|  |  |  |
| --- | --- | --- |
|  | | |
|  |  |  |

|  |  |
| --- | --- |
| GIEM  Naming and Design Rules.  Date: 28 Oct 2011  Status: Consultation Draft  Version: 0.1  Document Sponsor: Brian Stonebridge | |
|  | |
| attention_pms This document and its attachments are  **Unclassified** | direction_pms |
|  | For further information or questions, please contact the Department of Human Services on 02 61551525 |

ENDORSEMENT/VERSION CONTROL

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CURRENT VERSION | | Consultation Draft 0.1 | DATE | 28 Oct 2011 |
| AUTHORS | | | | |
| Steve Capell |  | |  | |
|  | | | | |
| APPROVAL |  | |  | |
| Scott Wallace |  | |  | |

VERSION CONTROL

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Status /Version** | **Revision date** | **Name of author/ modifier** | **Change summary** | **Distributed to** |
| Consultation Draft 0.1 | 28 Oct 2011 | Steve Capell | First release | GovShare |
|  |  |  |  |  |
|  |  |  |  |  |

Table of contents

[1 Introduction 4](#_Toc307765165)

[1.1 Purpose 4](#_Toc307765166)

[1.2 Audience 4](#_Toc307765167)

[2 Naming Rules 5](#_Toc307765168)

[2.1 Data Element Naming Convention 5](#_Toc307765169)

[2.2 Representation Terms 6](#_Toc307765170)

[2.3 Message & Service Naming Convention 7](#_Toc307765171)

[2.4 Namespace Naming Convention 7](#_Toc307765172)

[2.5 Filename & Schema Location Convention 8](#_Toc307765173)

[3 Design Rules 10](#_Toc307765174)

[3.1 UML Profile 11](#_Toc307765175)

[3.1.1 UML Class 11](#_Toc307765176)

[3.1.2 UML Aggregation 12](#_Toc307765177)

[3.1.3 UML Generalisation 12](#_Toc307765178)

[3.1.4 UML Package 13](#_Toc307765179)

[3.2 XML Design Rules 13](#_Toc307765180)

[3.2.1 UML to XML Mapping Rules 14](#_Toc307765181)

[3.2.2 XML Schema Sample 14](#_Toc307765182)

[3.3 XBRL Design Rules 15](#_Toc307765183)

[3.3.1 UML to XBRL Mapping Rules 16](#_Toc307765184)

[3.3.2 XBRL Taxonomy Sample 17](#_Toc307765185)

List of Figures

[Figure 1 – Naming Rules Context 5](#_Toc307765186)

[Figure 2 - GIEM Design Rules Context 10](#_Toc307765187)

[Figure 3 - XML Schema Sample 15](#_Toc307765188)

[Figure 4 - XBRL Element Definitions Sample 18](#_Toc307765189)

[Figure 5 - XBRL Label Linkbase Sample 18](#_Toc307765190)

[Figure 6 - XBRL Presentation Linkbase Sample 18](#_Toc307765191)

List of Tables

Table 1 - GEIM Representation Terms 7

Table 2 - GIEM Data Element to UML Class Mapping 11

Table 3 - XML Design Rules 14

Table 4 - XBRL design rules 16

1. Introduction

The Government Information Exchange Methodology (GIEM) provides agencies with a suite of tools and methods to help them produce rigorous data exchange specifications for their specific project needs. GIEM is not a data standard in itself, rather it is a method to help leverage the plethora of (often overlapping) national and international standards in order to address specific interchange needs. GIEM comprises three documents:

* **GIEM Development Methodology** describes how to produce an interchange specification. It provides standard modelling templates and includes worked examples.
* **GIEM Naming & Design Rules** (this document) ensures that interchange specifications are consistent across projects. It provides rules for naming of data elements, namespaces, and XML schema design rules.
* **GIEM Governance Framework** provides confidence and trust in the interchange specification. It describes a set of standard project roles, describes the steps in the development process, provides voting rules & issue resolution procedures, and defines a conformance testing framework.
  1. Purpose

The purpose of the GIEM naming conventions is to provide consistency.

* Consistent data element naming rules, allowing data elements from multiple sources to be merged into a single taxonomy.
* Consistent namespace naming rules, allowing information standards to be developed without risk of name collisions.
* Consistent service naming, allowing meaningful government wide service oriented architectures.
* Consistent UML, XML, or XBRL representations, allowing different projects to create interoperable technical representations.

A common challenge for information exchange projects is how to re-use information concepts from different standards and in different representations and still deliver a seamless and consistent interchange specification. The GIEM naming and design rules address this problem.

* 1. Audience

The target audience for this document is:

* Team members (both government and industry) of any project that includes a requirement to develop a standard data exchange technical specification.

Readers should be familiar with the concepts presented in the GIEM Development Methodology document.

1. Naming Rules

The diagram below provides context for each naming rule:

* A “Namespace” provides a globally unique container for one or more data elements.
* Data Elements (typically from multiple namespaces) are used to construct “Messages”.
* A “Service Interface” delivers a specific business function and typically includes a request and a response message.
* Elements, namespaces, messages and service specifications are all contained in files that have a “filename” and a “location”.



Figure – Naming Rules Context

## Data Element Naming Convention

The GIEM naming convention follows the recommendations of a widely recognised meta-data standard (ISO-11179). An element name is constructed as a dot separated list of terms:

***Qualifier\_ObjectClass.[Qualifier\_Property]\*.Representation***

Where

* ObjectClass represents a classification for a related group of concepts (eg "Address" or "WageItems")
* Property represents a more granular concept within the object class (eg "postcode" or "overtime")
* Representation defines the basic type of the data element (eg "code", "text", "date", etc). The allowed list of representation terms is defined in the core data type library.
* The property term can be repeated if the underlying conceptual data model is nested. (eg "Organisation.Address.Postcode.Identifier")
* Both the ObjectClass an the property can be prepended with a qualifier term to restrict meaning (eg "Postal\_Address.Line1.Text"

Element names are unique within a namespace. Therefore the namespace qualified name of any element is also a globally unique identifier for the data element.

**Data Element Meta-Data**

Data elements need more than a meaningful and globally unique identifier to be useful. Each data element MUST also have:

* **A definition.** This is a string of text that provides a concise definition for the data concept.
* **A value domain.** This defines the allowed values for the data element. Allowed values can be defined by a data type (eg “Date”) or an enumeration (eg Person.Gender.Code is allowed to be "Male" or "Female" or "Indeterminate").

And may optionally have:

* Alias. This is one or more alternative names for the same concept. eg "Address.Locality.Text" is also known as "Suburb".

## Representation Terms

GIEM representation terms (often called “classwords” by data modelers) are drawn from a controlled vocabulary because they represent foundation types of concept. The GIEM representation term list is given here.

|  |  |  |
| --- | --- | --- |
| **Representation Term** | **Definition** | **Example usage** |
| Text | A character string (I.e. a finite set of characters) generally in the form of words of a language. | Product.Description.Text |
| Identifier | A character string used to identify and distinguish uniquely, one instance of an object in an identification scheme from all other objects within the same scheme. | Organisation.ABN.Identifier |
| Code | A character string (letters, figures or symbols) that for brevity and / or language independence may be used to represent or replace a definitive value or text of a Property. Not to be confused with Identifier which represents a specific instance of an object. | Person.Birth\_Country.Code |
| Numeric | A number as an integer, decimal, or real number. | Account.InterestRate.Numeric |
| Measure | The numeric value determined by measuring an object. The unit of measure is drawn from UN/ECE Rec 20 (eg KGM = “Kilogram”) | Shipping\_Package.Weight.Measure |
| Quantity | A counted number of non-monetary units possibly including fractions. The counting unit, if applicable, is drawn from UN/ECE Rec 21. | Product.PackageLimit.Quantity |
| Monetary | A number of monetary units specified in a currency where the unit of currency is indicated using 3 alpha codes of UNECE Rec 9 / ISO-4217. | Payslip.GrossPay.Amount |
| BinaryObject | A set of finite-length sequences of binary octets. This Representation Term shall also be used for Data Types representing graphics, pictures, sound, video, etc. | Product.DGCertificate.BinaryObject |
| Indicator | A binary indicator with allowed values of True or False. | Product.GSTExemption.Indicator |
| Date | A particular point in the progression of time in ISO-8601 "zulu" timezone corrected date-time format CCYYMMDDhhmmssZhhmm | PayrollRun.PeriodEnd.Date |

Table - GEIM Representation Terms

## Message & Service Naming Convention

A ***service*** performs some action on a business object and so is named accordingly:

***<Object>.<Action>***

Examples

* FinancialAccount.BalanceEnquiry
* PurchaseOrder.Create

A service is invoked with a requesting message and may optionally provide a responding message. Therefore ***messages*** are named as an extension of the service name.

***<Object>.<Action>.<Intent>***

Examples

* FinancialAccount.BalanceEnquiry.Request
* Employment.ScheduleOfDeductions.Report

## Namespace Naming Convention

A namespace is a globally unique identifier for a related group of concepts. Global uniqueness is normally achieved through use as a URI with a domain name prefix that is owned by the issuing organisation. Namespaces in GIEM use the URL notation but do not necessarily represent a resolvable location.

The GIEM namespace convention is:

***<dnsDomain>/<group>/[<classification>/]\*<name>.<version>[.<type>]***

Where:

* <dnsDomain> is:
  + http://xml.gov.au/ - for all data elements
* <group> is one of:
  + "fdtn" for all data elements that are not domain or agency specific (eg core data types)
  + "icls" for data elements classified into a functional business domain independant of agency (eg payroll types)
  + <agency code> for agency specific reports & services (eg "csa" or "centrelink").
* <classification> is a repeating code that defines the position of the data element within a hierarchical classification scheme.
  + For business domain namespaces this will be taken from the Australian Government Architecture Business Reference Model (AGA BRM) <brmL1>/<brmL2>.
  + For agency reports / services, this will be an agency specific <category> scheme - normally a logical business area within an agency.
* <name> is a meaningful name for the schema:
  + For domain schema, normally a functional name for the granular domain.
  + For agency schema, normally the report / message / service name.
* <version> is
  + a major / minor version identifier of the form "xx.yy" for domain schema
  + a date made of YYYYMM for agency forms, services and common objects.
* <type> is optional but if present should be one of:
  + "data" - for element definition schema
  + "code" - for enumerated codelist schema
  + "dtyp" - for core / primitive type definitions.
  + "message" - for the message schema
  + "service" - for service definitions

**Examples:**

http://xml.gov.au/icls/employment/payroll.01.00.data

http://xml.gov.au/csa/assessment/financialAccount.balanceEnquiry.request.200904

http://xml.gov.au/csa/assessment/financialAccount.balanceEnquiry.200904

**Namespace Meta-Data**

Namespaces MUST include the following meta-data:

* **Name** – A meaningful short name
* **Namespace URI** – the globally unique URI following the namespace naming convention.
* **Version** – major / minor version.
* **Owner** – the organisation that is responsible for maintenance of the namespace.

Namespaces MAY also include any meta-data drawn from AGLS (see http://www.agls.gov.au/ ) and any additional meta-data needed for target technology code generation.

## Filename & Schema Location Convention

There is a very close relationship between a namespace and a filename & location. They are not identical because there can be many files in one namespace and may be hosted at different physical internet locations.

The complete file & location URL convention is:

***<RepositoryURL>/<status>/<group>/[<classification>/]\*<name>[.<type>].<version><fileExtension>***

* <RepositoryURL> is the internet location of the repository server that is hosting the files. For GovDex hosting, that is <http://xml.gov.au/schema>
* <status> is one of:
  + "draft" for all all schema under development and not yet endorsed as dhs standards.
  + "final" for all endorsed and implementable (ie "fit for build") schema.
* <fileExtension> is the letters representing the filetype - eg ".xsd", ".wsdl", ".xls", ".csv" etc.
* <type> is the same as for the namespace but can take can take the following additional values
  + "sample" for sample data files
  + "guide" for implementation guides or other user documentation

All other elements are the same as for namespace naming conventions.

**Examples:**

http://xml.gov.au/schema/final/icls/employment/payroll.01.00.data.xmi

http://xml.gov.au/schema/final/icls/employment/payroll.01.00.data.rtf

http://xml.gov.au/schema/draft/csa/assessment/financialAccount.balanceEnquiry.request.200904.message.xsd

http://xml.gov.au/schema/final/csa/assessment/financialAccount.balanceEnquiry.200904.service.wsdl

http://xml.gov.au/schema/csa/assessment/financialAccount.balanceEnquiry.200904.guide.pdf.

# Design Rules

The GIEM naming conventions are not specific to any technical representation. However a practical data exchange specification should leverage some consistent design time notation and allow deployment to one or more technologies for implementation. The GIEM design rules currently define one design time notation and two deployment technology options:

* A UML (Unified Modelling Language) profile for design time models.
* XML Schema design rules for SOA type implementations.
* XBRL Taxonomy design rules for Reporting type implementations.



Figure - GIEM Design Rules Context

There is no reason why alternative design notations and additional implementation technologies cannot also be supported by GIEM.

The UML profile and XML design rules presented here use the GIEM Labour Relations / Payroll reference model as an example to illustrate the rules. The reference model is available at <https://govshare.gov.au/xmlui/handle/10772/6430>.

The UML profile and XML / XBRL design rules are deliberately kept short and simple in order to improve readability of this document. Projects that wish to use GIEM for their data interchange specification development should request GIEM training from AGIMO and should study the detailed GIEM reference models and example implementations available from GovShare.

For background and detailed information on UML, XSD Schema and XBRL, readers should refer to the corresponding specification website. However some simple guidance is offered here to help projects decide whether they should use XSD or XBRL for deployment of their specifications.

* XSD Schema are typically the technology of choice for Service Oriented Architectures (SOA) where implementers aim to build interfaces for a specific data exchange purpose (eg “create.purchaseOrder” or “validate.identityDocument” or “lodge.exportDeclaration”). Messages are usually deeply nested with a single root element such as <PurchaseOrder>
* XBRL Taxonomies are typically the technology of choice for broad information distribution where many organisations distribute facts to a wide audience, but all facts are drawn from the same taxonomy. The most common use case is corporate financial reports. Messages are usually a very flat list of facts, where each fact is understandable in isolation of others.

In most cases, XML Schema will be the appropriate technology. However an advantage of creating information models using the GIEM UML profile is that both XML Schema and XBRL implementations can be created from the same model.

## UML Profile

The GIEM UML profile provides guidance on how to develop GIEM data models using UML notation. UML is the preferred notation for GIEM designs because

* It is a mature global standard (<http://www.uml.org/>)
* It is widely understood by modellers and has broad tool support.
* UML diagrams are more easily understood (than XML or XBRL) by non-technical project team members.
* Most UML tools can generate technology specific representations including XML Schema.

### UML Class

|  |  |
| --- | --- |
|  | The “Payslip” object is used to illustrate the basic GIEM data element mapping to UML. |

|  |  |  |
| --- | --- | --- |
| **GIEM Data Element** | **UML Mapping** | **Example** |
| Object class | UML class name | Payslip |
| Property and Representation | UML attribute name | GrossPay.Amount (“GrossPay” is the property and “Amount” is the representation. |
| Value domain | UML attribute data type | monetary |
| Definition | UML attribute note | The financial amount of the total gross pay for the payroll event. |
| Qualifier terms | Qualified class name or attribute name | Employee\_Person (“Person” is the object class and “Employee” is the qualifier. |

Table - GIEM Data Element to UML Class Mapping

### UML Aggregation

|  |  |
| --- | --- |
|  | A UML aggregation, indicated by the hollow diamond symbol, can be read as “includes”. For example, a Payslip “includes” one or more WageItems. The aggregation target becomes a property of the source. In the example below, Payslip has 8 simple properties (eg Payslip.NetPay.Amount) and one complex property (Payslip.WageItem.Details). |
|  |  |

### UML Generalisation

|  |  |
| --- | --- |
|  | A UML generalisation, indicated by a hollow triangle, can be read as “is a type of”. For example, “Employee\_Person” is a type of “Person”. The child type (eg “Employee\_Person”) inherits all the properties of the parent. Therefore, in the example below, “Employee\_Person.Birth.Date” is a valid GIEM data concept because the Birth.Date property is inherited by Employee\_Person. |

### UML Package

UML classes are contained in a UML package. Therefore a UML class maps to a GIEM Namespace. Namespace meta-data are maintained as UML Tagged values against the corresponding UML package as shown in the example below.



## XML Design Rules

Since both UML Classes and XML Schema are designed to represent data in an object-oriented way, there is a fairly simple and straightforward mapping between UML and XML Schema. GIEM follows an industry standard design pattern called “venetian blind” where each class is generated as a re-usable XSD complex type and the schema contains only one root element.

### UML to XML Mapping Rules

|  |  |
| --- | --- |
| **UML Construct** | **Default XSD Production Rules** |
| Package | One XML schema file is generated for each UML package. XML Schema attributes such as “targetNamespace” map to UML package tagged values of the same name.  If the UML package contains elements that reference other elements in any other packages, then the XML Schema must include corresponding <import> elements. |
| Class | A complexType definition is generated for each UML class in the package. The element name and type are the same as the class name. An XSD sequence model group is generated to contain UML attributes generated as elements. |
| Attribute | An XSD element is declared for each UML attribute. The element name is set to that of the UML attribute name. The minOccurs and maxOccurs attributes are set to reflect the attribute cardinality. |
| Aggregation | An XSD element is declared for each aggregation owned by a class. The element name is set to that of the association role. The minOccurs and maxOccurs reflect the cardinality of the association. |
| Generalization | For each UML generalisation, an extension element is generated with the base attribute set to the base class name. The UML attributes of the child class are then appended to an all model group within the extension element. |
| <<enumeration>> | A simpleType element is declared for the enumeration class with the name attribute set to the class name. A restriction element is generated with base set to string. Each of the class attributes are appended to the restriction element as XSD enumeration elements with value set to the UML attribute name. |
| Note | An XSD annotation containing and XSD documentation is generated for each UML class note and UML attribute note. |

Table - XML Design Rules

### XML Schema Sample

The following SD schema snippet is generated from the payroll model used for the UML profile example. The points to note are:

* The schema targetNamespace attribute maps to the GIEM namespace / UML package.
* The schema includes namespace declarations and <import> elements for each referenced package. In this example, the payroll model “employee\_person” class references the more abstract “person” class from the GIEM party demographics reference model.
* A <documentation> element is created for each corresponding UML note – thereby providing a fully documented schema.
* A <complexType> is generated for the corresponding UML Class. It includes an XML <element> declaration for each attribute of the class.
* Each <element> includes a type definition which corresponds to either a GIEM core data type or an enumerated code list. For example, the snippet below shows how the general StateTerritory\_code from the GIEM party demographics model (which is based on AS4590) is used as the XSD type for the “EmploymentJurisdiction.Code” element.

|  |
| --- |
| <?xml version="1.0"?>  <**xs:schema** targetNamespace=**"http://xml.gov.au/icls/lr/lrpy/payroll.01.01.data"** xmlns="http://xml.gov.au/icls/lr/lrpy/payroll.01.01.data" xmlns:lrpy.pay.01.01="http://xml.gov.au/icls/lr/lrpy/payroll.01.01.data" xmlns:pyde.01.01="http://xml.gov.au/dhs/icls/py/pyde/partyDemographics.01.01.data" xmlns:cdt="http://xml.gov.au/dhs/fdtn/cdt.01.01.dtyp" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:lrpy.pay.dt.01.01="http://xml.gov.au/dhs/icls/lr/lrpy/payroll.01.01.code" xmlns:pyde.dt.01.01="http://xml.gov.au/dhs/icls/py/pyde/partyDemongraphics.01.01.code">  <xs:annotation>  <xs:documentation>The payroll information domain is a reference model for payroll related information elements. it is designed to map closely to industry payroll software systems. Government data collection requirements in the payroll domain should be derived from this model. Some derivations will be quite simple (eg child support deduction amount for an employee for a period). others will be more complex (eg state payroll tax obligation for an employer for a period).  </xs:documentation>  </xs:annotation>  ...  ...  <**xs:complexType** name="**Employee\_Person**">  <xs:annotation>  <xs:documentation>The employee person element carries personal identification information and links to other payroll related elements. An employee may have an employement relationship with multiple employers. "Employee\_Person" is a type of "Person" and inherits all properties of "Person"  </xs:documentation>  </xs:annotation>  <xs:complexContent>  <**xs:extension** base="pyde.01.01:Person">  <xs:sequence>  <**xs:element** name="**EmployeeID.Identifier**" type="**cdt:identifier**" minOccurs="1" maxOccurs="1">  <xs:annotation>  <xs:documentation>The Employee ID, also known as "Payroll ID" is a unique identifier for an employee within a single employers payroll system. This identifier is usually meaningful only to the emplyment relationship.  </xs:documentation>  </xs:annotation>  </xs:element>  <**xs:element** name="**EmploymentJurisdiction.Code**" type="**pyde.dt.01.01:StateTerritory\_code**" minOccurs="0" maxOccurs="1">  <xs:annotation>  <xs:documentation>The Australian State or Territory of primary employment. Set to "overseas" if not employed within Australia. This element is used for payroll tax jurisdiction calculations. it's value is controlled via a code list.  </xs:documentation>  </xs:annotation>  </xs:element> |

Figure - XML Schema Sample

## XBRL Design Rules

XBRL represents concepts as a flat data dictionary without any hierarchy. Therefore the mapping from structured UML information models to XBRL is essentially a flattening exercise. XBRL also separates meta-data such as definitions from the element declaration. Instead meta-data is contained in a separate file called a linkbase which references the schema containing element declarations.

Note that XBRL generated from GIEM UML models using the design rules presented here will be conformant to the SBR (see [www.sbr.gov.au](http://www.sbr.gov.au)) definitional taxonomy architecture.

### UML to XBRL Mapping Rules

|  |  |
| --- | --- |
| **UML Construct** | **Default XBRL Production Rules** |
| Package | Each UML package will map to at least three XBRL Taxonomy files:   * One XBRL schema file containing flattened element declarations. Schema attributes such as “targetNamespace” map to UML package tagged values of the same name. * One XBRL Label linkbase file is generated containing all element definitions as defined in the UML Notes field. * One XBRL Presentation linkbase containing the hierarchical relationships between data elements (eg that “wageItem” is a child of “payslip”)   If the UML package contains elements that reference other elements in any other packages, then the XML Schema must include corresponding <import> elements. |
| Class | An Abstract XBRL element definition is created for each UML class.. |
| Attribute | An XBRL element declaration is created for each UML attribute. The name of the element is the fully expanded pathname of the UML attribute. In addition   * a label link is added to the label linkbase file and populated with the UML note. * a presentation link is added to the presentation linkbase file relating the schema element to the Abstract element representing the UML class. |
| Aggregation | Aggregation is a “contains” relationship and so XBRL generation will follow the tree to the parent class (possibly over multiple aggregations) to create a flattened XBRL element name.  For example, “WageItem” is contained in “Payslip” therefore the “HoursWorked.Numeric” attribute of “WageItem” is expanded to become XBRL element <Payslip.WageItem.HoursWorked.Numeric>. |
| Generalization | Generalisation is a “type of” relationship and the child inherits all properties of the parent. Thererfore the XBRL generation rules are to create an XBRL element for all attributes of a class plus all attributes of any parent class.  For example, the property “Birth.Date” of class “person” is inherited by “Employee\_Person” therefore the XBRL will include an element <Employee\_Person.Birth.Date> |
| <<enumeration>> | The same as XML Schema: A simpleType element is declared for the enumeration class with the name attribute set to the class name. A restriction element is generated with base set to string. Each of the class attributes are appended to the restriction element as XSD enumeration elements with value set to the UML attribute name. |
| Note | A label link entry is created in the label linkbase file for every UML note. |

Table - XBRL design rules

### XBRL Taxonomy Sample

The snippets below show fragments of the XBRL files generated from the UML Payroll model. Points to note are:

* There is one schema file and at least two linkbase files.
* The schema element is similar to the structured XML in as much as it must declare all related namespaces. It also declares the xbrl instance namespace.
* There is an <xsd:appinfo> element in the schema file that references the label and presentation linkbase files.
* An abstract element is created for each UML class. It is required in order to provide links to definitions and to allow presentation linkbases to navigate the hierarchy.
* Concrete elements are generated for each UML attribute. The name of the elements are the full pathname to the corresponding UML attribute including class name plus any containing class names indicated via UML aggregation. XBRL element names are usually much longer than corresponding structured XML element names due to the flattening.
* Each element has an id attribute (eg “DE46”) which is used by the linkbase files to reference the element.
* The label linkbase essentially links the definition text from the UML note to the correct element using the id.
* The presentation linkbase essentially contains a set of parent-child relationships that re-create the UML structure.

|  |
| --- |
| <xsd:schema xmlns:xbrli="**http://www.xbrl.org/2003/instance**" xmlns:link="http://www.xbrl.org/2003/linkbase" xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:lrpy.pay.01.01="http://xml.gov.au/icls/lr/lrpy/payroll.01.01.data" xmlns:lrpy.pay.dt.01.01="http://xml.gov.au/icls/lr/lrpy/payroll.01.01.code" xmlns:pyde.dt.01.01="http://xml.gov.au/icls/py/pyde/partyDemongraphics.01.01.code" attributeFormDefault="unqualified" elementFormDefault="qualified" targetNamespace="**http://xml.gov.au/icls/lr/lrpy/payroll.01.01.data**" xmlns:xsd="http://www.w3.org/2001/XMLSchema">  <xsd:annotation>  <xsd:appinfo>  <**link:linkbaseRef** xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase" xlink:href="**payroll.01.01.labLink.xml**" xlink:role="http://www.xbrl.org/2003/role/**labelLinkbase**Ref" xlink:type="simple" />  <**link:linkbaseRef** xlink:arcrole="http://www.w3.org/1999/xlink/properties/linkbase" xlink:href="**payroll.01.01.presLink.xml**" xlink:role="http://www.xbrl.org/2003/role/**presentationLinkbase**Ref" xlink:type="simple" />  </xsd:annotation>  </xsd:appinfo>  <xsd:element id="CLASS7\_detail" xbrli:periodType="instant" **abstract="true"** name="**Employee\_Person.Details.Abstract**" substitutionGroup="xbrli:item" type="xbrli:stringItemType" />  <xsd:element id="**DE46**" xbrli:periodType="duration" block="substitution" name="**Employee\_Person.EmployeeID.Identifier"** substitutionGroup="xbrli:item" type="xbrli:stringItemType" />  <xsd:element id="**DE48**" xbrli:periodType="duration" block="substitution" name="**Employee\_Person.EmploymentJurisdiction.Code**" substitutionGroup="xbrli:item" type="pyde.dt.01.01:StateTerritory\_code" />  <xsd:element id="DE123" xbrli:periodType="duration" block="substitution" name="**Payslip.WageItem.Description.Text**" substitutionGroup="xbrli:item" type="xbrli:stringItemType" />  <xsd:element id="DE124" xbrli:periodType="duration" block="substitution" name="**Payslip.WageItem.HoursWorked.Numeric**" substitutionGroup="xbrli:item" type="xbrli:decimalItemType" /> |

Figure - XBRL Element Definitions Sample

|  |
| --- |
| <link:loc xlink:label="loc\_DE46" xlink:href="**payroll.01.01.data.xsd#DE46**" xlink:type="locator" />  <link:labelArc xlink:order="1.0" xlink:arcrole="http://www.xbrl.org/2003/arcrole/**concept-label**" xlink:from="loc\_DE46" xlink:to="lbl\_DE46" xlink:type="arc" />  <link:label xlink:label="lbl\_DE46" xlink:role="http://sbr.gov.au/fdtn/sbr.01.02.tech/businessDefinition" xlink:type="resource" xml:lang="en">***The Employee ID, also known as "Payroll ID" is a unique identifier for an employee within a single employers payroll system. This identifier is usually meaningful only to the emplyment relationship.***  </link:label>  <link:loc xlink:label="loc\_DE48" xlink:href="**payroll.01.01.data.xsd#DE48**" xlink:type="locator" />  <link:labelArc xlink:order="1.0" xlink:arcrole="http://www.xbrl.org/2003/arcrole/**concept-label**" xlink:from="loc\_DE48" xlink:to="lbl\_DE48" xlink:type="arc" />  <link:label xlink:label="lbl\_DE48" xlink:role="http://sbr.gov.au/fdtn/sbr.01.02.tech/businessDefinition" xlink:type="resource" xml:lang="en">***The Australian State or Territory of primary employment. Set to "overseas" if not employed within Australia. This element is used for payroll tax jurisdiction calculations. it's value is controlled via a code list.***  </link:label> |

Figure - XBRL Label Linkbase Sample

|  |
| --- |
| <link:loc xlink:label="**loc\_CLASS7\_detail**" xlink:href="payroll.01.01.data.xsd#CLASS7\_detail" xlink:type="locator" />  <link:loc xlink:label="loc\_DE46" xlink:href="**payroll.01.01.data.xsd#DE46"** xlink:type="locator" />  <presentationArc xlink:type="arc" xlink:arcrole="http://www.xbrl.org/2003/arcrole/**parent-child**" xlink:from="**loc\_CLASS7\_detail**" xlink:to="loc\_DE46" order="26" xmlns="http://www.xbrl.org/2003/linkbase" />  <link:loc xlink:label="loc\_DE47" xlink:href="**payroll.01.01.data.xsd#DE47**" xlink:type="locator" />  <presentationArc xlink:type="arc" xlink:arcrole="http://www.xbrl.org/2003/arcrole/**parent-child**" xlink:from="**loc\_CLASS7\_detail"** xlink:to="loc\_DE47" order="27" xmlns="http://www.xbrl.org/2003/linkbase" /> |

Figure - XBRL Presentation Linkbase Sample