

Application  
Internet-Draft  
Intended status: Informational  
Expires: April 15, 2013

Y. Doi  
TOSHIBA Corporation  
October 12, 2012

**EXI Messaging Requirements**  
**draft-doi-exi-messaging-requirements-00**

Abstract

EXI (Efficient XML Interchange) is a specification on efficient encoding of XML. EXI is useful if an application requires XML based message exchange but no sufficient resource is available. However, schema-informed mode of EXI needs some out-of-band coordination between communicating nodes. This document discusses requirement on use of schema-informed EXI as a message exchange format on the Internet systems.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 15, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4](#).e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

<a href="#">1.</a>	<a href="#">Introduction</a>	<a href="#">3</a>
<a href="#">2.</a>	<a href="#">Schema Negotiation for Schema-Informed EXI Messaging</a>	<a href="#">3</a>
<a href="#">2.1.</a>	<a href="#">Content-Type and Schema Identification</a>	<a href="#">3</a>
<a href="#">2.2.</a>	<a href="#">Client-Driven Schema Negotiation</a>	<a href="#">4</a>
<a href="#">2.3.</a>	<a href="#">Server-Driven Schema Negotiation</a>	<a href="#">4</a>
<a href="#">2.4.</a>	<a href="#">Publisher-Driven Schema Negotiation</a>	<a href="#">4</a>
<a href="#">3.</a>	<a href="#">Security Considerations</a>	<a href="#">4</a>
<a href="#">4.</a>	<a href="#">IANA Considerations</a>	<a href="#">5</a>
<a href="#">5.</a>	<a href="#">Normative References</a>	<a href="#">5</a>
	<a href="#">Author's Address</a>	<a href="#">5</a>



## **1. Introduction**

EXI[W3C.REC-exi-20110310] (Efficient XML Interchange) is a specification on efficient encoding of XML. EXI is useful if an application requires XML based message exchange but no sufficient resource is available, such as environments discussed in [\[I-D.shelby-core-coap-req\]](#). However, EXI needs some out-of-band coordination between communicating nodes.

The target of this document is not to discuss EXI spec itself. This document discusses how to use it as a message exchange format (a presentation layer) on the Internet systems to support development and deployment of EXI systems in the Internet including constrained nodes.

## **2. Schema Negotiation for Schema-Informed EXI Messaging**

In short, EXI has two grammar modes: Schema-informed and Built-in.

Because schema-informed grammar can make smallest messages in most cases, some applications may want to make use of schema-informed grammar as its message format.

To decode an EXI message, the sender and receiver should have exactly same schema. However, the way to negotiate schema used in a communication is not yet well defined.

To use EXI as application message encodings, clients and servers should have a way to coordinate the schema used in the communication. This is similar to content negotiation defined in HTTP[RFC2616]. This section describes schema negotiation cases based on common communication pattern.

### **2.1. Content-Type and Schema Identification**

To negotiate schema, an application must have a way to identify a schema.

A content-type may use schema-informed EXI as its encoding. Each content-type should define how to identify a schema used in a communication. The identifier (schemaId) may have internal structure to indicate backward compatibility.

A good practice is to have schema version number (Major.Minor) as a schema ID. Between minor modifications, schemas should have backward compatibility (a node with schema 4.3 shall have schema 4.0, 4.1 and 4.2). Between major modifications, schema should not have it (a node



with schema 4.3 may not have schema 1.x, 2.x and 3.x). Note that schemaId is local identifier space that belongs to a content-type. There is no need to have global schema ID registry.

On schema negotiation, a receiver of a message declares a set of acceptable schema IDs and a sender selects a schema ID among the given set. The selected schema ID should be in schemaId field of EXI option header.

## **2.2. Client-Driven Schema Negotiation**

Client-driven schema negotiation is the way that a client decides actual schema version used in a communication. This happens in POST or PUT style communications. In [\[RFC2616\]](#), try-and-redirect style of client-driven content negotiation is described. Similar way should be possible in schema negotiation. However, it may be simpler to have a way to declare a server's acceptable schema set.

As an alternative, a server (or a resource on a server) may declare its available schema set via some service discovery mechanisms. Candidates are such as DNS-SD[I-D.cheshire-dnsext-dns-sd] TXT resource records or media type in link format[I-D.nottingham-http-link-header][] that represents a resource. If an application can assume a client does service discovery before using the service, it may assume the client knows server's schema set.

## **2.3. Server-Driven Schema Negotiation**

Server-driven schema negotiation is the way that a server decides actual schema version used in a communication. In HTTP, schema negotiation in GET requests should do server-driven negotiation. In [\[RFC2616\]](#), Accept: header is defined to make server-driven content negotiation. Schema negotiation can be piggybacked on it by using some content type parameter to carry acceptable schema ID set.

## **2.4. Publisher-Driven Schema Negotiation**

CoAP[I-D.ietf-core-coap] and some other protocols may have publish-subscribe (observer) pattern in communication. In this case, a subscriber should give its acceptable schema ID set to a publisher as it registers its subscription request.

## **3. Security Considerations**

No particular security concern is raised by this document. Applications should be able to detect malformed input as usual.



#### **4. IANA Considerations**

This document has no actions for IANA.

#### **5. Normative References**

- [I-D.cheshire-dnsext-dns-sd]  
Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", [draft-cheshire-dnsext-dns-sd-11](#) (work in progress), December 2011.
- [I-D.ietf-core-coap]  
Shelby, Z., Hartke, K., Bormann, C., and B. Frank, "Constrained Application Protocol (CoAP)", [draft-ietf-core-coap-11](#) (work in progress), July 2012.
- [I-D.nottingham-http-link-header]  
Nottingham, M., "Web Linking", [draft-nottingham-http-link-header-10](#) (work in progress), May 2010.
- [I-D.shelby-core-coap-req]  
Shelby, Z., Stuber, M., Sturek, D., Frank, B., and R. Kelsey, "CoAP Requirements and Features", [draft-shelby-core-coap-req-02](#) (work in progress), October 2010.
- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", [RFC 2616](#), June 1999.
- [W3C.REC-exi-20110310]  
Kamiya, T. and J. Schneider, "Efficient XML Interchange (EXI) Format 1.0", World Wide Web Consortium Recommendation REC-exi-20110310, March 2011, <<http://www.w3.org/TR/2011/REC-exi-20110310>>.





Author's Address

Yusuke Doi  
TOSHIBA Corporation  
Komukai Toshiba Cho 1  
Saiwai-Ku  
Kawasaki, Kanagawa 2128582  
JAPAN

Phone: +81-45-342-7230

Email: [yusuke.doi@toshiba.co.jp](mailto:yusuke.doi@toshiba.co.jp)

URI:

