XDI RDF Model Proposal

V5, May 21, 2007

This document is a proposal to the OASIS XDI Technical Committee for XDI to use the RDF graph model and an XRI-based RDF vocabulary called XDI RDF.

Notes on Terminology

New Term	Former Term	Stands For	Represents
XDI RDF	RDFX	XDI RDF	A proposed new data model for XDI based on RDF and XRIs
ATI	V18 or 3L	Authority/Type/Instance	The 3-level authority/type/instance data model reflected in the V17 and V18 XDI schema proposals
X3 (originally proposed by Bill Barnhill)	--	XDI RDF 3-part notation	A simple variant of N3 syntax for representing XDI RDF graphs in plain text

Motivations

The motivations for this new XDI data model are:

- *Reach full fidelity with the RDF graph model.* With the XDI RDF model, the object of a subject/predicate assertion may be either an XRI reference to another XDI subject, or a data element representing a literal value, but not both. This is the same limitation imposed by the RDF graph model, but it was not imposed by the ATI (authority/type/instance) model, which permits data nodes to be associated with subject, predicate, and object nodes.
- *Clean separation of RDF predicates and RDF objects.* In the XDI RDF model, all XDI addresses representing RDF statements about concrete data objects consist of two XRI segments (an XRI representing the RDF subject and an XRI representing the RDF predicate). All XDI addresses representing RDF statements about other XDI resources consist of three XRI segments (the third segment being an XRI identifying the target XDI subject or object). In the ATI model, the third XRI segment did not clearly correspond to an RDF object, as it could contain both an XRI and concrete data. This can lead to ambiguity about how to model different data structures.
- *Explicit representation of references and links.* In the XDI RDF model, refs and links are now expressed using XRIs, so they have their own explicitly XDI addresses.
- *Simplified human understanding.* With the XDI RDF model, 100% of the XDI graph (i.e., all XDI documents from all XDI authorities) can now be represented in a single four-column table (where for each row there is an entry in either the third or fourth column but not both). This is a simple and intuitive structure for developers and information architects to understand, and corresponds closely to RDF N3 notation.
- *Simpler and more expressive dictionary definitions.* The XDI RDF model uses four core predicate XRIs and one core object XRI to support the construction of basic XDI dictionary definitions with self-describing schemas and datatypes. Higher-level constraints (cardinality, sequence, etc.) can be added to the XDI $ Dictionary or other dictionaries using these five core XRIs.
- *Simplied link contract structure.* The XDI RDF model simplifies and generalizes the model for how any XDI authority can control access, exchange, and rights to any XDI data for which they are authoritative.

The XDI RDF Model

In the XDI RDF model, all data is both identified and described using RDF subject/predicate/object statements encoded as XRIs. Each XRI consists of three segments, each of which is itself an XRI. Note that both the second and third segment may be empty.

First segment (RDF Subject)	Second segment (RDF Predicate)	Third segment (RDF Object)	Data element	Identifies & Describes
XRI	Empty	Empty	Empty	The set of all XDI RDF statements rooted in an XDI subject.
XRI	XRI	Empty	Empty	A shorthand for defining that the predicate XRI is a member of the set of XRIs that describe the subject XRI, i.e., set/member relationships.
XRI	XRI	Empty	Non-empty	An XDI RDF statement whose object is a literal (represented as an XDI data element)
XRI	XRI	XRI	N/A	An XDI RDF statement whose object is another XDI resource

Notes:

- All simple XDI objects representing either literal values (if the data element is not empty) or set membership (if the data element is empty or absent) are addressable with a two-segment (subject/predicate) XRI.
- All complex XDI objects, to any degree of complexity, are defined using three-segment XRIs that describe the aggregation or composition relationships between the XDI subject and other XDI subjects.
- All XDI dictionary entries consist entirely of three-segment XRIs because they define only structure and not concrete data.

The Four Core XDI RDF Predicates

The XDI RDF model uses four core predicates defined in the XDI $ Dictionary to express fundamental XDI resource relationships.

XDI RDF Predicate	Relationship Type	RDF/RDFS/OWL Analogs
$is	Equivalence	No direct equivalent for XDI RDF subject equivalence. `owl:equivalentClass` for XDI RDF object equivalents.
isa	Inheritance	`owl:class` for for XDI RDF subjects. `rdf:type` for XDI RDF objects except those rooted in the XDI RDF **$type** space, which would be `rdf:parseType`
$has	Aggregation	`exns:hasXXX`, where `exns` is the specified namespace and XXX is the property name
hasa	Composition	`exns:hasxxx`, where `exns` is the specified namespace and `xxx` is the property name

The Four XRI Global Context Symbols

Beginning with XRI Syntax 3.0, all XRIs will be rooted in four global contexts corresponding to the XDI $ Dictionary entries discussed in the preceeding section. (Note: in XRI Syntax 2.0, the ! symbol was used as both a global context symbol and a local context symbol. The XRI TC has reached concensus to deprecate its use as a global context symbol beginning in XRI 3.0.)

GCS Symbol	XDI Dictionary	Description
$	$	The abstract root of all XDI RDF graph syntax (which is why $ is the root of the XRI and XDI $ Dictionaries).
=	$is	The abstract XDI RDF root subject relationship.
@	$has	The abstract XDI RDF root predicate relationship.
+	$a	The abstract XDI RDF root object relationship. (In XDI RDF, XRIs rooted in the +space represent resources that are targets of isa or hasa predicates.)

XDI RDF Addressing Rules

The structure of XRIs used to address XDI RDF documents and all elements within them are based on the four core predicates. They are summarized in the following table:

XDI RDF Predicate	Applies to XDI address segment	Addressing Rule
$has	First segment	Represented by subsegment delegation within the first segment of an XDI XRI (the subject segment).
hasa	Second segment	Represented by subsegment delegation wthin the second segment of an XDI XRI (the predicate segment).
$is	Second segment	Appears literally as an XDI RDF predicate value to assert an equivalence relationship.
isa	Second segment	Appears literally as an XDI RDF predicate value to assert an inheritance relationship.

See the sections below for extensive examples.

The Five Core XDI RDF Protocol Operations

The XDI protocol supports four core operations on the XDI RDF graph itself and one abstract operation on XDI data described by the graph. These are summarized in the following table:

XDI RDF Graph Operation	CRUD Equivalent	Description
$get	read	Read one or more rows from the XDI RDF table.
$add	create	Write one or more new rows to the XDI RDF table.
$mod	update	Change a data element value in a row of the XDI RDF table.
$del	delete	Delete one or more rows from the XDI RDF table.
$do	N/A	Perform an operation on XDI data elements whose API is defined by XDI subject rooted on this XRI.

Declaring XRIs for these explicit XDI protocol operations establishes the basis for permissioning in XDI link contracts (see examples below).

XDI RDF Tables

Because the XDI RDF model enables all XDI resources to be identified and described using either three segment XRIs or two-segment XRIs plus a literal data value, the content of XDI documents can be represented with 100% fidelity using a single four-column table. It is trivial to transform from this XDI RDF table format into a valid XDI document and vice versa.

The following sections are example XDI RDF tables. Notes about these examples:

- The XRIs in this table use the proposed XRI 3.0 direct concatenation syntax where global cross-references to XRI authorities do not require enclosing parentheses if they consist of only a single segment. See http://wiki.oasis-open.org/xri/XriCd02/XriAbnf2dot1.
- $* is a proposed XDI $ Dictionary entry for cardinality constraints on $has or hasa relationships. The absence of this predicate indicates no restraints on cardinality (i.e., zero-or-more). The presence of this predicate imposes the same cardinality constraints as currently supported in ABNF:
  - $*n = exactly n
  - $*n-n = from n to n, i.e., *1-3 = 1, 2, or 3
  - $*n-* = n or more
- / as the first character of an XRI cross-reference means the cross-reference is relative to the current XDI subject.
- Blank shaded rows are inserted between different XDI subjects only for improved readability – they have no meaning.
- XDI subjects and predicates are not repeated in the table when they are identical to the previous subject or predicate.

XDI Dictionary Definitions

The following XDI RDF table provides an example of XDI dictionary definitions, where all subjects are in the $ or + space and all predicates are all rooted on one of the four core XDI RDF predicates. Note that this example has no entries in the data column because a pure machine-readable XDI dictionary definition only defines relationships between XDI resources. However for human-readablity, a typical XDI dictionary may also include XDI predicates such as $type+text (representing that the value of the data element was of content type text). Rows using this data value would contain a human-readable text description of the dictionary entry.

Note that, if needed, a single XDI document can transmit both XDI content (i.e., XDI subjects rooted in the = and @ spaces) together with the relevant XDI dictionary definitions used by this content (XDI subjects rooted in the $ or + space). In the examples below, this would simply be a matter of combining some or all of this first XDI RDF table with the subsequent XDI RDF tables.

Note also that this example is limited simply to attributes of a person. The scope of XDI dictionaries is, like human-language dictionarys, bounded only by the context of the dictionary.

Subject	Predicate	Object	Literal
+	$has	person	
+person	hasa	$uri	
		+home	
		+work	
		+friend	
		+spouse	
		+name	
		+email	
		+phone	
+home	isa	$has+person	
+work	isa	$has+person	
+friend	isa	$has+person	
+name	isa	$type+xsi+string	
+person+name	isa	$type+xsi+string	
	hasa$*1	+first	
		+last	
		+legal	
		+preferred	

	hasa	+middle	
		+nickname	
+email	isa	$type+xsi+string	
	hasa+person	+home	
		+work	
		+primary	
		+alt	
		+preferred	
+phone	isa	$type+xsi+string	
	hasa	+country.code	
		+area.code	
		+number	
		+extension	
	hasa+person	+home	
		+work	
		+primary	
		+alt	
		+preferred	
phone+country.code	isa	$type+xsi+short	
	$type+xsi+enumeration	1	
		20	
		30	
		…	
phone+area.code	isa	$type+xsi+short	
Phone+number	isa	$type+xsi +postiveinteger	
phone+extension	isa	$type+xsi+short	

XDI I-Cards ("Contexts" or "Personas")

The following XDI RDF tables represent typical XDI documents that might be shared between individuals as XDI business cards or "i-cards". In the Higgins Project[1], an i-card is a user interface icon/metaphor that represents a source of information about one digital subject[2] from one Higgins context.[3]

The first section below represents the **=drummond** context and includes many literal data values. The second section represents the **=drummond+home** context and consists mostly of cross-references to the literal data values in the **=drummond** context. If the **=drummond+home** context was being shared, the XDI document would include only the **=drummond+home** rows plus the **=drummond** rows referenced by the **=drummond+home** cross-references (the XRIs in the object column enclosed in parentheses). The same would be true if only the **=drummond+work** context was being shared.

Note that this XDI content uses the XDI dictionary definitions established in the previous section.

Subject	Predicate	Object	Literal
=drummond	$is	=drummond.reed	
	isa	+person	
	$has	+home	
		+work	
	$uri		http://equals drummond.name
	+blog	(/$uri)	
	+person+name		Drummond Reed
	+person+name+first		Drummond
	+person+name+middle		Shattuck
	+person+name+last		Reed
	+person+name+legal		Drummond Shattuck Reed
	+person+name +preferred	(/+person+name)	
	+person+name +nickname		Drum
	+email	(/+email$v*2)	

[1] http://www.eclipse.org/higgins/

[2] http://idgang.idcommons.net/moin.cgi/Lexicon

[3] In Higgins, one context contains information about N>=1 digital subjects. For example, the XRI =drummond when augmented with a service type (e.g. OpenID) is considered in Higgins to uniquely identify a context containing information about N=1 digital subject (Drummond). A Higgins "Context Provider" plug-in could be created which will map the underlying OpenID data model into the Higgins data model and present it as an i-card to the user.

	+email$v*1		drummond@raincity.com
	+email$v*2		drummond.reed@gmail.com
	+email+home	(/+email)	
	+email+work		drummond.reed@cordance.net
=drummond+home	isa	+person	
	+person+name	(=drummond/+person+name)	
	+person+name+first	(=drummond/+person+name+first)	
	+person+name+last	(=drummond/+person+name+last)	
	+person+name+nickname	(/+person+name+nickname*1)	
		(/+person+name+nickname*2)	
	+person+name+nickname*1	(=drummond/+person+name+nickname)	
	+person+name+nickname*2		Shad
	+email	(=drummond/+email)	
	+email+home	(=drummond/+email+home)	
=drummond+work	isa	+person	
	+person+name	(=drummond/+person+name)	
	+person+name+first	(=drummond/+person+name+first)	
	+person+name+last	(=drummond/+person+name+last)	
	+email	(=drummond/+email+work)	
	+email+work	(=drummond/+email+work)	

XDI Link Contracts

A particularly attractive feature of the XDI RDF model is the simplicity of XDI link contracts. Link contracts are themselves fully addressable XDI documents that control access, authorization, and usage rights to other XDI documents. In essence a link contract is directly analogous to a real-world contract, for example a legal contract between companies (such as a non-disclosure agreement) or a "social contract" between people. The basic structure of a link contract is a **hasa** relationship called **$contract** that describes a relationship between two XDI subjects. This **$contract** can be further refined to subtypes, such as **$contract+person** and **$contract+org**. The OASIS XDI TC defines only the basic **$contract** container structure for link contracts; the human-readable text and machine-readable attributes of specific link contracts will be defined by standards bodies, governments, industry consortia, or other communities of usage such as the Identity Commons Identity Rights Agreements Working Group (http://wiki.idcommons.net/moin.cgi/IdentityRightsAgreementsCharter).

Following are several examples of how link contracts can be applied to personal data sharing relationships. They all reference an instance of a hypothetical personal data sharing link contract defined by Identity Commons (**@idcommons/$contract+person**). Note that this reference to an externally-defined link contract only governs the non-machine-readable data sharing policies. The machine-processible access and authorization policies are expressed using XDI RDF statements about the XDI protocol operations described above.

Note also how **$has** relationships are used to create groups such as **+friend**, **+friend+soccer**, and **+spouse**. Typically link contracts are then associate with these groups. Each group member then inherits the permissions granted to that group.

Subject	Predicate	Object	Data
=drummond	$has	+friend	
		+friend+soccer	
		+spouse	
	=andy.dale	=drummond+friend	
	=les	=drummond+friend	
		=drummond+friend +soccer	
	=rainyday*bob	=drummond+friend +soccer	
	=hip.hop	=drummond+friend +soccer	
	=mvr	=drummond+spouse	
=drummond+friend	isa	+friend	
	=andy.dale		
	=les		
	$contract	(@idcommons/	

		$contract+person)	
	$get	(=drummond +home/*)	
		(=drummond +work/*)	
=drummond +friend+soccer	isa	=drummond+friend	
	=les		
	=rainyday*bob		
	=hip.hop		
	$contract	(@idcommons/ $contract+person)	
	$get	(=drummond +home/+email)	
=drummond+spouse	isa	+spouse	
	=mvr		
	$contract	(@idcommons/ $contract+person)	
	$get	(=drummond*/*)	
	$add	(=drummond*/*)	

X3 Notation

XDI TC member Bill Barnhill has suggested XDI RDF tables can be represented in plain text using a notation very similar to the N3 (Notation 3) format used with RDF. This "X3" format would simply express each of the XDI RDF statements as XRIs.

Note that, for readability, subjects, predicates, and objects are all indented on different lines, and repeated subjects and predicates are not shown (just as in the XDI RDF tables above). Also, when literals are shown, they appear after the three slashes separating the three parts of an XDI address, and they are not themselves part of the address, rather the target of the address.

The following examples are X3 notation for the XDI RDF tables from the preceeding two sections.

```
=drummond
      /$is
            /=drummond.reed
      /$is$a
            /+person
      /$has
            /+home
            /+work
            /+friend
            /+friend+soccer
            /+spouse
      /$uri
            //"http://equalsdrummond.name"
      /+blog
            /(/$uri)
      /+person+name
            //"Drummond Reed"
      /+person+name+first
            //"Drummond"
      /+person+name+last
            //"Reed"
      /+person+name+middle
            //"Shattuck"
      /+person+name+legal
            //"Drummond Shattuck Reed"
      /+person+name+preferred
            /(/+person+name)
      /+person+name+nickname
            //"Drum"
      /+email
            /(/+email$v*2)
      /+email$v*1
            //"drummond.reed@rainycity.net"
      /+email$v*2
            //"drummond.reed@gmail.com"
      /+email+home
            /(/email)
      /+email+work
```

```
            //"drummond.reed@cordance.net"
      /=andy.dale
            /=drummond+friend
      /=les
            /=drummond+friend
            /=drummond+friend+soccer
      /=rainyday*bob
            /=drummond+friend+soccer
      /=hip.hop
            /=drummond+friend+soccer
      /=mvr
            /=drummond+spouse
=drummond+home
      /$is$a
            /+person
      /+person+name
            /(=drummond/+person+name)
      /+person+name+first
            /(=drummond/+person+name+first)
      /+person+name+last
            /(=drummond/+person+name+last)
      /+person+name+nickname
            /(=drummond/+person+name+nickname*1)
            /(=drummond/+person+name+nickname*2)
      /+person+name+nickname*1
            /(=drummond/+person+name+nickname)
      /+person+name+nickname*2
            //"Shad"
      /+email
            /(=drummond/+email)
      /+email+home
            /(=drummond/+email+home)
=drummond+work
      /$is$a
            /+person
      /+person+name
            /(=drummond/+person+name)
      /+person+name+first
            /(=drummond/+person+name+first)
      /+person+name+last
            /(=drummond/+person+name+last)
      /+email
            /(=drummond/+email+work)
      /+email+work
            /(=drummond/+email+work)
=drummond+friend
      /$is$a
            /+friend
      /=andy.dale/
      /=les/
      /$contract
            /(@idcommons/$contract+person)
      /$get
            /(=drummond+home/*)
            /(=drummond+work/*)
=drummond+friend+soccer
      /$is$a
```

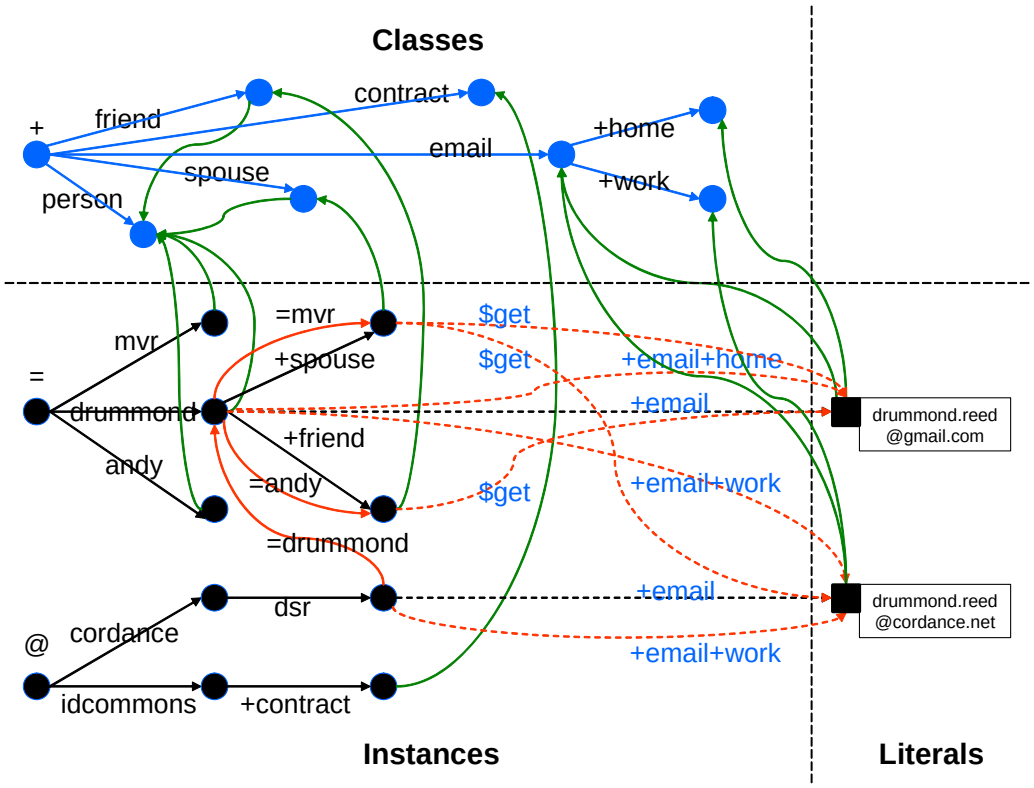
```
            /=drummond+friend
      /=les/
      /=rainyday*bob/
      /=hip.hop/
      /$contract
            /(@idcommons/$contract+person)
      /$get
            /(=drummond+home/+email)
=drummond+spouse
      /$is$a
            /+spouse
      /=mvr/
      /$contract
            /(@idcommons/$contract+person)
      /$get
            /(=drummond*/*)
      /$add
            /(=drummond*/*)
```

An Example XDI Graph

XDI documents, XDI RDF tables, and X3 text blocks can also be represented as visual graphs. Following is a key to reading the graph:

- subject node (instances)
- object node (classes)
- data
- $has (instance → instance relationship)
- $is (equivalence relationship)
- hasa (class → subclass relationship)
- isa (instance → class relationship)
- instance → literal relationship
- instance → literal equivalence relationship

Following is a example graph of a typical XDI document containing a subset of the XDI RDF statements from the preceeding XDI RDF tables. Note that the upper left-hand quadrant represents the XDI dictionary space, i.e., it illustrates XDI RDF object definitions or "classes". The lower left-hand quadrant represents the XDI RDF instance space, and the green arrows up to the XDI dictionary space represent the instance/class relationships. The lower right-hand quadrant represents the literal data space, with green arrows up to the dictionary space representing isa relationships to the simple data types defined there (note that further details of these definitions are not shown).



Appendix A: The XDI RDF Schema V2

Note: the only change from V1 was to abbreviate the subject, predicate, and object element names as spelling the words out tended to hinder rather than enhance readability of XDI documents.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!-- Proposed XDI XDI RDF schema v2, posted 2007-02-06 -->
<schema xmlns="http://www.w3.org/2001/XMLSchema"
xmlns:xdi="http://xdi.oasis-open.org" targetNamespace="http://xdi.oasis-open.org"
elementFormDefault="qualified">
	<element name="xdi">
		<complexType>
			<sequence>
				<element ref="xdi:s" maxOccurs="unbounded"/>
			</sequence>
		</complexType>
	</element>
	<element name="s">
		<complexType>
			<sequence>
				<element ref="xdi:p" minOccurs="0"
maxOccurs="unbounded"/>
			</sequence>
			<attribute name="xri" type="string" use="required"/>
		</complexType>
	</element>
	<element name="p">
		<complexType>
			<choice>
				<element ref="xdi:o" minOccurs="0"
maxOccurs="unbounded"/>
				<element ref="xdi:data" minOccurs="0"/>
			</choice>
			<attribute name="xri" type="string" use="required"/>
		</complexType>
	</element>
	<element name="o">
		<complexType>
			<attribute name="xri" type="string" use="required"/>
		</complexType>
	</element>
	<element name="data">
		<complexType mixed="true"/>
	</element>
</schema>
```

Appendix B: An Example XDI Document

Following is an example XDI document using the XDI RDF V2 schema intended to illustrate a small set of XDI personal i-cards and associated link contracts taken from the example XDI RDF tables above.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Example instance document using XDI RDFX schema v1, posted 2007-02-15 -->
<xdi:xdi xmlns:xdi="http://xdi.oasis-open.org"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
        <xdi:s xri="=">
                <xdi:p xri="$has">
                        <xdi:o xri="drummond"/>
                </xdi:p>
        </xdi:s>
        <xdi:s xri="=drummond">
                <xdi:p xri="$is">
                        <xdi:o xri="=drummond.reed"/>
                </xdi:p>
                <xdi:p xri="$is$a">
                        <xdi:o xri="+person"/>
                </xdi:p>
                <xdi:p xri="$has">
                        <xdi:o xri="+home"/>
                        <xdi:o xri="+work"/>
                        <xdi:o xri="+friend"/>
                        <xdi:o xri="+spouse"/>
                </xdi:p>
                <xdi:p xri="+email">
                        <xdi:o xri="(/+email+home)"/>
                        <xdi:o xri="(/+email+work)"/>
                </xdi:p>
                <xdi:p xri="+email+home">
                        <xdi:data>drummond.reed@gmail.com</xdi:data>
                </xdi:p>
                <xdi:p xri="+email+work">
                        <xdi:data>drummond.reed@cordance.net</xdi:data>
                </xdi:p>
                <xdi:p xri="=andy.dale">
                        <xdi:o xri="=drummond+friend"/>
                </xdi:p>
                <xdi:p xri="=les">
                        <xdi:o xri="=drummond+friend"/>
                        <xdi:o xri="=drummond+friend+soccer"/>
                </xdi:p>
```

```
        <xdi:p xri="=rainyday*bob">
                <xdi:o xri="=drummond+friend"/>
        </xdi:p>
        <xdi:p xri="=hip*hop">
                <xdi:o xri="=drummond+friend"/>
        </xdi:p>
        <xdi:p xri="=mvr">
                <xdi:o xri="=drummond+spouse"/>
        </xdi:p>
</xdi:s>
<xdi:s xri="=drummond+home">
        <xdi:p xri="+email">
                <xdi:o xri="(=drummond/+email+home)"/>
        </xdi:p>
</xdi:s>
<xdi:s xri="=drummond+work">
        <xdi:p xri="+email">
                <xdi:o xri="(=drummond/+email+work)"/>
        </xdi:p>
</xdi:s>
<xdi:s xri="=drummond+friend">
        <xdi:p xri="$is$a">
                <xdi:o xri="+friend"/>
        </xdi:p>
        <xdi:p xri="$has">
                <xdi:o xri="+soccer"/>
        </xdi:p>
        <xdi:p xri="=andy.dale"/>
        <xdi:p xri="=les"/>
        <xdi:p xri="$contract">
                <xdi:o xri="(@idcommons/$contract+person"/>
        </xdi:p>
        <xdi:p xri="$get">
                <xdi:o xri="(=drummond+home/*)"/>
        </xdi:p>
</xdi:s>
<xdi:s xri="=drummond+friend+soccer">
        <xdi:p xri="$is$a">
                <xdi:o xri="+friend+soccer"/>
        </xdi:p>
        <xdi:p xri="=les"/>
        <xdi:p xri="=rainyday*bob"/>
        <xdi:p xri="=hip.hop"/>
        <xdi:p xri="$contract">
                <xdi:o xri="(@idcommons/$contract+person"/>
        </xdi:p>
```

```
            <xdi:p xri="$get">
                <xdi:o xri="(=drummond/+email+home)"/>
            </xdi:p>
        </xdi:s>
        <xdi:s xri="=drummond+spouse">
            <xdi:p xri="$is$a">
                <xdi:o xri="+spouse"/>
            </xdi:p>
            <xdi:p xri="=mvr"/>
            <xdi:p xri="$contract">
                <xdi:o xri="(@idcommons/$contract+person"/>
            </xdi:p>
            <xdi:p xri="$get">
                <xdi:o xri="(=drummond*/*)"/>
            </xdi:p>
            <xdi:p xri="$add">
                <xdi:o xri="(=drummond*/*)"/>
            </xdi:p>
        </xdi:s>
</xdi:xdi>
```

Appendix C: Revision History

V2

- Shortened the <subject>, <predicate>, and <object> element names in the XDI RDF schema to make instance documents easier to read.
- Revised the XDI RDF diagram to illustrate link contracts instead of dictionaries.
- Provided a more typical example of an XDI instance document including examples of link contracts.
- Added a phone number example to the XDI RDF tables in the XDI Dictionary Definitions section to show a complex object definition.

V3

- New terminology (see Terminology section below).
- Updated the XDI RDF model section to specify all XDI RDF statement forms that can be expressed with XRIs.
- Replaced the core XDI RDF predicate names with their simpler English equivalents (suggested by Bill Barnhill, Paul Trevithick, and Laurie Rae, among others).
- Changed **$data** to **$type** (per suggestion from Bill Barnhill) and removed it from the core predicate table (because it is only used as an object).
- Added information about the RDF/RDFS/OWL equivalents for the four core XDI RDF predicates.

- Updated the link contract examples to show a simpler way to express group membership.
- Added section show X3 syntax.

V4

- Updated terminology table to use “ATI” instead of “3L”.
- Simplified table listing options for XDI RDF model.
- Adjusted text and examples to use proposed XRI 2.1 syntax.
- Updated changed terminology in graph examples.

V5

- Clarified references to Higgins and Higgins contexts in the XDI I-Cards section.
- Updated references to XRI Syntax 2.1 to XRI Syntax 3.0.
- Miscellaneous editing throughout for readability.
- Moved revision history to Appendix C

To-Dos for Future Versions

- *Request from Laurie Rae*: Show separate example XDI RDF documents for dictionary entries and instances of those entries.
- *Request from Paul Trevithick*: Show the exact representation of several example XDI RDF documents in all four formats (XML, table, X3, and graph). (The current document shows slightly different data sets in all four formats.)