Customer Profile Exchange (CPExchange) Specification

Kathy Bohrer and Bobby Holland, editors

October 20, 2000
Version 1.0

Copyright

IDEAlliance, Graphic Communications Association (GCA), Printing Industries of America (PIA), and the members of the CPExchange Network (collectively and individually, "Presenters") make no representations or warranties, express or implied, including, but not limited to, warranties of merchantability, fitness for a particular purpose, title, or non-infringement. Presenters make no representation or warranty that the contents of this document are free from error or suitable for any purpose of any user, nor that implementation of such contents will not infringe any third party patents, copyrights, trademarks or other rights. By making use of this document, the user assumes all risks and waives all claims against Presenters.

In no event shall Presenters be liable to user (or other person) for direct, indirect, special or consequential damages arising from or related to any use of this document, including, without limitation, lost profits, business interruption, loss of programs, or other data on your information handling system or otherwise, even if Presenters are expressly advised of the possibility of such damages.

Abstract

This document presents the **Customer Profile Exchange (CPExchange)** specification for the privacy-enabled global exchange of customer profile information. It uses the **Extensible Markup Language (XML)** to format messages transmitted under standard Internet protocols. This specification includes a Customer Information model and a Privacy Information model. Future versions of this specification will include an Operations Information model.

The information models contained in this specification facilitate customer profile transport. The models include the metadata that associates data protection (privacy) to customer profiles. A future version will define operations for query, delivery and update of the customer profile information. The specification builds on the **W3C XML Schema** and the **W3C P3P specifications**.

Acknowledgements

The CPExchange Working Group acknowledges the members of the CPExchange Network and other industry experts who have contributed directly or indirectly to the creation or content of this document. The Working Group is particularly grateful to Vignette Corporation for the strawman proposal and to IBM for the contribution of their IAA XML Messaging Architecture and CRM UML object model. The Vignette strawman proposal and the IBM CRM object model formed the basis for the business information in this CPExchange specification. The IBM IAA XML Messaging architecture contributed the UML to XML mapping and 3-layer payload architecture.

Document Revision History:

Date	Author	Changes
1999.08.17	Paula Angerstein	Initial strawman proposal from Vignette.
1999.12.17	Paula Angertstein and Kathy Bohrer	Combined strawman with IBM IAA Party model.
2000.04.27	Kathy Bohrer	Various updates. Added sample XML profile document.
2000.05.23	Bobby Holland	Edited and revised. Added XML DTD and use cases
2000.06.20	Bobby Holland	Revised the Privacy sections
2000.08.07	Bobby Holland	Edited, revised per group
2000.08.28	Bobby Holland	Incorporatedlatest group revisions
2000.08.29	Bobby Holland	Incorporatedlatest revisions
2000.09.10	Kathy Bohrer	Incorporated group revisions. Added new tables, DTDs, and example.
2000.09.11	Bobby Holland	Reviewed with minor mods
2000.10.10	Kathy Bohrer	Fixed editorial errors documented by Paul Conn.
2000.10.13	Kathy Bohrer	Edits from network member review.
2000.10.18	Bobby Holland	Review changes
2000.10.19	Kathy Bohrer	Incorporated minor group edits.
2000.10.20		Final 1.0 specification.

Table of Contents

Introduction

The **Customer Profile Exchange (CPExchange)** specification is a new standard that takes into account the traditional customer-facing facets of an enterprise, such as customer support, call centers, sales tracking, marketing campaigns and order tracking. This standard builds common ground with emerging web-based **Internet Relationship Management (IRM)** and **Enterprise Relationship Management (ERM)** approaches to online business, decision support, and personalisation.

The CPExchange specification is a standard that will facilitate the privacy-enabled exchange of customer profile information. The CPExchange specification defines a data model for profile information that relates a customer or other business partner, to an enterprise. The CPExchange specification also defines metadata for associating privacy controls with various subsets of the profile information, as well as operations for query, delivery, and update of this information. This information is represented in **Extensible Markup Language (XML)**, which can be exchanged through messages.

Goals and Capabilities

The CPExchange specification is intended to support enterprise-wide architectures in which many applications use and update information relating to a customer. This specification differs from enterprise frameworks such as **Active Directory Services Interface (ADSI)**, **Java Naming and Directory Interface (JNDI)**, and **Lightweight Directory Access Protocol (LDAP)** in the following ways:

- It is independent of any platform, vendor, and application
- It provides a comprehensive view of the customer, not just as a user of a particular application, but also as an entity that interacts with multiple facets of an enterprise.
- It allows views of the customer's activities over time, providing a cumulative historical record of events that enhances the enterprise's understanding of the customer.
- It provides a granular privacy and authorization model that is optimized for aggregated and interchanged information as opposed to traditional monolithic storage models.
- It is designed to promote generation and collection of customer profile information in a fashion that promotes useful query and reporting for enterprise customer relationship management.

Use of this specification to describe profile information for people and organisations will promote interoperability between all applications in an enterprise that touch the same person or organisation. The CPExchange specification can be used to carry profile information about people or organisations that have other relationships to an enterprise, such as a supplier or employee. The term "customer" is used in this document for convenience, but can also refer to a person or organisation regardless of their particular relationship to the enterprise, unless otherwise noted.

The ability to exchange profile information throughout and between enterprises leverages the collective (enterprise) knowledge of a customer to provide service, support, and new products that tailored to a customer's needs and wishes. It allows both customers and enterprises to be more efficient in their contacts with each other, whether by customization of self-service web sites or by personalization of more direct call center or e-mail contacts.

Within an enterprise, no single application need use nor store all the information in a customer profile. Each application can aggregate the information through exchange of messages carrying profile data in the standard representation defined in this specification. Information exchanged is represented withXML, but it is expected that applications will map the XML into their proprietary data representations. Applications that support the CPExchange data model should be able to receive an XML-based CPExchange customer profile, map the elements into appropriate internal data representation and generate an XML-based CPExchange customer profile from their internal data representation. There is no requirement that a generated customer profile exactly match any CPExchange customer profile previously taken in; however, it is intended that a given application consistently map a given data item to the same CPExchange element.

While the CPExchange data model provides a base set of information elements that all CPExchange-compliant applications must understand, it provides for points of extensibility for application-specific information elements.

Future Directions

In the future, CPExchange will enhance this specification with definitions of operations that can be used to query and update CPExchange business information, and will also define additional business information. CPExchange will also define an XML Schema for its XML representation. More details on these future enhancements can be found in **Appendix A**.

Privacy Strategy

CPExchange is a privacy-enabling standard in that it supports the exchange of privacy information as metadata applied to business data. This means that various profile data elements may be associated with multiple privacy declarations. Privacy declarations are defined in a privacy header of a CPExchange XML document. The CPExchange privacy declarations carry policy information defined in the W3C P3P standard.

The current specification is intended to facilitate the business-to-business communication of the types of privacy metadata gathered by the privacy policy negotiations of related standards, such as P3P. The scope of the privacy declarations encompasses the exchange between the current sender and receiver of the data, without regard to the source, or originator, of the data. The latter aspect may be addressed in future version of this specification.

Documentation Conventions and Terminology

The XML defined by this specification must conform to the DTDs in the Appendices. However, the information represented in the XML is presented in this document as UML object model drawings, tables, and descriptive text.

The UML usage is explained in the Unified Modeling Language (UML) Usage section. The UML->XML Mapping section details the general rules used to represent the UML models in XML. By convention, all names of classes and attributes in the UML are mixed case. Class names start with an uppercase character, attributes start with a lowercase character. Primitive and simple datatypes follow the XML Schema conventions and start with lowercase.

Each UML drawing is focused on a particular class or set of classes. The drawing is followed by descriptive text for each class, describing its main responsibilities and any important information about its attributes.

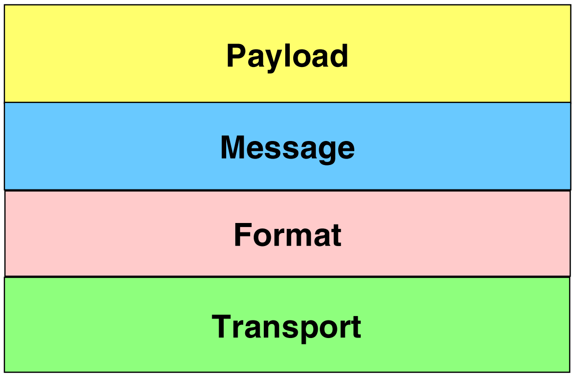
A table follows each class description, listing the XML Entity for each class, followed by the XML element for each object attribute of that class. Both inherited and added elements are listed. The "type" column gives the type of each XML Entity and element. The first class to use an enumerated type lists that enumerated type in its table. In this version of the standard enumerated types are documented in the UML drawings and tables but not represented in the DTDs or XML. The "type" column lists the valid set of values for that enumerated type.

Entity **or element or enumerated type**	**Description**	**Datatype or allowed values**

An example of a CPExchange XML document, including both business information and privacy control metadata is contained in the Appendices.

Technical Overview

The CPExchange specification is designed for applications that exchange messages. The architecture of a messaging system includes the following technical layers: **transport**, **data format**, **message structure** and **payload**.



CPExchange established the following requirements on its specifications:

1) platform/application/vendor independence

2) communications protocol independence

3) market acceptability

4) message envelope independence

5) operation independence

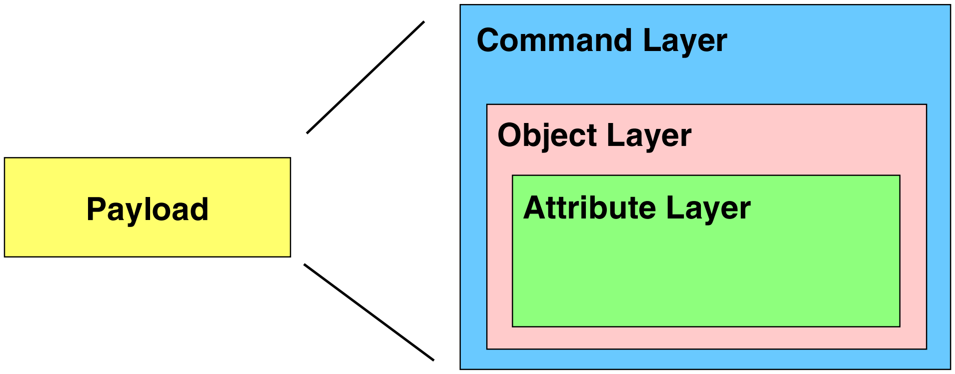
These requirements influence how CPExchange fits into each layer of the message system architecture.

The **transport** layer provides the basic communications infrastructure and low-level protocol. CPExchange specifications are transport independent. However, a primary transport is expected to be HTTP over TCP/IP.

The **format** determines the basic way that the bits carried through the communications infrastructure are interpreted. CPExchange fits into message systems that carry data formatted as XML.

The **message** layer defines the overall structure of a message, which is information common to the entire message. This information allows consumers and suppliers to identify the message uniquely, interpret the message properly, and determine whether a response is required. CPExchange is designed to be useable within different message layers, especially those that are emerging as standards for XML data exchange between or within businesses. Examples are BIZTALK, ebXML, ICE, SOAP and more domain specific message layers like those defined by OTA and IFX.

The **payload** is the information carried within the common message structure. The payload is unique to the business information that is being exchanged, the operations being requested, and the responses being returned. CPExchange defines payload information. The payload can be divided into three layers of information: commands, objects, and attributes.



Each command within a message contains information about an operation and the subject of that operation. The command layer provides enough information about the command for the receiving software to process it accurately. The operations include, but are not limited to, the usual data-centric operations to add, modify, delete, get (an instance) and search for information. Command definitions are not included in this version of the CPExchange specification, but in the future a minimum set of the data-centric operations will be added. More specific business activities can be added as extensions. Theobject layer defines the subjects of the commands.

Objects carry business information or metadata such as privacy control information. Objects provide a convenient way to address groups of attributes that are often used together. Objects in a message, or across messages from the same supplier, can be referred to through their object identifier. This uniquely identifies an object within the scope of its supplier. Objects with object identifiers are independent of one another, although relationships exist between them. Objects are represented in XML as elements that have sub-elements corresponding to their attributes. For more detailed information on the mapping between an object model and an XML representation see the section UML to XML Mapping.

The attribute layer of the message represents each piece of data within an object. An attribute may be a complex (object) or simple data type. Simple data types have no sub-elements in their XML representation. Attributes are represented in the XML messages as elements, not as XML-attributes of an element. Attributes may be required or optional in objects, and attributes may be allowed to occur multiple times in an object.

This CPExchange specification includes an information model of both business data and privacy metadata. The business data information model defines a collection of profile elements that describe people and organisations that have a business relationship with a particular company. This includes both static information about a person or organisation as well as a history of interactions with a person or organisation. The described people and organisations are often **customers**, but might have some other business relationship such as **supplier** or **employee**. The goal of the business data model is to define a common representation and meaning of a base set of profile data elements that can be interchanged between and used by a broad set of applications.

A business data model for a profile (person or organisation) contains descriptive information of a business entity, a record of encounters between that entity and the enterprise, the individual events or actions that occur in an encounter, and descriptions of items involved in the encounter. A profile can contain data on multiple encounters. A web site visit is an example of an encounter. A product reference is an example of an item associated with a purchasing event within a web site encounter. The CPExchange specification defines a core set of encounter types, action event types, and item types. Additional action event types will be defined by other groups and for specific industries.

Unified Modeling Language (UML) Usage

The objects and attributes of a CPExchange message payload are described informally in this document with **Unified Modeling Language (UML)** object model diagrams. The UML object diagrams capture the information and relationships that are then represented in XML format according to the CPEXML DTDs. UML class diagrams capture the object types (classes), their attributes, the attribute types, and relationships between classes.

Inheritance relationships show how one object class (subclass) extends another object class (superclass) to contain both the data of the superclass and add additional attributes. For example, most classes in CPExchange inherit from the Distinguishable superclass that defines attributes for an object identifier, a creation date, a last modification data, and reference to a privacy declaration.

Associations show how an object of one class references or contains other objects (of the same or a different class). Associations have cardinality and navigation characteristics. Cardinality defines how many objects of one end of the association are associated with how many objects on the other end of the association. A **0..1** cardinality says that an object has **0** or **1** objects associated with it. That means the association to the other object is optional. For example, a person object may have an associated occupation object. A **1..1** association would be a mandatory association to **1** other object. A **0..n** or **1..n** association would say an object is associated with up to **n** other objects. For example, a person object can have multiple nationality objects if they have dual citizenship or residences in multiple countries.

Associations also indicate navigation direction. No arrowhead on an association indicates that the association is navigable in both directions. That is, each object would include a sub-element to reference or contain the other associated object. For example, a person can be associated with multiple nationality objects. That association is navigable in both directions (no arrowhead), so that the person object has sub-elements that are the nationality objects, and the nationality object has a reference to the associated person object. Note that the association can only be represented by a sub-element that **is** the object in one direction, other associations must be done by reference to the other objects through their object identifiers. Where there is an arrowhead on the association, it is an indication that only the class pointing to the other class defines a sub-element for the association. For example, a contact point usage object has a reference to the party name object that provides the address information for a particular address contact point. However, a party name object does not have sub-elements for all the contact point usages with which it is associated.

UML to XML Mapping

The CPExchange payload representation is XML based. In the future it will be formally defined by an XML Schema. Currently, the XML representation is defined in XML DTD files. Some validation and datatype knowledge that can be expressed in an XML Schema will be lost in the **Document Type Definition (DTD)**. The XML data model is informally presented in tables in the following sections. These tables describe the XML data elements and datatypes of profile information. The appendices include the formal DTDs for the data model.

The XML representation is generated from the UML drawings according to a set of rules. These are described in the remainder of this section.

First, a set of primitive datatypes is defined to indicate how #PCDATA values should be constrained to match XML Schema Datatypes. Some of these are the built-in datatypes defined by the **XML Schema: Datatypes** standard. Others are CPExchange definitions of new XML Schema generated datatypes. The intent of the constraints imposed by each datatype is documented in this specification, or, in many cases other standards are referenced. The **XML 1.0 DTD** cannot express the datatype constraint. Instead the datatype is merely represented with a parameter entity reference. For example:

```
Primitive Types - they match the XML Schema Data Types
<!ENTITY % timeInstant "#PCDATA">
```

Each class may be represented as two parameter ENTITY definitions in the DTD. One ENTITY expresses the content of the class, and the other ENTITY expresses the XML attributes of the class. This second ENTITY is not required if the class does not add any new XML attributes to those of its superclasses. Subclass entities include the superclass entities. By convention, the name of content ENTITY is <ClassName>, where the class name is capitalized, the XML attribute ENTITY is <ClassNameAttrs>. For example:

```
Classes
<!ENTITY % Distinguishable      " oid, typeName?">
<!ENTITY % DistinguishableAttrs " eid ID #IMPLIED privacyControlIdref IDREF #IMPLIED">
Specialization/Inheritance
<!ENTITY % PersonName         " %PartyName;, firstName?, lastName">
<!ENTITY % anyDemographics      " personDemographics | OrganisationalDemographics">
```

For each class that can be instantiated, there is an ELEMENT defined in the DTD that includes the class ENTITY, and an ATTLIST that includes the attribute ENTITY. The name of the ELEMENT for a concrete class is <className> starting with a lowercase letter. For example:

```
Instances
<!ELEMENT personName (%PersonName;)>
   <!ATTLIST personName %BaseAttrs;>
```

Each attribute of a class is represented by an ELEMENT . If the attribute is an object in the UML drawing, then the corresponding ELEMENT is defined to include the ENTITY (or entities) of the corresponding class(es). The name is <attributeName> from the UML drawing. Where the ELEMENT is meant to use polymorphism to allow any subclass of a particular class, a special ENTITY is used which is defined as the “| “ of all the subclass entities. In the future, XML Schema types will allow the superclass to be specified as the type of the element.

```
UML Attributes (object or primitive type)
  <!-- properties -->
    <!ELEMENT preferredLanguage (%language;)>
    <!ELEMENT demographics (%anyDemographics;)>
```

An association by value is treated like an attribute of the class from which navigation is possible. Where the association is by value, the element name will be the same as the attribute in the UML drawing. If the cardinality of the association is more than 1, then an ELEMENT is defined that represents the collection of associated objects. This collection element has the name of the UML attribute. The sub-elements have the name of the singular form of the UML attribute, and are defined as instances of the associated class.

```
Association (by value)
 <!ELEMENT nationalities (nationality*)>
 <!ELEMENT nationality (%Nationality;)>
```

An association by reference to any object that inherits from Distinguishable is represented by an ELEMENT with a key data type. The value of the element must be the **oid** of the associated object. The associated object does not have to be in the same exchange document. Where the association is by reference, the names have an **Id** suffix, and the sub-elements in the group are key datatype values referencing the associated object. If the cardinality of the association is more than 1, then an ELEMENT is defined that represents the collection of associated objects. This collection element has a name **<attributeName>Ids**. The sub-elements have the name of the singular form of the UML attribute with an **Id** suffix, and are defined as key datatypes.

Wherever an ELEMENT is of a key datatype, the ELEMENT has an XML attribute called **referenceTo** which takes as a value the name of the class of the object that the element is referencing.

```
Association (by reference) for business objects that have an oid
 <!ELEMENT defaultNameIds (defaultNameId*)>
 <!ELEMENT defaultNameId (%key;)>
   <!ATTLIST defaultNameId referenceTo (PartyName) "PartyName">
```

Note that ID and IDREF are not used to implement associations by reference for business objects. That is because ID and IDREF are only useful for associating objects that exist in the same document. CPExchange is designed to support transfer of updates to objects, without having to resend all the associated objects. Therefore, object references are implemented in the XML using the **oid** sub-element of an object, rather then an object’s element ID (**eid**) XML attribute. In fact, CPExchange does not require a business information object ELEMENT to have a value for its **eid**, but it must have a value for its **oid**.

Associations by reference to objects that do not inherit from Distinguishable are done using either an XML IDREF or IDREFS attribute. The associated object must have an **eid** that is an XML ID attribute. Generally objects associated in this way are metadata that apply to business objects. The metadata objects do not have object oids. These objects can only be referenced within the scope of a document. In the UML drawings these associations are with a stereotype of **<<xml IDREF attribute>>**.

```
Association (by reference) for metadata object that have an eid, but no oid
for cardinality “0..1” or “1”
<!ENTITY % DistinguishableAttrs " eid ID #IMPLIED privacyControlIdref IDREF #IMPLIED">

or, for cardinality “0..n” or “1..n” objects
<!ENTITY % ExchangePartnerAttrs " controllingJurisdictionIdrefs IDREFS #IMPLIED">
```

Enumerated types are defined as either OpenEnum or ClosedEnum types. Closed enumerated types have a fixed set of valid values. OpenEnum types have an initial set of valid values that all implementations support, but can also have additional valid values defined by users of the standard. Each ELEMENT that is an enumerated type has an XML attribute **enumType** that names the valid set of values. This name has the form **<enum type name>Enum**. The valid values are documented in the UML drawings and tables of this document. In the future, the values will also be expressed in the XML Schema.

```
Enumerated Type
  <!ELEMENT preferredLanguage (%ClosedEnum;)>
        <!ATTLIST preferredLanguage enumType  (LanguageEnum) "LanguageEnum">
```

Datatypes

The following primitive and complex datatypes are used to constrain the **#PCDATA** content of elements and attributes. The primitive datatypes are defined in the **W3C XML Schema: Datatypes** standard. In some cases the **W3C XML Schema** standard references another standard for the exact syntax of the datatype representation.

Built-in Primitive Datatype	**Description**	**Standard Referenced**
float	Single precision floating point numbers	
double	Double precision floating point numbers	
decimal	Decimal numbers	
boolean	A true or false indicator	
binary	Binary data	
uriReference	Uniform Resource Locator, which identifies a Web resource	RFC 2396
recurringDuration	A recurring duration in time expressed as CCYY-MM-DDThh:mm:ss.sss	ISO 8601
timeDuration	A duration of time expressed as PnYnMnDTnH	ISO 8601

Built-in Primitive Datatype	Description	Standard Referenced
string		ISO 10646

Generated datatypes further constrain or extend a primitive type. The XML Schema: Datatypes standard defines a set of built-in generated types, and supports a way of expressing user-defined generated datatypes. The CPExchange standard defines certain generated datatypes, and these are indicated under the table column heading **Standard Referenced**.

Generated Datatype	Description	Standard Referenced
language	Natural language tokens	RFC 1766
integer		
nonPositiveInteger		
positiveInteger		
negativeInteger		
nonNegativeInteger		
long		
int		
short		
byte		
unsignedLong		
unsignedInt		
unsignedShort		
unsignedByte		
timeInstant	Date and time xpressed as CCYY-MM-DDThh:mm:ss.sss plus an optional time zone suffix	
timePeriod		
date		
recurringDate		
recurringDay		
month		
year		
century		

Generated Datatype	Description	Standard Referenced
time		
uuid	Universally Unique Identifier represented as 32 hexadecimal digits with optional embedded hyphens.	CPExchange, based on the Open Group CDE 1.1 UUID http://www.opengroup.org/onlinepubs/9629399/apdxa.htm
key	A string that uniquely identifies a data object within the scope of a single supplier.	CPExchange
currencyValue	A string with the ISO currency code, a colon separator, and a decimal representing an amount of money.	CPExchange, ISO 4217 (representation of currencies and funds)
ipAddress	A TCP/IP address.	CPExchange

key

All major object elements of CPExchange information, including those that can be independently queried and retrieved, contain an identifying **oid** element of the *key* datatype. Each key identifies an information object uniquely. It is an identifier that is unique in the CPExchange data space of a specific supplier. Some enterprises may have keys that are understood only by their organisation, while others may share keys across enterprises. The existence of the key in the CPExchange data allows a receiving enterprise to match data obtained previously with updates, or to recognize duplicate data from different sources that share or can match each other's keys.

```
<key>PERS000000355</key>
```

currencyValue

A currency value is represented as a string with a ISO currency code, followed by a "**:**" (colon), followed by a substring in decimal type format. For example, the following element is of type currencyValue representing $19.21.

```
<cost>USD:19.21</cost>
```

ipAddress

A TCP/IP address is represented by its dotted-decimal string value, or by a name that can be resolved in a TCP/IP network. For example:

```
<host>9.4.333.2</host>
```

Or

```
<host>austin.ibm.com</host>
```

Enum

An Enum is a string that must hold one of a defined set of valid values. Specializations of this type are the OpenEnum and the ClosedEnum.

OpenEnum

An OpenEnum is a specialization of the Enum type, that must hold one of a defined set of valid values where new valid values can be defined. A more specialized OpenEnum type is defined for a particular set of valid values. Generally the name of the specialized enumerated type is **<attribute name>Enum**. For example the valid values of the incomeLevel attribute are defined by the IncomeLevelEnum set of valid values. Users of CPExchange may define additional valid values.

User defined values should follow the naming convention **<inverted organisation domain name><value name>**, where the **<value name>** describes the new value and the **<inverted organisation domain name>** provides a prefix to prevent naming conflicts in user extensions to CPExchange. For example, an organisation with a domain name of foo.com could extend the set of values for the PartyRoletypeEnum with the new value com.foo.STUDENT.

ClosedEnum

A ClosedEnum is a specialization of the Enum type that must hold one of a defined set of valid values. The set of values is fixed and can not be extended except by CPExchange. A more specialized ClosedEnum type is defined for a particular set of valid values. Generally the name of the specialized enumerated type is **<attribute name>Enum**. For example the valid values of the gender attribute are defined by the GenderEnum set of valid values. This set of values is fixed by CPExchange and must not be extended with new values by users of CPExchange.

Complex Datatypes

The XML Schema: Structures standard also supports definition of complex datatypes for elements with mixed or element content. CPExchange defines a number of these using DTD Entities as described in the UML to XML Mapping section. The detail is included in subsequent sections.

In particular, CPExchange defines complex datatypes for names, addresses, and phone numbers, rather than using simple strings or some constraining datatype based on a string. Marking up component parts of this information provides the basis for rich query and sorting algorithms.

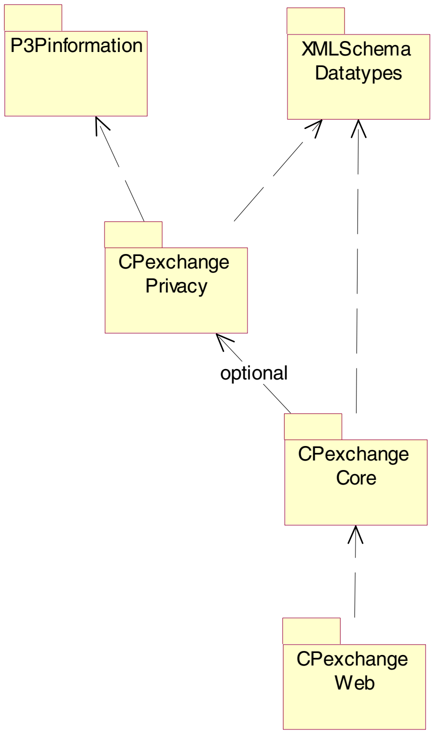
Extensibility Support

The CPExchange data model is designed to allow extensibility while providing consistency. Since the data model representation is XML-based, it has some inherent extensibility. The DTD mechanism of **entities** can be used to include the CPExchange DTDs into new DTDs. These new DTDs can have additional elements that will only be handled by applications that understand those new elements. CPExchange also supports a property element definition that allows application-specific sets of properties to be defined outside of CPExchange, and without requiring additional XML element definitions. This allows CPExchange applications to store, propagate, and display properties when the application does not have any specific knowledge or business processing for those properties.

CPExchange Information Categories

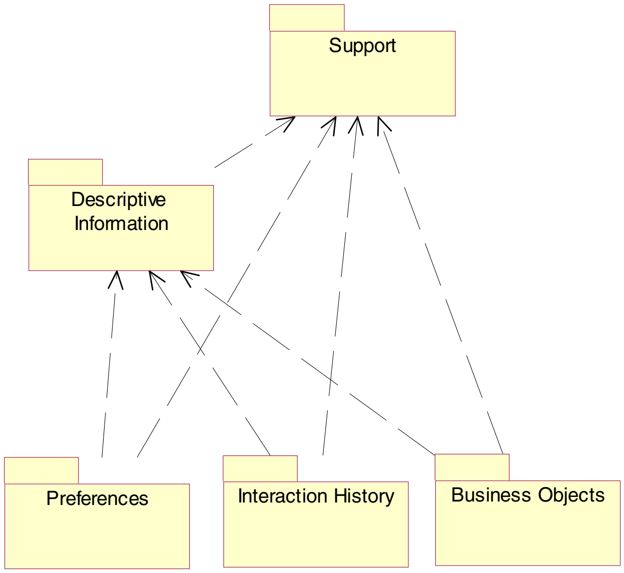
Profile information can be quite large. Therefore CPExchange organizes profile information into categories and subcategories. Suppliers can provide only certain categories of profile information, and subscribers can request certain categories, or query subsets of certain categories.

CPexchange Categories



At the highest level, the CPExchange categories build on categories that represent information defined in the XML Schema and P3P W3C Working Drafts. CPExchange defines both a Privacy category of metadata, a Core category of business information common across industries and channels, and a Web category that adds business information specific to web usage.

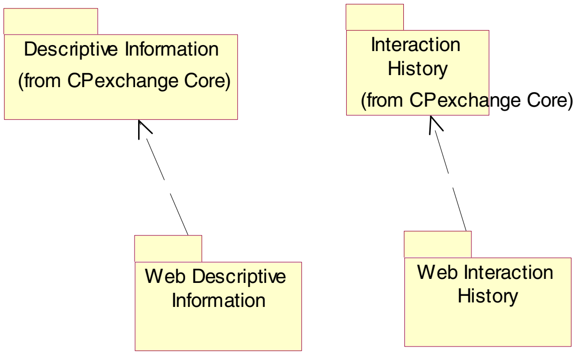
CPexchange Core Subcategories



The Core category contains the following subcategories:

- Support - data common to many objects
- Descriptive – static information about a role player, including identifying information, names, addresses, demographics, activities, relationships, and roles
- Preferences – name/value pairs describing preferences of a role player
- Interaction History - history of interactions between role players, optionally associated with cases that track a process or subject
- Business Objects – references to business data that is detailed in some back office or industry specific system

CPexchange Web



The Web category extends the Core category with two subcategories that provide specialized information used in web channels. Other vertical industry groups could extend the Core CPExchange category in a similar way.

- Web Interaction History – extension to Interaction History for web channels
- Web Descriptive Information – extension to Descriptive Information for web channels

The Core and Web category Profile business information revolve around a **role player**, which is either a party or a role played by a party. A party is either a person or an organisation. A profile exchange can contain descriptive information about a role player, the preferences of a role player, a record of encounters between role players, the individual events or actions that occur in an encounter, and descriptions of items involved in the encounter. A web site visit is an example of an encounter. A product reference is an example of a description of an item associated with a purchasing event within a web site encounter. CPExchange defines a core set of descriptive information including party roles, names, contact information, activity types, and demographics. CPExchange also defines a core set of encounter types, action event types, and business object types. The core information is applicable across many industries. The information model can be extended with properties and additional types for specific vertical industries.

Privacy Information

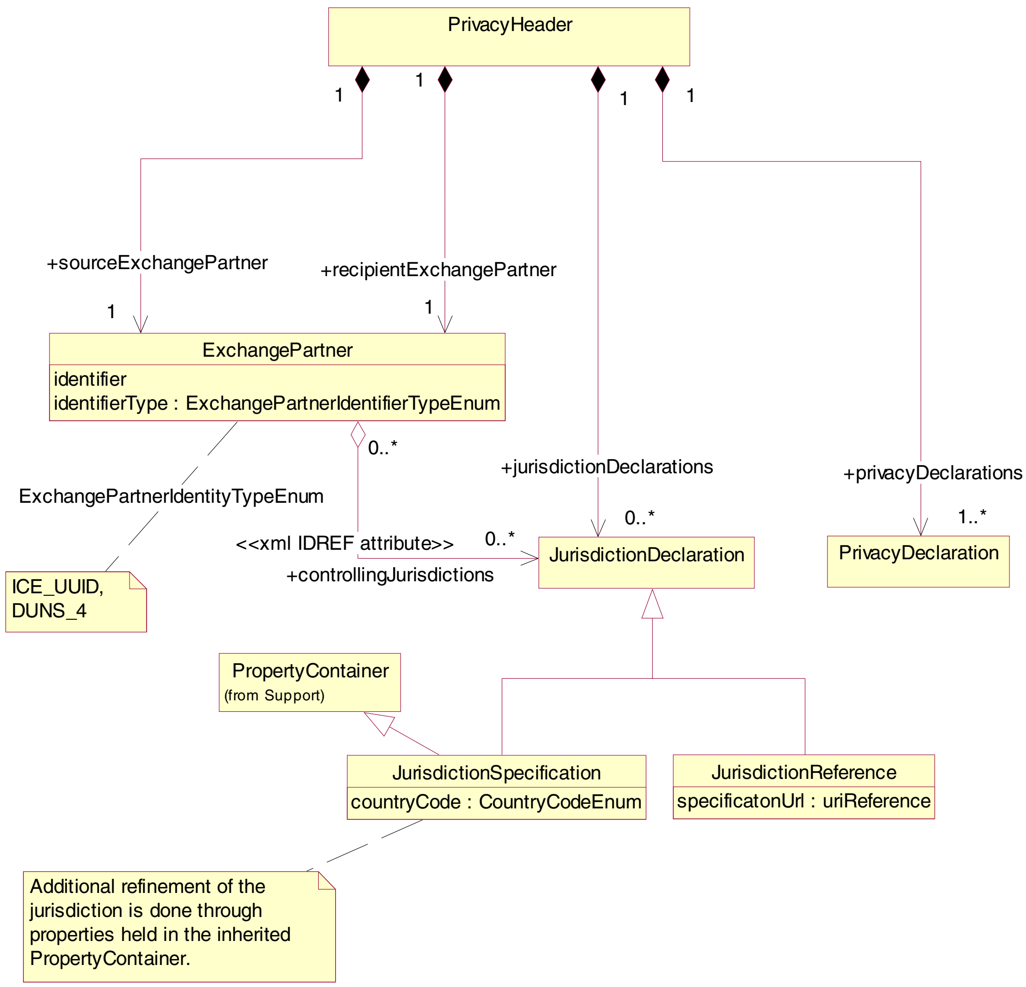
The Privacy Information Model extends the P3P 1.0 W3C Working Draft 18 October 2000 document to describe the privacy policies covering the profile information being exchanged. A CPExchange document maycontain a single policy header and one or more control groups of profile information associated with different policy declarations within the policy header.

The privacy objects described below are referenced when necessary within the XML document using XML ID attributes and IDREF attributes. These IDs are unique within a single XML document, and are not necessarily the same across multiple document exchanges. This is different from the business data objects which have an object identifier, “oid” object attribute (XML subelement), which identifies a business object across all document exchanges. This use of document scoped IDs for objects in the PrivacyCategory allows the privacy controls defined in this specification to be easily reused in other XML messages for other purposes. The PrivacyCategory depends only on the XML Schema data types and the CPexhange user defined simple datatypes, in particular, Property.

Privacy Policy Header

Each CPExchange data transfer mayinclude a PrivacyHeader that provides information identifying the sending and receiving exchange partners, information about the jurisdiction of each partner, and the privacy policies applicable to different pieces of data in the exchange.

Privacy Header



PrivacyHeader
1
1
1
1
+sourceExchangePartner
1
+recipientExchangePartner
1
ExchangePartner
identifier
identifierType : ExchangePartnerIdentifierTypeEnum
0..*
ExchangePartnerIdentityTypeEnum
+jurisdictionDeclarations
+privacyDeclarations
0..*
1..*
<<xml IDREF attribute>>
0..*
+controllingJurisdictions
JurisdictionDeclaration
PrivacyDeclaration
ICE_UUID,
DUNS_4
PropertyContainer
(from Support)
JurisdictionSpecification
countryCode : CountryCodeEnum
JurisdictionReference
specificatonUrl : uriReference
Additional refinement of the jurisdiction is done through properties held in the inherited PropertyContainer.

PrivacyHeader

The CPExchange Privacy Header is the container within a CPExchange document that holds information about the ExchangePartners, the Jurisdiction information applicable to those ExchangePartners and the PrivacyDeclarations that can be applied to data within the document.

ExchangePartner

Exchange Partner defines an entity that is either transmitting or receiving a CPExchange XML document. For any given data transfer there are two partners: a **Source Exchange Partner** and a **Recipient Exchange Partner**. An Identifier uniquely identifies the exchange partner. In a CPExchange Document both the Source and the Recipient Exchange Partners are declared.

JurisdictionDeclaration

A Jurisdiction Declaration contains information that identifies where to locate information that describes the jurisdiction to which the exchange partner(s) are subject. A Jurisdiction Declaration carries a unique ID within the XML document so it can be referred to in enumerating jurisdictions.

Each unique Jurisdiction is declared by reference or by specification.

JurisdictionSpecification

A JurisdictionSpecification presents descriptive information about the jurisdiction of an ExchangePartner. The descriptive information is held in the inherited PropertyContainer of the specification.

JurisdictionReference

A JurisdictionReference is a reference a document containing a JurisdictionSpecification. The specificationURL attribute points to the specification document.

PrivacyDeclaration

PrivacyDeclaration is a concrete subtype of PrivacyControl that supports policy information that is an extension of the P3P 1.0 W3C Working Draft 18 October 2000 document.

The PrivacyDeclaration aggregates specific policy characteristics that describe how data can be used (Purpose), how long data can be retained (Retention) and whether access may be granted to the individual's data (Access). Each declaration is given an ID within the XML document so that it may be referred to from Distinguishable business data elements in the XML document.

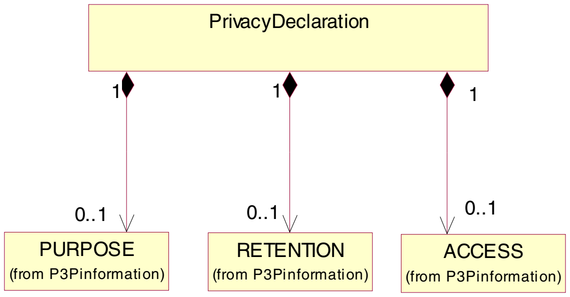
Entity or element	**Description**	**Datatype or allowed values**
PrivacyHeader	Container for all privacy controls specified	
1) sourceExchangePartner	Entity that is transmitting CPEXML document	ExchangePartner
2) recipientExchangePartner	Entity that is receiving CPEXML document	ExchangePartner
3) jurisdictionDeclarations	Collection of jurisdiction declarations for this PrivacyHeader	
a) jurisdictionDeclaration	A jurisdictionSpecification or a jurisdictionReference	JurisdictionDeclaration
4) privacyDeclarations	Collection of privacy declarations	
a) privacyDeclaration	Privacy specification that can be applied to "distinguishable" objects	PrivacyDeclaration
JursidictionDeclaration	Defines a jursidiction	
JurisdictionReference	Reference to a JurisdictionSpecification document	JurisdictionDeclaration

Entity or element	Description	Datatype or allowed values
1) specificationUrl	Web address to retrieve JurisdictionSpecification document	uriReference
JurisdictionSpecification	information that describes a Jurisdiction	JursidictionDeclaration, PropertyContainer
1) properties	Group of properties	
a) property	A named value.	Property
2) countryCode	Country of citizenship	2 character ISO 3166 code
ExchangePartnerIdentifierTypeEnum	Valid values for ExchangePartner "identifierType" sub element	ClosedEnum (ICE_UUID, DUNS_4)
ExchangePartner	Entity that is either transmitting or receiving the XML document	
1) identifier	identifier to uniquely identify the SourceExchangePartner	string
2) identifierType	Type of the identifier	ExchangePartnerIdentifierTypeEnum

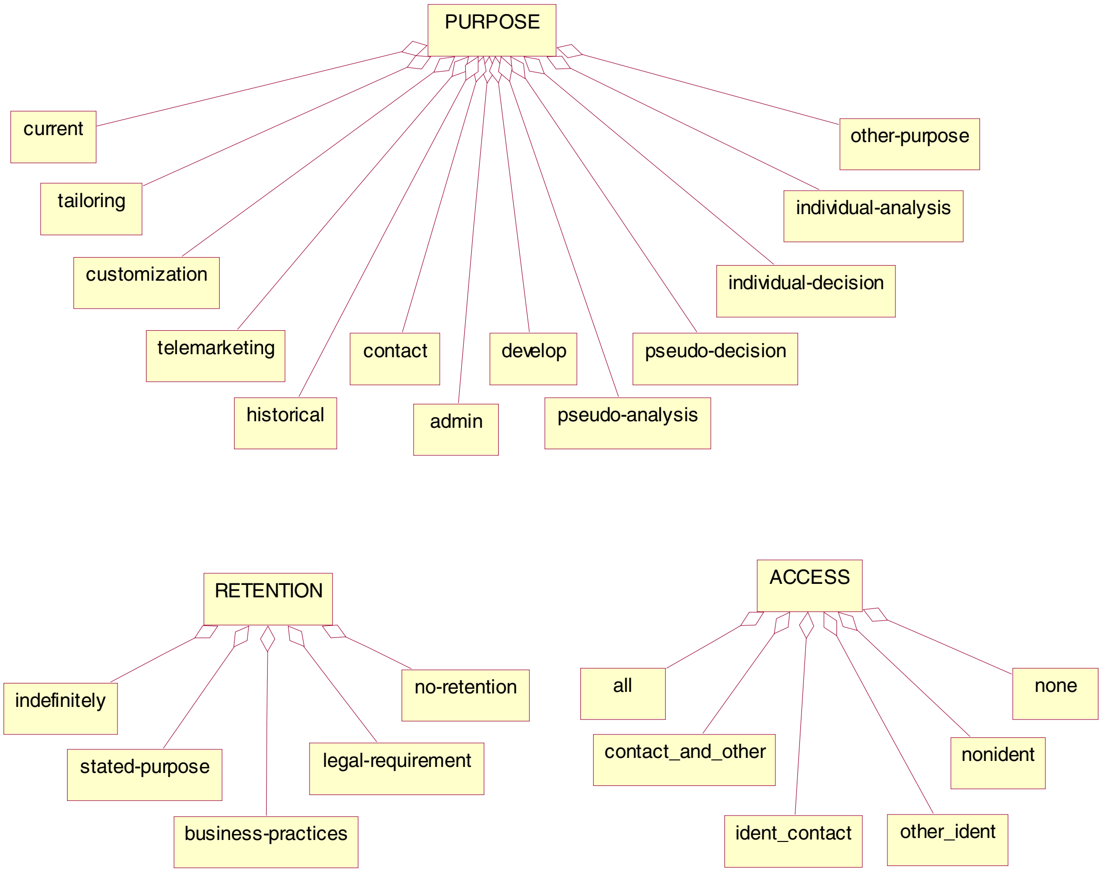
Privacy Declaration Information

A PrivacyDeclaration holds zero or one policy characteristic PURPOSE, RETENTION, and ACCESS objects. PURPOSE, ACCESS, and RETENTION are as defined as in the P3P 1.0 W3C Working Draft 18 October 2000 document..

Privacy Declaration Information



P3P Information (used in CPexchange), 10 October 2000



Purpose

Purpose gives the ways in which the data covered by this PrivacyDeclaration should be used.

The values that the PURPOSE sub-element position can take on are drawn from the P3P 1.0 W3C Working Draft 18 October 2000 document. In particular, one or more of the following sub-elements are allowed:

<current/> - completion and support of current activity

<admin/> - web site and system administration

<develop/> - research and development

<customization/> - affirmative customization

<tailoring/> - one-time tailoring

<pseudo-analysis/> - pseudonymous analysis

<pseudo-decision/> - pseudonymous decision

<individual-analysis/> - individual analysis

<individual-decision/> -individual decision

<contact/> - contacting visitors for marketing of services or products

<historical/> - historical preservation

<telemarketing/> - telephone marketing

<other-purpose>string</other-purpose> - other uses

Retention

Retention indicates the kind of retention policy that applies to the data covered by this Privacy Declaration

The values that the RETENTION sub-element position can take on are drawn from the P3P 1.0 W3C Working Draft 18 October 2000 document. In particular, one of the following sub-elements is allowed:

<no-retention/> - information is not retained for more than a brief period of time

<stated-purpose/> - information is retained to meet the stated purpose

<legal-requirement/> - information is retained to meet a stated purpose but the destruction of the information is governed by compliance with legal requirements

<business-practices/> - information is retained in accordance with a provider's business practices that establish a retention policy and destruction time table

<indefinitely/> - information is retained indefinitely

Access

Access describes whether access can be granted and how much access may be granted to the data covered by this PrivacyDeclaration.

The values that the PURPOSE sub-element position can take on are drawn from the P3P 1.0 W3C Working Draft 18 October 2000 document. In particular, one of the following sub-elements is allowed:

<nonident/> - identifiable data is not used

<all/> - access is given to all identifiable information

<contact_and_other/> - access is given to identifiable contact information, both online and physical, as well as other data linked to the identifiable entity

<ident_contact/> - access is given to identifiable contact information, both online and physical

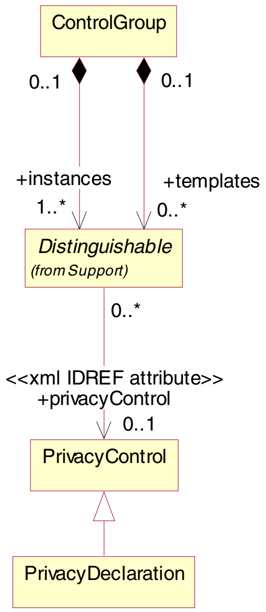
<other_ident/> - access is given to certain other data linked to an identifiable entity

<none/> - no access is given to identifiable information

Applying Privacy Control to Data Objects

The privacy policies are defined in PrivacyControl objects. The PrivacyControl objects in the PrivacyHeader can be associated with either groups of Distinguishable objects using a ControlGroup, or individual Distinguishable objects using a privacyControlId XML attribute on the Distinguishable object.

Privacy Control



ControlGroup

The ControlGroup collects a group of objects that share the same privacyControl default values. The default values are specified in a group of template objects. A template object provides the default privacyControl for all other objects of that same type in the ControlGroup. A template object is distinguished from a data object by being a sub-element of a templates element.

Both the template and instance objects are held as direct sub-elements of the templates and instances elements in the ControlGroup, rather than by reference elements. The PrivacyControl objects are referenced by a privacyControl attribute on a template object. Any instance of a Distinguishable object can override the template's default privacyControl value by having its own privacyControl attribute that references a different PrivacyControl object.

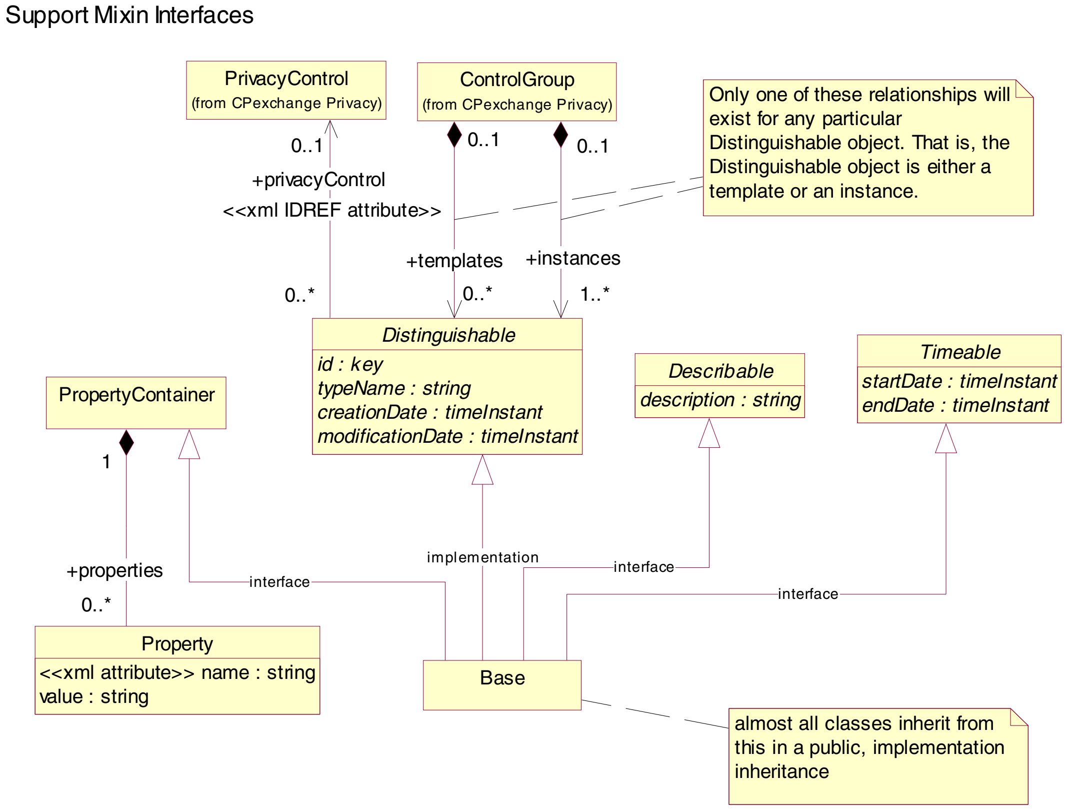
PrivacyControl

PrivacyControl is an abstract type to support the use of different privacy models. CPExchange uses the PrivacyDeclaration subtype of PrivacyControl.

PrivacyDeclaration is a concrete subtype of PrivacyControl that contains information on privacy policies based on the P3P 1.0 W3C Working Draft 18 October 2000 document.

Support Information

The classes defined here do not represent concrete information objects that can be queried and exchanged, but instead define common pieces of information that are inherited or used by many of the other classes described in subsequent sections.


Support Mixin Interfaces
PrivacyControl
(from CPexchange Privacy)
ControlGroup
(from CPexchange Privacy)
Only one of these relationships will exist for any particular Distinguishable object. That is, the Distinguishable object is either a template or an instance.
0..1
+privacyControl
<<xml IDREF attribute>>
0..1
0..1
+templates
+instances
0..*
0..*
1..*
Distinguishable
id : key
typeName : string
creationDate : timeInstant
modificationDate : timeInstant
Describable
description : string
Timeable
startDate : timeInstant
endDate : timeInstant
PropertyContainer
1
+properties
0..*
interface
implementation
interface
interface
Property
<<xml attribute>> name : string
value : string
Base
almost all classes inherit from this in a public, implementation inheritance

Distinguishable

The Distinguishable abstract class represents any object that can be uniquely identified from all other objects from the same supplier system. Almost all the concrete classes inherit from Distinguishable. Exceptions are Property, Preference, and ContentItemReference which are always accessed through some other containing concrete class.

The Distinguishable class provides a mandatory object id uniquely identifying an object within the scope of a set of objects provided by a single supplier system. This allows a receiver to match new updates to objects that were previously received. The Distinguishable class also provides optional attributes for the object's creation date, a last modification date, a typeName for further classification of an object within its class type, and a reference to a privacy declaration. The privacy declaration reference, if provided, overrides the privacy declaration referenced by an applicable template in the ControlGroup that holds this object.

Particular classes that support Distinguishable may have different allowed values for their typeName attribute. The enumerated list of valid values will be included in the information model as metadata. The naming convention of <class name>TypeEnum will be used to refer to the list of valid values for a particular class' typeName. For example, PartyRoleTypeEnum specifies the set of enumerated values for the typeName attribute of a PartyRole object.

It is assumed that the smallest amount of data that can be accessed independently is an object that supports Distinguishable information. For example, Party information could be queried and exchanged, and it would include any associated property data in that Party object. However, Party property data cannot be queried and retrieved independently, separated from its containing Party information because Property does not inherit from Distinguishable. A CPExchange supplier does not have to supply all of the Party information fields. Many fields of CPExchange objects are optional, and may not be supported by a particular supplier.

Distinguishable	Information to uniquely identify a queryable CPExchange object	
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OpenEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant

Timeable

The Timeable abstract class represents anything that has a time span. This class holds a start and end time instant. A time instant can have both a date and time including time zone information. Depending on the concrete subclass, this can be a time span during which the information is effective, or it can be the time duration of an activity, event or other occurrence.

Timeable	Information defining a range of time associated with a CPExchange object.	
1) startDate	A start date and time.	timeInstant
2) endDate	An ending date and time.	timeInstant

Describable

The Describable abstract class represents anything that can have a translatable text description. Describable can also encompass the storage of multiple text translations for different locales.

Describable	Information providing a translatable text description for a CPExchange object.	
1) description	A translatable text description.	string

Property Container

This specification does not expect to define all the information that an application might need in a profile. Instead, CPExchange defines profile attributes that seem to be commonly used in most application domains. CPExchange also defines a Property class that supports definition of new business specific attributes without requiring the definition of new object classes and their corresponding new XML elements.

The Property class represents a dynamically defined attribute that is associated with some other object. A Property object has a name and a value.

A PropertyContainer abstract class represents an object that has a collection of properties. Every object that supports Distinguishable also is a PropertyContainer, with one exception. The Preferences class does not support PropertyContainer, because the Preferences class performs the same function as a PropertyContainer for Preference properties.

The definition and usage of properties should be described in a document agreed upon by those participating in the interchange of the properties. If a property with a particular name is not understood by an application, the application may still store it, propagate it to subscribers, and present it to users. It would also be possible to apply natural language processing technology to mining of property information if the property names and values are language words and phrases.

User defined property names should follow the naming convention of <inverted organisation domain name><property name>, where the domain name prefix is provided to avoid conflicts in user extensions.

For example, to add a demographic characteristic of “hair color”, you might add the following property XML element to a profile <personDemographics> element:

```
<property name=”com.foo.hairColor”>chestnut</property>
```

The meaning of “hairColor” and potentially a list of possible strings that could be used within the value element would be defined somewhere outside of CPExchange.

PropertyContainer	Collection of properties	
1) properties	Group of properties	
a) property	A property, which is a named value.	Property

Property	A named value pair, where the name is an XML attribute	string

Base

The Base class is introduced to provide a single superclass for inheritance where multiple inheritance is not supported. This is the case with XML Schema types. Therefore, in XML Schema, Base inherits from Distinguishable, and aggregates the attributes of Describable, Timeable, and PropertyContainer, through aggregation in an XML Schema implementation. Other classes descend from Base with implementation (XML Schema) inheritance.

An instance of the Base class may appear in privacy ControlGroup as a template to indicate privacyControl that should apply to all subclasses of Base. However, the profile instances will always be of some concrete subclass of Base.

Base	Base type for most other classes	Distinguishable, Describable, Timeable, PropertyContainer
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OpenEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property

Descriptive Information

This category has classes that hold the basic descriptive information about a person, organisation, or about the roles played by a person or organisation. This includes identifying numbers, names, activities, demographic information plus information such as telephone numbers and addresses used to contact a person or organisation. Information in other profile categories is associated with the basic descriptive information that identifies a person or organisation.

Party Identification

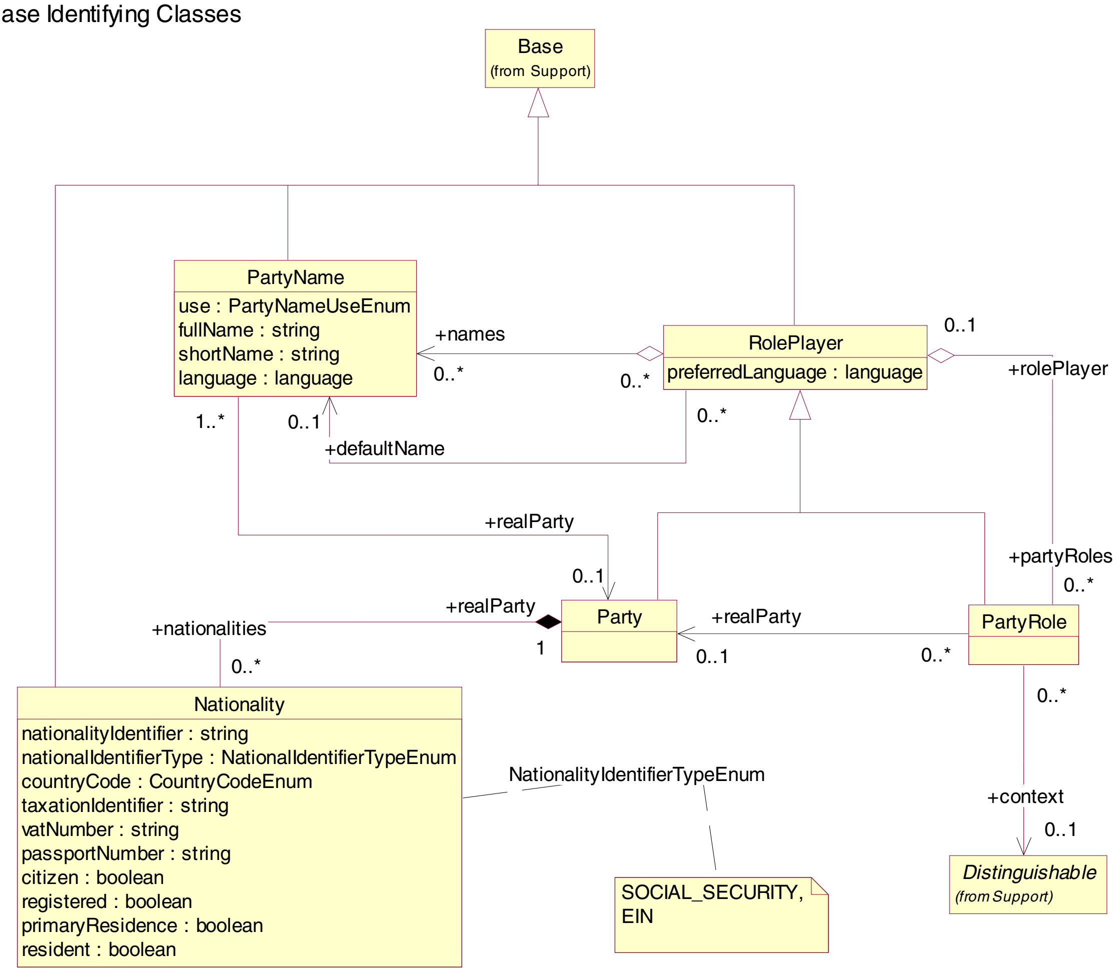
The following diagram shows the relationship between the base classes that identify a person, group of persons, or roles played by a person or group.

A person or organisation is identified by a Party object, or by one of its subclasses Person, or Organisation. A PartyRole object is used to associate CRM information with a role played by a person or organisation. Role information is not affected by the nature of the person or organisation playing that role. Examples of role types are employees, customers, and suppliers. Because both PartyRole and Party objects can have CRM information associated with them, they both inherit from an abstract RolePlayer class. Most relationships in the object model associate other CRM information with a RolePlayer object of any concrete subclass: Person, Organisation, or PartyRole.

Each RolePlayer can have multiple names (PartyName), used for different purposes, and have country specific identification information (Nationality). A person can have a name represented by the more detailed PersonName subclass of PartyName. A PersonName breaks down a name into distinct parts to facilitate querying, sorting, and formatting of the information.

All the classes described in this section can have business specific properties associated with them. Systems should document properties they define or support. Business specific extensions can also be accomplished by defining new subclasses. Generally new subclasses will be added for some specific roles, extensions to the Party and PartyName classes would generally be done with new properties.

Base Identifying Classes



RolePlayer

A RolePlayer object represents all types of people, singular or groups. RolePlayer is also the interface to a role so that the user of customer relationship management information can interact with whatever person or organisation is fulfilling that role at that time.

A RolePlayer object can represent a physical or legal person, a corporation, a group of persons, and any combination of these. In addition, this object can represent any role played by any of these persons, corporations or groups of persons, in the context of other parties, physical objects, events, that are of interest to a company.

RolePlayer is an abstract class providing a common type as the target of associations to other CRM information. The more specialized Person, Organisation, and PartyRole subclasses are used for concrete role-player information objects. These subclasses have additional appropriate attributes.

RolePlayer	Information to uniquely identify a customer or other business partner	Base

1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OpenEnum(see subclasses)
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) preferredLanguage	Default Language preference of role player	language
10) defaultNameId	Reference to PartyName	key
11) nameIds	All names of role player	
a) nameId	Reference to a PartyName	key
12) partyRoleIds	All roles of a role player	
a) partyRoleIds	reference to a party role.	key
13) defaultContactPointIds	Default contact points for this role player	
a) defaultContactPointId	Reference to a ContactPoint	key
14) contactPointUsageIds	All contact point usages for this role player	
a) contactPointUsageId	Reference to a ContactPointUsage	key
15) demographics	Demographic information for role player	PersonDemographics | OrganisationDemographics
16) partyActivities	Collection of Activities of this role player	
a) partyActivity	An activity of this role player	PartyActivity

Party

Party objects represent a common abstract subtype of RolePlayer for physical persons and groups of persons. Party is an abstract class used to associate country specific identifying information with a person or organisation. An example would be a social security number in the United States.

Person and Organisation are concrete subclasses of Party for representing individuals and groups of people.

Party	This specialization of RolePlayer represents a physical person or group of people.	RolePlayer
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OpenEnum(see subclasses)
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) preferredLanguage	Default Language preference of role player	language
10) defaultNameId	Reference to PartyName	key
11) nameIds	All names of role player	
a) nameId	Reference to a PartyName	key
12) partyRoleIds	All roles of a role player	
a) partyRoleIds	reference to a party role.	key
13) defaultContactPointIds	Default contact points for this role player	
a) defaultContactPointId	Reference to a ContactPoint	key
14) contactPointUsageIds	All contact point usages for this role player	
a) contactPointUsageId	Reference to a ContactPointUsage	key
15) demographics	Demographic information for role player	PersonDemographics | OrganisationDemographics
16) partyActivities	Collection of Activities of this role player	
a) partyActivity	An activity of this role player	PartyActivity
17) nationalities	Collection of nationalities of the party	
a) nationality	A nationality of the party	Nationality

Nationality

A Nationality object represents the nationality of a Party, including national and taxation identifying information, and citizenship status. A Party can have multiple nationalities.

Nationality objects appear as **sub-objects** of the associated Party object.

NationalityIdentifierTypeEnum	The enumerated values for the nationalIdentifierType attribute in Nationality objects.	OpenEnum (SOCIAL_SECURITY, EIN)
CountryCodeEnum	The enumerated values for the countryCode attribute in Nationality and also in PostalAddress objects.	ClosedEnum(2-char ISO country codes)
Nationality	Country specific information of a Party, including identifiers.	
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OpenEnum/not used
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) countryCode	Country of citizenship	CountryCodeEnum
10) nationalIdentifier	Social security or other identifying string	string
11) nationalIdentifierType	The type of the national identifier.	NationalIdentifierTypeEnum
12) taxationIdentifier	Taxation identification number	string
13) vatNumber	Value added tax number	string
14) passportNumber	Passport number	string
15) citizen	Indication of citizenship	boolean
16) registered	Indication of registration in country	boolean
17) primaryResidence	Whether this is country of primary residence	boolean
18) resident	Indication of residence in country	boolean
19) realPartyId	Reference to associated Party	key

PartyRole

PartyRole objects represent the role a party plays in a particular context. This class is a significant component of the model when it comes to understanding the relationship between a company and the parties it comes in contact with. PartyRole is a concrete class, but also is an extension point for creating subclasses for specialized roles that require additional information.

As an example, employee, customer, trustee, beneficiary, owner of and so on, are all roles a party can play in the context of a company such as a bank or insurance firm.

PartyRole objects are also used to represent relationships between two role players. For this purpose, one role player is represented by the rolePlayer attribute, and the other by the context attribute. PartyRole objects can be used in this way to represent organisational hierarchies and other object graphs.

PartyRoleTypeEnum	The enumerated values for the typeName attribute of a PartyRole object.	OpenEnum(MEMBER, SUBORGANISATION, PARENT_ORGANISATION, EMPLOYEE, OWNER, CONTACT, CUSTOMER, SUPPLIER, PARTNER, ORGANISATION_HEAD)
PartyRole	This specialization of RolePlayer represents a role or relationship.	RolePlayer
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	PartyRoleTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) preferredLanguage	Default Language preference of role player	language
10) defaultNameId	Reference to PartyName	key
11) nameIds	All names of role player	
a) nameId	Reference to a PartyName	key
12) partyRoleIds	All roles of a role player	
a) partyRoleIds	reference to a party role.	key

13) defaultContactPointIds	Default contact points for this role player	
a) defaultContactPointId	Reference to a ContactPoint	key
14) contactPointUsageIds	All contact point usages for this role player	
a) contactPointUsageId	Reference to a ContactPointUsage	key
15) demographics	Demographic information for role player	PersonDemographics | OrganisationDemographics
16) partyActivities	Collection of Activities of this role player	
a) partyActivity	An activity of this role player	PartyActivity
17) rolePlayerId	RolePlayer playing this role	key
18) contextId	An associated Distinguishable object	key
19) realPartyId	Party at end of role-rolePlayer chain	key

PartyName

A PartyName represents a name of a person or organisation used by a particular RolePlayer. PartyName is an abstract class for more specialized name objects.

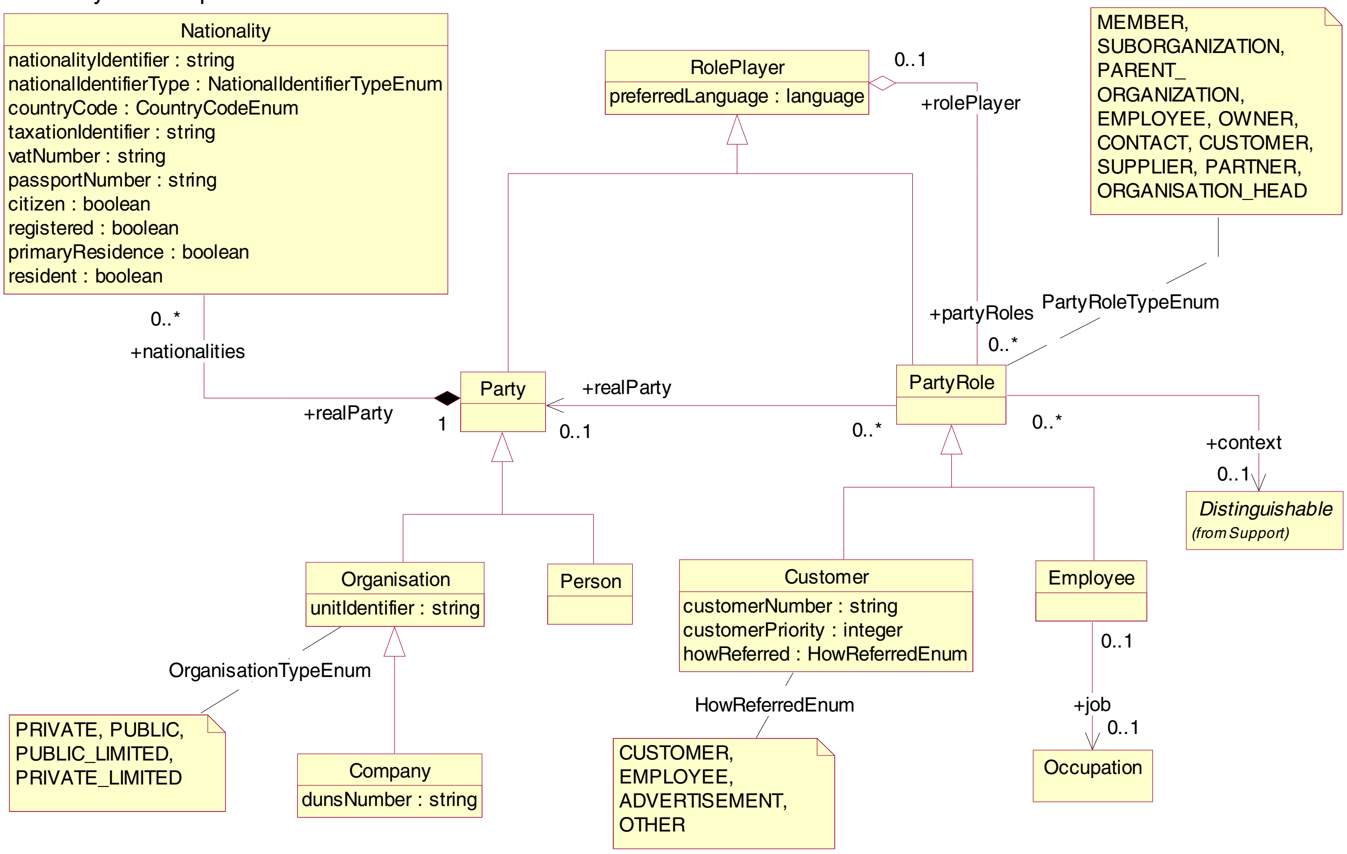
Any RolePlayer object may have one or more names associated with it, generally used in different situations. For example, a person might use a legal name for contracts that is different from the name they use in informal conversation.

A PartyName can also be used by multiple RolePlayer objects, but may be related to only one Party object. For example, when a person plays many roles, and wants to use the same name in all of these roles, this must be possible. However, this name remains specific to this single person, and should never be used by another person or organisation.

Element	**Description**	**Datatype or allowed values**
PartyNameUseEnum	Valid values for PartyName “use” sub element	OpenEnum(DEFAULT, LEGAL, BUSINESS, INFORMAL)
PartyName	Name of a person or Organisation	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	A further classification for this object within its class	OpenEnum/not used
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant

Element	Description	Datatype or allowed values
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) use	usage of name	PartyNameUseEnum
10) fullName	Full name	string
11) shortName	A shortened form of the name.	string
12) language	The language of the name.	language
13) realPartyId	The oid of the Party having this name.	key

Role Player Descriptive Information


Nationality
nationalityIdentifier : string
nationalIdentifierType : NationalIdentifierTypeEnum
countryCode : CountryCodeEnum
taxationIdentifier : string
vatNumber : string
passportNumber : string
citizen : boolean
registered : boolean
primaryResidence : boolean
resident : boolean
0..*
+nationalities
+realParty
1
RolePlayer
preferredLanguage : language
0..1
+rolePlayer
MEMBER, SUBORGANIZATION, PARENT_ ORGANIZATION, EMPLOYEE, OWNER, CONTACT, CUSTOMER, SUPPLIER, PARTNER, ORGANISATION_HEAD
PartyRoleTypeEnum
+partyRoles
0..*
Party
+realParty
0..1
0..*
PartyRole
0..*
+context
0..1
Distinguishable
(from Support)
Organisation
unitIdentifier : string
Person
Customer
customerNumber : string
customerPriority : integer
howReferred : HowReferredEnum
Employee
0..1
+job
0..1
Occupation
OrganisationTypeEnum
PRIVATE, PUBLIC, PUBLIC_LIMITED, PRIVATE_LIMITED
Company
dunsNumber : string
HowReferredEnum
CUSTOMER, EMPLOYEE, ADVERTISEMENT, OTHER

Person

Person objects represent human beings. Person is a concrete subclass of Party.

The Person class currently holds no predefined attributes that are unique from Party. However, it can have properties associated with it that are relevant only to people and not organisations. Its provision as a separate class also facilitates operations on people separate from organisations.

Since Person does not add any attributes or associations to Party, its sub-elements are the same as those of Party. However, the demographics element must be of type PersonDemographics and the typeName should not be used. See the table under Party for other details.

Organisation

Organisation objects represent groups of persons, legal persons, corporations, governments, agencies, departments, or any combination of these. Organisation is a concrete subclass of Party.

The Organisation class extends Party with an optional unitIdentifier that identifies an organisation unit within a larger organisation.

OrganisationTypeEnum	Valid values for Organisation "typeName" sub element	OpenEnum(PRIVATE, PUBLIC, PUBLIC_LIMITED, PRIVATE_LIMITED)
Organisation	This specialization of Party represents a group of people.	Party
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OrganisationTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A named value.	string
9) preferredLanguage	Default Language preference of role player	language
10) defaultNameId	Reference to PartyName	key
11) nameIds	All names of role player	
a) nameId	Reference to a PartyName	key
12) partyRoleIds	All roles of a role player	
a) partyRoleIds	reference to a party role.	key

13) defaultContactPointIds	Default contact points for this role player	
a) defaultContactPointId	Reference to a ContactPoint	key
14) contactPointUsageIds	All contact point usages for this role player	
a) contactPointUsageId	Reference to a ContactPointUsage	key
15) demographics	Demographic information for role player	OrganisationDemographics
16) partyActivities	Collection of Activities of this role player	
a) partyActivity	An activity of this role player	PartyActivity
17) nationalities	Collection of nationalities of the party	
a) nationality	A nationality of the party	Nationality
18) unitIdentifier	An identifier of an Organisational unit within a larger Organisation.	string

Company

Company objects represent a commercial organisation doing any kind of business, not only banking or insurance. Company is a concrete subclass of Organisation.

Company	This specialization of Organisation represents an Organisation that is recognized as a legal entity.	Organisation
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OrganisationTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) preferredLanguage	Default Language preference of role player	language
10) defaultNameId	Reference to PartyName	key
11) nameIds	All names of role player	
a) nameId	Reference to a PartyName	key
12) partyRoleIds	All roles of a role player	

a) partyRoleIds	reference to a party role.	key
13) defaultContactPointIds	Default contact points for this role player	
a) defaultContactPointId	Reference to a ContactPoint	key
14) contactPointUsageIds	All contact point usages for this role player	
a) contactPointUsageId	Reference to a ContactPointUsage	key
15) demographics	Demographic information for role player	OrganisationDemographics
16) partyActivities	Collection of Activities of this role player	
a) partyActivity	An activity of this role player	PartyActivity
17) nationalities	Collection of nationalities of the party	
a) nationality	A nationality of the party	Nationality
18) unitIdentifier	An identifier of an Organisational unit within a larger Organisation.	string
19) dunsNumber	The Dun and Bradstreet identifying number of a company.	string

Employee

Employee objects represent the role that a person plays for an employer (typically an Organisation). These objects hold information about an employee’s job responsibility and salary. Employee is a concrete subclass of PartyRole.

Employee	This specialization of PartyRole represents an employee.	PartyRole
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	PartyRoleTypeEnum = EMPLOYEE
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property

9) preferredLanguage	Default Language preference of role player	language
10) defaultNameId	Reference to PartyName	key
11) nameIds	All names of role player	
a) nameId	Reference to a PartyName	key
12) partyRoleIds	All roles of a role player	
a) partyRoleIds	reference to a party role.	key
13) defaultContactPointIds	Default contact points for this role player	
a) defaultContactPointId	Reference to a ContactPoint	key
14) contactPointUsageIds	All contact point usages for this role player	
a) contactPointUsageId	Reference to a ContactPointUsage	key
15) demographics	Demographic information for role player	PersonDemographics | OrganisationDemographics
16) partyActivities	Collection of Activities of this role player	
a) partyActivity	An activity of this role player	PartyActivity
17) rolePlayerId	Role player playing this role	key
18) contextId	Reference to the employer Organisation.	key
19) realPartyId	Party at end of role-rolePlayer chain	key
20) jobId	Reference to an Occupation object that has details about the employee's job.	key

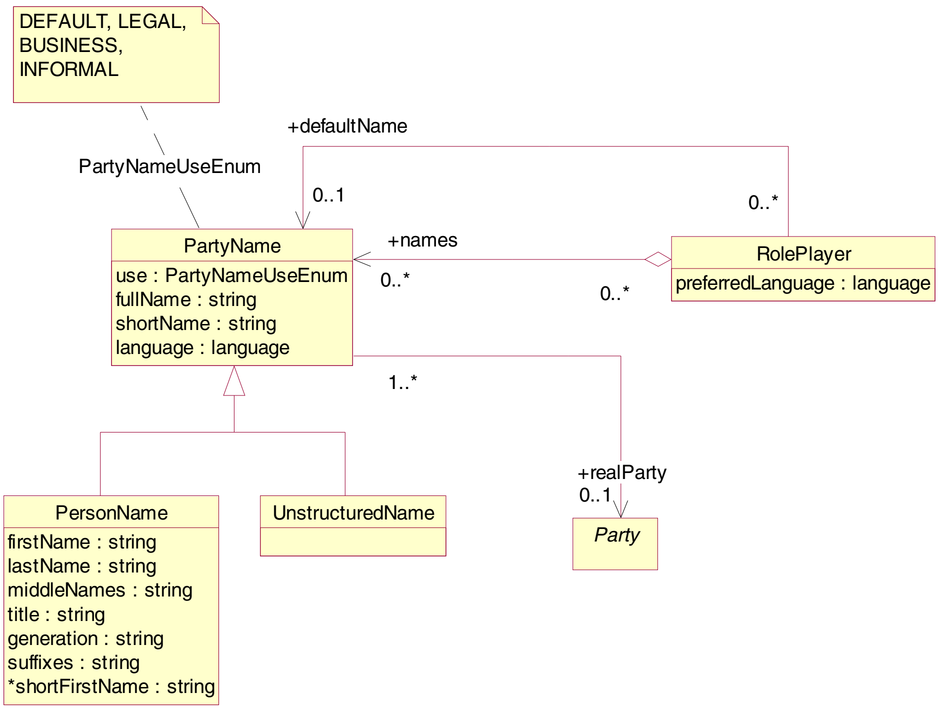
Customer

Customer objects represent the role that a person plays as a customer of a company. This class holds information about the customer including a customer identification number and a priority classification. Customer is a concrete subclass of PartyRole.

HowReferredEnum	The valid values of the howReferred attribute.	OpernEnum (CUSTOMER, EMPLOYEE, ADVERTISEMENT, OTHER)
Customer	This specialization of PartyRole representing a customer.	PartyRole
1) oid	A key identifying this object data in its source system	key

2) typeName	Further classification for this object within its class	PartyRoleTypeEnum = CUSTOMER
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) preferredLanguage	Default Language preference of role player	language
10) defaultNameId	Reference to PartyName	key
11) nameIds	All names of role player	
a) nameId	Reference to a PartyName	key
12) partyRoleIds	All roles of a role player	
a) partyRoleIds	reference to a party role.	key
13) defaultContactPointIds	Default contact points for this role player	
a) defaultContactPointId	Reference to a ContactPoint	key
14) contactPointUsageIds	All contact point usages for this role player	
a) contactPointUsageId	Reference to a ContactPointUsage	key
15) demographics	Demographic information for role player	PersonDemographics | OrganisationDemographics
16) partyActivities	Collection of Activities of this role player	
a) partyActivity	An activity of this role player	PartyActivity
17) rolePlayerId	Role player playing this role	key
18) contextId	An associated distinguishable object	key
19) realPartyId	Party at end of role-rolePlayer chain	key
20) customerNumber	A number identifying this customer within the context Organisation.	string
21) customerPriority	A prioritization of the value of this customer to the context Organisation	positive-integer between 1 (greatest value) and 10.
22) howReferred	Indicates how the customer was referred to the Organisation.	HowReferredEnum

Party Name Descriptive Information



PersonName

PersonName objects represent a name structure used to identify persons. PersonName is a concrete subclass of PartyName.

A PersonName extends PartyName to break a name down into commonly designated parts, like title, first name, middle name, etc. This facilitates more efficient sorting and searching on name information.

Element	**Description**	**Datatype or allowed values**
PersonName	Name of a person.	PartyName
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OpenEnum/not used
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string

Element	Description	Datatype or allowed values
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) use	usage of name	PartyNameUseEnum
10) fullName	Full name	string
11) shortName	A shortened form of the name	string
12) language	The language of the name.	language
13) realPartyId	The oid of the Party having this name.	key
14) firstName	First name.	string
15) middleNames	One or more middle names.	string
16) lastName	Last name	string
17) title	Title used at beginning of name.	string
18) generation	Number following last name that indicates generation using same name.	string
19) suffixes	Any additional information, except generation, used after last name.	string
20) shortFirstName	Shortened form of first name.	string

UnstructuredName

An UnstructuredName is a concrete subclass of PartyName that represents names that cannot be broken down into any common structure. This may be useful for Company and Organisation names, and may be used in other cases where the structure of a PersonName is not needed.

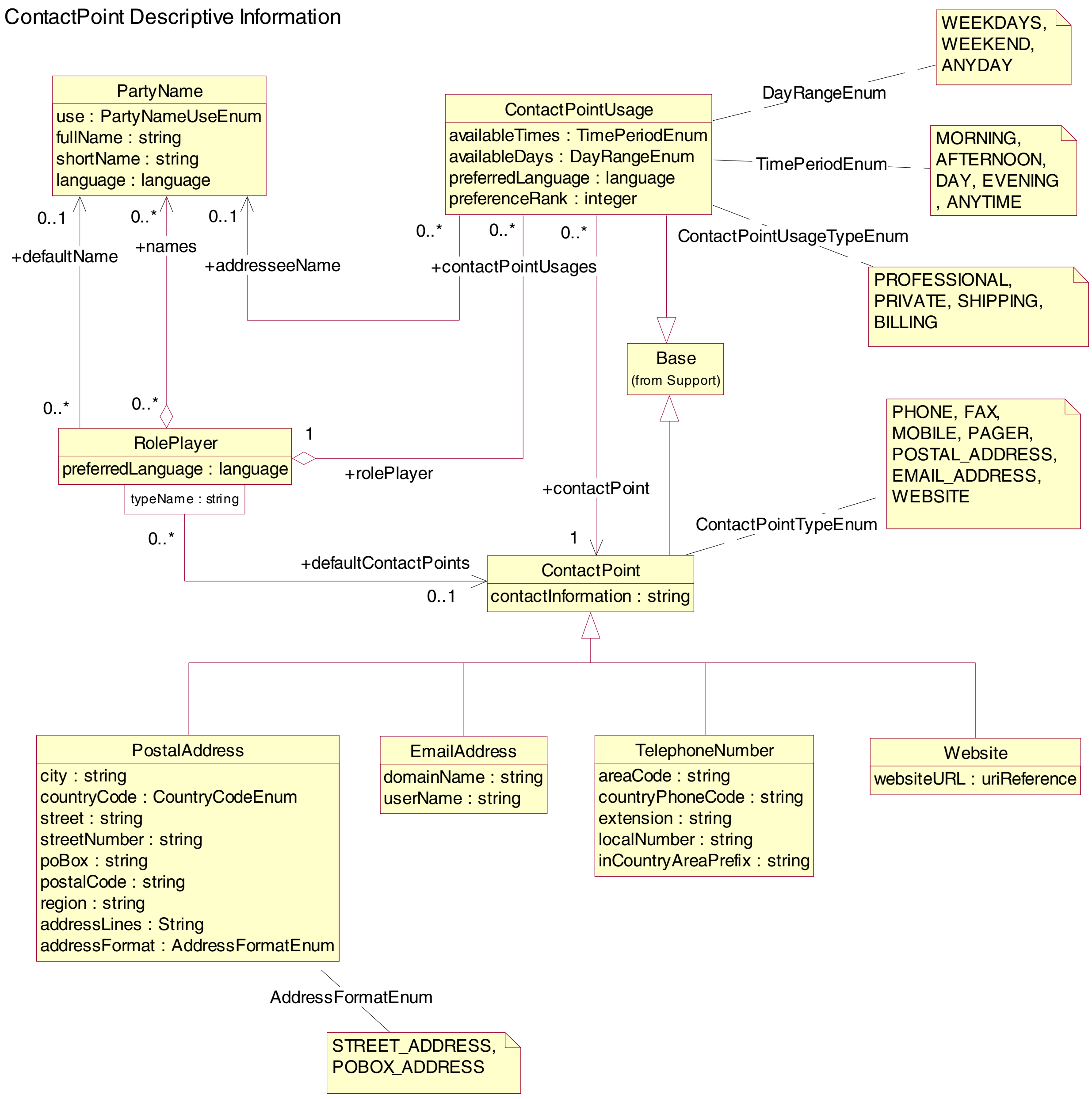
An UnstructuredName does not add any attributes or associations to its PartyName superclass. Therefore, its sub-elements are identical to those for a PartyName. See the PartyName table for details.

Contact Information

Contact information holds the data enabling contact with a RolePlayer, whether a person, an organisation, or a role. ContactPoint is a class defining common information for all forms of contact. PostalAddress, TelephoneNumber, EmailAddress, and Website are subclasses of ContactPoint. These specializations of

ContactPoint break down contact information into component parts to facilitate rich query and sorting algorithms. A RolePlayer may have many ContactPoint objects associated with it.

A ContactPoint is associated with one or more ContactPointUsage objects that describe what types of interactions should be initiated through that contact point, and when the contact point is usable. This allows the appropriate contact point information to be used for a particular purpose and at a particular time. Current, past, and even future contact information can be maintained. The ContactPointUsage can also be associated with a PartyName. This is useful when the RolePlayer has multiple names, and some are appropriately used in different interaction situations.



ContactPoint

A ContactPoint represents information used to contact any person, organisation or role, either physically or electronically. ContactPoints are valid for use within a specified time period.

In most applications, ContactPoint will be used as an abstract class, where concrete subclasses of ContactPoint hold the type specific structured information used to make the contact, like a telephone number, postal mail address, etc.

However, it is possible to use ContactPoint as a concrete class, since it does have an unstructured form of the contact information. In that case the **typeName** attribute can be used to identify the type of contact information and properties can be added to hold the appropriate information.

A ContactPointUsage object associates a ContactPoint with a RolePlayer. Therefore, a ContactPoint object can be associated with multiple RolePlayer objects through a ContactPointUsage. A single RolePlayer may also have multiple ContactPoint objects. The ContactPointUsage object has information specific to the use a particular ContactPoint with a particular RolePlayer such as the purpose of the contact information, appropriate PartyName to use in the contact, and effective times when the contact point should be used to reach the role player.

Element	**Description**	**Datatype or allowed values**
ContactPointTypeEnum	Enumerated values for typeName attribute of a ContactPoint.	OpenEnum (PHONE, FAX, MOBILE, PAGER, POSTAL_ADDRESS, EMAIL_ADDRESS, WEBSITE)
ContactPoint	Information for contacting a role player.	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ContactPointTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) contactInformation	A string containing the full contact information, not broken down into parts.	string

ContactPointUsage

ContactPointUsage objects represent the characteristics of the intended use of a ContactPoint for communicating with a particular RolePlayer. The ContactPointUsage object associates a single ContactPoint to a single RolePlayer.

The ContactPointUsage holds information about the purpose for which the contact point should be used, such as to conduct business or for shipping. The ContactPointUsage class also holds information about the times when the contact point should be used, such as weekdays or evenings. A PartyName can be associated with a ContactPointUsage to indicate which of multiple names is most appropriate for a particular communication. Other attributes include a preferred language and a preference ranking for this usage. When more than one ContactPointUsage object applies, the preference ranking can be used to choose the preferred method of contact.

ContactPointUsageEnum	The enumerated values for the typeName attribute of a ContactPointUsage object.	OpenEnum (PROFESSIONAL, PRIVATE, SHIPPING, BILLING)
TimePeriodEnum	Valid values for "availableTimes" attribute.	ClosedEnum (MORNING, AFTERNOON, EVENING, DAY, ANYTIME)
DayRangeEnum	Valid values for "availableDays" attribute.	ClosedEnum (WEEKDAYS, WEEKENDS, ANYDAY)
ContactPointUsage	How a contact point should be used to contact a RolePlayer.	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ContactPointUsageTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	

a) property	A property, which is a named value.	Property
9) contactPointId	The contactPoint for which this contact point usage applies	key
10) rolePlayerId	The rolePlayer to which this contact point usage applies	key
11) addresseeNameId	The name to be used in the contact	key
12) availableTimes	the time-of-day period(s) during which a contact can be made.	TimePeriodEnum
13) availableDays	the day-of-week type during which the contact can be made	DayRangeEnum
14) preferredLanguage	Language to use in contact	language
15) preferenceRank	Ranking of this usage with respect to other available usages. An integer from 1 to 10, with 1 the most preferred.	positive-integer

TelephoneNumber

TelephoneNumber objects represent a communication point as defined by the telecommunication services. TelephoneNumber extends ContactPoint to provide a structure of the commonly designated parts of a phone number. This allows the correct number to be dialed based on the location of the caller, for example, without the country code prefix.

Element	**Description**	**Datatype or allowed values**
TelephoneNumber	A telephone number.	ContactPoint
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ContactPointTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property

Element	Description	Datatype or allowed values
9) contactInformation	A string containing the full contact information, not broken down into parts.	string
10) countryPhoneCode	Country phone code used when dialing from another country	string
11) areaCode	Region or area portion of phone number used when calling from outside that region.	string
12) inCountryAreaPrefix	Prefix to areaCode, used when dialing within the country	string
13) localNumber	Local number, which follows any areaCode dialed.	string
14) extension	An extension number.	string

PostalAddress

PostalAddress objects represent a communication point as defined by the postal services. PostalAddress extends ContactPoint to provide a structure of the commonly designated parts of a postal mail address. This facilitates more efficient sorting and searching on various parts of the address information.

Element	Description	Datatype or allowed values
CountryCodeEnum	Enumerated values for countryCode attribute of PostalAddress	ClosedEnum (2-char ISO country codes)
AddressFormatEnum	Enumerated values for addressFormat attribute.	OpenEnum (STREET_ADDRESS, POBOX_ADDRESS)
PostalAddress	Information for contacting a role player.	ContactPoint
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ContactPointTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property

Element	Description	Datatype or allowed values
9) contactInformation	A string containing the full contact information, not broken down into parts.	string
10) addressFormat	Specifies the intepretation of the addressLines	AddressFormatEnum
11) addressLines	Additional address lines, preceeding any street number or PO Box	string
12) streetNumber	Street number	string
13) street	Street name	string
14) poBox	Post office box number	string
15) city	City, town, village, hamlet or other municipality	string
16) region	State, province, or region	string
17) countryCode	Country specified by 2 char ISO code	CountryCodeEnum
18) postalCode	Zip code or postal code	string

EmailAddress

EmailAddress objects represent an electronic communication point used for routing e-mail to a particular party. EmailAddress extends ContactPoint to provide a structure of the commonly designated parts of an electronic mail address. This facilitates more efficient sorting and searching on various parts of the e-mail address information.

Element	Description	Datatype or allowed values
EmailAddress	Detailed electronic mail contact information.	ContactPoint
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ContactPointTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	

Element	Description	Datatype or allowed values
a) property	A property, which is a named value.	Property
9) contactInformation	A string containing the full contact information, not broken down into parts.	string
10) domainName	The domain portion of an email address	string
11) userName	The user name portion of an email address.	string

Website

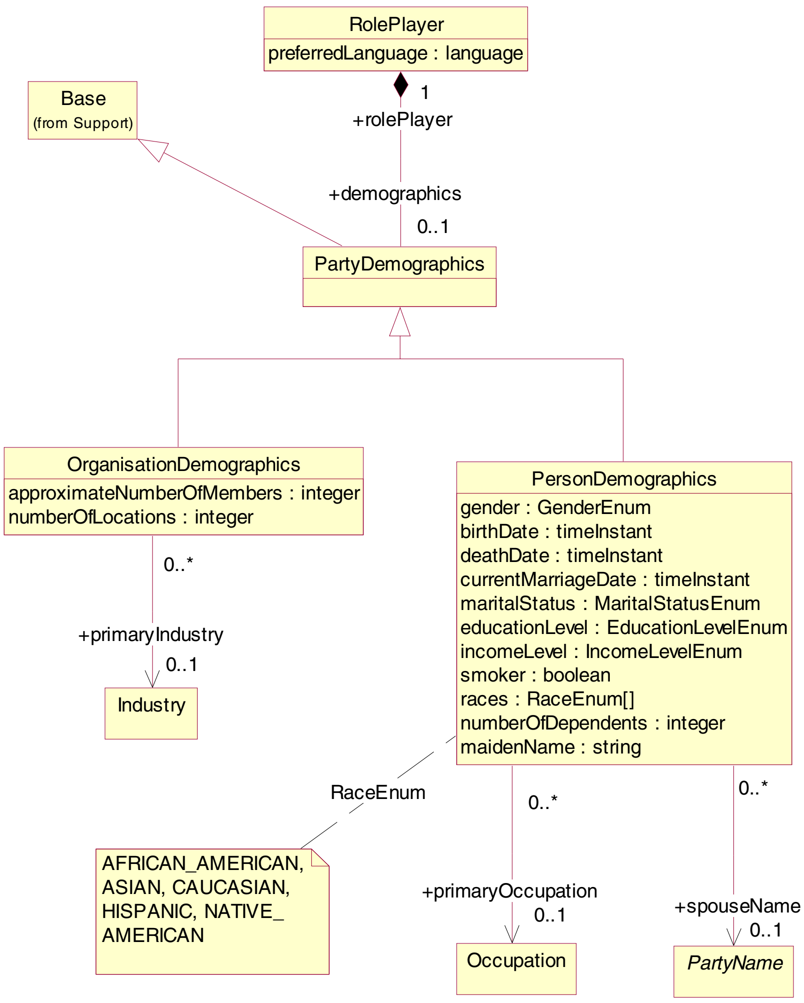
Website represents a party's Internet web site address. Website extends ContactPoint to provide the URL of a web site.

Element	Description	Datatype or allowed values
Website	Detailed website contact information	ContactPoint
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ContactPointTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) contactInformation	A string containing the full contact information, not broken down into parts.	string
10) websiteURL	The url of the website.	uriReference

Demographics

Demographic information contains general characteristics of a person that are widely applicable to business processes, personalization, and market targeting. More complex information on activities and life events are defined in additional classes in the model.

Demographics Descriptive Information



PartyDemographics

The PartyDemographics class represents demographic information that is useful on all types of RolePlayer objects. Currently this is an abstract class that simply establishes the relationship between a RolePlayer and one of the concrete subclasses of PartyDemographics: PersonDemographics or OrganisationDemographics. Generally, demographics will be associated with a Party subclass (Person or Organisation) rather than with a PartyRole.

PersonDemographics

The PersonDemographics class subclasses PartyDemographics to add information about characteristics particular to a person. A fairly large number of attributes are defined. Examples are gender, birthdate, number of dependents, and income level.

IncomeLevelEnum	List of values used to specify a Person’s income level	OpenEnum (LOW_INCOME, MIDDLE_INCOME, HIGH_INCOME, VERY_HIGH_INCOME)
EducationLevelEnum	List of values used to specify a Person’s achieved level of education	OpenEnum (ELEMENTARY_GRADUATE, SECONDARY_SCHOOL_DEGREE, UNIVERSITY_DEGREE, POST_GRADUATE_DEGREE, ASSOCIATE_DEGREE, OTHER, UNKOWN)
GenderEnum	List of values to designate a Person’s gender	ClosedEnum (MALE, FEMALE, UNKNOWN)
MaritalStatusEnum	List of values to designate a Person’s marital status	ClosedEnum (SINGLE, MARRIED, DIVORCED, WIDOWED, UNKNOWN)
RaceEnum	List of values to designate a Person’s race	OpenEnum (AFRICAN_AMERICAN, ASIAN, CAUCASIAN, HISPANIC, NATIVE_AMERICAN)
PersonDemographics	Demographics specific to a Person	PartyDemographics
1) oid	A key identifying this object data in its source system	key
2) typeName	A further classification for this object within its class	OpenEnum/not used
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property

9) rolePlayerId	Reference to the role player that has these demographics	key
10) gender		GenderEnum
11) birthDate		timeInstant
12) deathDate		timeInstant
13) maritalStatus		MaritalStatusEnum
14) currentMarriageDate	Date the Person's current marriage began	timeInstant
15) spouseNameId	Reference to a PartyName for the Person's spouse	key
16) maidenName	Last name of Person prior to marriage	string
17) numberOfDependents	Number of dependents a Person claims to have	integer
18) educationLevel		EducationLevelEnum
19) incomeLevel		IncomeLevelEnum
20) smoker		boolean
21) races	Collection of races to which the Person belongs	
a) race		RaceEnum
22) primaryOccupationId	Reference to an Occupation that is considered primary	key

OrganisationDemographics

The OrganisationDemographics class subclasses PartyDemographics to add information about characteristics particular to a group of people. This includes the approximate number of participants in the group, the number of locations, and information about the primary industry sector of this organisation.

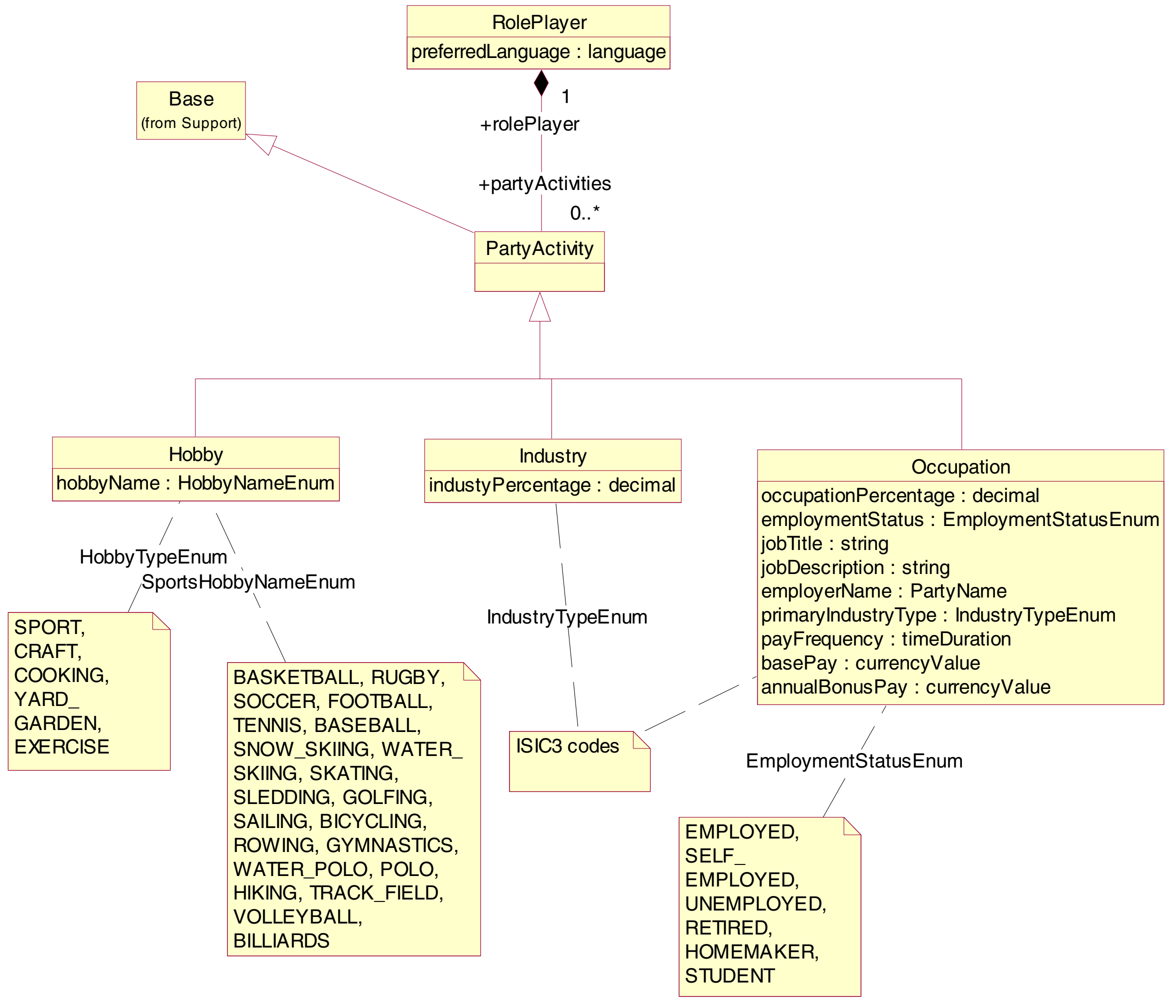
OrganisationDemographics	Demographics specific to an organization	PartyDemographics
1) oid	A key identifying this object data in its source system	key
1) typeName	Further classification for this object within its class	OpenEnum/not used
2) creationDate	Creation timestamp	timeInstant
3) modificationDate	Last modification timestamp	timeInstant
4) description	A translatable text description.	string
5) startDate	A start date and time.	timeInstant

6) endDate	An ending date and time.	timeInstant
7) properties	Group of properties	
a) property	A property, which is a named value.	Property
8) rolePlayerId	Reference to the role player that has these demographics	key
9) approximateNumberOfMembers	Estimated number of members of organization	integer
10) numberOfLocations	Number of physical sites within this organisation	integer
11) primaryIndustryId	Reference to an Industry	key

Party Activities

Information about a party's regular activities can be useful in personalizing interactions with that party, and also in targeting products and marketing to parties that will be most likely to want them. Therefore, activity information is used much like demographic information and preference information. New subclasses of the base PartyActivity class can be created to represent activities that require additional attributes. Alternatively, properties can be added to PartyActivity objects. Both techniques will be used, as applications and systems extend the core information model to more specific business requirements.

Activity Descriptive Information


RolePlayer
preferredLanguage : language
Base
(from Support)
1
+rolePlayer
+partyActivities
0..*
PartyActivity
Hobby
hobbyName : HobbyNameEnum
Industry
industyPercentage : decimal
Occupation
occupationPercentage : decimal
employmentStatus : EmploymentStatusEnum
jobTitle : string
jobDescription : string
employerName : PartyName
primaryIndustryType : IndustryTypeEnum
payFrequency : timeDuration
basePay : currencyValue
annualBonusPay : currencyValue
HobbyTypeEnum
SportsHobbyNameEnum
IndustryTypeEnum
EmploymentStatusEnum
SPORT, CRAFT, COOKING, YARD_ GARDEN, EXERCISE
BASKETBALL, RUGBY, SOCCER, FOOTBALL, TENNIS, BASEBALL, SNOW_SKIING, WATER_ SKIING, SKATING, SLEDDING, GOLFING, SAILING, BICYCLING, ROWING, GYMNASTICS, WATER_POLO, POLO, HIKING, TRACK_FIELD, VOLLEYBALL, BILLIARDS
ISIC3 codes
EMPLOYED, SELF_ EMPLOYED, UNEMPLOYED, RETIRED, HOMEMAKER, STUDENT

PartyActivity

The PartyActivity objects represent activities in which a party participates, or has participated in the past. The PartyActivity class supports the Base interface, so it has a start and stop date. Subclasses of PartyActivity provide more specific information about hobbies, occupations, and industries. The typeName of an Activity, inherited from the class Distinguishable, is used to further describe or name the activity. Each subclass of a PartyActivity would have its own set of valid typeName values.

Hobby

A Hobby object represents an activity that would be considered a hobby of a RolePlayer. Hobbies would include sports, crafts, cooking, gardening, or creative arts in which a person participates. Valid values of the typeName attribute of a Hobby are listed in the HobbyTypeEnum. The hobbyName attribute can be used to further classify the hobby within its typeName value. An example is given for hobbies of typeName **SPORTS**. Valid values of the hobbyName attribute are listed in the corresponding **<hobbyTypeName>HobbyNameEnum**.

Element	Description	Datatype or allowed values
HobbyTypeEnum	Valid values for the typeName attriute of a Hobby object.	OpenEnum (SPORTS, CRAFT, COOKING, YARD_GARDEN, EXERCISE)
HobbyNameEnum	Further classification of a HobbyTypeEnum value.	OpenEnum(see subclasses <hobby type name>HobbyNameEnum
SportsHobbyNameEnum	Valid values for the hobbyName attribute of a Hobby object with a typeName of SPORTS.	OpenEnum (BASKETBALL, RUGBY, SOCCER, FOOTBALL, TENNIS, BASEBALL, SNOW_SKIING, WATER_SKIING, SKATING, SLEDDING, GOLFING, SAILING, BICYCLING, ROWING, GYMNASTICS, WATER_POLO, POLO, HIKING, TRACK_FIELD, VOLLEYBALL, BILLIARDS)
Hobby	Describes a role player's participation in a hobby	PartyActivity
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	HobbyTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) rolePlayerId	Reference to role player involved in this activity	key
10) hobbyName	The particular hobby referred to by this element.	HobbyNameEnum

Occupation

An Occupation object represents the activity of a RolePlayer working at some job. The Occupation class includes job title, employer name, retirement status, and the percentage of the role player's total work time spent in this occupation. The basePay attribute indicates payment received periodically from this occupation, and the payFrequency attribute specifies how often this payment is received. The valid values for the typeName attribute are listed in the OccupationTypeEnum.

OccupationTypeEnum	Valid values for the typeName attribute of an Occupation object.	OpenEnum (…)
EmploymentStatusEnum	Valid values for the employmentStatus attributet.	ClosedEnum (EMPLOYED, SELF_EMPLOYED, UNEMPLOYED, RETIRED, STUDENT, HOMEMAKER)
IndustryTypeEnum	The ISIC3 code for the primaryIndustryType attribute.	ClosedEnum (Any ISIC3 code)
Occupation	This specialization of PartyActivity represents an occupational role which the party fulfills.	PartyActivity
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OccupationTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) rolePlayerId	Reference to role player involved in this activity	key
10) occupationPercentage	The percentage of time that the RolePlayer spends involved with this occupation.	decimal
11) employmentStatus	The status of the occupation	EmploymentStatusEnum
12) jobTitle	Title of the occupation.	string
13) jobDescription	Description of the occupation.	string
14) employerNameId	Reference to PartyName of the employer.	PartyName
15) primaryIndustryType	Industry in which the occupation acts.	IndustryTypeEnum
16) payFrequency	Time frequency at which the periodic payments are received.	timeDuration
17) basePay	Amount of periodic payments for this occupation	currencyValue
18) annualBonusPay	Any bonus payments within a year.	currencyValue

Industry

An Industry object represents the business industries in which the RolePlayer is active, and the percentage of a business' revenue that comes from that industry. Normally an Industry object is associated with an Organisation rather than a Person or PartyRole. The valid typeName values are listed in the IndustryTypeEnum. These match the ISIC3 standardized codes.

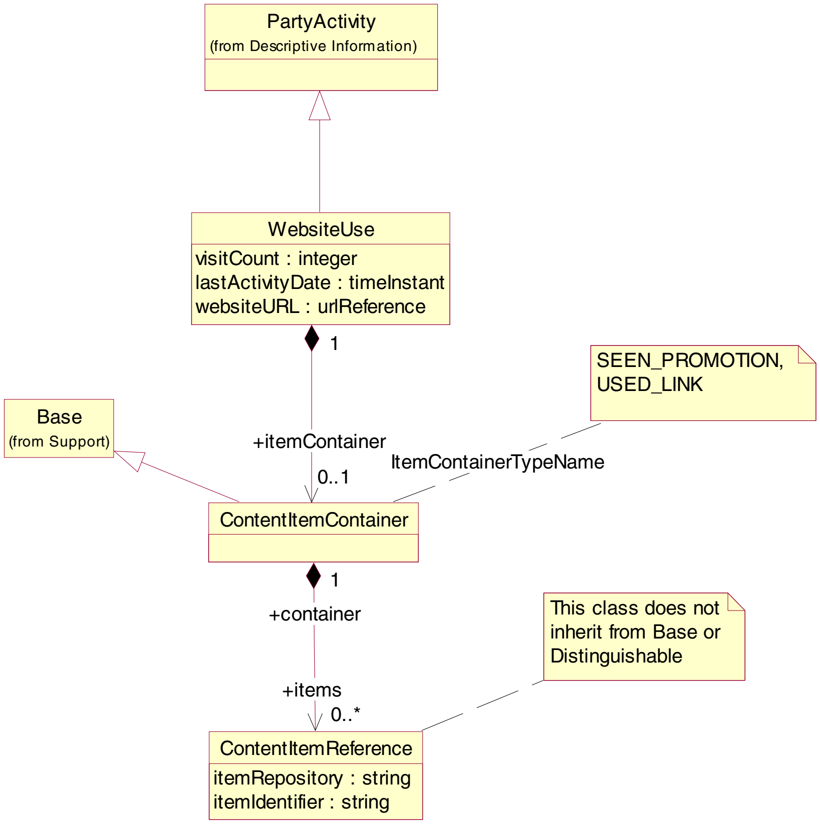
IndustryTypeEnum	The ISIC3 code for the primaryIndustryType attribute.	ClosedEnum (Any ISIC3 code)
Industry	Represents a rolePlayer's participation in business within a particular industry.	PartyActivity
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	IndustryTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) rolePlayerId	Reference to role player involved in this activity	key
10) industryPercentage	Percentage of a the rolePlayer's revenue that comes from this industry	decimal

Web Site Use

Web site use contains summary information related to what a user has done in a web session. Detailed actions of the user are tracked in either the web logs or in ActionEvent objects. ActionEvent objects are described in the following Interaction History section.

The WebSiteUse summary information includes the number of times the user has visited the site, the date of the user's last session as well as the most frequently used list of various web content items that the company wants to track. These frequently used lists can be used for personalization of the website, to make it more productive for a particular user.

Web Descriptive Information

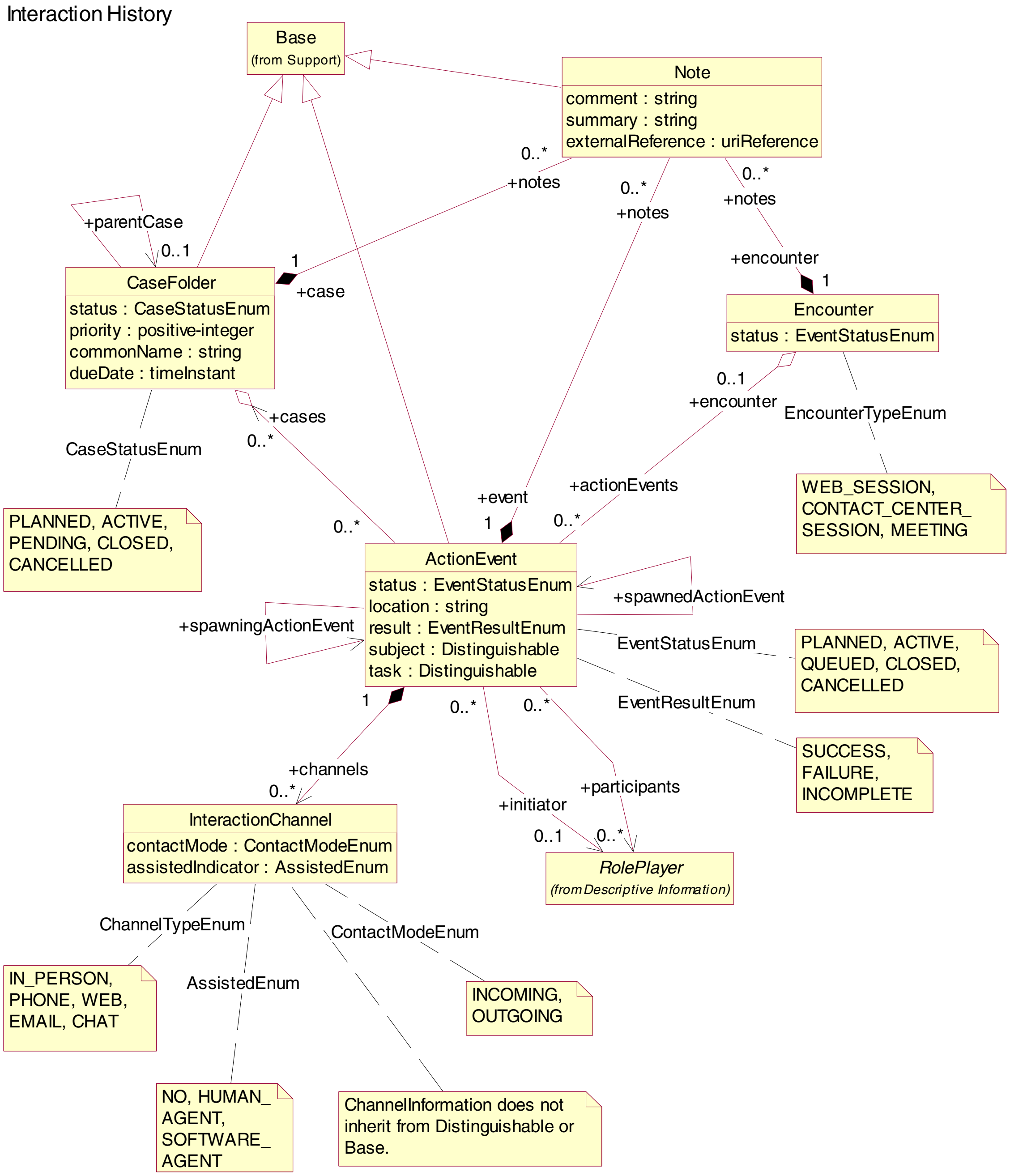


Element	**Description**	**Datatype or allowed values**
WebsiteUse	Tracks the use of a website by a user	PartyActivity
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ContactPointTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) rolePlayerId	Reference to role player involved in this activity	key
10) websiteURL	The uri of the website	uriReference

Element	Description	Datatype or allowed values
11) visitCount	the number of times the site/page has been visited	integer
12) lastActivityDate	the date/time of the last visit	timeinstant
ItemContainerTypeEnum	Valid values for the typeName attribute of a ContentItemContainer object.	OpenEnum (SEEN_PROMOTION ,USED_LINK)
ContentItemContainer		Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ItemContainerTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) items	Collection of content item references	
a) item	Reference to a content item	ContentItemReference
ContentItemReference	Information needed to reference a content item	
1) itemRepository	Identifies the repository of the content item	string
2) itemIdentifier	Identifier of the content item within the repository	string

Interaction History Information

Interaction history information records the history of actions taken by any party or organisation. There are three main data objects: **action event**, **encounter**, and **case folder**. The action event tracks a single action that occurs at one particular time for some duration. An encounter tracks a series of actions that occur within the same session. That session could be a web site session, a call to a call center, or a meeting. In addition, action events can be associated with other business objects, and in particular can be associated with case folders. A case folder could represent a sales opportunity, an order, a service request, etc. Each of these three main elements is meant to be extensible with more specific information.



ActionEvent

An ActionEvent object represents the smallest unit of activity or transaction provided. This object tracks a single action that occurs at one particular time for some duration. An example is the tracking of an individual business event within a company that represents an action initiated by a person and occurring over a single time period. An ActionEvent is associated with at most one encounter.

An ActionEvent can be in one of several states: created, planned, queued, active, cancelled, or closed. In all states except the **PLANNED** state, an ActionEvent is associated with an Encounter.

An ActionEvent has a single initiator. If an action is part of an Encounter that involves multiple participants, a copy of the ActionEvent will also be associated with each participant, including the initiator. ActionEvents are also associated with an Encounter, and any number of other objects such as a CaseFolder.

An ActionEvent may cause another action to be initiated that is associated with a different Encounter. An ActionEvent has a reference to any such subsequent ActionEvent (spawnedEventId) and also to the ActionEvent that spawned it - if there was one. This supports following the thread of events that have different participants and different encounters. An example would be a customer call that started in a support center with a problem request on a personal computer product, and was then followed by transferring the customer to a sales agent to order more memory for the computer.

Element	**Description**	**Datatype or allowed values**
ActionEventTypeEnum	Valid values for the typeName attribute of ActionEvent objects. These may be subclass dependend.	OpenEnum (…)
EventStatusEnum	Valid values for the status attribute of ActionEvent and Encounter objects.	ClosedEnum (PLANNED, ACTIVE, QUEUED, CLOSED, CANCELLED)
EventResultEnum	Valid values for the result attribute of ActionEvents. The values can be subclass specific.	OpenEnum(SUCCESS, FAILURE, INCOMPLETE)
ActionEvent	A single interaction between parties on a single subject for one contiguous time period.	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ActionEventTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant

Element	Description	Datatype or allowed values
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) status	The status of the interaction	EventStatusEnum
10) location	Where the interaction took place	string
11) result	The result of the interaction	EventResultEnum
12) subjectId	Reference to a Distinguishable object that is the subject of the event, often a CaseFolder or a RolePlayer.	key
13) taskId	Reference to a Distinguishable object that is the subject of the event, often a CaseFolder.	key
14) caseIds	Collection of reference to CaseFolders related to this interaction.	
a) caseId	Reference to an CaseFolder for a process related to this event.	key
15) initiatorId	Reference to the RolePlayer that initiated this interaction.	key
16) participantIds	Collection of participants in this event	
a) participantId	Reference to a RolePlayer that took place in this interaction.	key
17) encounterId	Reference to the Encounter that this event is part of.	key
18) spawnedActionEventId	Next event in the same encounter, if any.	key
19) spawningActionEventId	Previous event in the same encounter, if any.	key
20) notes	Collection of notes attached to this event	
a) note	Note attached to this event	Note
21) channels	Collection of channels used in this event	
a) channel	A channel used in this event	InteractionChannel

InteractionChannel

The InteractionChannel represents the usage of a device channel in an action event.

Element	Description	Datatype or allowed values
ChannelTypeEnum	Valid values for the typeName attribute of InteractionChannel objects.	OpenEnum (IN_PERSON, EMAIL, WEB, PHONE, CHAT, UNKNOWN)
ContactModeEnum	Valid values for the contactMode attribute of InteractionChannel objects.	OpenEnum (INCOMING, OUTGOING, UNKNOWN)
AssistedEnum	Valid values for the assistedIndicator attribute of InteractionChannel objects.	OpenEnum (NO, HUMAN_AGENT, SOFTWARE_AGENT, UNKNOWN)
InteractionChannel	Characteristics of a channel used in an interaction	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ActionEventTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) contactMode	Specifies the directional mode of the interaction, with respect to the owner of the associated business process.	ContactModeEnum
10) assistedIndicator	Specifies whether the interaction was assisted, and how.	AssistedEnum

Encounter

An encounter represents a single contact with an enterprise, particularly from the customer’s point of view. The encounter contains information about an interaction between the customer and the organisation. An encounter may hold more detailed contact history information as a collection of individual action events. The encounter data allows exchange of the aggregate information about a session with multiple events.

Examples of encounters inside an enterprise would include a **meeting**, a **customer call**, **receipt of a fax**, an **outgoing e-mail**, or a **web site visit**.

Encounters involving only a single person for a single action would have only one ActionEvent and one participant. Encounters involving a customer and a call center agent for one request would have one ActionEvent and two participants. A meeting Encounter that involved a series of presentations could have an ActionEvent for each presentation and a number of participants.

Element	**Description**	**Datatype or allowed values**
EncounterTypeEnum	Valid values for the typeName attribute of an Encounter object.	OpenEnum(WEB_SESSION, CALL_CENTER_SESSION, MEETING)
Encounter	Represents a physical connection between interacting parties, and tracks the action events that occur during that connection.	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	EncounterTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) status		EventStatusEnum
10) actionEventIds	Collection of references to action events that occurred within this encounter	
a) actionEventId	Reference to ActionEvent that was part of this encounter	key
11) notes	Collection of notes attached to this encounter	
a) note	Note attached to this encounter	Note

CaseFolder

A CaseFolder is a thread of a sustained dialog with a particular Party. The CaseFolder object tracks the activity related to a business process. This grouping of action events is entirely dependent on the business model of the company setting up the case folders. There might be a case folder for every order, every service request, every sales campaign, etc.

A CaseFolder can also represent a business process that involves multiple parties. A Case generally has ActionEvents associated with it for each distinct event in time that is part of the sustained dialogue, or business process, represented by the CaseFolder. Action events can be associated with more than one case folder.

CaseFolder objects have start and end dates, and can have due dates. CaseFolder objects also have a status that indicates the state of a business process in its life cycle such as ACTIVE or CLOSED.

CaseFolder objects can have hierarchical relationships. This means that a CaseFolder can have child cases. For example, a sales campaign case folder might have child CaseFolders for each party that is a target of the campaign. Individual ActionEvents are associated with one or more case folders. Examples of CaseFolder objects are a sales opportunity, an order, a problem report on a product, etc.

Element	**Description**	**Datatype or allowed values**
CaseFolderTypeEnum	Valid values for the typeName attribute of CaseFolder objects. These may be subclass dependend.	OpenEnum (…)
CaseStatusEnum	Valid values for the status attribute of CaseFolder objects.	CloseEnum (PLANNED, ACTIVE, PENDING, CLOSED, CANCELLED)
CaseFolder	Represents a business process or task and tracks relevant interactions between involved parties	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	CaseFolderTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) dueDate	Date on which the processing for this case should be completed.	timeInstant
10) commonName	A short descriptive name for this case.	string
11) status	The status of the processing for this case.	CaseStatusEnum
12) priority	A value from 1(highest) to 10.	positive-integer

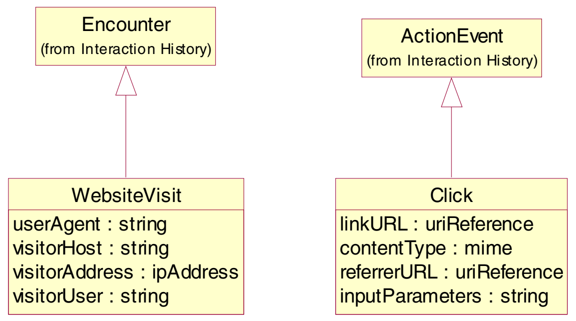
Element	Description	Datatype or allowed values
13) parentCaseId	Reference to a parent case that this is part of, if any	key
14) notes	Collection of attached notes	
a) note		Note

Note

A Note contains text or other media that is attached to some other object, such as an Encounter, ActionEvent or CaseFolder. A Note may only be attached to one of these object, not one of each type. The note can contain both text information and a reference to external data through its URL.

Element	Description	Datatype or allowed values
Note	Text information added to either an event, case, or encounter.	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OpenEnum/not used
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) summary	summary of note contents	string
10) externalReference	URL referencing an external document	uriReference
11) comment	text of note	string
12) caseId	Reference to the CaseFolder object , if any, to which this note is attached	key
13) encounterId	Reference to the Encounter object , if any, to which this note is attached	key
14) actionEventId	Reference to the ActionEvent object , if any, to which this note is attached	key

Web Interaction History



WebSiteVisit

A web site visit is a special type of an encounter that records the interaction of a person at a web site. It is associated with more detailed click information that records the sequence of URLs of pages visited by the user.

Element	Description	Datatype or allowed values
WebSiteVisit	Specialization of Encounter for a web session.	Encounter
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	EncounterTypeEnum = WEB_SESSION
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	A start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) status		EventStatusEnum
10) actionEventIds	Collection of references to action events that occurred within this encounter	

Element	Description	Datatype or allowed values
a) actionEventId	Reference to ActionEvent that was part of this encounter	key
11) notes	Collection of notes attached to this encounter	
a) note	Note attached to this encounter	Note
12) visitorAddress		ipAddress
13) visitorHost		string
14) visitorUser		string
15) userAgent		string

Click

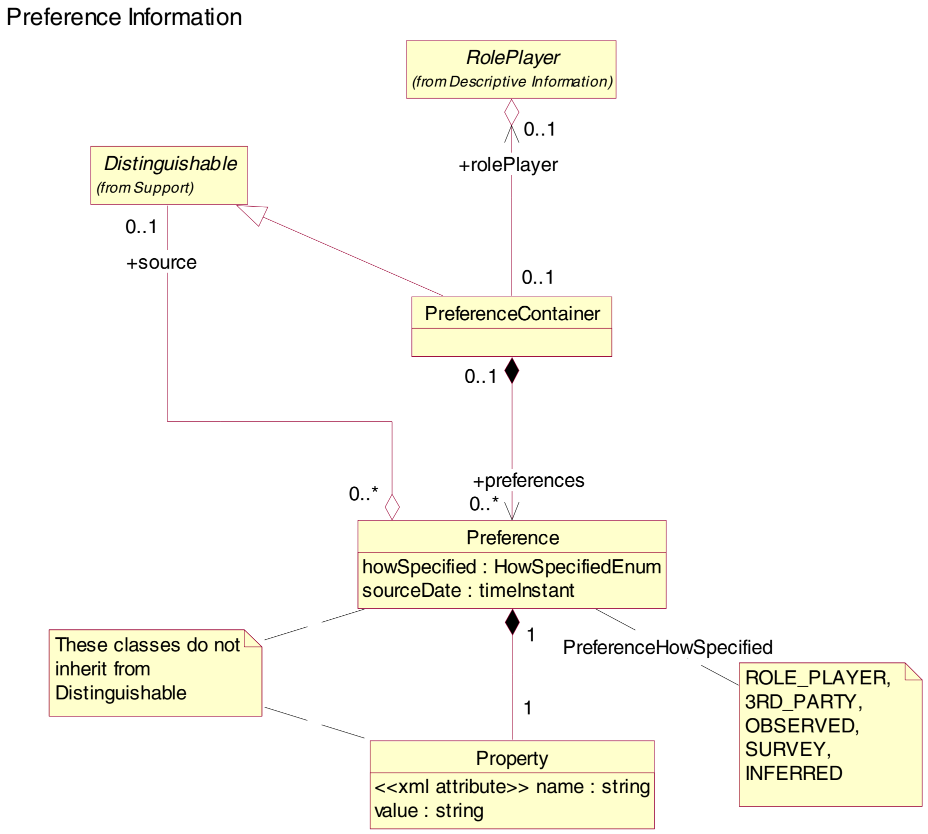
The click information is a special type of action event associated with a user following a hyperlink to a different page on a Website.

Element	Description	Datatype or allowed values
Click	A specialisation of ActionEvent that represents a web "hit".	ActionEvent
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ActionEventTypeEnum = CLICK
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	starting date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) status	The status of the interaction	EventStatusEnum
10) location	Where the interaction took place	string
11) result	The result of the interaction	EventResultEnum

Element	Description	Datatype or allowed values
12) subjectId	Reference to a Distinguishable object that is the subject of the event, often a CaseFolder or a RolePlayer.	key
13) taskId	Reference to a Distinguishable object that is the subject of the event, often a CaseFolder.	key
14) caseIds	Collection of reference to CaseFolders related to this interaction.	
15) caseId	Reference to an CaseFolder for a process related to this event.	key
16) initiatorId	Reference to the RolePlayer that initiated this interaction.	key
a) participantIds	Collection of participants in this event	
17) participantId	Reference to a RolePlayer that took place in this interaction.	key
a) encounterId	Reference to the Encounter that this event is part of.	key
18) spawnedActionEventId	Next event in the same encounter, if any.	key
19) spawningActionEventId	Previous event in the same encounter, if any.	key
20) notes	Collection of notes attached to this event	
a) note	Note attached to this event	Note
21) channels	Collection of channels used in this event	
a) channel	A channel used in this event	InteractionChannel
22) linkUrl	The url of the target page.	uriReference
23) contentType	The content type of the document referenced by the url	mime
24) referrerURL	The url of the page from which the link occured	uriReference
25) inputParameters	The input parameters passed to the url.	string

Preference Information

Preference information captures in a very generic way a set of properties that describe the customer's preferences, either those that have been explicitly provided, or those that are inferred from past behavior. Preferences could be inferred through analysis of CPExchange interaction history information as well as other data sources.



PreferenceGroup

The PreferenceGroup class represents a collection of preferences associated with a particular RolePlayer. Each preference is represented in a Preference object.

Preference

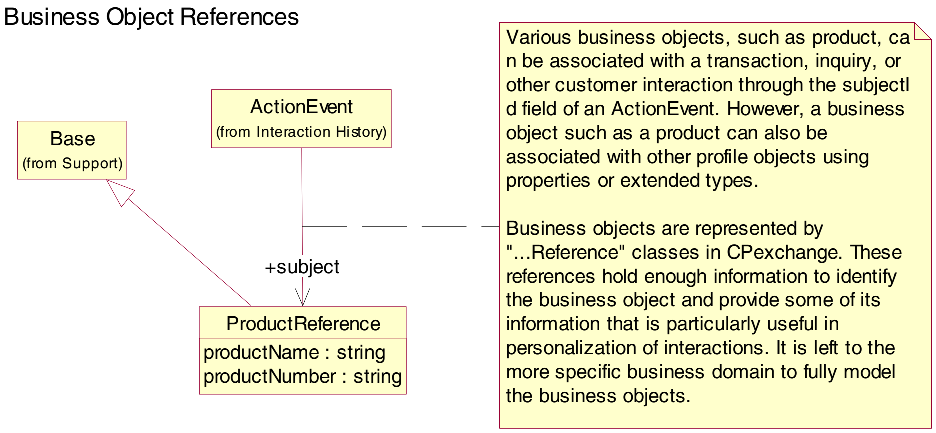
The Preference class combines a property with additional information indicating how the preference was determined and the source of the information used to determine the preference.

Element	Description	Datatype or allowed values
HowSpecifiedEnum	Valid values for howSpeccified attribute of a Preference.	OpenEnum (ROLE_PLAYER, 3RD_PARTY, OBSERVED, SURVEY, INFERRED)

Element	Description	Datatype or allowed values
PreferenceGroup	Collection of a role player’s preferences	Distinguishable
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	OpenEnum/not used
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) rolePlayerId	Reference to role player involved in this activity	key
6) preferences	Collection of preferences	
a) preference	A single preference	Preference
Preference	Information describing a personal preference.	
1) property	A property, which is a named value	Property
2) howSpecified	An indication of how the preference value was determined	HowSpecifiedEnum
3) sourceId	Reference to distinguishable object that is the source of the preference value	key
4) sourceDate	The date the value was specified	timeInstant

Business Object Information

Customer interactions are often associated with business objects that may represent business processes or information used in business processes. It is expected that specific industries will define XML for exchange of process and object information in their domain. This standard defines references to a few business objects that would be commonly associated with customer interactions. These business object references contain only a small amount of information about the particular business object. The information is expected to be enough to identify the more complete business object defined in another system or by a vertical industry, and, in some cases, to contain the most commonly used information about that object.



Various business objects, such as product, can be associated with a transaction, inquiry, or other customer interaction through the subjectId field of an ActionEvent. However, a business object such as a product can also be associated with other profile objects using properties or extended types.

Business objects are represented by "...Reference" classes in CPexchange. These references hold enough information to identify the business object and provide some of its information that is particularly useful in personalization of interactions. It is left to the more specific business domain to fully model the business objects.

ProductReference

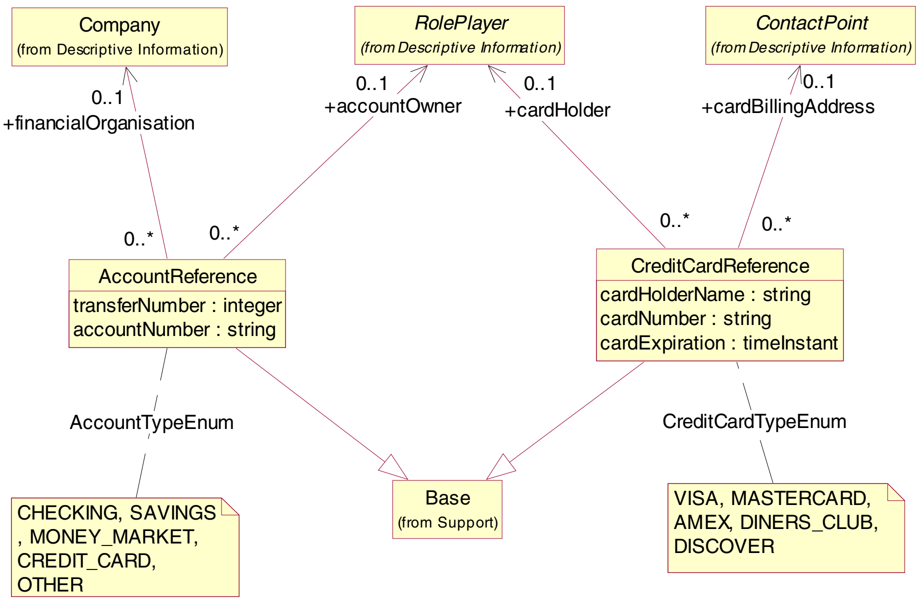
A ProductReference represents a reference to a product of some sort. A ProductReference has a productName that identifes the product generically, such as a model number. A ProductReference also may have a productNumber that identies a particular instance of a product, such as a Stock Keeping Unit (SKU) or serial number.

Element	Description	Datatype or allowed values
ProductTypeEnum	Reserved for future classification of products.	Not used - reserved
UnitTypeEnum	Valid values of product unit types.	OpenEnum (EACH, …)
ProductReference	Basic information to reference a product	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	ProductTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) productName	The name or model of this product.	string
10) productNumber	The productSKU or serial number	string

Payment information

Payment information contains information about accounts with financial institutions and credit cards owned or used by a person, organisation, or role. This could include accounts internal to a company as well as accounts at other institutions.

Payment Business Object Information



AccountReference

The AccountReference class represents identifying information for an account with a financial institution that is owned by one or more parties, or roles. This class does not represent the actual account, which would have much more information including balance and transaction information.

AccountReference objects may represent internal accounts or accounts held at other institutions.

Element	**Description**	**Datatype or allowed values**
AccountTypeEnum	Valid values for the typeName attribute of an AccountReference object..	OpenEnum (CHECKING, SAVINGS, MONEY_MARKET, CREDIT_CARD, OTHER)
AccountReference	Basic information to reference an account	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	AccountTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant

Element	Description	Datatype or allowed values
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) tranferNumber	Bank routing (ABA or T/R) 9 digit number of financial orgainisation	integer
10) accountNumber	Account number within the finanicial organisation	string
11) financialOrganisationId	Reference to the financial Organisation object	key
12) accountOwnerId	Reference to the account owner RolePlayer object	key

CreditCard

CreditCard contains the information about a credit card held by a party or role that is commonly needed to charge a purchase to a credit card.

Element	Description	Datatype or allowed values
CreditCardTypeEnum	Valid values for the typeName attribute of a CreditCard object.	OpenEnum (VISA, MASTER_CARD, DISCOVER, AMEX, DINERS_CLUB)
CreditCardReference	Basic information needed to charge to a credit card	Base
1) oid	A key identifying this object data in its source system	key
2) typeName	Further classification for this object within its class	CreditCardTypeEnum
3) creationDate	Creation timestamp	timeInstant
4) modificationDate	Last modification timestamp	timeInstant
5) description	A translatable text description.	string
6) startDate	start date and time.	timeInstant
7) endDate	An ending date and time.	timeInstant
8) properties	Group of properties	
a) property	A property, which is a named value.	Property
9) cardHolderName	The name on the credit card	string
10) cardNumber	The number on the credit card	string
11) cardExpirationDate	The expiration date of the credit card	timeInstant

Element	Description	Datatype or allowed values
12) cardHolderId	Reference to the RolePlayer that is the card holder	key
13) cardBillingAddressId	Reference to a PostalAddress used for card billing.	key

Appendix A: Future Directions

Commands

The CPExchange commands will describe how enterprise applications request, deliver, and update customer profiles. It will support a model wherein a particular application may be the authority for certain data elements, use data elements from other sources, and be totally uninterested in other data elements. In this model, a complete customer profile may never exist; certain aspects of it will be created dynamically as requested from the authority application. It will also be possible to design an aggregation application that creates a physical repository of customer profiles and manages their caching and refresh.

The CPExchange commands will include a query response model for specifying the customer profiles or data elements to be delivered, their delivery, and their update. A design goal is to be independent of the transport mechanism used to interchange these requests and responses.

The CPExchange commands may contain a methodology for registration (unique identification) of applications within an enterprise (and eventually a trading network), as well as a mechanism to "declare" which aspects of the customer profile an application supports and can update.

Detailed Command Requirements

The following detailed command requirements have been identified by examining the CPExchange use cases:

- **(r0-request/response support)** The standard should support a consumer request for CPExchange data from a supplier, and receipt of the data in a response.

- **(r1-profile subset exchanged)** The standard should support notifying a supplier of what subset of profile data to send, and notifying a consumer of what data will be received.

- **(r2a- value based selection criteria)** The standard should support a consumer's ability to specify which objects should be returned based on selection criteria for matching object attribute values, as is supported for querying databases and directory services.

- **(r2b- class based return criteria)** The standard should allow a consumer to specify a subset of the profile to be returned based on the object class (e.g. XML element names or XML Schema element types corresponding to object class). This is different from r2a, which involves also looking at values of object attributes.

- **(r2c- attribute template return criteria** The standard should allow a consumer to specify a subset of the properties of objects that should be returned. This is in combination with r2b above, which would allow specifying which object types to return, and in combination with r2a which would allow objects with attributes of certain values to be returned. This could be based on a template of the XML element corresponding to an object, and attributes on the sub-elements to indicate whether that sub-element should be returned. For example, this would allow only <postalCode> to be returned in <postalAddress> objects, without returning all the other attributes/sub-elements of the <postalAddress>.

- **(r11- generic query support)** The standard should support a generic query of a subset of profile data based on both object/element type and values would be best - especially for reporting applications and services.

- **(r8- limited query support)** If a full query on CPExchange attribute values cannot be supported, the standard should provide a way to tag attributes as **queryable**. **Queryable** attributes could be supplier defined, or CPExchange defined.

- **(r12- query by id only)** If more generic approaches to sub-setting profile responses (see r2a,r2b,r2c,r11,r8) are not supported, then the standard must provide a way to constrain the response to profile information for a particular user (e.g. Query on at least the key or some attribute of a party object).

- **(r3- publish/subscribe support)** The standard should enable a consumer to subscribe to notifications containing profile updates from a supplier. Part of the subscription information should specify either immediate notification on change, or periodic notification of a batch of changes to the profile information.

- **(r4- specify valid responses)** The standard should provide the ability to specify which responses can result from different requests, including which subset of the CPExchange data is returned.

- **(r5- CPX request with other response)** The standard should enable a message request to use CPExchange even though the response may NOT use CPExchange data.

- **(r10- other request with CPX response)** The standard should allow a request that is not defined by CPExchange to result in a response that could be a CPML XML document.

- **(r6- user defined requests)** The standard should support User-defined requests that encompass CPExchange data. These requests could specify very domain specific operations to be carried out using the CPExchange data.

- **(r7- CPX requests)** The standard should define a set of generic specific request operations. In a **publish** notification or in a response to a request for data, the CPExchange data can imply a **modify** or **create** of that data. At **subscription** time or in a request, **read** or **query** criteria might be supplied in a QBE (Query By Example) template of CPExchange elements. Transaction oriented standards could specify all the basic CRUD operations: create, read, update, and delete.

- **(r9- use of CPX within other XML documents)** The standard should support the use of CPX data elements in non-CPX XML documents, probably through definition of a namespace. For example, anywhere a postal address is needed in an XML document, the CPExchange XML representation of a <postalAddress> should be useable.

The following other requirements, concerns, and approaches have been logged based on discussions:

- **(request supplier schema)** The standard should enable a consumer to find out which CPX objects a supplier provides, and which extensions to CPX the supplier supports.

- **(identify required extensions)** The standard should identify certain extensions as **required**, meaning that a consumer must know how to handle these extensions or the exchange may be error-prone and should not be attempted.

- **(subset agreements)** The requirements to exchange a subset of available profile information could be satisfied in many different ways:

  - this could be determined by agreements between consumers and suppliers outside of this standard - either offline or online,

  - consumers could say what they want on subscription only,

  - consumers could be allowed to say what they want on each individual request,

  - suppliers could say what they provide in response to a request for their "schema" and maybe also provide notifications on schema updates,

  - suppliers could pass back the schema they provide on every response/notification - probably in bitmap form to keep this small,

  - there could be a negotiation on what the consumer wants and the supplier is willing to provide.

- **(optional full query)** The standard should support Query-By-Example based on key values, and make QBE using other attribute values be an optional part of the standard.

- **(partial object return)** The standard should support templates for which attributes of an object should be returned. This can be easily combined with a QBE approach to selecting the objects to return.

- **(schema change support)** The standard should support versioning and controls that enable exchange partners to know the versions of the schema in use.

- **(don't define protocol)** The standard will only define privacy and profile data, and leave the transport protocol completely open to application definition.

- **(4 layer architecture)** The standard should support a 4 layer architecture: message, command, object, and attribute. The message layer may be outside the scope of CPExchange, allowing for the use of many different transport protocols to be used to exchange CPX information. Examples might be BIZTALK, ICE, and ebXML. The command layer needs to allow for non-CPExchange operations, but might also include some basic operations that all CPExchange implementers support such as **modify/create** and **read/query**. The object and attributes layers are part of the UML model of the profile information. While CPExchange operations would use the object layer (which contains the attributes), other XML documents might find it useful to use just the attribute layer.

- **(consumer/supplier identity)** The standard should identify who the consumer or supplier is, for auditing purposes only - just having a unique id may not be enough.

- **(notification broadcasts)** The standard should support Broadcasts that carry enough information so suppliers can prevent sending notifications to consumers that already have that information (e.g. Avoid cycles)

Objects and Attributes

An XML Schema definition of CPExchange documents will be provided. Providing a definition of CPExchange XML using XML Schema, (when it becomes a final W3C recommendation) will allow the XML to be more easily constrained to the desired object model and to fully define some types that are currently constrained only by the documentation in this specification. In particular, specialized enumerated types can be explicitly defined along with their valid values. It should also be easier to modularly use portions of the CPExchange standard, since XML Schemas are easier to reuse and combine than DTDs.

The object model itself will also be enhanced as experience and continued work identifies additional cross-industry profile information. In particular, a representation of a household with relationships between members of the household and their roles in the household could be useful in many industries.

Work will also be done with interested vertical industry groups to ensure that these groups can effectively use the CPExchange specification for cross-industry information and extend it as desired.

Appendix B: DTDs

Complete Customer Profile Exchange DTD

```
<!-- CPEXMLv1 Information: complete Customer Profile Exchange XML Version 1 Information  with
all categories of data defined in the CPexchange standard -->

<!-- include all the CPexchange categories data -->
<!ENTITY % types SYSTEM " cpexmlv1types.dtd">
%types;
<!ENTITY % support SYSTEM " cpexmlv1sup.dtd">
%support;
<!ENTITY % name SYSTEM " cpexmlv1nam.dtd">
%name;
<!ENTITY % contact SYSTEM " cpexmlv1cont.dtd">
%contact;
<!-- the WEB category brings in cpexmlv1int.dtd and cpexmlv1role.dtd -->
<!ENTITY % web SYSTEM " cpexmlv1web.dtd">
%web;
<!ENTITY % preference SYSTEM " cpexmlv1pref.dtd">
%preference;
<!ENTITY % busobjs SYSTEM " cpexmlv1buso.dtd">
%busobjs;


<!-- override definition of what objects can be used as privacy control templates and instances -->
<!ENTITY % CPEXMLv1ALL " %CPEXMLv1NAM; | %CPEXMLv1CONT; | %CPEXMLv1ROLE; |
%CPEXMLv1INT; | %CPEXMLv1WEB; | %CPEXMLv1BUSO; | %CPEXMLv1PREF; ">
<!ENTITY % templates " ((distinguishable | %CPEXMLv1ALL;)*) ">
<!ENTITY % instances " ((%CPEXMLv1ALL;)*) ">

<!-- include the CPexchange Privacy category data -->
<!ENTITY % p3p SYSTEM " cpexmlv1p3pv1.dtd">
%p3p;
<!ENTITY % privacy SYSTEM " cpexmlv1priv.dtd">
%privacy;

<!-- define root document element -->

<!ELEMENT CPEXMLv1 ( %CPEXMLv1PRIV; )>
```

CPExchange P3P Privacy Subset DTD

```
<!-- CPEXMLv1P3Pv1 Information - Customer Profile Exchange information used from P3P Version
1. This file has no dependencies on other files.

To use this file, declare this file as an ENTITY and include it.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the elements as
desired. This file imposes no structure on the use of these elements.
-->

<!-- beginning of P3P definitions -->
<!--  *********** ACCESS ***********  -->
<!ELEMENT ACCESS (nonident
   | ident_contact
   | other_ident
   | contact_and_other
   | all
   | none)>
<!ELEMENT nonident EMPTY>
<!ELEMENT ident_contact EMPTY>
<!ELEMENT other_ident EMPTY>
<!ELEMENT contact_and_other EMPTY>
<!ELEMENT all EMPTY>
<!ELEMENT none EMPTY>

<!--  *********** PURPOSE ***********  -->
<!ELEMENT PURPOSE ((current
   | admin
   | develop
   | contact
   | customization
   | tailoring
   | individual-analysis
   | individual-decision
   | pseudo-analysis
   | pseudo-decision
   | historical
   | telemarketing
   | other-purpose)+,
   EXTENSION*)>
<!ELEMENT current EMPTY>
<!ELEMENT admin EMPTY>
<!ELEMENT develop EMPTY>
<!ELEMENT contact EMPTY>
<!ELEMENT customization EMPTY>
<!ELEMENT tailoring EMPTY>
<!ELEMENT individual-analysis EMPTY>
<!ELEMENT individual-decision EMPTY>
<!ELEMENT pseudo-analysis EMPTY>
<!ELEMENT pseudo-decision EMPTY>
<!ELEMENT historical EMPTY>
<!ELEMENT telemarketing EMPTY>
```

```
<!ELEMENT other-purpose (#PCDATA)>

<!--  *********** RETENTION ***********  -->
<!ELEMENT RETENTION ((no-retention
   | stated-purpose
   | legal-requirement
   | indefinitely
   | business-practices),
   EXTENSION*)>
<!ELEMENT no-retention EMPTY>
<!ELEMENT stated-purpose EMPTY>
<!ELEMENT legal-requirement EMPTY>
<!ELEMENT indefinitely EMPTY>
<!ELEMENT business-practices EMPTY>

<!--  *********** RECIPIENT ***********  -->
<!ELEMENT RECIPIENT ((ours
   | same
   | other-recipient
   | delivery
   | public
   | unrelated)+,
   EXTENSION*)>
<!ELEMENT ours EMPTY>
<!ELEMENT same EMPTY>
<!ELEMENT other-recipient EMPTY>
<!ELEMENT delivery EMPTY>
<!ELEMENT public EMPTY>
<!ELEMENT unrelated EMPTY>

<!--  *********** EXTENSION ***********  -->
<!ELEMENT EXTENSION (#PCDATA)>
<!ATTLIST EXTENSION
    optional (yes | no) "yes" >


<!-- end of P3P definitions -->
```

CPExchange XML Schema Datatypes

```
<!-- CPEXMLv1TYPES - Customer Profile Exchange XML Schema Datatypes include file. To use
declare this file as an ENTITY and include it in the dtd that uses these definitions. -->

<!-- primitive XML Schema data types -->

<!ENTITY % real "#PCDATA">
<!ENTITY % boolean "#PCDATA">
<!ENTITY % language "#PCDATA">
<!ENTITY % timeInstant "#PCDATA">
<!-- ISO 8601 format,CCYY-MM-DDThh:mm:ss.sssz -->
<!ENTITY % timeDuration "#PCDATA">
<!-- ISO 8601 format, CCYY-MM-DDThh:mm:ss.sssz -->
<!ENTITY % recurringInstant "#PCDATA">
<!ENTITY % language "#PCDATA">
<!-- RFC 1766 format -->
<!ENTITY % string "#PCDATA">
<!-- ISO 10646 unicode -->
<!ENTITY % uriReference "#PCDATA">
<!-- Uniform Resource Locator, IETF RFC 1738 -->


<!-- generated builtin XML Schema data types -->

<!ENTITY % decimal " %real;">
<!ENTITY % integer " %decimal;">
<!ENTITY % non-negative-integer " %integer;">
<!ENTITY % non-positive-integer " %integer;">
<!ENTITY % positive-integer " %non-negative-integer;">
<!ENTITY % negative-integer " %non-positive-integer;">
<!ENTITY % date " %recurringInstant;">
<!ENTITY % time " %recurringInstant;">
<!ENTITY % mime " %string;">


<!-- user defined types representing enumerated element values -->

<!ENTITY % Enum "#PCDATA">
<!ENTITY % OpenEnum " %Enum;">
<!ENTITY % ClosedEnum " %Enum;">
<!ENTITY % CountryCodeEnum " %ClosedEnum;">

<!-- user defined simple data types -->
<!ENTITY % key " %string;">
<!-- string value is any supplier-specific object identifier -->

<!ENTITY % currencyValue " %string;">
<!-- format is a [ISO 3-character country code]:[decimal number] -->

<!ENTITY % ipAddress " %string;">
<!-- format is a either the dotted-decimal string or the complete tcp host.domain name -->

<!ENTITY % numericPriority " %integer;">
```

```
<!-- format is an integer between 1 and 10, with 1 being the highest priority -->

<!-- property extensibility type -->
<!ENTITY % Property " %string;">
<!ENTITY % PropertyAttrs " name CDATA #REQUIRED enumType CDATA #IMPLIED">
```

CPExchange Privacy Category DTD

```
<!-- CPEXMLv1PRIVacy Information - Customer Profile Exchange XML Privacy category data
include file. This file depends on:
.........cpexmlv1p3pv1.dtd       CPExchange P3P subset
.........cpexmlv1types.dtd       XML Schema and CPExchange Types
.........cpexmlv1sup.dtd         CPExchange Support category
To use this file, declare this file and the listed files as ENTITYs. Include the listed
files in the order shown, followed by this file.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the CPEXMLv1PRIV
ENTITY that lists the top level elements in this file.

To restrict what elements can appear in a ControlGroup, the "templates" and "instances"
ENTITYs can be redefined before including this file.
-->

<!--  define hierarchies of classes where any class may be used (e.g.used polymorphically) -->
<!ENTITY % anyJurisdictionDeclaration " jurisdictionSpecification | jurisdictionReference">

<!-- the following ENTITY should be included in the document root element, or some other element
-->
<!ENTITY % CPEXMLv1PRIV " privacyHeader?, controlGroup*">

<!-- define the individual classes with their properties -->

<!-- The templates and instances entities should be overridden in higher level dtds to restrict the
elements that can appear in a ControlGroup -->
<!ENTITY % templates " ANY">
<!ENTITY % instances " ANY">
<!ENTITY % ControlGroup " templates?, instances">
   <!-- properties -->
      <!ELEMENT templates %templates;>
      <!ELEMENT instances %instances;>
<!ELEMENT controlGroup (%ControlGroup;)>

<!ENTITY % PrivacyHeader "
sourceExchangePartner,recipientExchangePartner,jurisdictionDeclarations, privacyDeclarations">
   <!-- properties -->
      <!ELEMENT jurisdictionDeclarations ((%anyJurisdictionDeclaration;)*)>
      <!ELEMENT privacyDeclarations (privacyDeclaration*)>
<!ELEMENT privacyHeader (%PrivacyHeader;)>

<!ENTITY % ExchangePartner " identifier, identifierType">
   <!-- properties -->
      <!ELEMENT identifier (%string;)>
      <!ELEMENT identifierType (%OpenEnum;)>
         <!ATTLIST identifierType enumtype (ExchangePartnerIdentityTypeEnum)
"ExchangePartnerIdentityTypeEnum">
<!ENTITY % ExchangePartnerAttrs " controllingJurisdictionIdrefs IDREFS #IMPLIED">
<!ELEMENT sourceExchangePartner (%ExchangePartner;)>
      <!ATTLIST sourceExchangePartner %ExchangePartnerAttrs;>
<!ELEMENT recipientExchangePartner (%ExchangePartner;)>
```

```
        <!ATTLIST recipientExchangePartner %ExchangePartnerAttrs;>

<!ENTITY % JurisdictionDeclaration "">
<!ENTITY % JurisdictionDeclaractionAttrs " eid ID #REQUIRED">

<!ENTITY % JurisdictionSpecification " %PropertyContainer;, countryCode? ">
    <!-- properties -->
<!ELEMENT jurisdictionSpecification (%JurisdictionSpecification;)>
     <!ATTLIST jurisdictionSpecification %JurisdictionDeclaractionAttrs;>

<!ENTITY % JurisdictionReference " specificationUrl ">
    <!-- properties -->
        <!ELEMENT specificationUrl (%uriReference;)>
<!ELEMENT jurisdictionReference (%JurisdictionReference;)>
     <!ATTLIST jurisdictionReference %JurisdictionDeclaractionAttrs;>

<!ENTITY % PrivacyControl "">
<!ENTITY % PrivacyControlAttrs " eid ID #REQUIRED">

<!ENTITY % PrivacyDeclaration " ACCESS,PURPOSE,RETENTION">
<!ELEMENT privacyDeclaration (%PrivacyDeclaration;)>
    <!ATTLIST privacyDeclaration %PrivacyControlAttrs;>
```

CPExchange Support DTD

```
<!-- CPEXMLv1SUPport - Customer Profile Exchange XML Support Data include file. To use declare
this file as an ENTITY and include it in the dtd that uses these definitions. -->

<!ENTITY % types SYSTEM "cpexmlv1types.dtd">
%types;

<!-- no defined root document element -->

<!-- CPexchange complex Support category data types and common elements -->

<!ENTITY % Timeable " startDate?,endDate?">
  <!-- properties -->
    <!ELEMENT startDate (%timeInstant;)>
    <!ELEMENT endDate (%timeInstant;)>

<!ENTITY % Distinguishable " oid,typeName?,creationDate?,modificationDate?">
  <!-- properties -->
    <!ELEMENT oid (%key;)>
    <!ELEMENT typeName (%OpenEnum;)>
      <!ATTLIST typeName enumtype CDATA #REQUIRED>
    <!ELEMENT creationDate (%timeInstant;)>
    <!ELEMENT modificationDate (%timeInstant;)>
<!ENTITY % DistinguishableAttrs " eid ID #IMPLIED privacyControlIdref IDREF #IMPLIED">
<!ELEMENT distinguishable (%Distinguishable;)>
  <!ATTLIST distinguishable %DistinguishableAttrs;>

<!ENTITY % Describable " description?">
  <!-- properties -->
    <!ELEMENT description (%string;)>

<!ENTITY % PropertyContainer " properties?">
  <!-- properties -->
    <!ELEMENT properties (property*)>
    <!ELEMENT property (%Property;)>
      <!ATTLIST property %PropertyAttrs;>

<!ENTITY % Base " %Distinguishable;,%Describable;,%Timeable;,%PropertyContainer;">
<!ENTITY % BaseAttrs " %DistinguishableAttrs;">


<!-- common elements -->

<!ELEMENT websiteURL (%uriReference;)>
<!ELEMENT countryCode (%CountryCodeEnum;)>
   <!ATTLIST countryCode enumtype (CountryCodeEnum) "CountryCodeEnum">
<!ELEMENT language (%language;)>
<!ELEMENT preferredLanguage (%language;)>
<!ELEMENT realPartyId (%key;)>
   <!ATTLIST realPartyId referenceTo  (Party) "Party">
<!ELEMENT rolePlayerId (%key;)>
   <!ATTLIST rolePlayerId referenceTo  (RolePlayer) "RolePlayer">
```

CPExchange Name Information DTD

```
<!-- CPEXMLv1NAMe Information - Customer Profile Exchange XML Name information from the
Descriptive Information category data include file. This file depends on:
.........cpexmlv1types.dtd       XML Schema and CPExchange Types
.........cpexmlv1sup.dtd         CPExchange Support category
To use this file, declare this file and the listed files as ENTITYs. Include the listed
files in the order shown, followed by this file.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the CPEXMLv1BUSO
ENTITY that lists the top level elements in this file.-->

<!--  define hierarchies of classes where any class may be used (e.g.used polymorphically) -->
<!ENTITY % anyPartyName " partyName | personName |  unstructuredName">

<!-- the following ENTITY should be included in the document root element, or some other element
-->
<!ENTITY % CPEXMLv1NAM " %anyPartyName;">

<!-- define the individual classes with their properties -->
<!ENTITY % PartyName " %Base;,use?,fullName,shortName?,language?,realPartyId?">
   <!-- properties -->
      <!ELEMENT use (%OpenEnum;)>
         <!ATTLIST use enumtype (PartyNameUseEnum | ContactPointUseEnum) #REQUIRED>
      <!ELEMENT fullName (%string;)>
      <!ELEMENT shortName (%string;)>
<!ELEMENT partyName (%PartyName;)>
   <!ATTLIST partyName %BaseAttrs;>

<!ENTITY % PersonName " %PartyName;,firstName,middleNames?,lastName,title?,generation?,
suffixes?,shortFirstName?">
   <!-- properties -->
      <!ELEMENT firstName (%string;)>
      <!ELEMENT middleNames (%string;)>
      <!ELEMENT lastName (%string;)>
      <!ELEMENT title (%string;)>
      <!ELEMENT generation (%string;)>
      <!ELEMENT suffixes (%string;)>
      <!ELEMENT shortFirstName (%string;)>
<!ELEMENT personName (%PersonName;)>
   <!ATTLIST personName %BaseAttrs;>

<!ENTITY % UnstructuredName " %PartyName;">
<!ELEMENT unstructuredName (%UnstructuredName;)>
   <!ATTLIST unstructuredName %BaseAttrs;>
```

CPExchange Contact Information DTD

```
<!-- CPEXMLv1CONTact Point Information - Customer Profile Exchange XML Contact Point
information from the Descriptive Information category data include file. This file
depends on:
.........cpexmlv1types.dtd       XML Schema and CPExchange Types
.........cpexmlv1sup.dtd         CPExchange Support category
To use this file, declare this file and the listed files as ENTITYs. Include the listed
files in the order shown, followed by this file.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the CPEXMLv1CONT
ENTITY that lists the top level elements in this file.-->

<!--  define hierarchies of classes where any class may be used (e.g.used polymorphically) -->
<!ENTITY % anyContactPoint " contactPoint | postalAddress | telephoneNumber | emailAddress |
website">

<!-- the following ENTITY should be included in the document root element, or some other element
-->
<!ENTITY % CPEXMLv1CONT " contactPointUsage | %anyContactPoint; ">

<!-- define the individual classes with their properties -->

<!ENTITY % ContactPoint " %Base;,contactInformation">
   <!-- properties -->
      <!ELEMENT contactInformation (%string;)>
<!ELEMENT contactPoint (%ContactPoint;)>
    <!ATTLIST contactPoint %BaseAttrs;>

<!ENTITY % PostalAddress "
%ContactPoint;,addressFormat?,addressLines?,streetNumber?,street?,poBox?,city,
region?,countryCode,postalCode">
   <!-- properties -->
      <!ELEMENT addressLines (%string;)>
      <!ELEMENT poBox (%string;)>
      <!ELEMENT streetNumber (%string;)>
      <!ELEMENT street (%string;)>
      <!ELEMENT city (%string;)>
      <!ELEMENT region (%string;)>
      <!ELEMENT postalCode (%string;)>
      <!ELEMENT addressFormat (%OpenEnum;)>
         <!ATTLIST addressFormat enumType  (AddressFormatEnum) "AddressFormatEnum">
<!ELEMENT postalAddress (%PostalAddress;)>
   <!ATTLIST postalAddress %BaseAttrs;>

<!ENTITY % TelephoneNumber " %ContactPoint;,countryPhoneCode?,inCountryPrefix?,
areaCode?,localNumber,extension?">
   <!-- properties -->
      <!ELEMENT localNumber (%string;)>
      <!ELEMENT areaCode (%string;)>
      <!ELEMENT inCountryPrefix (%string;)>
      <!ELEMENT countryPhoneCode (%string;)>
      <!ELEMENT extension (%string;)>
```

```
<!ELEMENT telephoneNumber (%TelephoneNumber;)>
   <!ATTLIST telephoneNumber %BaseAttrs;>

<!ENTITY % EmailAddress " %ContactPoint;,domainName,userName">
   <!-- properties -->
      <!ELEMENT domainName (%string;)>
      <!ELEMENT userName (%string;)>
<!ELEMENT emailAddress (%EmailAddress;)>
   <!ATTLIST emailAddress %BaseAttrs;>

<!ENTITY % Website " %ContactPoint;,websiteURL">
<!ELEMENT website (%Website;)>
   <!ATTLIST website %BaseAttrs;>

<!ENTITY % ContactPointUsage " %Base;,contactPointId,rolePlayerId,addresseeNameId?,
availableTimes?,availableDays?, preferredLanguage?, preferenceRank?">
   <!-- properties -->
      <!ELEMENT addresseeNameId (%key;)>
         <!ATTLIST addresseeNameId referenceTo (PartyName) "PartyName">
      <!ELEMENT contactPointId (%key;)>
         <!ATTLIST contactPointId referenceTo (ContactPoint) "ContactPoint">
      <!ELEMENT availableTimes (%ClosedEnum;)>
         <!ATTLIST availableTimes enumType  (TimePeriodEnum) "TimePeriodEnum">
      <!ELEMENT availableDays (%ClosedEnum;)>
         <!ATTLIST availableDays enumType  (DayRangeEnum) "DayRangeEnum">
      <!ELEMENT preferenceRank (%string;)>
<!ELEMENT contactPointUsage (%ContactPointUsage;)>
    <!ATTLIST contactPointUsage %BaseAttrs;>
```

CPExchange Role Information DTD

```
<!-- CPEXMLv1ROLE Information - Customer Profile Exchange XML Role Category Information
include file. This file depends on:
.........cpexmlv1types.dtd       XML Schema and CPExchange Types
.........cpexmlv1sup.dtd         CPExchange Support category
To use this file, declare this file and the listed files as ENTITYs. Include the listed
files in the order shown, followed by this file.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the CPEXMLv1ROLE
ENTITY that lists the top level elements in this file.-->

<!--  define hierarchies of classes where any class may be used (e.g.used polymorphically) -->
<!ENTITY % anyPartyDemographics " partyDemographics | organisationDemographics |
personDemographics">
<!ENTITY % anyRolePlayer " rolePlayer | party | person | organisation |company | partyRole |
customer | employee">
<!ENTITY % anyPartyActivity " partyActivity | hobby | occupation | industry">

<!-- the following ENTITY should be included in the document root element, or some other element
-->
<!ENTITY % CPEXMLv1ROLE " %anyRolePlayer; | %anyPartyDemographics; | %anyPartyActivity;">

<!-- define the individual classes with their properties -->
<!ENTITY % RolePlayer " %Base;,preferredLanguage?,defaultNameId?,nameIds?,partyRoleIds?,
defaultContactPointIds?,contactPointUsageIds?,( %anyPartyDemographics;)?,partyActivities?">
   <!-- properties -->
      <!ELEMENT nameIds (nameId*)>
      <!ELEMENT nameId (%key;)>
         <!ATTLIST partyNameId referenceTo (PartyName) "PartyName">
      <!ELEMENT defaultNameId (%key;)>
         <!ATTLIST defaultNameId referenceTo (PartyName) "PartyName">
      <!ELEMENT partyRoleIds (partyRoleId*)>
      <!ELEMENT partyRoleId (%key;)>
         <!ATTLIST partyRoleId referenceTo (PartyRole) "PartyRole">
      <!ELEMENT contactPointUsageIds (contactPointUsageId*)>
      <!ELEMENT contactPointUsageId (%key;)>
         <!ATTLIST contactPointUsageId referenceTo  (ContactPointUsage) "ContactPointUsage">
      <!ELEMENT defaultContactPointIds (defaultContactPointId*)>
      <!ELEMENT defaultContactPointId (%key;)>
         <!ATTLIST defaultContactPointId referenceTo  (ContactPoint) "ContactPoint">
      <!ELEMENT partyActivities ( %anyPartyActivity;)*>
<!ELEMENT rolePlayer (%RolePlayer;)>
   <!ATTLIST rolePlayer %BaseAttrs;>

<!ENTITY % Party " %RolePlayer;,nationalities?">
   <!-- properties -->
      <!ELEMENT nationalities (nationality*)>
<!ELEMENT party (%Party;)>
   <!ATTLIST party %BaseAttrs;>

<!ENTITY % Nationality " %Base;,countryCode,nationalIdentifier?,nationalIdentifierType?,
taxationIdentifier?,vatNumber?,passportNumber?,citizen?,registered?,primaryResidence?,
resident?,realPartyId">
   <!-- properties -->
      <!ELEMENT nationalIdentifier (%string;)>
      <!ELEMENT nationalIdentifierType (%OpenEnum;)>
         <!ATTLIST nationalIdentifierType enumtype (NationalIdentifierTypeEnum)
"NationalIdentifierTypeEnum">
      <!ELEMENT taxationIdentifier (%string;)>
      <!ELEMENT vatNumber (%string;)>
      <!ELEMENT passportNumber (%string;)>
      <!ELEMENT citizen (%boolean;)>
      <!ELEMENT registered (%boolean;)>
      <!ELEMENT primaryResidence (%boolean;)>
      <!ELEMENT resident (%boolean;)>
<!ELEMENT nationality (%Nationality;)>
   <!ATTLIST nationality %BaseAttrs;>

<!ENTITY % Person " %Party;">
<!ELEMENT person (%Person;)>
   <!ATTLIST person %BaseAttrs;>

<!ENTITY % Organisation " %Party;,unitIdentifier?">
   <!-- properties -->
      <!ELEMENT unitIdentifier (%string;)>
<!ELEMENT organisation (%Organisation;)>
   <!ATTLIST organisation %BaseAttrs;>

<!ENTITY % Company " %Organisation;,dunsNumber?">
   <!-- properties -->
      <!ELEMENT dunsNumber (%string;)>
<!ELEMENT company (%Company;)>
   <!ATTLIST company %BaseAttrs;>

<!ENTITY % PartyRole " %RolePlayer;,contextId?,rolePlayerId?,realPartyId?">
   <!-- properties -->
      <!ELEMENT contextId (%key;)>
         <!ATTLIST contextId referenceTo (Distinguishable) "Distinguishable">
<!ELEMENT partyRole (%PartyRole;)>
   <!ATTLIST partyRole %BaseAttrs;>

<!ENTITY % Customer " %PartyRole;,customerNumber?,customerPriority?,howReferred?">
   <!-- properties -->
      <!ELEMENT customerNumber (%string;)>
      <!ELEMENT customerPriority (%numericPriority;)>
      <!ELEMENT howReferred (%OpenEnum;)>
         <!ATTLIST howReferred enumtype (HowReferredEnum) "HowReferredEnum">
<!ELEMENT customer (%Customer;)>
   <!ATTLIST customer %BaseAttrs;>

<!ENTITY % Employee " %PartyRole;,jobId?">
   <!-- properties -->
      <!ELEMENT jobId (%key;)>
         <!ATTLIST jobId referenceTo (Occupation) "Occupation">
<!ELEMENT employee (%Employee;)>
   <!ATTLIST employee %BaseAttrs;>
```

```
<!ENTITY % PartyDemographics " %Base;, rolePlayerId">
<!ELEMENT partyDemographics (%PartyDemographics;)>
    <!ATTLIST partyDemographics %BaseAttrs;>

<!ENTITY % OrganisationDemographics "%PartyDemographics;,
approximateNumberOfMembers?,numberOfLocations?,primaryIndustryId?">
   <!-- properties -->
      <!ELEMENT approximateNumberOfMembers (%integer;)>
      <!ELEMENT numberOfLocations (%integer;)>
      <!ELEMENT primaryIndustryId (%key;)>
         <!ATTLIST primaryIndustryId referenceTo (Industry) "Industry">
<!ELEMENT organisationDemographics (%OrganisationDemographics;)>
   <!ATTLIST organisationDemographics %BaseAttrs;>

<!ENTITY % PersonDemographics " %PartyDemographics;,gender?,birthDate?,deathDate?,
maritalStatus?,currentMarriageDate?,spouseNameId?,maidenName?,numberOfDependents?,
educationLevel?,incomeLevel?,smoker?,races?,primaryOccupationId?">
   <!-- properties -->
      <!ELEMENT gender (%ClosedEnum;)>
         <!ATTLIST gender enumtype (GenderEnum) "GenderEnum">
      <!ELEMENT birthDate (%timeInstant;)>
      <!ELEMENT deathDate (%timeInstant;)>
      <!ELEMENT currentMarriageDate (%timeInstant;)>
      <!ELEMENT maritalStatus (%ClosedEnum;)>
         <!ATTLIST maritalStatus enumtype (MaritalStatusEnum) "MaritalStatusEnum">
      <!ELEMENT educationLevel (%OpenEnum;)>
         <!ATTLIST educationLevel enumtype (EducationLevelEnum) "EducationLevelEnum">
      <!ELEMENT incomeLevel (%OpenEnum;)>
         <!ATTLIST incomeLevel enumtype (IncomeLevelEnum) "IncomeLevelEnum">
      <!ELEMENT smoker (%boolean;)>
      <!ELEMENT races (race+)>
      <!ELEMENT race (%ClosedEnum;)>
         <!ATTLIST race enumtype (RaceEnum) "RaceEnum">
      <!ELEMENT numberOfDependents (%integer;)>
      <!ELEMENT primaryOccupationId (%key;)>
         <!ATTLIST primaryOccupationId referenceTo (Occupation) "Occupation">
      <!ELEMENT spouseNameId (%key;)>
         <!ATTLIST spouseNameId referenceTo (PartyName) "PartyName">
      <!ELEMENT maidenName (%string;)>
<!ELEMENT personDemographics (%PersonDemographics;)>
   <!ATTLIST personDemographics %BaseAttrs;>

<!ENTITY % PartyActivity " %Base;,rolePlayerId">
<!ELEMENT partyActivity (%PartyActivity;)>
    <!ATTLIST partyActivity %BaseAttrs;>

<!ENTITY % Hobby " %PartyActivity;,hobbyName?">
   <!-- properties -->
      <!ELEMENT hobbyName (%OpenEnum;)>
         <!ATTLIST hobbyName enumtype (HobbyNameEnum) "HobbyNameEnum">
<!ELEMENT hobby (%Hobby;)>
   <!ATTLIST hobby %BaseAttrs;>
```

```
<!ENTITY % Occupation " %PartyActivity;,occupationPercentage?,employmentStatus?,jobTitle?,
jobDescription?,
employerNameId?,primaryIndustryType?,(payFrequency,basePay,annualBonusPay)?">
   <!-- properties -->
      <!ELEMENT occupationPercentage (%decimal;)>
      <!ELEMENT employmentStatus (%ClosedEnum;)>
         <!ATTLIST employmentStatus enumtype (EmploymentStatusEnum)
"EmploymentStatusEnum">
      <!ELEMENT jobTitle (%string;)>
      <!ELEMENT jobDescription (%string;)>
      <!ELEMENT employerNameId (%key;)>
         <!ATTLIST employerNameId referenceTo (PartyName) "PartyName">
      <!ELEMENT primaryIndustryType (%key;)>
         <!ATTLIST primaryIndustryType referenceTo (IndustryTypeEnum) "IndustryTypeEnum">
      <!ELEMENT payFrequency (%timeDuration;)>
      <!ELEMENT basePay (%currencyValue;)>
      <!ELEMENT annualBonusPay (%currencyValue;)>
<!ELEMENT occupation (%Occupation;)>
   <!ATTLIST occupation %BaseAttrs;>

<!ENTITY % Industry " %PartyActivity;,industryPercentage?">
   <!-- properties -->
      <!ELEMENT industryPercentage (%decimal;)>
<!ELEMENT industry (%Industry;)>
   <!ATTLIST industry %BaseAttrs;>
```

CPExchange Preferences DTD

```
<!-- CPEXMLv1PREFerence Information - Customer Profile Exchange XML Preference Information
category data include file. This file depends on:
.........cpexmlv1types.dtd        XML Schema and CPExchange Types
.........cpexmlv1sup.dtd          CPExchange Support category
To use this file, declare this file and the listed files as ENTITYs. Include the listed
files in the order shown, followed by this file.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the CPEXMLv1PREF
ENTITY that lists the top level elements in this file.-->

<!--  define hierarchies of classes where any class may be used (e.g.used polymorphically) -->
<!ENTITY % anyPreference " preference">

<!-- the following ENTITY should be included in the document root element, or some other element
-->
<!ENTITY % CPEXMLv1PREF " preferenceContainer ">

<!-- define the individual classes with their properties -->
<!ENTITY % PreferenceContainer " %Distinguishable;, rolePlayerId?,preferences?">
   <!-- properties -->
      <!ELEMENT preferences ((%anyPreference;)*)>
<!ELEMENT preferenceContainer (%PreferenceContainer;)>
    <!ATTLIST preferenceContainer %DistinguishableAttrs;>

<!ENTITY % Preference " property, howSpecified?,sourceId?,sourceDate?">
   <!-- properties -->
      <!ELEMENT howSpecified (%OpenEnum;)>
      <!ELEMENT sourceId (%key;)>
         <!ATTLIST sourceId referenceTo CDATA #REQUIRED>
      <!ELEMENT sourceDate (%timeInstant;)>
<!ELEMENT preference (%Preference;)>
```

CPExchange Business Object DTD

```
<!-- CPEXMLv1BUSiness Object Information - Customer Profile Exchange XML Business
Object Information category data include file. This file depends on:
.........cpexmlv1types.dtd       XML Schema and CPExchange Types
.........cpexmlv1sup.dtd         CPExchange Support category
To use this file, declare this file and the listed files as ENTITYs. Include the listed
files in the order shown, followed by this file.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the CPEXMLv1BUSO
ENTITY that lists the top level elements in this file.-->

<!--  define hierarchies of classes where any class may be used (e.g.used polymorphically) -->
<!ENTITY % anyProductReference " productReference">
<!ENTITY % anyCreditCardReference " creditCardReference">
<!ENTITY % anyAccountReference " accountReference">

<!-- the following ENTITY should be included in the document root element, or some other element
-->
<!ENTITY % CPEXMLv1BUSO " %anyProductReference; | %anyCreditCardReference; |
%anyAccountReference;">

<!-- define the individual classes with their properties -->
<!ENTITY % ProductReference " %Base;, productName, productNumber?">
   <!-- properties -->
      <!ELEMENT productName (%string;)>
      <!ELEMENT productNumber (%string;)>
<!ELEMENT productReference (%ProductReference;)>
    <!ATTLIST productReference %BaseAttrs;>

<!ENTITY % AccountReference "
%Base;,accountNumber,transferNumber,financialOrganisationId?,accountOwnerId?">
   <!-- properties -->
      <!ELEMENT accountNumber (%timeInstant;)>
      <!ELEMENT transferNumber (%integer;)>
      <!ELEMENT financialOrganisationId (%key;)>
         <!ATTLIST financialOrganisationId referenceTo (Organisation) "Organisation">
      <!ELEMENT accountOwnerId (%key;)>
         <!ATTLIST accountOwnerId referenceTo (RolePlayer) "RolePlayer">
<!ELEMENT accountReference (%AccountReference;)>
    <!ATTLIST accountReference %BaseAttrs;>

<!ENTITY % CreditCardReference " %Base;,
cardHolderName,cardNumber,cardExpiration,billingAddressId?,cardHolderId? ">
   <!-- properties -->
      <!ELEMENT cardHolderName (%string;)>
      <!ELEMENT cardNumber (%string;)>
      <!ELEMENT cardExpiration (%timeInstant;)>
      <!ELEMENT billingAddressId (%key;)>
         <!ATTLIST billingAddressId referenceTo (PostalAddress) "PostalAddress">
      <!ELEMENT cardHolderId (%key;)>
         <!ATTLIST cardHolderId referenceTo (RolePlayer) "RolePlayer">
<!ELEMENT creditCardReference (%CreditCardReference;)>
```

```
<!ATTLIST creditCardReference %BaseAttrs;>
```

CPExchange InteractionHistory DTD

```
<!-- CPEXMLv1INTeraction History Information - Customer Profile Exchange XML Interaction
History Information category data include file. This file depends on:
.........cpexmlv1types.dtd       XML Schema and CPExchange Types
.........cpexmlv1sup.dtd         CPExchange Support category
To use this file, declare this file and the listed files as ENTITYs. Include the listed
files in the order shown, followed by this file.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the CPEXMLv1INT
ENTITY that lists the top level elements in this file.-->

<!--  define hierarchies of classes where any class may be used (e.g.used polymorphically) -->
<!ENTITY % anyActionEvent " actionEvent">
<!ENTITY % anyCaseFolder " caseFolder">
<!ENTITY % anyEncounter " encounter">

<!-- the following ENTITY should be included in the document root element, or some other element
-->
<!ENTITY % CPEXMLv1INT " %anyActionEvent; | %anyCaseFolder; | %anyEncounter; | note">

<!-- define the individual classes with their properties -->
<!ENTITY % ActionEvent " %Base;,status,location?,result?,subjectId?,taskId?,caseIds?,
initiatorId?,participantIds?,encounterId?,spawnedActionEventId?,spawningActionEventId?,
notes?,channels?">
  <!-- properties -->
    <!ELEMENT status (%OpenEnum;)>
      <!ATTLIST status enumType (EventStatusEnum | CaseStatusEnum) #REQUIRED>
    <!ELEMENT location (%string;)>
    <!ELEMENT result (%OpenEnum;)>
      <!ATTLIST result enumType (ResultEnum) "ResultEnum">
    <!ELEMENT encounterId (%key;)>
      <!ATTLIST encounterId referenceTo (Encounter) "Encounter">
    <!ELEMENT initiatorId (%key;)>
      <!ATTLIST initiatorId referenceTo (RolePlayer) "RolePlayer">
    <!ELEMENT subjectId (%key;)>
      <!ATTLIST subjectId referenceTo (Distinguishable | CaseFolder) "Distinguishable">
    <!ELEMENT taskId (%key;)>
      <!ATTLIST taskId referenceTo (Distinguishable | CaseFolder) "Distinguishable">
    <!ELEMENT participantIds (participantId*)>
    <!ELEMENT participantId (%key;)>
      <!ATTLIST participantId referenceTo (RolePlayer) "RolePlayer">
    <!ELEMENT caseIds (caseId*)>
    <!ELEMENT caseId (%key;)>
      <!ATTLIST caseId referenceTo (CaseFolder) "CaseFolder">
    <!ELEMENT spawningActionEventId (%key;)>
      <!ATTLIST spawningActionEventId referenceTo (ActionEvent) "ActionEvent">
    <!ELEMENT spawnedActionEventId (%key;)>
      <!ATTLIST spawnedActionEventId referenceTo (ActionEvent) "ActionEvent">
    <!ELEMENT channels (interactionChannel*)>
    <!ELEMENT notes (note*)>
<!ELEMENT actionEvent (%ActionEvent;)>
   <!ATTLIST actionEvent %BaseAttrs;>
```

```
<!ENTITY % CaseFolder " %Base;,dueDate?,commonName?,status,priority?,
parentCaseId?,notes?">
  <!-- properties -->
    <!ELEMENT dueDate (%timeInstant;)>
    <!ELEMENT commonName (%string;)>
    <!ELEMENT priority (%integer;)>
    <!ELEMENT parentCaseId (%key;)>
      <!ATTLIST parentCaseId referenceTo (CaseFolder) "CaseFolder">
<!ELEMENT caseFolder (%CaseFolder;)>
   <!ATTLIST caseFolder %BaseAttrs;>

<!ENTITY % Encounter " %Base;,status,actionEventIds?,notes?">
  <!-- properties -->
    <!ELEMENT actionEventIds (actionEventId*)>
    <!ELEMENT actionEventId (%key;)>
      <!ATTLIST actionEventId referenceTo (ActionEvent) "ActionEvent">
<!ELEMENT encounter (%Encounter;)>
   <!ATTLIST encounter %BaseAttrs;>

<!ENTITY % Note " %Base;, summary,externalReference?,comment?,(caseId,
encounterId,actionEventId)?">
  <!-- properties -->
    <!ELEMENT comment (%string;)>
    <!ELEMENT summary (%string;)>
    <!ELEMENT externalReference (%uriReference;)>
<!ELEMENT note (%Note;)>
   <!ATTLIST note %BaseAttrs;>

<!ENTITY % InteractionChannel " contactMode, assistedIndicator?">
  <!-- properties -->
    <!ELEMENT contactMode (%OpenEnum;)>
      <!ATTLIST contactMode enumType (ContactModeEnum) "ContactModeEnum">
    <!ELEMENT assistedIndicator (%OpenEnum;)>
      <!ATTLIST assistedIndicator enumType (AssistedEnum) "AssistedEnum">
<!ELEMENT interactionChannel (%InteractionChannel;)>
```

CPExchange Web DTD

```
<!-- CPEXMLv1WEB Information - Customer Profile Exchange XML Web Information category data
include file.
This file depends on:
.........cpexmlv1types.dtd       XML Schema and CPExchange Types
.........cpexmlv1sup.dtd         CPExchange Support category
To use this file, declare this file and the listed files as ENTITYs. Include the listed
files in the order shown, followed by this file.

To incorporate the elements in this file into a higher level document structure, the
root, (or some other) element of the new document should include the CPEXMLv1WEB
ENTITY that lists the top level elements in this file.

WARNING: This file directly includes the following files, so they cannot be included again:
.........cpexmlv1int.dtd
.........cpexmlv1role.dtd
-->

<!--  define hierarchies of classes where any class may be used (e.g.used polymorphically) -->
<!ENTITY % anyActionEvent " actionEvent | click">
<!ENTITY % anyEncounter " encounter | websiteVisit">
<!ENTITY % anyPartyActivity " partyActivity | hobby | occupation | industry | websiteUse">

<!ENTITY % roles SYSTEM "cpexmlv1role.dtd">
%roles;
<!ENTITY % interaction SYSTEM "cpexmlv1int.dtd">
%interaction;

<!-- the following ENTITY should be included in the document root element, or some other element
-->
<!ENTITY % CPEXMLv1WEB " websiteVisit | click | websiteUse  ">

<!-- define the individual classes with their properties -->
<!ENTITY % WebsiteVisit " %Encounter;, visitorAddress?,visitorHost?,visitorUser?,
userAgent?">
   <!-- properties -->
      <!ELEMENT visitorAddress (%ipAddress;)>
      <!ELEMENT visitorHost (%string;)>
      <!ELEMENT visitorUser (%string;)>
      <!ELEMENT userAgent (%string;)>
<!ELEMENT websiteVisit (%WebsiteVisit;)>
    <!ATTLIST websiteVisit %BaseAttrs;>

<!ENTITY % Click " %ActionEvent;, linkURL,contentType?,referrerURL?,inputParameters?">
   <!-- properties -->
      <!ELEMENT linkURL (%uriReference;)>
      <!ELEMENT contentType (%mime;)>
      <!ELEMENT referrerURL (%uriReference;)>
      <!ELEMENT inputParameters (%string;)>
<!ELEMENT click (%Click;)>
    <!ATTLIST click %BaseAttrs;>

<!ENTITY % ContentItemContainer " %Base;,items?">
```

```
   <!-- properties -->
      <!ELEMENT items (contentItemReference*)>
<!ELEMENT contentItemContainer (%ContentItemContainer;)>
    <!ATTLIST contentItemContainer %BaseAttrs;>

<!ENTITY % ContentItemReference " itemRepository, itemIdentifier">
   <!-- properties -->
      <!ELEMENT itemRepository (%string;)>
      <!ELEMENT itemIdentifier (%string;)>
<!ELEMENT contentItemReference (%ContentItemReference;)>

<!ENTITY % WebsiteUse "
%PartyActivity;,websiteURL,visitCount?,lastActivityDate?,itemContainer?">
   <!-- properties -->
      <!ELEMENT visitCount (%integer;)>
      <!ELEMENT lastActivityDate (%timeInstant;)>
      <!ELEMENT itemContainer (%ContentItemContainer;)>
<!ELEMENT websiteUse (%WebsiteUse;)>
    <!ATTLIST websiteUSe %BaseAttrs;>
```

Appendix C: Sample Profile data

The following CPExchange profile data was created by hand to show a simple example of how the data will be represented in XML. This does not show the protocol header or commands that would also be included in a real data exchange document.

The data that is provided in this sample represents a single person, her contact phone numbers, email address, demographic information, a very limited transaction history for the most recent, and next CPExchange data model subgroup meetings. Some privacy control information is also included.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE CPEXMLv1 SYSTEM "cpexmlv1.dtd" >
<CPEXMLv1>
 <privacyHeader>
  <sourceExchangePartner controllingJurisdictionIdrefs="JD0001">
   <identifier>1234567890123456789012345678912</identifier>
   <identifierType>UUID</identifierType>
  </sourceExchangePartner>
  <recipientExchangePartner controllingJurisdictionIdrefs="JD0002 JD0003">
   <identifier>1234567890123456789012345678913</identifier>
   <identifierType>UUID</identifierType>
  </recipientExchangePartner>
  <jurisdictionDeclarations>
   <jurisdictionReference eid="JD0001">
    <specificationUrl>http://www.amazon.de/CPEX/2000/jurisdiction.xml</specificationUrl>
   </jurisdictionReference>
   <jurisdictionSpecification eid="JD0002">
    <properties>
     <property name="name">Amazon</property>
     <property name="postalCode">32894</property>
    </properties>
    <countryCode>DE</countryCode>
   </jurisdictionSpecification>
   <jurisdictionSpecification eid="JD0003">
    <properties>
     <property name="name">Galois</property>
     <property name="postalCode">11000</property>
    </properties>
    <countryCode>FR</countryCode>
   </jurisdictionSpecification>
  </jurisdictionDeclarations>
  <privacyDeclarations>
   <privacyDeclaration eid="PVDC00000001">
    <ACCESS>
     <none/>
    </ACCESS>
    <PURPOSE>
     <current/>
     <admin/>
     <contact/>
    </PURPOSE>
```

```
      <RETENTION>
        <indefinitely/>
      </RETENTION>
    </privacyDeclaration>
    <privacyDeclaration eid="PVDC00000002">
      <ACCESS>
        <none/>
      </ACCESS>
      <PURPOSE>
        <current/>
        <admin/>
      </PURPOSE>
      <RETENTION>
        <no-retention/>
      </RETENTION>
    </privacyDeclaration>
  </privacyDeclarations>
</privacyHeader>
<controlGroup>
  <templates>
    <distinguishable privacyControlIdref="PVDC00000001">
      <oid/>
    </distinguishable>
  </templates>
  <instances>
    <partyName>
      <oid>NAME00000001</oid>
      <use enumtype="PartyNameUseEnum">DEFAULT</use>
      <fullName>Kathy Bohrer</fullName>
      <shortName>Kathy</shortName>
    </partyName>
    <personName>
      <oid>NAME00000011</oid>
      <use enumtype="PartyNameUseEnum">LEGAL</use>
      <fullName>Kathryn Ann Bohrer</fullName>
      <firstName>Kathryn</firstName>
      <middleNames>Ann</middleNames>
      <lastName>Bohrer</lastName>
      <shortFirstName>Kathy</shortFirstName>
    </personName>
    <person>
      <oid>PRSN00000001</oid>
      <defaultNameId>NAME00000001</defaultNameId>
      <nameIds>
        <nameId>NAME00000001</nameId>
      </nameIds>
      <contactPointUsageIds>
        <contactPointUsageId>CPTU00000002</contactPointUsageId>
        <contactPointUsageId>CPTU00000001</contactPointUsageId>
        <contactPointUsageId>CPTU00000003</contactPointUsageId>
      </contactPointUsageIds>
      <personDemographics>
        <oid>PDEM00000001</oid>
        <rolePlayerId>PRSN00000001</rolePlayerId>
```

```
          <gender>F</gender>
          <birthDate>1952-08-19T</birthDate>
          <maritalStatus>MARRIED</maritalStatus>
          <currentMarriageDate>1978-01-01T</currentMarriageDate>
          <spouseNameId>NAME00000002</spouseNameId>
          <smoker>false</smoker>
        </personDemographics>
        <partyActivities>
          <occupation>
            <oid>PACT00000001</oid>
            <rolePlayerId>PRSN0000001</rolePlayerId>
            <occupationPercentage>100</occupationPercentage>
            <employmentStatus>RETIRED</employmentStatus>
            <jobTitle>Distinguished Engineer</jobTitle>
            <employerNameId>NAME10000001</employerNameId>
          </occupation>
          <hobby>
            <oid>PACT00000002</oid>
            <typeName enumtype="HobbyTypeEnum">SPORTS</typeName>
            <startDate>1994-12-27T</startDate>
            <rolePlayerId>PRSN00000001</rolePlayerId>
            <hobbyName>SAILING</hobbyName>
          </hobby>
          <websiteUse>
            <oid/>
            <rolePlayerId>PRSN00000001</rolePlayerId>
            <websiteURL>http://www.cpexchange.org</websiteURL>
            <visitCount>31</visitCount>
            <lastActivityDate>2000-08-02T</lastActivityDate>
          </websiteUse>
        </partyActivities>
        <nationalities>
          <nationality privacyControlIdref="PVDC00000002">
            <oid>NATL00000001</oid>
            <countryCode>US</countryCode>
            <nationalIdentifier>223-435-9999</nationalIdentifier>
            <nationalIdentifierType>SOCIAL_SECURITY</nationalIdentifierType>
            <realPartyId>PRSN00000001</realPartyId>
          </nationality>
        </nationalities>
      </person>
      <contactPointUsage>
        <oid>CPTU00000001</oid>
        <typeName enumtype="ContactPointUsageTypeEnum">BUSINESS</typeName>
        <contactPointId>ADDR00000001</contactPointId>
        <rolePlayerId>PRSN00000001</rolePlayerId>
      </contactPointUsage>
      <postalAddress>
        <oid>ADDR00000001</oid>
        <contactInformation>3711 Del Robles Dr.
Austin, TX 78727
USA</contactInformation>
        <street>Del Robles Drive</street>
        <city>Austin</city>
```

```
  <region>Texas</region>
  <countryCode>US</countryCode>
  <postalCode>78727</postalCode>
</postalAddress>
<contactPointUsage>
  <oid>CPTU00000002</oid>
  <typeName enumtype="ContactPointUsageTypeEnum">BUSINESS</typeName>
  <contactPointId>PHON00000001</contactPointId>
  <rolePlayerId>PRSN00000001</rolePlayerId>
</contactPointUsage>
<telephoneNumber>
  <oid>PHON00000001</oid>
  <contactInformation>512-791-3058</contactInformation>
  <countryPhoneCode>001</countryPhoneCode>
  <inCountryPrefix>1</inCountryPrefix>
  <areaCode>512</areaCode>
  <localNumber>791-3058</localNumber>
</telephoneNumber>
<contactPointUsage>
  <oid>CPTU00000003</oid>
  <typeName enumtype="ContactPointUsageTypeEnum">BUSINESS</typeName>
  <contactPointId>EMAIL00000001</contactPointId>
  <rolePlayerId>PRSN00000001</rolePlayerId>
</contactPointUsage>
<emailAddress>
  <oid>EMAIL00000001</oid>
  <contactInformation>bohrer@us.ibm.com</contactInformation>
  <domainName>us.ibm.com</domainName>
  <userName/>
</emailAddress>
<partyName>
  <oid>NAME00000002</oid>
  <use enumtype="PartyNameUseEnum">LEGAL</use>
  <fullName>Frederick John Tydeman</fullName>
</partyName>
<partyName>
  <oid>NAME10000001</oid>
  <use enumtype="PartyNameUseEnum">LEGAL</use>
  <fullName>International Business Machines Corporation</fullName>
  <shortName>IBM</shortName>
</partyName>
<creditCardReference privacyControlIdref="PVDC00000002">
  <oid>CRED00000001</oid>
  <typeName enumtype="CreditCardTypeEnum">VISA</typeName>
  <cardHolderName>Kathryn A. Bohrer</cardHolderName>
  <cardNumber>88799822832</cardNumber>
  <cardExpiration>2002-02-31T</cardExpiration>
  <billingAddressId>ADDR00000001</billingAddressId>
  <cardHolderId>PRSN00000001</cardHolderId>
</creditCardReference>
<productReference>
  <oid/>
  <productName>The Pilot’s Wife</productName>
  <productNumber>ISBN0316601950</productNumber>
  </productReference>
  <actionEvent>
   <oid>EVNT00000001</oid>
   <typeName enumtype="ActionEventTypeEnum">EXTERNAL_MEETING</typeName>
   <description>Subgroup meetings in Alexandria.</description>
   <startDate>2000-04-26T09:00:00-5</startDate>
   <endDate>2000-04-26T17:00:00-5</endDate>
   <status enumType="EventStatusEnum">PLANNED</status>
   <subjectId>CASE00000001</subjectId>
   <encounterId>ENC00000001</encounterId>
   <channels>
    <interactionChannel>
     <contactMode>WEB</contactMode>
     <assistedIndicator>COLLABORATIVE</assistedIndicator>
    </interactionChannel>
   </channels>
  </actionEvent>
  <actionEvent>
   <oid>EVNT00000001</oid>
   <typeName enumtype="ActionEventTypeEnum">EXTERNAL_MEETING</typeName>
   <description>Working Group meeting in Alexandria.</description>
   <startDate>2000-04-27T09:00:00-5</startDate>
   <endDate>2000-04-27T17:00:00-5</endDate>
   <status enumType="EventStatusEnum">PLANNED</status>
   <encounterId>ENC00000001</encounterId>
  </actionEvent>
  <encounter>
   <oid>ENC00000001</oid>
   <description>CPexchange Working Group meeting in Alexandria.</description>
   <status enumType="EventStatusEnum">CLOSED</status>
  </encounter>
  <caseFolder>
   <oid>CASE00000001</oid>
   <description>Customer Profile Exchange XML standard effort.</description>
   <startDate>1999-08-06T</startDate>
   <dueDate>2000-09-01T</dueDate>
   <commonName>CPexchange</commonName>
   <status enumType="CaseStatusEnum">ACTIVE</status>
  </caseFolder>
  <websiteVisit>
   <oid>ENC00000002</oid>
   <status enumType="EventStatusEnum">CLOSED</status>
   <visitorHost>kb.watson.ibm.com</visitorHost>
   <visitorUser>bohrer</visitorUser>
  </websiteVisit>
  <click>
   <oid/>
   <typeName enumtype="ActionEventTypeEnum">CLICK</typeName>
   <status enumType="EventStatusEnum">CLOSED</status>
   <encounterId>ENC00000002</encounterId>
   <linkURL>http://www.cpexchange.org</linkURL>
  </click>
 </instances>
</controlGroup>
```

```
</CPEXMLv1>
```

Appendix D: Use Cases

Use Case: Medical Contact

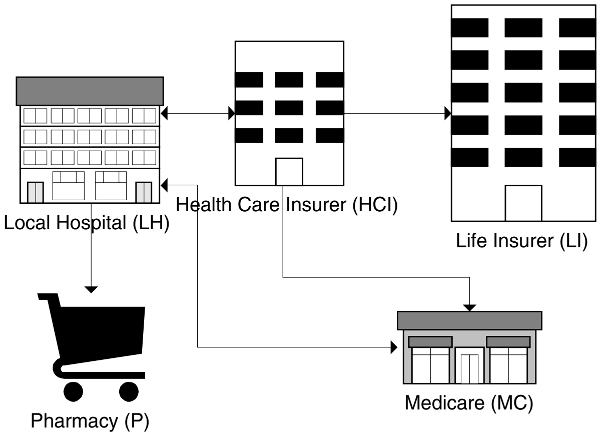
Revision History:

Date	Author	Changes
2000.01.31	Matthew Doering	Initial specification
2000.05.23	Bobby Holland	Change title and formatting for inclusion in spec v0.6
2000.08.09	Brad Husick	add P to LH interaction

Abstract:

A **Patient (P)** is admitted to a local **Hospital** for emergency surgery. The **Patient** has private insurance but is also covered by **Medicare**. There are 5 entities that will need to share information about the consumer: the local **Hospital** (**LH**), the **Health Care Insurer** (**HCI**), **Medicare** (**MC**), local **Pharmacy** (B), and HCI's parent company a **Life Insurance Company** (**LIC**). The hospital needs to verify that the patient has coverage and coverage amounts. The insurers need to confirm that the procedures being performed are appropriate for the patient (for example, males are not covered for hysterectomies). The insurers then need basic reports. The pharmacy needs coverage info and drug interaction warnings and the life insurance company needs to cross sell.

Interactions:



P to LH – Admission.

- Sent – Patient registration data, including SSN, Name, Address, Insurance data

LH to MC – Request for verification of coverage.

- Sent – SSN and/or Medicare ID, Name, Address, DRG (Diagnostic Research Group), HCI ID
- Received – Coverage verification flag

LH to MC – Request for payment

- Sent – SSN and/or Medicare ID, Name, Address, DRG, Charges
- Received – EFT confirmation number

MC to LH – Monthly report of Medicare activity

- Sent – request for monthly report
- Received – SSN and/or Medicare ID, DRGs, demographic (Gender, age), HCI ID, HCI Covered amounts, Medicare amounts for multiple consumers
- LH to MC - Request for verification of coverage
- Sent – SSN, Group/Policy ID, Name, Address, DRG (Diagnostic Research Group), HCI ID
- Received – Coverage verification flag, Service levels ($)LH to MCI – Request for payment
- Sent – SSN, Group/Policy ID, Name, Address, DRG, Charges
- Received – EFT confirmation number

MCI to LH – Monthly report of Insurance activity/Fraud reporting

- Sent – request for monthly report
- Received – SSN, Group/Policy ID, DRGs, All demographic
- Sent – SSN, Group/Policy ID, Fraud flag

MC to HCI – Payment and Coverage Verification

- Sent – request for monthly report
- Received – Hospital ID, SSN, DRG, Date, coverage amounts

P to HCI/MC – Request for coverage and co-pay

- Sent – SSN, Group/Policy ID, RX#, RX id, prescribing DR. #
- Receive – SSN, Group/Policy ID, RX #, co-pay amount

LI to HCI – Request for coverage activity for Cross Sell

- Sent – Request of all coverage requests last n days
- Received – All demographic, coverage activity, DRG

Use Case: Customer with Corporate Profile

Revision History:

Date	Author	Changes
2000.03.13	Matthew Doering	Initial specification
2000.05.23	Bobby Holland	Modified formatting and insertion into spec v0.6
2000.08.09	Brad Husick	revisions

Abstract:

A **Business Traveler** (**BT**) needs to rent a car at a major airport location. This **Reservation** will be made through the business traveler's local **Travel Agent** (**TA**). The **Business Traveler** is covered under the **Parent Company**'s (**PC**) corporate **Rental Contract** that also has latitude for personal preferences. In this case the corporate **Rental Contract** allows for a small to mid-sized car, waive all insurance, and no fuel option. This **Reservation** will also use a promotional code from the **Airline Partner** (**AP**) that allows a 1 car class upgrade. The **Business Traveler** can specify a car type in the small to mid-size range and a smoking/non-smoking preference. The **Rental Company** (**RC**) will want to track the **Business Traveler**'s activity by the specific traveler as well as aggregated into a corporate activity register. The rental company needs to report back to the company all activity under that corporate contract. In addition the rental company has a co-marketing agreement with certain airline partners and will want to validate frequent flyer numbers and program participation.

Interactions:

TA to RC – Reservation request.

- Sent – PC agreement ID, requested date/time, location, duration
- Received – PC profile and BT allowable options
- Sent – BT ID and preferences for allowable options (wants mid-size, smoking), Frequent Flyer Number and Airline ID/Flight Number, Promo ID
- Received – Confirmation of reservation with agreed preferences

RC to AP – Report of Frequent Flyer activity

- Sent – BT's Frequent Flyer Number, date/time/Flight Number, Promo ID

RC to PC – Report on Rental Activity

- Sent – BT’s ID, rental details and allowed preferences and preference override for the Promo ID

Use case: Online rewards company

Revision History:

Date	Author	Changes
2000.01.31	Unknown	Initial specification
2000.05.23	Bobby Holland	Change title and formatting for inclusion in spec v0.6
2000.08.09	Brad Husick	revisions

Abstract

This use case covers the interactions between an example online rewards company and various actors. A **Consumer** **(C)** wishes to accrue reward points from an **Online Rewards Company (ORC)**, earned by shopping at various **Vendor** websites **(V)**. The **Online Rewards Company** offers the **Consumer** targeted content when the **Consumer** redeems accumulated points. The targeting is determined from analysis by a **Data Analysis** firm **(DA)**.

C to ORC (Registration)

- Send registration: name, address, email, phone #, occupation, etc.
- Preferences: interests, vendors
- Behavior: frequency, recency, monetary

C to ORC (Notification of Purchase from Vendor)

- RC requests C purchase history from V.
- Send: Name, shipping address, coupon.
- Receive transaction/product info: UPC, ISBN, geography of purchase transaction, channel, return/purchase, unit cost, size, date, time

ORC to DA (Analysis)

- Send: ORC data collected on C (demographics, purchase history, behavioral data) and promotion/offer information
- Receive scoring of RC customer: score and key.

ORC enhances content

- Send: scoring information with key.

ORC serves content

- Send: key to recommendation engine.
- Receive: content identifier to indicate ad/editorial/product to show.

Appendix E: Relationship to Other Standards

The following standards are related to the CPExchange specification.

W3C

XML is used as the encoding standard for the interchange of a Customer Profile. The notion of data typing is used in accordance with forthcoming specifications for an XML Schema Language.

The XML Schema provides a standard way to describe XML content more precisely than is possible with just a Document Type Definition (DTD). CPExchange will provide both XML Schema and DTD definitions of its XML content.

> (XML) T. Bray, J. Paoli, C. M. Sperberg-McQueen (Eds.). "Extensible Markup Language (XML) 1.0 Specification." World Wide Web Consortium, Recommendation. 10 February 1998.

> (XML-Schema1) H. Thompson, D. Beech, M. Maloney, and N. Mendelsohn (Eds.). "XML Schema Part 1: Structures" World Wide Web ConsortiumWorking Draft. 7 April 2000.

> (XML-Schema2) P. Biron, A. Malhotra (Eds.). "XML Schema Part 2: datatypes" World Wide Web Consortium Working Draft. 7 April 2000.

P3P supports a limited set of characteristics defining a Web visitor, with an extensibility mechanism. P3P uses Resource Description Framework (RDF) XML as its interchange format. CPExchange will provide a mapping between its information elements and P3P's data schema.

> (RDF) O. Lassila and R. Swick (Eds.). "Resource Description Framework (RDF) Model and Syntax Specification" World Wide Web Consortium Recommendation. 22 February 1999.

A subset of P3P's specification of privacy levels and policies is used to specify the desired privacy handling of CPExchange data being transferred.

> (P3P) Cranor, L., Langheinrich, M., Marchiori, M., Presler-Marshall, M. and J. Reagle, (Eds.) "The Platform for Privacy Preferences 1.0 (P3P1.0) Specification" World Wide Web Consortium Working Draft. 18 October 2000.

ebXML

The Electronic Business XML (ebXML) initiative Core Components Working Group is currently reviewing this specification. The ebXML Technical Architecture Working Group is currently using the CPExchange UML->XML mapping rules to determine what mapping rules ebXML will recommend and use.

Appendix F: Glossary

A	
ADSI	Active Directory Services Interface
B	
C	
CRM	Customer Relationship Management
CPEXML	CPExchange Markup Language
CXML	Commerce XML is a new set of document type definitions (DTD) for the XML specification
D	
DSML	Directory Service Markup Language is an effort to represent standard LDAP objects and attributes in XML
DTD	Document Type Definition
E	
EBXML	Electronic Business XML initiative
ECML	E-commerce Markup Language
ERM	Enterprise Relationship Management
I	
ICE	Information Content Exchange
IETF	Internet Engineering Task Force
IRM	Internet Relationship Management

ISO	International Standards Organisation
J	
JNDI	Java Naming Directory Interface provides applications written in the Java programming language with a unified interface to multiple naming and directory services in the enterprise http://java.sun.com/a-z/index.html#J
L	
LDAP	Lightweight Directory Access Protocol
P	
P3P	W3C Privacy Preferences Project
R	
RDF	Resource Description Framework
RFC	Request For Comment
S	
SKU	Stock Keeping Unit
SOAP	Simple Object Access Protocol
U	
UML	Unified Modeling Language
URL	Uniform Resource Locator
W	
W3C	World Wide Web Consortium

X	
XML	Extensible Markup Language http://www.xml.com/pub/98/10/guide1.html
XML-EDI	Extensible Markup Language for Electronic Data Interchange
XML-RPC	Extensible Markup Language for Remote Procedure Calls
XSL	Extensible Style Language
XSLT	Extensible Stylesheet Language Transformations