people prefer mobile phones over desktop or laptop systems. It is projected that the mobile data traffic is supposed to increase by 700% by 2021 [15].

Increase in the number of mobile users and data traffic, at the same time lack of adequate support infrastructures; underline the need of a lightweight data exchange mechanism in mobile wireless environments.

The basic objectives of this research are to develop an alternative data exchange format for mobile and wireless environments that possess the desirable qualities of existing formats at the same time less verbose and low processing overhead. Existing data exchange formats are designed and developed for wired communication environments where document verbosity and its processing time is less concerned. Reusing such technologies in a resource constraint environment reduces the efficiency, wastes the resources and are not advisable.

Many researchers have targeted on measures to enhance the performance of XML and JSON. Binary XML and BSON are examples of such enhancements. Also, researches are carried out to develop alternative data exchange formats including ProtocolBuffer developed by Google Inc. [16]. Though binary formats are better than the non-binary formats, they are not human readable like XML. This paper introduces a new data exchange format called Lightweight XML (LXML) that is expected to be well suited for mobile and wireless environments. The paper is organized as follows: Section 2 outlines the state of the art. Section 3 is about the desirable characteristics for a data exchange format. Section 4 discusses the proposed format. Section 5 discusses document handling in LXML. In Section 6, performance

evaluation is discussed. Section 6 concludes the findings.

3. Literature Review

The term data exchange refers to the set of data communicated between two applications. In a scenario where there are multiple data sources, different data categories, variety of data attributes, numerous data distributors and assessors, the data exchange formats play a vital role in communication. They mainly bridge the gap between the providers and data acquirers.

The major concerns when designing an exchange format are interoperability and data transfer. The available data exchange formats can be broadly classified as binary and non-binary formats used to represent big data. Since binary formats are not human readable, emphasis is given to non-binary formats. This section surveys the existing non binary data exchange mechanisms available in literature.

Applications mostly rely on two standard formats for structuring and interchanging data - XML and JSON. While XML is a mark-up language based on a set of specifications, JSON is purely a data format for JavaScript applications.

3.1. XML

XML is a simplified mark-up language derived from the Standard Generalized Markup Language (SGML). XML is used to specify the structure and content of data in a text based document. It is a generalized data exchange format between applications on the internet. XML is considered as the basic data exchange format for heterogeneous ubiquitous environments such as business to business (B2B) transactions, web services, Web Service Description Language (WSDL) and personalised content delivery [17].

An XML document contains a set of user defined mark ups that encloses the data referred as tags. The organization of the XML document is controlled by the Document Type Declaration (DTD) that precisely defines the XML.

XML plays an important role in numerous areas in computing such as data communication, data storage and retrievals. Programmable web services are based on XML. It is used as a standard message format for Simple Object Access Protocol (SOAP). Mobile based distributed applications use XML messaging for communicating with remote devices and servers over wireless networks.

XML is a mark-up language that describes the organization and content of a document with structured and semi-structured data. Even though XML focuses only on document operations in structured and semi structured data, its pitfalls in dealing with semi-structured data is yet to be addressed [18]. The important advantages of XML include: simplifies the data exchange between devices and databases, improves data availability in heterogeneous systems, human readable, supports Unicode and is based on international standards, and self-documenting structure with strict syntax and schemas [19,20].

XML uses a hierarchical structure for organizing data using tags. XML documents are dynamic, extensible and are easy to manage changes.

Human readability of XML is due to the abundant use of tags supported by it. In XML, data is organized in a hierarchical structure using tags. There will be a root tag that encloses all the whole document. Inner tags that are enclosed inside the root tag will be a data tag holding the atomic data or a container tag that contains another tag. The container tags may nest other containers (repetitive groups) or data tags. This nesting can go to any number of levels.

The verbose nature of XML is due to the use of tags. Removing tags can reduce the size of the document and hence decrease the processing and transmission time to a large extent [17,20].

3.2. Issues of XML in RCNs

Applications of XML documents can be viewed in different scenarios such as a platform neutral mechanism data exchange, platform independent database storage and retrieval option, configuration and metadata retrievals. Even though XML is the de-facto standard in platform independent data exchange, its use in especially resource constrained environments has many limitations [17,19-23].

- XML documents are verbose in nature due to the use of a large number of descriptive tags [21]. Every data item is enclosed between open and closing tags. In a normal case, a XML document contains less amount of data when compared to the total size of the document. In worst cases, the amount of data can shrink up to $1/5^{th}$ of the total file size.
- The redundant tags increase in file size and thus may cause inefficiency in parsing the file and transmission of files through a network. Parsing a large XML file requires more memory and consumes much processor time. The file size is positively correlated to the cost of network transmission. Due to this compression techniques are to be employed to reduce the file size prior to transmission.
- Designing data structures for handling XML is quite hard as it encourages non-relational data structures. It enforces a large overhead in accessing information.
- XML does not directly map to object models as in JSON. It creates certain confusions while parsing in JavaScript friendly environments.

3.3. JSON

JSON is a light weight, human readable, text based data exchange format. The JSON format considerably relies on the concepts of arrays and lists available with JavaScript. JSON is not a document format or a mark-up language like XML but simply a schema less representation of structured data. Here data is presented in the form of a key-value pair.

JSON is widely supported in many programming languages, databases and web services. JSON includes a minimal set of data types and is more generic in nature. It is a good option for porting data between applications developed in different languages [19].

Even though JSON is a concise text based and flexible data interchange format supported by many programming languages and relational databases, it possesses many shortcomings such as: absence of schema - since the schema helps to check the validity and well forms of the document, the changes of data corruption is higher in JSON due to this reason, JSON is more generic. It supports only IEEE-754 double precision format for numbers. Similarly, there is no specific format for representing date and time, poor readability for large documents - JSON does not support comments, not extensible as XML, verbosity, unavailability of namespaces, difficulty to represent data other than plain texts, and security issues - JSON are commonly used with REST based web services due to its easiness in implementation. Sniffing JSON strings can easily figure out the object properties and alter the values. Such security breaches are also a concern for using this format [9,19,23,24].

3.4. YAML

YAML is a data transmission and serialization standard used by programming languages. YAML is a lightweight and human readable standard used in applications that store, process and transmit data [25]. YAML presents data in both text format and by using native data structures.

Both JSON and YAML are data interchange formats having many similarities in common. While JSON focuses on simplicity, universality and the ease of processing, YAML focuses on data serialization using some native arbitrary structures.

XML and JSON are not related in any manner. XML is a generalised mark-up language whereas YAML is a serialization language. YAML can be considered as the superset of JSON [26]. XML focuses on structuring and imposes many constraints in that sense. There are no such constraints in YAML.

The advantages of YAML includes human readable, easy to use and implement, extensible, portable between applications and programming languages, availability of native data structures that matches with agile programming languages, availability of consistent models to support generic tools, and schema awareness [26].

Every programming language has a certain level of comfort with YAML, yet it is not as popular as XML or JSON because of the following reasons [27]: it provides different views to represent data and its relationships, it is very complex when processing the document, it relies much on indentation formatting, a minor change in intent may result in processing failures, it does not support annotation information and block comments in the document, and it does not preserve order for key-value pairs in map types.

3.5. Protocol Buffers and Massage Pack

The protocol Buffers[16] and Massage Pack[28] are two recently data exchange formats developed as an alternative to XML and JSON. These formats considerably reduce document size and hence the amount of data communicated can be increased. Protocol Buffer is a platform and language independent format for serialising structured data developed by Google Inc. Message Pack is a compact form of JSON that attempts to serialize arbitrary data structures with type tags. It provides no structure validation and it is not schema based [28].

The limitations of existing formats clearly describe the need of a data exchange format that retains the extensible nature of XML and is more efficient while transmission and processing. Otherwise, applications that make use of XML will strain in networks with smaller bandwidth and fewer infrastructure resources.

The literature survey clearly underlines the need of an alternate messaging format in wireless environments due to the following reasons:

- Verbose nature: XML is the most common mechanism used to transmit data which is verbose in nature [29]. Due to the heavy and redundant use of tags and other elements, the standard formats such as XML and JSON are considered as verbose in nature [30]. Mobile business applications deal with transactions of varying sizes. When verbosity increases, the energy consumption and transmission costs increase dramatically in a mobile environment. To reduce the verbosity of the XML the following approaches can be used: compression and decompression techniques at sending and receiving ends, and use of middleware components to enhance performance by reducing the transmission impacts. Both these methods are costly in a resource constraint environment. Another solution is to use JSON, which is a promising format. But it has many limitations [31-33] for its use in the constrained wireless mobile environment. Thus, it can be concluded that XML and JSON are not suitable in resource constrained wireless environments.
- Processing: The libraries used for processing XML consumes much memory and CPU cycles. Use of such libraries hinders the processing speed of an XML document as the file size increases. Even though object

mapping techniques are available with JSON, the document structure creates complications in processing and hence affects the processing performance [29,34].

- Extensibility: XML format is extensible due to the ability to add additional tags, attributes with the help of CDATA sections. But it increases the verbosity of the document and reduces the content density to a large extent. JSON formats are inextensible and do not support namespaces. It is simply a data interchange format [3].
- Environmental Restrictions: Though the standard formats are suitable in traditional environments, the diverse nature of wireless environments imposes many restrictions that hinder their use in constrained wireless environments.

3.6. Desirable Characteristics of a Data Exchange Format

Considering the restrictions that are inherent in a mobile environment, the most important quality that a data interchange format should possess is to reduce the bulkiness of data or verbosity of the document. The desirable qualities that a data interchange format can be outlined as follows [27]: (Structure: There should be a common structure for the document, that is concise, human readable and at the same time easily parable. The structure should pertain to the data relationships such as hierarchical and tree, Interoperability: A format should be language independent, concise and should be easy to extract the data under any platform, Extensible: The format should follow a free-form structure and should not impose any specific constraints such as the need of fixed tags as in html. Also, the format should be capable of accommodating features that could be added later, Schema awareness: the document should have associated schema to validate the well form and semantic properties of the document, support to text, binary and other types of data, and based on existing standards and capable of coexisting with them.

All these criteria may not be applicable in all scenarios but certain criteria will overrun others depending on the need of the situation.

4. Methodology

This is applied research that attempts to refine the existing prominent data exchange format without losing its qualities to suit resource constraint networks. As more business is being migrated to the mobile platform due to the current pandemic scenario, the chances for upgrading infrastructure in the wireless environment is a far possibility to accommodate the exponential growth in data generation. One possible attempt is to change the technology used to exchange data. This underlines the relevance of this research that enhances the way the data is communicated in a wireless environment.

Real production data from business application is considered for evaluating the performance of new message format and analysing it with the existing prominent formats like XML and JSON. In this evaluation, six parameters are chosen that are critical in wireless communication arena are considered. The sample documents are classified into three categories based on size to generate three scenarios - applications that send and receive large amounts of data, applications that handle medium sized data chunks and small sized chunks. Here platforms and network conditions such as bandwidth, buffer size, and hardware architectures are kept constant for sending and receiving the data during the test. The results thus obtained provide quantitative comparisons and qualitative insights into various data exchange formats against LXML that helps to optimize the data communication. Thus, obtained quantitative measures can be used for capturing the performance of different data exchange formats.

```
<products>
        <manufacturer>sonis corp</manufacturer>      → level 0
        <prodyear>2020</prodyear>                    → level 0
        <product>                                    → level 0
                <name>television</name>              → level 1
                <date>
                        <month>Jan</month>           → level 2
                        <year>2020</year>            → level 2
                </date>
                <price>1000</price>                  → level 1
        </product>
        <product>
                <name>refrigarator</name>
                <date>
                        <month>Jan</month>
                        <year>2020</year>
                </date>
                <price>1500</price>
        </product>
</products>
```

Fig.1. An XML document with its level numbers for tags

5. Lightweight XML (LXML) - an Alternative Data Exchange Format

In this section, a data exchange format called LXML is proposed as an alternative to XML. It's an XML-like format. In the LXML format, all the tags in the XML document are replaced by level numbers, except the root tag. Root tag is kept as such to identify the document or transaction. The first tag inside the root tag is given at level 0. All its siblings are also considered at the same level. A nesting inside the tag increases the level number by 1 (Refer Fig. 1).

Now, to create the message, the root tag is kept as such without using angle brackets or closing tags. This is followed by a pattern like **<level Number> data**. A sample XML format and its corresponding LXML format is as shown in Fig.2.

```
<product>                                          product
    <name>television</name>                        <00>television
    <manufacturer>sonis corp</manufacturer>        <00>sonis corp
    <price>100</price>                             <00>1000
</product>
```

(a) XML format (b) LXML format

Fig.2. XML document and its corresponding LXML format

For nested tags only level numbers are specified as shown in Fig.3.

```
<products>                                         Products
    <product>                                      <00>
        <name>television</name>                        <01>television
        <manufacturer>sonis corp</manufacturer>        <01>sonis corp
        <price>1000</price>                            <01>1000
    </product>                                     <00>
    <product>                                          <01>refrigerator
        <name>refrigarator</name>                      <01>abc corp
        <manufacturer>abc corp</manufacturer>          <01>1500
        <price>1500</price>
    </product>
</products>
```

(a) XML Format (b) LXML format

Fig.3. XML document with nested tags and its corresponding LXML format

There will be situations, when a collection of nested tags and atomic tags repeatedly occurs in the document as shown in Fig.4. A shift from nested tag to atomic tags can be identified by a decrease in the level numbers as shown in Fig.4.

```
<invoice>                                          Invoice
    <manufacturer>                                 <00>
        <name>ABC Corp</name>                          <01>ABC Corp
        <brand>series A</brand>                        <01>series A
        <address>streat No 1, New Colony</price>       <01>streat No 1, New Colony
    </manufacturer>                                <00>
    <products>                                         <01>
        <product>                                          <02>A01
            <name>A01</name>                               <02>1400
            <price>1400</price>                            <02>15
            <units>15</units>                          <01>
        </product>                                         <02>A02
        <product>                                          <02>1450
            <name>A02</name>                               <02>10
            <price>1450</price>
            <units>10</units>
        </product>
    </products>
</invoice>
```

(a) XML Format (b) LXML format

Fig.4. XML document containing nested and atomic tags and its corresponding LXML format

5.1. Handling Attributes

Atomic tags as well as container tags will have attributes associated with the tags to specify specific properties. To associate the attributes in the LXML format, attributes are given to corresponding level numbers that designate a tag. Fig.5. illustrates the attributes given to both types of tags- atomic and container tags.

```
<products>
        <product series='A'>
                <name  grade='A1'>television</name>
                <manufacturer>sonis corp</manufacturer>
                <price>1000</price>
        </product>
        <product  series='B'>
                <name>refrigarator</name>
                <manufacturer>abc corp</manufacturer>
                <price>1500</price>
        </product>
</products>
```

(a) XML Format

```
Products
<00  series='A'>
    <01 grade='A1'>television
    <01>sonis corp
    <01>1000
<00   series='B'>
     <01 grade='A2'>refrigerator
     <01>abc corp
     <01>1500
```

(b) LXML format

Fig.5. XML document with attributes and its corresponding LXML format

5.2. Schema Awareness and Extensibility

Adherence to schema is helpful in structuring and organizing a document. It is also helpful in processing and extracting data contained in the document. LXML uses a reference XML for extracting and processing the data contained in the document. The reference XML is defined for each LXML document that defines the mapping rules associated with it. The reference XML contains the exact tag names as given in the corresponding LXML document but the data and nested tags can be removed. The reference XML acts as the schema for LXML. It needs to be defined once; any number of LXML records could be added as per the schema.

```
<invoice>
    <manufacturer>
        <name>ABC Corp</name>
        <brand>series A</brand>
        <address>St New Colony</ address >
    </manufacturer>
    <products>
        <product>
            <name>A01</name>
            <price>1400</price>
            <units>15</units>
        </product>
        <product>
            <name>A02</name>
            <price>1450</price>
            <units>10</units>
        </product>
        <product>
            <name>A0n</name>
            <price>1550</price>
            <units>100</units>
```

```
invoice
<00>
<01>ABC Corp
<01>series A
<01>St , New Colony
<00>
<01>
    <02>A01
    <02>1400
    <02>15
<01>
    <02>A02
    <02>1450
    <02>10
<01>
    <02>A0n
    <02>1550
    <02>100
```

LXML format

```
<invoice>
    <manufacturer>
        <name></name>
        <brand></brand
        <address></address>
    </manufacturer>
    <products>
        <product>
            <name></name>
            <price></price>
            <units></units>
        </product>
    <products>
</invoice>
```

Reference XML

Fig.6. An XML document, its corresponding LXML format and reference XML

An invoice XML document with a large number of product details and corresponding LXML format and reference XML is shown in Fig.6. The reference XML thus defined is independent of the size or number of records available in the LXML document and remains unchanged.

LXML strictly adheres to its schema on document structure. Also LXML does not require any additional tags to store schema related data. In LXML, the validity and structure is compared with the mapping document, which is already defined for a document; any mismatch can be easily traced and isolated.

In LXML, all data is defaulted to string and other types should be specified as attributes to the corresponding tags while defining the reference XML. This is a drawback of LXML schema - the inability to specify name spaces and data types for each tag.

6. LXML Document Handling

An LXML document can be generated easily from the XML document or can be independently created according to the user's needs. An LXML document generated from an XML is expected to be easier, less verbose and less error-prone compared to XML. The steps in generating and processing a LXML is represented in the Fig.7.

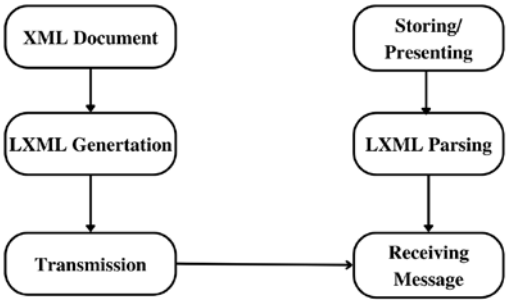


Fig.7. Steps in Document Handling

6.1. LXML Generation

LXML document is generated from the XML document. It exploits the hierarchical representation of data in an XML to avoid the excessive use of tags (Fig.8).

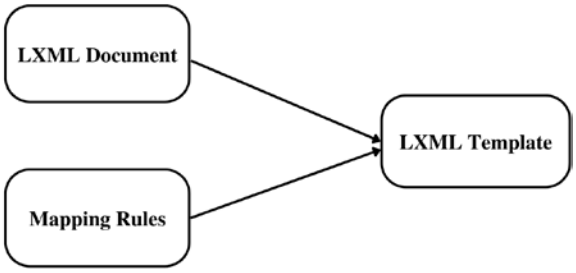


Fig.8. Generating LXML document

6.2. LXML Parsing

XML parsing is the process of extracting information from an XML document. There are different XML parsers available such as DOM parser and SAX parser. Each of them scans through the document, breaking into smaller and smaller chunks to fetch the data. Even though data representation using XML is simple, the processing of XML documents to extract the data is very expensive in terms of processor time and memory requirements [35].

Parsing an XML document and converting it into objects is a difficult and time consuming operation due to the strict inherent structure restrictions and validation techniques [22]. Also the DOM libraries used for parsing and processing the XML document consumes much memory. About 99% of this processing overhead is incurred while parsing the XML document [36-38]. The document size and the use of more attributes to XML tags contribute in reducing the parsing performance. In addition to this, formatting data as required in XML schema can also consume memory. LXML parsing can be performed in two steps: (i) Creating data objects (DO), and (ii) mapping.

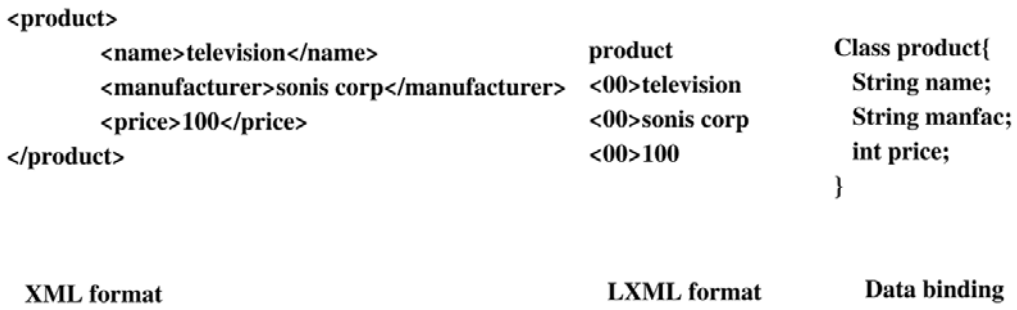


Fig.9. The XML, LXML and the corresponding DO binding class

Data Object (DO) is a class or a user defined data type with a number of data members corresponding to the number of distinct tags in the XML document. There should be a one to one relationship between tags and the members

in the DO for extraction. In addition to the data members, the DO contains setter and getter methods for each data member. A sample XML document with its LXML alternative and data binding class is shown in Fig.9 and Fig.11.

An LXML document may correspond to multiple DOs. A single LXML document with multiple DOs and binding class is depicted in Fig.11.

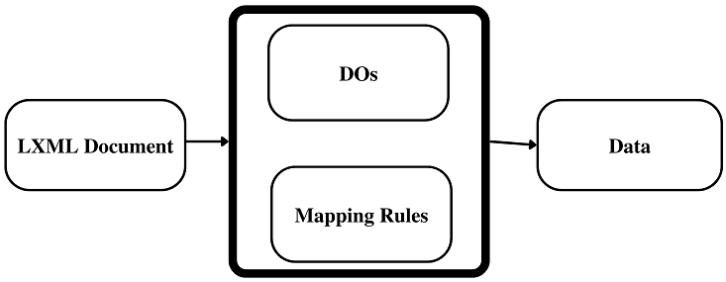


Fig.10. Extracting data from LXML document

Mapping or binding is the process of assigning each LXML value to a data member in the DO. The mapping rules given in the reference document is the basis for data extraction as depicted in Fig.10. The mapping is done based on level numbers. LXML data with same level numbers can be split and dynamically assigned to the DO easily. An LXML document with repeating groups will generate an array of DOs. The DOs later can be easily stored to a relational database or presented to the user interface as required.

```
<invoice>
    <manufacturer>
        <name>ABC Corp</name>
        <brand>series A</brand>
        <address>St New Colony</ address >
    </manufacturer>
    <products>
        <product>
            <name>A01</name>
            <price>1400</price>
            <units>15</units>
        </product>
        <product>
            <name>A02</name>
            <price>1450</price>
            <units>10</units>
        </product>
    </products>
</invoice>
```

(a) XML Format

```
Invoice
<00>
  <01>ABC Corp
  <01>series A
  <01>St , New Colony
<00>
   <01>
     <02>A01
     <02>1400
     <02>15
   <01>
     <02>A02
     <02>1450
     <02>10
```

(b) LXML format

```
Class Manufacturer{
    String mname;
    String brand;
    String address;
}

Class Product{
    String pname;
    float price;
    int units;
}
```

(c) Data Binding

Fig.11. The XML, LXML and the corresponding DO binding class

The algorithm to parse the LXML document is given below

Input: Data in XML form
Output: LXML String
1. Set level number l as 0
2. If the document has a non-empty root node
Add node to the resultant LXML
3. For each child node T in the XML document, do
3.2. If T is a simple node
Extract the data from the node and add to the LXML as <level No> data
3.3 If T is a container node
Increase the level number
Repeat step 3 for each child of T
Decrease level number
4. Stop

7. Performance Evaluation

The primary necessity for an improved messaging format is to increase the efficiency of data transmission. This is very critical in resource constraint networks like mobile messaging environments [39]. The performance of the proposed LXML method is evaluated considering the following performance parameters: verbosity, contentdensity,parsing overhead, serialization &deserialization time, marshalling &un-marshalling time, and transmission time.

Verbosity is a factor of document size. As the verbosity of a document increases, transmission overhead also increases. So a less verbose format is more suitable for data exchange. Content density is the ratio of data contained against the total size of the document. A format with content density 1 is considered as more efficient. Adding more meta-data like information pushes the content density towards 0.

Processing speed is more important in wireless environment as mobile devices have less processing power compared to static devices [40,41]. So the amount of time required for process the document should be minimal. Serialising and de-serialising of objects is performed before and after transmission. A minimal serialization and de-serialising time is considered as apt in resource constraint environment. Similarly, a minimal time requirement for marshalling and un-marshalling the document is the expected behaviour.

Transmission time is the time required for transporting the data. Transmission time is critical as it depends on the amount data, bandwidth and the network infrastructure. The transmission time should be kept minimal and to reduce the time, amount of data should be kept minimal as bandwidth and infrastructure is out of hand.

For the ease of evaluation, documents are classified into four categories depending on the size as shown in Table 1, based on the discussion in [36].

Table 1. Data Sets for Experiments

	File Size	**No of Records**
Very Small	In KBs	1-20
Small	Less than 0.5 MB	100-1000
Medium	0.5 MB to 1 MB	1000-20000
Large	Above 1 MB	100000-150000

The sample file sizes opted in each category are as given in Table 2.

Table 2. Sample file sizes for performance evaluation

Format	**Very Small (Size in KB)**	**Small (KB)**	**Medium (KB)**	**Large (KB)**
XML	4.57	376	603	2190
JSON	3.90	314	689	2307
LXML	2.52	209	309	1342

7.1. Experiment Setup

The environment to test the proposed format is configured using a laptop and a mobile phone. The laptop has a core i3 processor with 4GB RAM and Microsoft Windows 10 as operating system. The mobile phone has Android version 9 as the operating system. The laptop has JDK version 1.8 installed. JABX toolkit and Jackson API [42] are used for marshalling and un-marshalling XML and JSON objects. SoapUI tool is used for simulating servers. The experiment is carried out with sample files of varying sizes. The data set is chosen with at most care and each experiment is repeated at least three attempts and average value is taken to ensure the accuracy and the reliability of the simulation.

7.2. Verbosity

The verbosity of prominent data exchange formats such as XML and JSON are compared against LXML. Medium, small and large types of data sets, as mentioned in table 1, are prepared in XML, JSON and LXML formats and their verbosities are compared (Refer Fig.12, Fig.13 and Fig.14, respectively). Here the size of the document is taken in Y axis and the messaging formats are plotted in X axis.

Irrespective of the document category, the size is very less for LXML and hence it can be inferred that LXML has the least verbosity compared to XML and JSON for all the data sets.

Since schema defines the document and is important for processing, the schema size also matters. The schema size is independent of the number of data objects included in the document. Verbosity of XML Schema and LXML Schema is also compared for the sample provided in Fig.6 (Table 3). It can be seen that LXML Schema is less verbose than XML Schema.

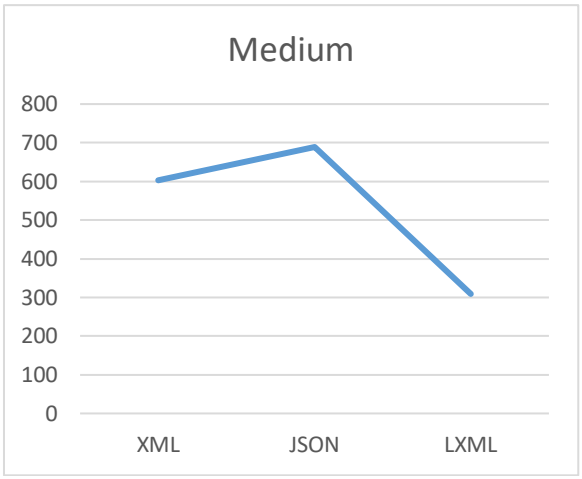


Fig.12. Verbosity comparison for medium sized documents

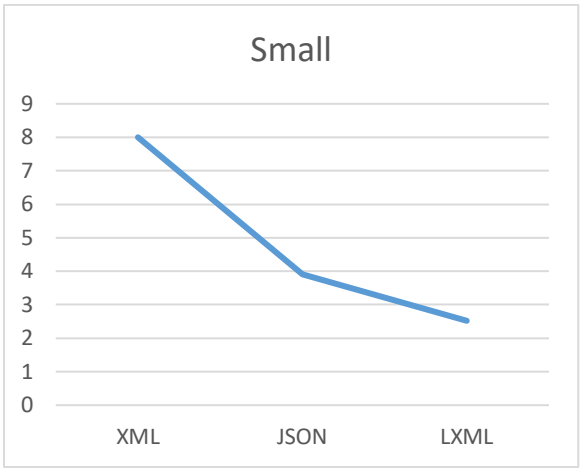


Fig.13. Verbosity comparison for small sized documents

Fig.14. Verbosity comparison for large sized documents

Table 3. Schema Size: XML v/s LXML

XML Schema Size	1.34 KB
LXML Schema Size	246 bytes

7.3. Content Density

Content density is the ratio of total amount of data available in the document to the total size of the document [43,44]. The amount of data includes the data available inside the tags and attributes. Any information apart from data constitute the residues necessary to facilitate the transmission.

Table 4. Comparison of Content Density for XML, JSON and LXML

Data Set	XML	JSON	LXML
Very small	0.406	0.521	0.814
Small	0.423	0.534	0.815
Medium	0.337	0.438	0.767
Large	0.464	0.585	0.859

Content density values of XML, JSON and LXML formats for all categories of data sets are provided in Table 4. Graphical representation of this table is as provided in Fig. 15. Here content density is plotted in Y-axis. Content density ranges from 0 to 1. A format is considered to be efficient when its content density is equal to 1. From the graph it is clear that the content density values LXML is close to 1 for all the data sets. Hence it can be inferred that LXML is the most compact messaging format among these formats (Fig.15).

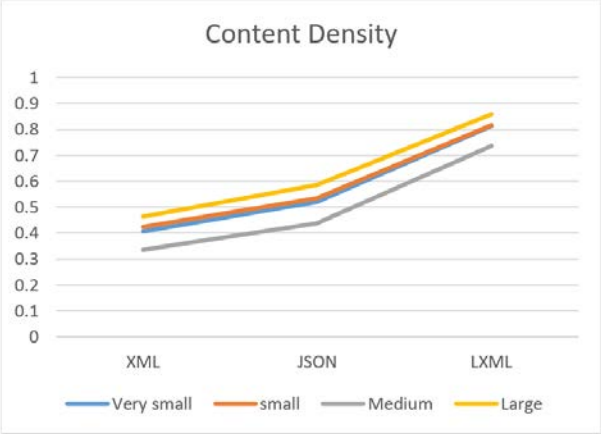


Fig.15. Data exchange formats vs content density

7.4. Parsing Overhead

Parsing is the process of reading a document and extracting its content. Performance of parsing the XML and JSON depends on the technology used. Parsing an XML document using JavaScript is more efficient than JSON; but in the case of querying, JSON has an edge over XML [45]. To parse XML documents, DOM parser is used and Java API JsonParser[46] is used to parse JSON. A standalone Java program is prepared to parse LXML. The average time consumed (in ms) in parsing XML, JSON and LXML documents consisting of different record count is as shown in Table 5.

Table 5. Parsing overhead for XML, JSON and LXML

No of Records	Average time consumed (in ms)		
	XML	JSON	LXML
1000	129	90	79
5000	534	326	261
10000	1320	779	731

Here Y axis shows the parsing time required in ms. It can be inferred that LXML parsing is very simple when compared with parsing of XML and JSON (Fig.16). This is due to the fact that unlike XML, LXML does not have processing overhead in terms of processor time and memory.

7.5. Serialization and Deserialization Time

Serialization and deserialization are important operations performed on data exchange formats such as XML and JSON. Serialization converts an object to a serial data, say binary form that can be readily transmitted [3,47]. The reverse process is termed as deserialization. The executional speed in serializing and de-serializing is an important performance criteria in data interchange formats[48]. This performance evaluation is carried out using Xstream [49] and JSON libraries [50] for XML and JSON respectively. The time required to serialize the objects at the serving side is positively correlated to the number of objects [23].

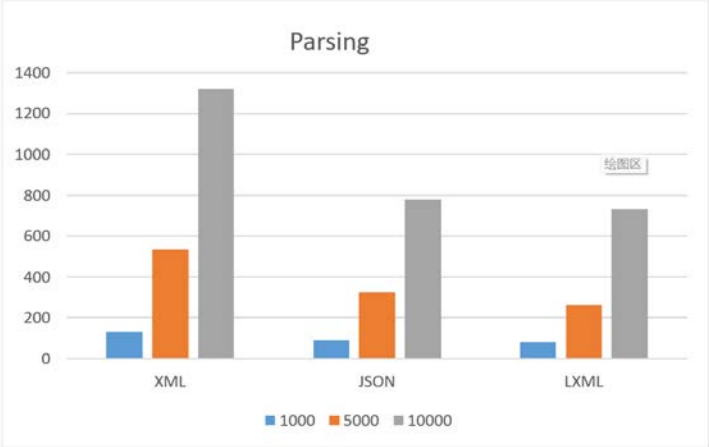


Fig.16. Data Exchange Formats v/s Parsing Overhead

Table 6. Serialization and deserialization time (in ms) for XML, JSON and LXML

Type of Processing	Format	Average Time Elapsed (ms)		
		Small	Medium	Large
Serialization (average time consumed in ms)	XML	2.860	3465	7214
	JSON	3.146	5356	10812
	LXML	2.421	2954	5824
De-serialization (average time consumed in ms)	XML	5.320	4654	9405
	JSON	4.405	9476	19423
	LXML	4.224	4320	8640

The average serialization and deserialization time consumed (in ms) for XML, JSON and LXML documents consist of small, medium, and large datasets, respectively, are as shown in Table 6. It can be observed that the proposed format has an advantage over XML and JSON in terms of time required for serializing and de-serialising for documents in the small category (Fig.17). As the serialization and deserialization process directly impact the transmission of the document, clearly LXML has an edge.

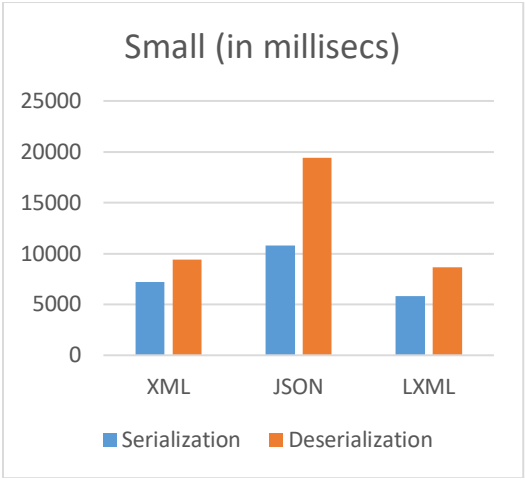


Fig.17. Serialization and deserialization time (in ms) for XML, JSON and LXML - small dataset

In medium and large files, LXML has only a narrow edge over XML in deserialization (Fig.18 and Fig.19). But it has a clear performance difference when compared with JSON.

7.6. Marshalling and Un-marshalling Time

Similar to JSON and XML, LXML relies on data objects (DO) while processing the documents. Marshalling and un-marshalling DOs are necessary while processing and transmitting documents. Marshalling time is the time consumed to convert an object to a stream of data and Un-marshalling time is the time taken for converting a stream of data back

to an object is [19]. Time taken for marshalling and Un-marshalling of XML, JSON and XML documents consisting of small datasets is as shown in Table 7.

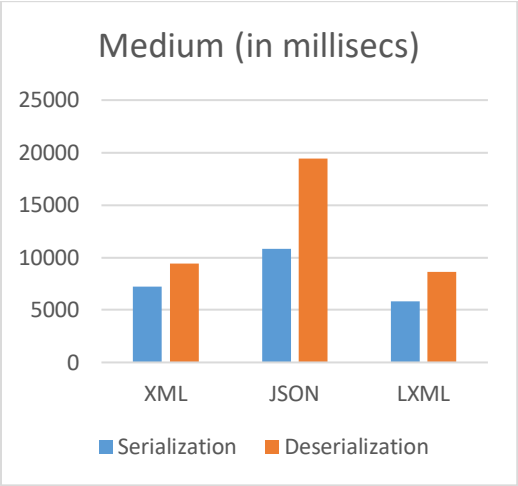


Fig.18. Serialization and deserialization time (in ms) for XML, JSON and LXML - medium dataset

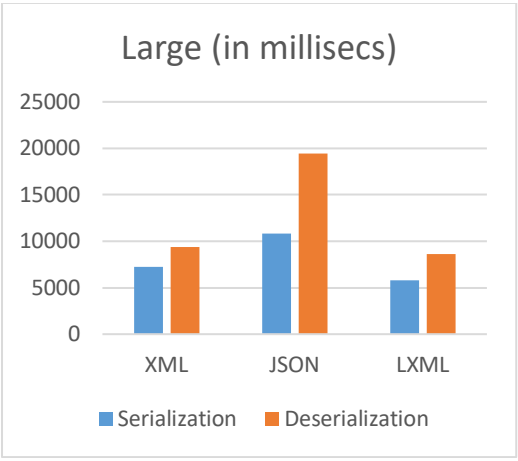


Fig.19. Serialization and deserialization time (in ms) for XML, JSON and LXML - large dataset

Table 7. Marshalling and un-marshalling footprints

	XML	JSON	LXML
File Size in KB (small)	8	3.9	2.52
Memory in KB (marshal)	553	91.57	73.24
Memory in KB (Unmarshal)	143	121.38	96.34
Execution Time in millisec (marshal)	0.024	0.0108	0.0102
Execution Time in millisec (Unmarshal)	0.044	0.0141	0.0143

Marshalling and un-marshalling is purely dependent on the document, its number of tags, number of nodes, type of tags, and size of the document. It can be observed that the memory footprints and execution time for marshalling and un-marshalling of LXML is close to that of JSON and far better than XML (Fig.20).

7.7. Transmission Time

Transmission time depends on the size of the document. Transmission time in WSN can be calculated using the formula [51].

$$\text{Transmission Time} = \frac{\text{Size}}{\text{K}} + \frac{\text{D}}{\text{C}} \quad (1)$$

Where, K is calculated using the formula,

$$K = \frac{2*\text{Size}}{\text{Bit Rate}(R)} \quad (2)$$

Where, Size is the packet size in bits, R is Bit Rate (in bps) and D is the distance between nodes in meter, C is the velocity of light for wireless communication (m/s).

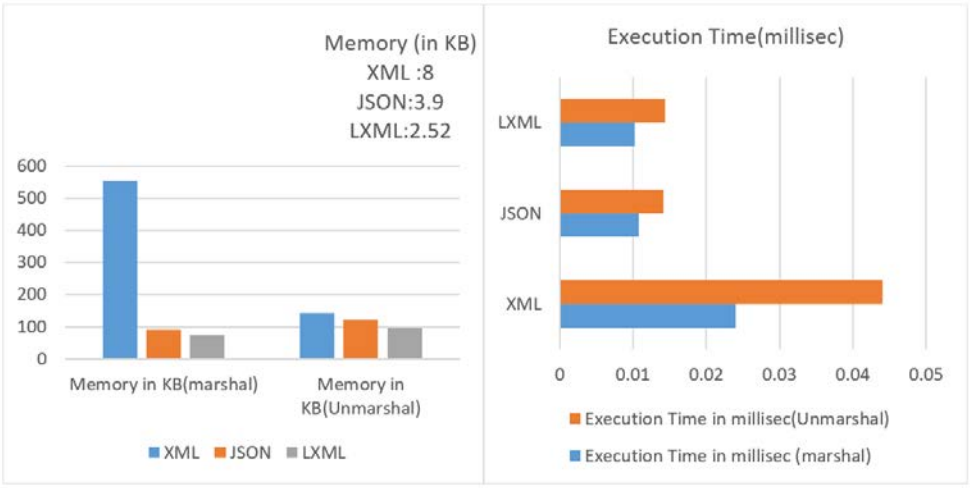


Fig.20. Marshalling and Un-marshalling time (small dataset) - XML, JSON and LXML

The transmission time for various messaging formats is calculated by simulating a network with two nodes 100 KMs apart. Data of different sizes is being sent assuming the speed of signal as 1, 00,000 KM per sec. The transmission time is tabulated assuming there is no propagation delay and the rate of data transfer is 1Mbps.

Time for transmission of XML, JSON and XML documents using datasets of small and medium categories, respectively, are as shown in Table 8.

Table 8. Transmission time (ms) for XML, JSON and LXML

Format	No. of Records	Transmission time(ms)
XML	1000 (Small)	0.223
	5000 (Medium)	0.837
	10000 (Medium	1.690
JSON	1000 (Small)	0.128
	5000 (Medium)	0.492
	10000 (Medium	1.231
LXML	1000 (Small)	0.109
	5000 (Medium)	0.32
	10000 (Medium)	0.764

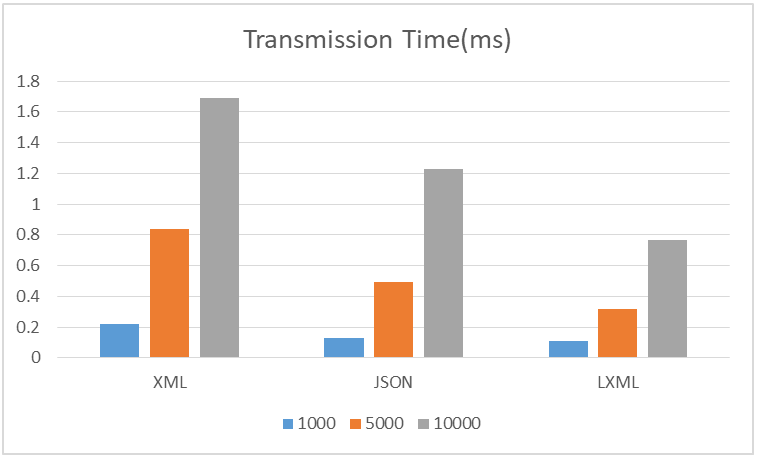


Fig.21. Graph representing transmission times

It can be inferred that LXML has a performance advantage over XML and JSON in terms of transmission time (Fig.21).

8. Result and Discussion

Despite their advantages, the widely used data exchange formats such as XML and JSON have several performance overheads in transmission and processing, especially in resource constraint networks. These overheads accelerate the energy consumption and drains the mobile battery as well.

In this work, a messaging proposed called LXML is proposed and the proposed messaging format is compared with the prominent data exchange formats XML and JSON using six parameters. Among these six performance parameters, document size, content density and processing time is more critical in a wireless mobile environment.

The LXML format drastically reduces the amount of data to be communicated. There is a reduction of 40 to 48% in document size when compared with XML and 30-40% with JSON. The LXML schema is 4.6 times less verbose than XML schema. The content density of LXML averages to 80-85% of the document size. This will definitely improve the transmission overhead. At the same time the processing LXML never incur any additional overhead.

Experiments reveal that the LXML format has a clear edge over the other formats; hence it can be inferred that LXML is most suited for resource constrained wireless environments. It is natural that implementing the LXML format may require development of new APIs; but it can be achieved by making minor changes in current XML APIs.

9. Conclusions

In this paper, an alternative data exchange scheme known as LXML is proposed to address the limitation of existing data exchange formats. The proposed LXML format represents data in a hierarchical manner and is derived from XML format. It possesses all the qualities of XML; but it is less verbose than XML. Performance of the proposed format is evaluated and compared with widely used existing messaging formats such as XML and JSON for six parameters such as document size, content density, parsing overhead, serialization and deserialization time, marshalling & un-marshalling time, and transmission time. It is found that LXML is performing much better than XML and JSON. The huge difference in the document size and density make this model the most appropriate one in a wireless environment. It reduces the amount of non-data that are forced to transmit along with documents each time. Hence it can be concluded that the proposed LXML format can address the verbosity related problems faced by formats such as XML and JSON in resource constraint networks.

A less verbose alternative for XML and JSON is a need of time as many conventional business models are shifted to mobile platforms. Such a paradigm shift without change in technology will not attain expected outcome. The LXML format integrated with the transaction models can improve the data transmission and processing efficiency. The suitability of this format for inter app communications, communication in distributed applications, client server communication and cross languages data sharing that includes heterogeneous platforms and devices is to be experimentally tested and proved. The application of LXML format with respect to configuration files and log files is yet to be identified. These are directions for future work.

References

[1] IBM, “https://www.ibm.com/topics/mobile-workforce” https://www.ibm.com/topics/mobile-workforce, accessed on March 20

[2] Philip A. Bernstein and Eric Newcomer, Principles of Transaction Processing-A volume in The Morgan Kaufmann Series in Data Management Systems, Book • Second Edition • 2009

[3] G. Wang, "Improving Data Transmission in Web Applications via the Translation between XML and JSON," 2011 Third International Conference on Communications and Mobile Computing, Qingdao, 2011, pp. 182-185, doi: “10.1109/CMC.2011.25”.

[4] G. H. Forman and J. Zahorjan, "The Challenges of Mobile Computing," IEEE Computer, 27, 1994, pp. 38-47.

[5] A. Holzinger, A. Nischelwitzer, and M. Meisenberger, “Mobile phones as a challenge for mlearning: examples for mobile interactive learning objects (MILOs)”, in Proceedings of the 3rd IEEE International Conference on Pervasive Computing and Communications Workshops, 2005, pp. 307- 311.

[6] Nipan Maniar, Emily Bennett, Steve Hand and George Allan, the effect of mobile phone screen size on video based learning JOURNAL OF SOFTWARE, VOL. 3, NO. 4, APRIL 2008.

[7] Boonkrong, Sirapat & Pham Cao, Dinh. (2013). The Comparison of Impacts to Android Phone Battery between Polling Data and Pushing Data. 10.13140/2.1.4731.7447.

[8] Kangasharju, Jaakko. (2007). Efficient Implementation of XML Security for Mobile Devices. 134-141, doi: “10.1109/ICWS.2007.81”.

[9] Lee, Hsun-Ming and Mehta, Mayur R. (2013) "Defense Against REST-based Web Service Attacks for Enterprise Systems," Communications of the IIMA: Vol. 13: Iss. 1, Article 5. Available at: http://scholarworks.lib.csusb.edu/ciima/vol13/iss1/5

[10] Kuyoro, Shade & Ibikunle, Oluwasegun & Awodele, Frank & Okolie, Samuel. (2012). Security Issues in Web Services. International Journal of Computer Science and Network Security. 12.

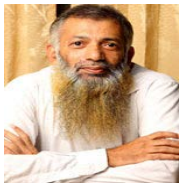
[11] “Mobile devices have overtaken the human population” retrieved from https://www.newstalk.com/news/mobile-devices-have-overtaken-the-human-population-499358, (accessed on Aug 2018)

[12] "DIGITAL AROUND THE WORLD" retrieved from https://datareportal.com/global-digital-overview(accessed on March 2021).
[13] Deyan Georgiev, "Smartphone Statistics", https://review42.com/resources/smartphone-statistics/ June 2021
[14] Aleksandar S. "What Percentage of Internet Traffic Is Mobile in 2021?", https://techjury.net/blog/what-percentage-of-internet-traffic-is-mobile, November 2021.
[15] Insider Intelligence "Mobile data will skyrocket 700% by 2021", https://www.businessinsider.com/mobile-data-will-skyrocket-700-by-2021-2017-2?IR=T, Feb 9, 2017.
[16] G. Kaur and M. Fuad. An evaluation of protocol buffer. In Proceedings of the IEEE Southeast Con 2010 (SoutheastCon), pages 459–462, 2010.
[17] Diao Y, Franklin M (2003) Query processing for high-volume XML message brokering. In: Proceedings of the 28th international conference on very large database, pp 261–272
[18] Suciu D. (2000) Semistructured Data and XML. In: Tanaka K., Ghandeharizadeh S., Kambayashi Y. (eds) Information Organization and Databases. The Springer International Series in Engineering and Computer Science, vol 579. Springer, Boston, MA, doi: "10.1007/978-1-4615-1379-7_2"
[19] Saurabh Zunke, Veronica D'Souza, JSON vs XML: A Comparative Performance Analysis of Data Exchange Formats, International Journal of Computer Science and Network, Volume 3, Issue 4, August 2014
[20] P. P. Abdul Haleem, M. P. Sebastian, An energy-conserving approach for data formatting and trusted document exchange in resource-constrained networks. Knowl. Inf. Syst. 32(3): 559-587 (2012).
[21] Tamayo, Alain & Granell, Carlos & Huerta, Joaquín. (2011). Dealing with large schema sets in mobile SOS-based applications. 10.1145/1999320.1999336.
[22] Nicola, Matthias & John, Jasmi. (2003). XML parsing: A threat to database performance. International Conference on Information and Knowledge Management, Proceedings. 175-178. 10.1145/956863.956898.
[23] Muzafer H. Saracevic, Emrus Azizovic, Munir Sabanovic "Comparative analysis of AMF, JSON and XML technologies for data transfer between the server and the client" retrieved from https://www.researchgate.net/publication/311750440, Oct 2016.
[24] Jorstad, Ivar & Bakken, Elias & Johansen, Tor Anders. (2008). Performance Evaluation of JSON and XML for Data Exchange in Mobile Services. 237-240.
[25] P P Abdul Haleem and M P Sebastian (2014), An alternative approach for XML messaging, International Journal of Advanced Research ISSN NO 2320-5407, Volume 2, Issue 1, 251-294
[26] Ben Kiki O. YAML Ain't Markup Language (YAML) Version 1.2. Available athttps://yaml.org/spec/1.2/spec.html, (Accessed March 2021).
[27] P. Greenfield, M. Droettboom, E. Bray ASDF: A new data format for astronomy, Astronomy and Computing Volume 12, September 2015, Pages 240-251, doi: "10.1016/j.ascom.2015.06.004".
[28] K. Maeda. Comparative survey of object serialization techniques and the programming support. Journal of Communication and Computer, 9:920 – 928, 2012.
[29] Huang, Xu & Ridgewell, Alexander & Sharma, Dharmendra. (2006). A Dynamic Threshold Technique for XML Data Transmission on Networks. 1163-1167, doi: "10.1007/11893011_147".
[30] Jonathan Freeman, "What is JSON? A better format for data exchange", https://www.infoworld.com/article/3222851/what-is-json-a-better-format-for-data-exchange.html, OCT 25, 2019
[31] S. Ghandeharizadeh, C. Papadopoulos, M. Cai, and K. K. Chintalapudi, Performance of Networked XML-Driven Cooperative Applications", In Proceedings of the Second Inter-national Workshop on Cooperative Internet Computing Hong Kong, China, August 2002.
[32] Alexander Ridgewell, Xu Huang, and Dharmendra Sharma, "Evaluating the Size of the SOAP for Integration in B2B", the Ninth International Conference on Knowledge-Based Intelligent Information & Engineering Systems Melbourne, Australia, September, 2005. Part IV, pp.29.
[33] Yajuan Yao, Lijuan Shi, "Study of Index Mechanism for GML Data", International Journal of Education and Management Engineering, vol.2, no.2, pp.13-20, 2012.
[34] K. Naresh kumar, Ch. Satyanand Reddy, N.V.E.S. Murthy, "Fuzzy-Based XML Knowledge Retrieval Methods in Edaphology", International Journal of Intelligent Systems and Applications, Vol.8, No.5, pp.55-64, 2016.
[35] Y.S. Alone, V.M. Deshmukh, an overview of XML Parsing using Prefetching algorithm, International Journal of Computer Science and Applications, Vol. 6, No.2, Apr. 2013, ISSN: 0974-1011.
[36] Nicola, Matthias & John, Jasmi. (2003). XML parsing: A threat to database performance. International Conference on Information and Knowledge Management, Proceedings. 175-178. 10.1145/956863.956898.
[37] Abdeslem DENNAI, Sidi Mohammed BENSLIMANE, "Semantic Indexing of Web Documents Based on Domain Ontology", International Journal of Information Technology and Computer Science, vol.7, no.2, pp.1- 11, 2015.
[38] Yiqun Chen, Jinyin Cao, "TakeXIR: a Type-Ahead Keyword Search Xml Information Retrieval System", International Journal of Education and Management Engineering, vol.2, no.8, pp.1-5, 2012.
[39] K. Maeda, "Performance evaluation of object serialization libraries in XML, JSON and binary formats," 2012 Second International Conference on Diital Information and Communication Technology and it's Applications (DICTAP), Bangkok, 2012, pp. 177-182, doi: "10.1109/DICTAP.2012.6215346".
[40] Balasubramanian, N., Balasubramanian, A. and Venkataramani, A. (2009). Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications. Department of Computer Science, University of Massachusetts Amherst.
[41] Mobile Computing Available at https://www.ibm.com/topics/mobile-technology (accessed on May 2020)
[42] Jackson JSON Processor Available at http://jackson.codehaus.org/ (accessed on May 2020).
[43] White G, Kangasharju J, Brutzman D, Williams S (2007) Efficient XML interchange measurements note. Available at http://www.w3.org/TR/exi-measurements// (accessed on May 2020).
[44] Suchit A. Sapate," Effective XML Compressor: XMill with LZMA Data Compression", International Journal of Education and

Management Engineering, Vol.9, No.4, pp.1-10, 2019.
[45] Jan Stenberg "XML Can Give the Same Performance as JSON", https://www.infoq.com/news/2013/08/xml-json-performance/, accessed on March 2021.
[46] JSON, Available at http://www.json.org/ (accessed on May 2020).
[47] JSON Serialization Usage, Available at http://flexjson.sourceforge.net/ (accessed on May 2020).
[48] Mohamed Shameer M C, Abdul Haleem P, P, A study on the requirements of a transaction model in mobile environment, communicated, unpublished.
[49] Xstream, XStream, Available at http://xstream.codehaus.org/ (accessed on May 2020).
[50] google-gson-A Java library to convert JSON to Java objects and vice-versa, http://code.google.com/p/google-gson/ (accessed on May 2020).
[51] Esmaeili, Sobhan. "Re: How can I find a transmission time formula for WSN?" Retrieved from: https://www.researchgate.net/post/How_can_I_find_a_transmission_time_formula_for_WSN/5e3c8d933d48b73dc26ac563/citation/download (accessed on May 2021).21.

Authors' Profiles

M. C. Mohammed Shameer, Assistant Professor in Postgraduate Department of Computer Science, Farook College (Autonomous) Kozhikode, Kerala, India. He received his Master's degree in Computer Applications (MCA) from University of Kerala (2007) and worked in various IT organizations including IBM prior to joining Farook College. Published 5 papers in several national and international conferences and journals and authored one book. The broad area of my research is mobile computing.



P. P. Abdul Haleem, received his B E (Computer Science and Engineering) degree from Bangalore University (1994), and M Tech (Computer Science and Engineering) and PhD degrees from National Institute of Technology Calicut, India in 2005 and 2011, respectively. He has two decades of experience in academics and software development. Currently he is working as Assistant Professor at Postgraduate Department of Computer Science & Research Centre, Farook College (Autonomous), Kozhikode, India. He has published papers in several national and international conferences and journals. His research interests include Mobile Computing and Data Formatting Schemes.

Yazik K. Puthenpediyakkal, is currently working as senior software engineer with Zalando SE, Berlin, Germany. He has over 15 years of experience as Technical Architect and other leadership roles in agile software development. He received his Master's in Computer Applications (2007) from College of Engineering, Trivandrum, Kerala. India, and Bachelors in Information Technology (2004) from University of Calicut. His areas of interest include networking, cloud computing, services and security.

How to cite this paper: M. C. Mohammed Shameer, P. P. Abdul Haleem, Yazik K. Puthenpediyakkal, "A lightweight Data Exchange Format for Mobile Transactions", International Journal of Computer Network and Information Security(IJCNIS), Vol.15, No.3, pp.47-64, 2023. DOI:10.5815/ijcnis.2023.03.04