Product Engineering and Technology Group

eCollege Annotated Guide to the IMS Enterprise Specification and eCollege Extensions

4900 South Monaco Street
Denver, CO 80237

CONTENTS

1. INTRODUCTION 1

1.1. OVERVIEW AND SCOPE 1
1.2. DOCUMENT ORGANIZATION 1
1.3. BRIEF INTRODUCTION TO XML 2
1.4. DEFINITIONS 4

2. ORGANIZATION OF THE COMPLETE IMS ENTERPRISE V1.1 WITH ECOLLEGE EXTENSIONS 6

2.1. <ENTERPRISE> ELEMENTS 6
2.2. <PROPERTIES> ELEMENTS 6
2.3. <PERSON> ELEMENTS 6
2.4. <GROUP> ELEMENTS 8
2.5. <MEMBERSHIP> ELEMENTS 9
2.6. <SOURCEDID> ELEMENTS 9
2.7. EXTENSIONS 10

3. IMS ELEMENTS CONSUMED BY THE ECOLLEGE API FOR SIS 11

3.1. <ENTERPRISE> ELEMENTS 11
3.2. <PROPERTIES> ELEMENTS 11
3.3. <PERSON> ELEMENTS 12
3.4. <GROUP> ELEMENTS 19
3.5. <MEMBERSHIP> ELEMENTS 20
3.6. COMMON <SOURCEDID> ELEMENTS 23

4. ECOLLEGE EXTENSIONS 23

4.1. <RESULT> ELEMENTS 24
4.2. <PERSONPROPERTY> ELEMENTS 25

5. SCHEMA NAMESPACES AND URIS 26

5.1. IMS ENTERPRISE V1.1 26

6. MORE ABOUT SOURCEIDS 27

6.1. WHAT IS SOURCEDID USED FOR WITHIN ECOLLEGE? 27
6.2. IS SOURCEID ASSIGNED FOR DELIMITED FILES? 27
6.3. ARE THERE LIMITATIONS WITH USING BOTH THE BATCH AND WEB SERVICES APIS CONCURRENTLY? 27
6.4. HOW IS THE SOURCEID MAPPED TO A USER IN THE ECOLLEGE SYSTEM? 28

7. QUICK REFERENCE 29

1. Introduction

1.1. Overview and scope

This document provides a detailed description of how eCollege uses the IMS Enterprise Specification v1.1 both to take in information from a Student Information System (SIS) and to provide information in return. This is not a complete discussion of the IMS Enterprise specification, nor does it address the differences between the 1.01 and 1.1 versions. For complete information on the work being done by IMS Global Learning Consortium and on the IMS Enterprise specification, please visit http://www.imsglobal.org.

While readers of this document are expected to know and understand XML and XML Schemas, a brief refresher is provided.

In the section detailing the IMS Enterprise elements that are relevant to eCollege's implementation, we have provided a description of how eCollege interprets the data provided by that element, and, in particular, any differences in required information.

1.2. Document organization

Section 2 contains an overview of the major sections of the IMS Enterprise v1.1 specification. Diagrams of the elements in each section are provided, along with a brief description of the general purpose of the section. Detailed discussions of the exact use of each element, the data each contains, and how that information is consumed by the eCollege API for SIS are included in Section 3. Any element found in the diagrams in Section 2 but which is not discussed in Section 3 should be assumed to be ignored by the eCollege system. Providing optional elements which are not consumed by the eCollege system will not result in any errors.

There are, however, some IMS optional elements that **are required** by the eCollege API for SIS. Section 2, which is taken directly from the IMS Enterprise v1.1 specification, refers to these elements as optional elements. In Section 3, however, these elements are clearly marked as ***eCollege required,*** and include an explanation of exactly why the API for SIS requires the information to function correctly.

The eCollege extensions to the IMS specification are also discussed in each section.

The description of each element consumed by the eCollege API for SIS includes the following information:

- *Diagram:* a visual summary of the element, its attributes, and its children. Generally, diagrams are only provided for elements with children.
- *Description:* a description of the purpose of the element in the IMS Schema
- *eCollege implementation:* any information on how the eCollege implementation differs from the general description
- *Data type:* what type of data is allowed in the element
- *Multiplicity:* how many of the element are allowed in the parent element, and whether the element is optional
- *Attributes:* a list of attributes in the element, along with a short description and data type information for each attribute
- *Elements:* a list of child elements that the element may contain. The child elements will be detailed in element sub-sections of their own under their parent element's section.

- *Example:* a fragment of an XML document showing the element, its attributes, and children with data. Generally, examples are provided for any element which has children; further descendants will be indicated by an ellipsis (…).

In addition, extension elements include information about:

- *Document type:* indicates if the element is expected incoming, outgoing, or log documents

- *IMS parent:* the name(s) of the IMS Enterprise Schema elements where the extension may be used. All extensions must appear in an extension element, so that may be assumed. For example, the full path for container for an extension to `person` is `enterprise/person/extension`.

1.3. Brief introduction to XML

A basic understanding of XML is critical to successfully creating an IMS Enterprise-compliant document. This section is intended only to provide a minimal understanding for those new to XML or for those who need a brief refresher. A full description of the XML language can be found at http://www.w3.org/TR/2004/REC-xml11-20040204.

Basically, XML is a means of representing relational data through markup in a text file. The actual markup looks very much like HTML, although the grammar rules for an XML document are more strict. An XML document is made up of *elements* – words or names contained within angle brackets, which in turn contain the data. The structure of the document – the names of the elements and their relation to one another – is intended to provide the meta-data for the actual content.

1.3.1. Elements

The basic rules for elements are:

- There must be a single root element for the XML document (in this case, `<enterprise>`)
- Every element must be closed. This can take the form of a pair of tags, one to open and one to close `<enterprise>…</enterprise>,` or a single self-closing tag `<enterprise/>`
- Element names are case sensitive. `<enterprise/>` is not the same element as `<Enterprise/>`. In the IMS specification, all names are all lowercase. All eCollege extensions follow this convention.
- There are a number or rules about what can and can't be done with the name of an element, all covered by the W3C's specification. Of note here is the fact that a name cannot contain spaces. The convention in the IMS specification is to simply remove the space, rather than replace it with some other character, so, `phone number` becomes `phonenumber`, for example.

1.3.2. Attributes

In addition to elements containing information, attributes can be used to provide some additional meta-data. Typically attributes don't contain data, but rather information about the data – for example, the element `<phonenumber>` might have an attribute `type` which is used to tell you that the number is a home, work, fax, or cell phone number. The rules for attributes are:

- All attribute values must be contained in double quotation marks. Where `<table width=640>…</table>` is acceptable in HTML, in XML it must be written as `<table width="640">…</table>`
- All attributes must have a value. While you might be familiar with `<select multiple>…</select>` in HTML, in XML it must be something like `<select multiple="true">…</select>`
- Just like element names, attribute names cannot contain spaces.

1.3.3. White space

XML generally treats all white space as something to ignore. Generally, any multiple spaces, tabs, carriage returns, line breaks, etc. are collapsed down to a single space. Between the end of one element and the start of another, XML doesn't even bother with the single space. So what we might write as

```
<children>
     <person>
              Owen,
              A Monk
     </person>
     <person>
              Edmond,
              Earl of Richmond
     </person>
     <person>
              Jasper,
              Earl of Pembroke,
              Duke of Bedford
     </person>
     <person>
              Margaret
     </person>
</children>
```

for convenience and readability, the XML parser sees as

<children>person>Owen, A Monk</person><person>Edmond, Earl of Richmond</person><person>Jasper, Earl of Pembroke, Duke of Bedford</person><person>Margaret</person></children>

(except without the line breaks, which the word processor added). There are ways to force white space to be included, but generally you should assume that all white space will be collapsed.

1.3.4. Reading XSD diagrams

All of the diagrams in this document are standard diagrams generated from the XML Schema Definition (XSD) for the IMS Enterprise v1.1 plus eCollege's extensions. Following is a very brief summary of the basic diagramming conventions used in this document.

Elements: square boxes with a blue diamond in the upper left corner indicate elements.

Attributes: rounded boxes with a blue circle in the upper left corner indicate attributes.

Data types: the word in the gray bar at the bottom of the box indicates the general data type of the element or attribute.

Children: all child elements and attributes of an element are shown to its right. Elements are bracketed by red lines, while attributes are contained within a red box.

Multiplicity: the number of times an element or attribute may appear is indicated by the symbol in a circle to the left of the box for the element.

Question Mark – optional element – may appear zero or one times
Asterisk – optional element – may appear zero or many times
No circle – required element – must appear once and only once
D – optional, with a default value – if not defined the default is assumed
Plus sign – required element – must appear at least once; may appear many times

1.3.5. Data types

`elements` – the element may only contain other elements `integer` – any whole number value, positive, negative, and including 0. Decimals are not allowed, even if the number evaluates to a whole number. Legal values might be `10`, `0`, or `-2345678901234567890`. Illegal values include `A`, `2.0`, and `3.1415926`.

`NMTOKEN` – an XML 1.0 "Nmtoken" (Name token) – usually this indicates an enumeration where each item is composed of characters allowed in an XML name as a single token (no spaces allowed).

`positiveInteger` – an `integer` whose value is greater than `0` (`0` is not allowed).

`string` – character data. The number following indicates the maximum number of characters allowed, so string 256 indicates that the string may be 1-256 characters long. No following number means that the size of the string is unlimited.

1.4. Definitions

API

An Application Program Interface (API) is defined as a set of routines, protocols, and tools for building software applications. It is intended to provide a stable interface for other software that insulates the software from changes in the underlying implementation.

Client Sort String

A string value used to designate an Enrollable Node in the eCollege system. The client sort string is a dotted series of short identifiers used to define a point in a hierarchy (for example, "INSTITUTION.SCHOOL.DEPARTMENT.") *Note that the client sort string always ends in a dot.*

Course Call Number

An identifier provided by the EP which is used to identify a course in the eCollege LMS for the purposes of enrolling students. A course in the eCollege LMS may have more than one Course Call Number. The mapping between the Course Call Number and the course is maintained by eCollege.

Enrollable Node

A point in the eCollege system at which a user can be enrolled. One EP may have many enrollable nodes.

EP

An Educational Partner is an eCollege customer.

IMS

The IMS Global Learning Consortium develops and promotes the adoption of open technical specifications for interoperable learning technology. Several IMS specifications have become worldwide *de facto* standards for delivering learning products and services. IMS specifications and related publications are made available to the public at no charge from www.imsglobal.org**.** No fee is required to implement the specifications.

IMS Enterprise v1.1

A specific XML schema for passing learner and enrollment information as defined by the IMS and supported by the eCollege API for SIS. Often referred to as the IMS schema.

Incoming document

Any document sent to the eCollege API for SIS by an EP.

LMS

Learning Management System

Log document

A document generated by eCollege in response to an incoming document. The Log document is the incoming document plus information about the processing performed by eCollege, generally using the `result` extension element.

SIS – Student Information System

This is the back-end system the EP uses to track student information. This may be a home-grown solution, or a third-party vendor. In the case of a third-party vendor, eCollege may already have experience integrating with their system; your Client Services consultant can provide more information about vendors with whom such out-of-the box integration is available.

XML – Extensible Markup Language

XML is a pared-down version of SGML designed especially for Web documents. It allows designers to create their own customized tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations. Groups such as IMS may define a standard schema for exchange of data.

XSD – XML Schema Definition

XSD is an XML-based format for describing the acceptable structure of an XML document based on standards set by the W3C.

2. Organization of the complete IMS Enterprise v1.1 with eCollege extensions

2.1. <enterprise> Elements

Description: The enterprise element is the root of the XML document, and is required by the IMS Enterprise Specification. Its children (comments, properties, person, group, and membership) are summarized in the sections below. The person, group, and membership elements comprise the meat of the data, while properties provides general information about the entire document.

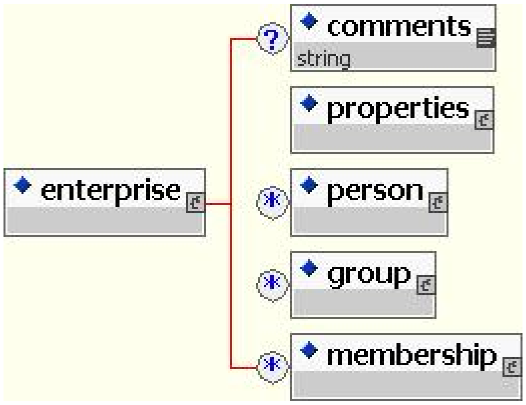


Fig. 1 – <enterprise> element

2.2. <properties> Elements

Description: Provides general information about the packaging of the data in the document, and information about the source and target systems in the data exchange. Information here applies to the entire document. eCollege does not support any extensions to the properties element.

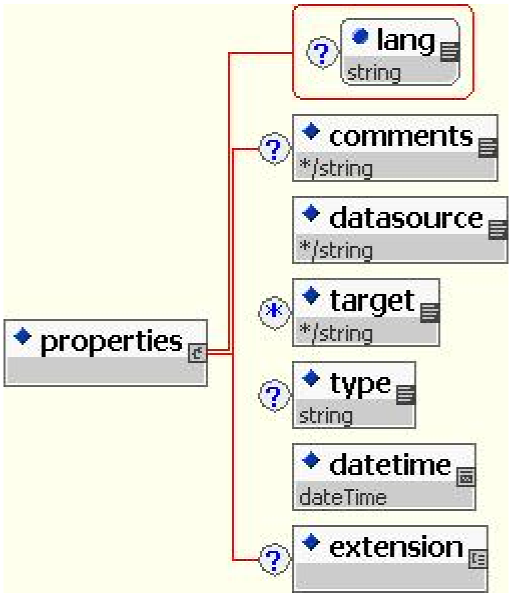


Fig. 2 – <properties> element

2.3. <person> Elements

Description: Each person element describes one user on whom the eCollege API for SIS is going to operate. The child elements provide all the necessary information about who that person is, but do not define the user’s relationship to the EP, enrollments in courses, etc.

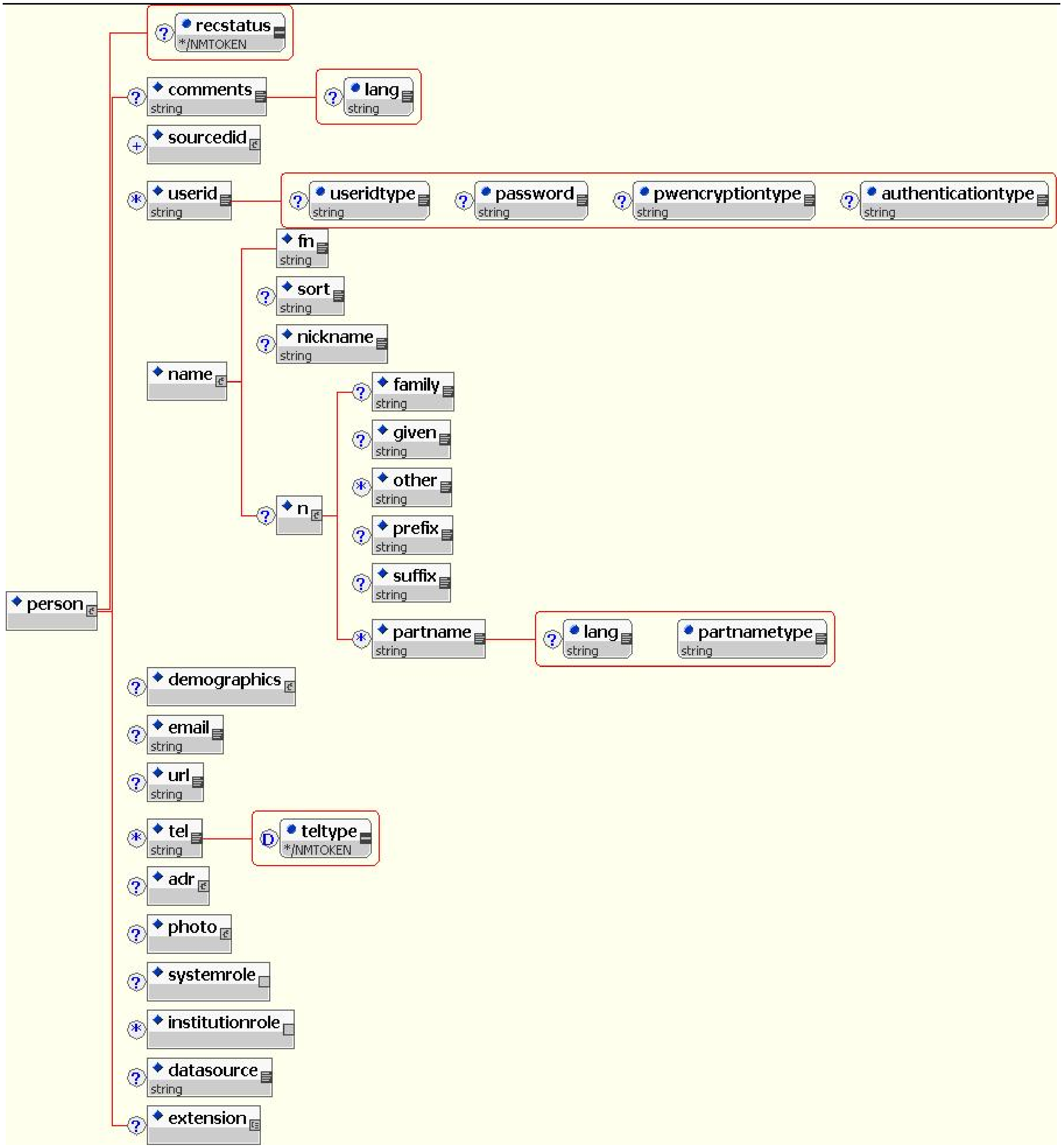


Fig. 3 – <person> element

Note that the <institutionrole> and <systemrole> elements are not consumed by the eCollege API for SIS. Because eCollege allows for an EP to be divided up into several layers of organization, as well as allows users to have different roles in each part of the organization, without some sort of organizational context, <systemrole> and <institutionrole> are meaningless concepts. Instead, system and institution roles are handled by specific <membership> relationships. Please see the membership elements in Section 3 for specific information on how the eCollege API for SIS handles this information.

2.4. <group> Elements

Description: A generic container for things that can be used to group other things. In the eCollege system, groups are either Courses or Enrollable Nodes.

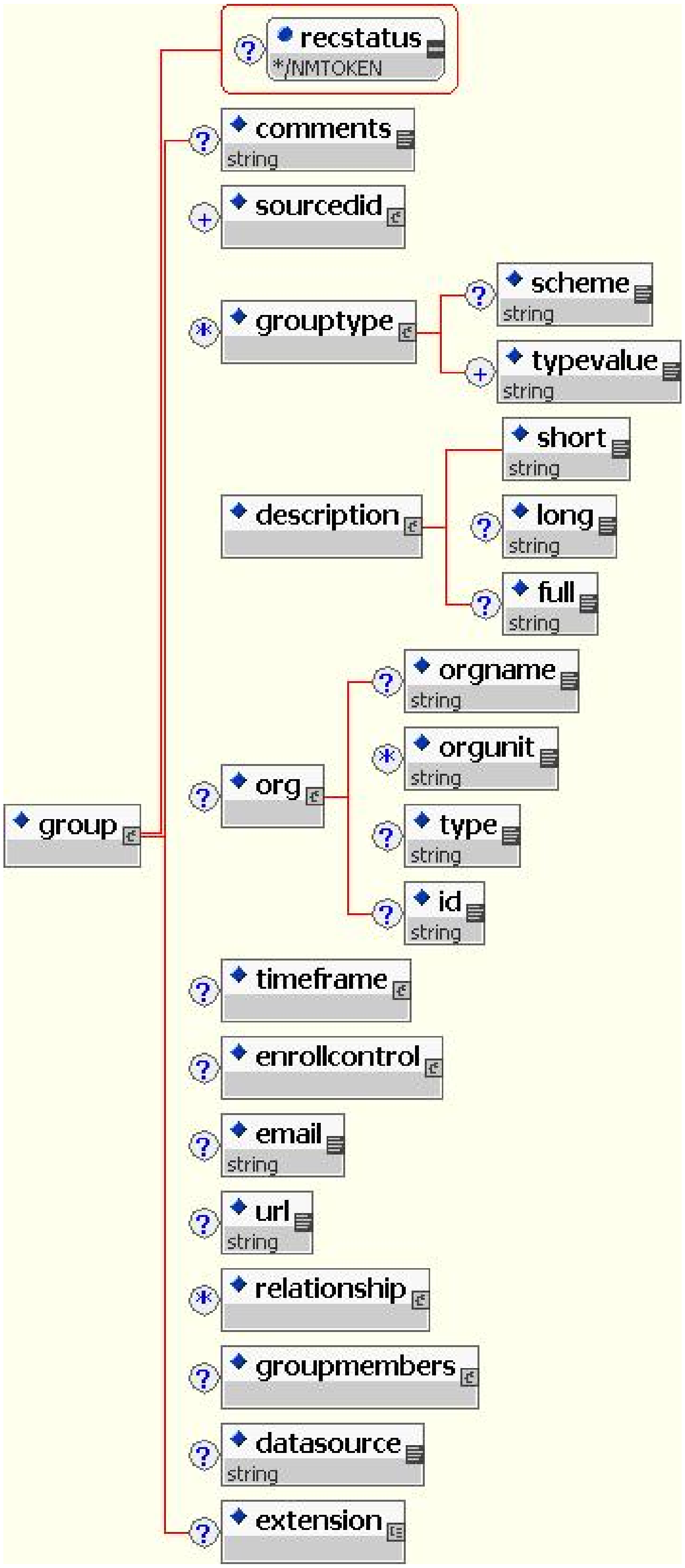


Fig. 4 – <group> element

2.5. <membership> Elements

Description: A container for defining the relationship between the member of a group and the group. Members of groups can be persons or groups. A group or person can be a member of any number of groups. The group and member are identified by their sourcedids.

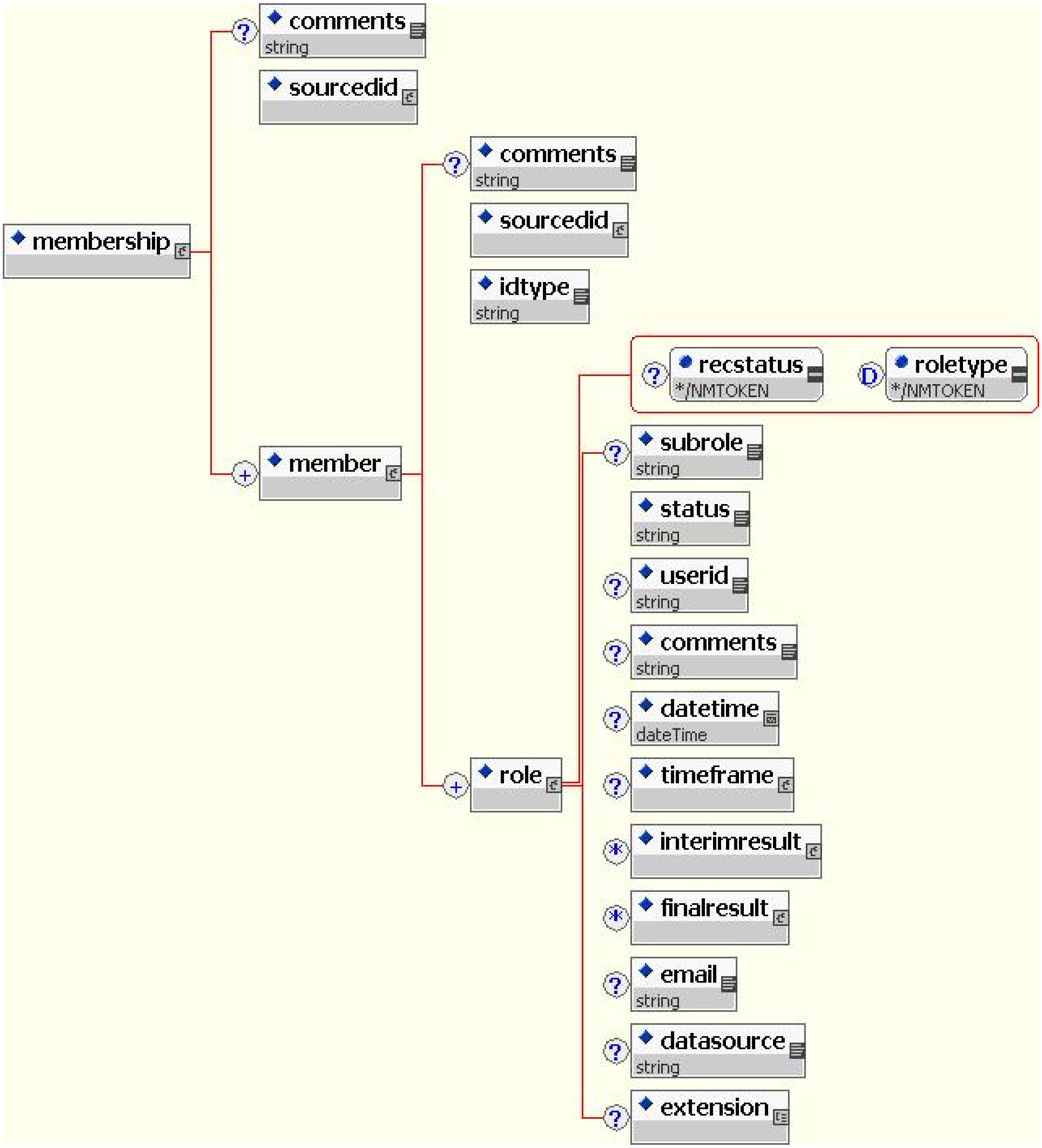


Fig. 5 – <membership> element

2.6. <sourcedid> Elements

Description: The ID of the data object as defined by the source system. The sourcedid is key to relating data elements to one another in the membership data structure. The sourcedid must uniquely identify the person or group in the document.

Note: The SourceID represents (or maps to) a single user in the eCollege system. An individual with multiple roles is not supported meaning, you cannot have multiple logins and passwords for a single user.

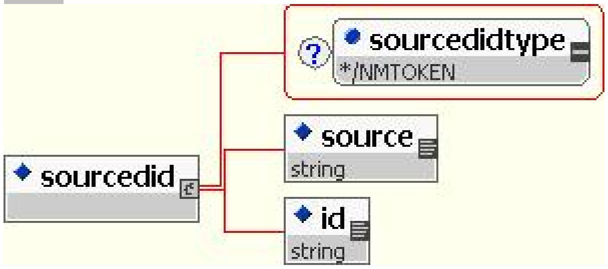


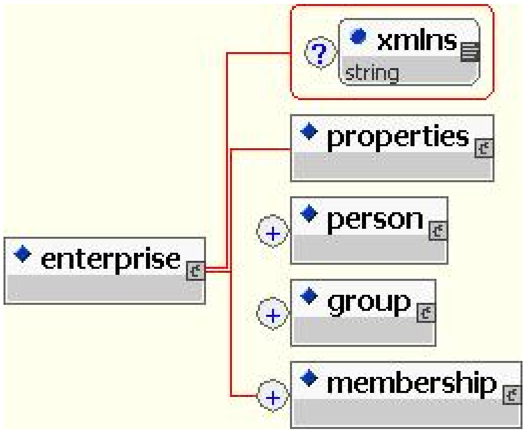
Fig. 6 – <sourcedid> element

2.7. Extensions

All of the major data structures, `properties`, `person`, `group`, and `membership` allow for custom extensions to the IMS Enterprise v1.1 specification within the context of the descendant `extension` element. The eCollege system will ignore any and all extensions of which it is not aware. eCollege does make use of a small set of its own extensions to accept and report more information about persons and to report back information about any actions taken on incoming requests. Details about all eCollege extensions are provided in Section 3, below.

3. IMS elements consumed by the eCollege API for SIS

3.1. <enterprise> Elements



Description: The `enterprise` element is the root of the document and the container for all the data objects of the IMS Enterprise Schema.

eCollege Implementation: The `person`, `group`, and `membership` elements are each optional under the IMS specification, but logically at least one of them must be present. For the purposes of eCollege, there must be at least one instance of each of the three elements. Because a person cannot exist within the eCollege system without being in a defined context beyond just the EP, those context(s) are defined in the `group` elements, and the relation of the `person` to the `group` is defined in a `membership` element. As a result, all three must be present in every IMS Enterprise document sent to eCollege, and are present in every document created by eCollege.

Data type: Elements

Multiplicity: Single, Required

Attributes:

- `xmlns` (optional) – the enterprise element should reference the namespace for the document: http://schemas.ecollege.com/ims_epv1p1.xsd

Elements:

- `properties`
- `person`
- `group`
- `membership`

Example:

```
<enterprise
xmlns="http://schemas.ecollege.com/Common/2006/01/ims_epv1p1.xsd">
     <properties>...</properties>
     <person>...</person>
     <group>...</group>
     <membership>...</membership>
</enterprise>
```

3.2. <properties> Elements

Description: Information about the entire document and the data being exchanged between the SIS and the eCollege API.

eCollege Implementation: eCollege ignores all elements within `properties`.

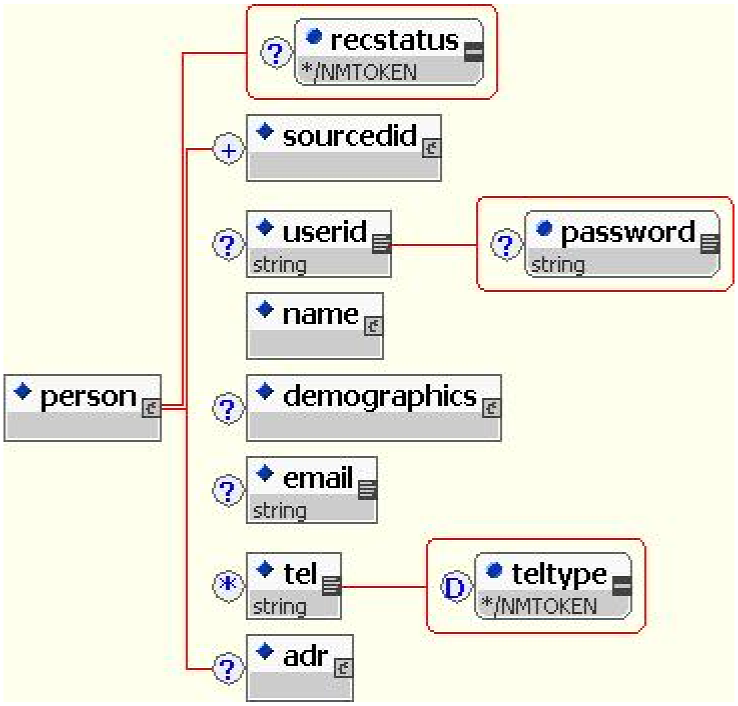
Data type: Elements

Multiplicity: Single, Required

Attributes:

- lang (not currently used) – identifies the language used for the properties. Vocabulary is based on the ISO639 standard.

3.3. <person> Elements



Description: The container for information about a particular person relevant to the eCollege LMS.
eCollege Implementation: The majority of the information provided in the IMS specification is ignored by eCollege (see the child elements list). Extensions to the person element are supported to allow the EP and eCollege to transfer specific information necessary for the EP and information agreed upon between the two organizations.This element is required in every document sent to the eCollege API for SIS so that eCollege can identify and/or create the user.
Data type: Elements
Multiplicity: Single, **eCollege Required**
Attributes:

- recstatus (optional and ignored by eCollege)

Elements:

- sourcedid
- name
- email
- tel
- adr

Example:

```
<person>
    <sourcedid>
        <source>Muppet University</source>
        <id>KERM148</id>
    </sourcedid>
    <userid password=”BeingGreen”>Kfrog1</userid>
    <name>...</name>
    <email>frogk@Disney.com</email>
    <tel teltype=”Voice”>303-555-1212</tel>
    <adr>...</adr>
    <extension>…</extension>
</person>
```

3.3.1. <sourcedid>

Description: The identifier of the `person` as defined by the source system. The `sourcedid` must uniquely identify the person within the document so that it can be used as a referential key between one or more `member` elements and the `person`.

eCollege Implementation: The eCollege SIS API will permanently associate the provided `sourcedid` with the user in our datastore.

Note: The SourceID represents (or maps to) a single user in the eCollege system. An individual with multiple roles is not supported meaning, you cannot have multiple logins and passwords for a single user. However, a given user can be associated with more than one `sourcedid`, as long as they are from *different sources.* Because associating pre-existing users with a `sourcedid` for the first time is a complicated process, please see the Developer's Guide/SampleCode/Use cases and test data/Sample code help topics for details and example.

Data type: Elements
Multiplicity: Single, Required
Attributes: See Section 3.6 for details of this common data structure.
Elements: See Section 3.6 for details of this common data structure.

3.3.2. <userid>

Description: The `person`'s ID to access the eCollege LMS.

eCollege Implementation: Also often called the "Login ID", the `userid` is the unique value that eCollege uses to identify a `person` within the context of an EP in the eCollege LMS. Because the IMS Enterprise Schema specifies this element as optional, the eCollege SIS for API will attempt to identify a `person` by the `sourcedid/id` value if, and only if, the `userid` is absent. *Also note that the string lengths for the `userid` and `password` are more restrictive that the IMS Enterprise Schema.*

Because the `userid` is the account a person uses to access the eCollege LMS, and that account is what is associated with enrollments (through the `membership` structure), while it is possible for a physical person to have more than one login that they might use, it is **not** possible for a `person` element to have more than one `userid`. As a consequence, while the IMS specification for multiplicity is Many, Optional, the eCollege implementation is Single, Optional. Because we cannot guarantee that the first userid element will always be used, no more than one userid element should be included when sending data to eCollege.

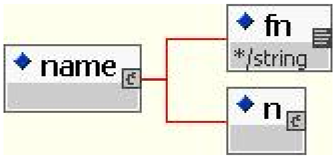
.

Data type: string 255 – invalid characters are: **`%] [+<>";'=:/|\`** and any white space .
Multiplicity: Single, **eCollege Required**
Attributes: eCollege currently supports only the `password` attribute.

- `password` (optional) – the password used to validate the person when logging in to the eCollege LMS. The password may be left blank if the person does not need to be created in the eCollege LMS, or the EP has opted to have random passwords generated. The length can be up to 50 characters. Invalid characters are: **`%] [+<>";'=:/|\_`** and any white space .

Elements: None

3.3.3. <name> Elements



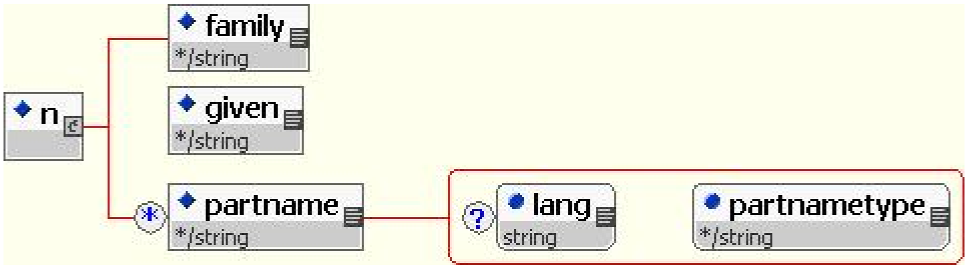
Description: Data element for the name of the person.
eCollege Implementation: No additional information.
Data type: Elements
Multiplicity: Single, Required
Attributes: None
Elements: None
Example:

```
<name>
    <fn>Mr. Kermit The Frog</fn>
    <n>…</n>
</name>
```

3.3.3.1. <fn>

Description: The complete formatted `name` of the `person.`
eCollege Implementation: eCollege SIS for API does not make use of this value.
Data type: string 256
Multiplicity: Single, Required
Attributes: None
Elements: None

3.3.3.2. <n> Elements



Description: The `name` of the `person` broken into all of its distinct components.
eCollege Implementation: The eCollege SIS for API uses the elements of the `n` element to obtain all of the naming information about a `person`. As a consequence, this element—while optional in the IMS Enterprise specification—is Required for the eCollege API for SIS.
Data type: Elements
Multiplicity: Single, **eCollege Required**
Attributes: None
Elements: None
Example:

```
<n>
    <family>Frog</family>
    <given>Kermit</given>
    <partname partnametype=”Middlename”>The</partname>
</n>
```

3.3.3.2.1. <family>

Description: The family name of the `person`. Because the IMS Enterprise specification is culture neutral, this is not necessarily the last name.
eCollege Implementation: While optional in the IMS Enterprise specification, this element is Required for the eCollege API for SIS. The string length is also more restrictive than the IMS Enterprise Schema.
Data type: string 40
Multiplicity: Single, **eCollege Required**

Attributes: None
Elements: None

3.3.3.2.2. <given>

Description: The given name of the person. Because the IMS Enterprise specification is culture neutral, this is not necessarily the first name.
eCollege Implementation: While optional in the IMS Enterprise specification, this element is Required for the eCollege API for SIS. The string length is also more restrictive than the IMS Enterprise Schema.
Data type: string 40
Multiplicity: Single, **eCollege Required**
Attributes: None
Elements: None

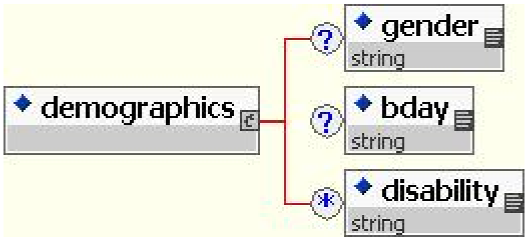
3.3.3.2.3. <partname>

Description: Component parts of the name, individually identified. This element allows greater flexibility in the organization of a name and sharing name information for a person.
eCollege Implementation: Currently only the partname identified as "Middlename" by its partnametype is consumed by the SIS for API. Other supplied partnames will be ignored.
Data type: string 256
Multiplicity: Multiple, Optional
Attributes:

- lang (optional) – string 128 – language used for the partname. Vocabulary is based on the ISO639 standard. eCollege only supports the English language, value "EN".
- partnametype (required) – string 64 – The type component of the name. Only "Middlename" is supported by the API for SIS.

Elements: None

3.3.4. <demographics>



Description: Container for the person's demographic information.
eCollege Implementation: No additional information.
Data type: Elements
Multiplicity: Single, Optional
Attributes: None
Elements:

- gender
- bday
- disability

3.3.4.1. <gender>

Description: Gender of the person.
eCollege Implementation: This relates to the extended user property Gender.

Data type: string 255
Multiplicity: Single, Optional
Attributes: None
Elements: None

3.3.4.2. <bday>

Description: Birth date of the person.
eCollege Implementation: This relates to the extended user property BirthDay.
Data type: string 255
Multiplicity: Single, Optional
Attributes: None
Elements: None

3.3.4.3. <disability>

Description: Disability information for the person.
eCollege Implementation: This relates to the extended user property Disability.
Data type: string 255
Multiplicity: Multiple (however, eCollege only uses the first instance), Optional
Attributes: None
Elements: None

3.3.5. <tel>

Description: Telephone number used to contact the person.
eCollege Implementation: eCollege typically has two phone numbers: a day-time and night-time number, although these are not implemented for every EP, and some EPs may have additional numbers, such as parent or emergency contact numbers. Because the `tel` element does not include any additional contextual information, the first number found will be assumed to be the day-time phone, and any additional numbers are ignored. If the number is not actually part of the user properties for the EP, the value will be ignored.
Data type: String 32
Multiplicity: Many, Optional
Attributes:

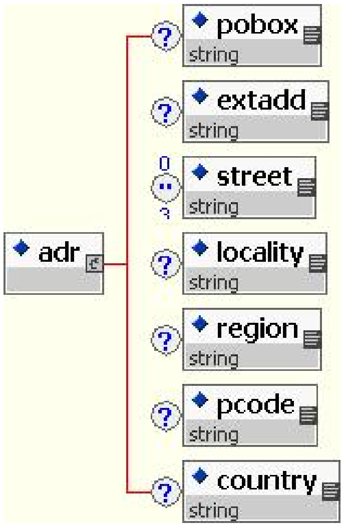
- `teltype` (enumeration) – indicates the type of phone number. May be sent as either the string or the numeric identifier. (1=Voice; 2=Fax; 3=Mobile; 4=Pager). Defaults to Voice. Note that eCollege only takes the first[t] <tel> with a teletype attribute of either “Voice” or “1”..

Elements: None

3.3.6. <email>

Description: Email address used to contact a person.
eCollege Implementation: No additional information.
Data type: String 256
Multiplicity: Single, **eCollege Required**
Attributes: None
Elements: None

3.3.7. <adr> Elements



Description: Container for all of the parts of the `person`'s shipping/delivery address. Normally there is only one address per `person`.
eCollege Implementation: No additional information.
Data type: Elements
Multiplicity: Single, Optional
Attributes: None
Elements:

- `pobox`
- `extadd`
- `street`
- `locality`
- `region`
- `pcode`
- `country`

3.3.7.1. <pobox>

Description: Post office box number address component.
eCollege Implementation: eCollege composites this value into the street address if the extended user property for the street address is enabled for the EP.
Data type: string 32
Multiplicity: Single, Optional
Attributes: None
Elements: None

3.3.7.2. <extadd>

Description: Extra, non-street address information, such as apartment or suite number.
eCollege Implementation: eCollege composites this value into the street address if the extended user property for the street address is enabled for the EP.
Data type: string 128
Multiplicity: Single, Optional
Attributes: None
Elements: None

3.3.7.3. <street>

Description: The actual street address.

eCollege Implementation: eCollege composites this value into the street address if the extended user property for the street address is enabled for the EP.
Data type: string 128
Multiplicity: Multiple (Maximum of two instances), Optional
Attributes: None
Elements: None

3.3.7.4. <locality>

Description: The locality component of the address, typically the City.
eCollege Implementation: eCollege uses this value if the extended user property for the city is enabled for the EP.
Data type: string 64
Multiplicity: Single, Optional
Attributes: None
Elements: None

3.3.7.5. <region>

Description: The region component of the address, typically the State or Province.
eCollege Implementation: eCollege uses this value if the extended user property for the state is enabled for the EP.
Data type: string 64
Multiplicity: Single, Optional
Attributes: None
Elements: None

3.3.7.6. <pcode>

Description: The postal code component of the address. Actual format may vary depending on country.
eCollege Implementation: eCollege uses this value if the extended user property for the postal code is enabled for the EP.
Data type: string 32
Multiplicity: Single, Optional
Attributes: None
Elements: None

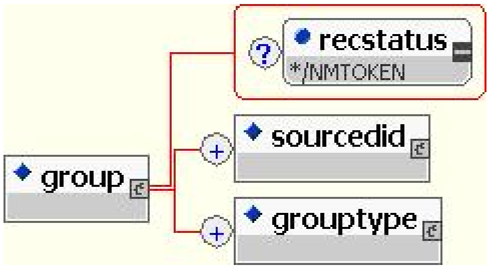
3.3.7.7. <country>

Description: The country component of the address. Format is based on ISO3166.
eCollege Implementation: eCollege uses this value if the extended user property for the country is enabled for the EP.
Data type: string 64
Multiplicity: Single, Optional
Attributes: None
Elements: None

3.3.8. Special note about <institutionrole> and <systemrole>

The `institutionrole` and `systemrole` elements were added to the IMS Enterprise v1.1 Schema to incorporate and standardize some very common extensions to the 1.01 Schema. The eCollege system allows for a more complex organization in which a user can have different roles in different parts of the organization. As a result, `institutionrole` and `systemrole` are meaningless concepts to the eCollege system without additional contextual information. For eCollege and the API for SIS, a `person`’s role within a given part of the organization is determined by the relationship set up between the `person` and one or more `groups` in the `membership` element.

3.4. <group> Elements



Description: The container for all of the information about a group and its relationship to other groups. A group can be a collection of individuals, a set of curriculum definitions, or any other collection of relevant objects. The group structure is a convenient abstract container for any collection of common objects.

eCollege Implementation: eCollege uses the `group` element to represent an area in which a user can be enrolled – typically this is a course, although less frequently it may be an Enrollable Node. (Nodes are a concept unique to eCollege, relating to hierarchical administrative structures. An eCollege Client Services Consultant can help determine when it is appropriate to enroll a user specifically in a node.)

The actual enrollment is described by the `membership` elements.

While the IMS Enterprise Schema considers the `group` element optional, in the context of a document for the eCollege API for SIS it is Required. Since the document describes enrollments (`membership`) of users (`person`) in courses (`group`), the data is meaningless without at least one instance of each.

Data type: Elements

Multiplicity: Many, **eCollege Required**

Attributes:

- `recstatus` (optional) – because courses are not created using the API for SIS, this attribute is ignored.

Elements:

- `sourcedid`
- `grouptype`

Example:

```
<group>
    <sourcedid>
        <source>Muppet University</source>
        <id>19791007</id>
    </sourcedid>
    <grouptype>
        <typevalue>Call Number</typevalue>
    </grouptype>
</group>
```

3.4.1. <sourcedid>

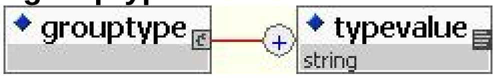
Description: The identifier of the `group` as defined by the source system. The `sourcedid` must uniquely identify the `group` with the document so that it can be used as a referential key between one or more `membership` elements and the `group`.

eCollege Implementation: The `sourcedid` for a course is expected to have the Course Call Number as the id and the EP's SIS system as the source. Since Course Call Number is a value generated by the EP and stored in the eCollege system at the time of course creation, that value is already mapped to a course in the eCollege system. If it is necessary to provide Enrollable Node information, the `sourcedid` is the eCollege system, and the source should be ECLG while the id is the Client Sort String of the enrollable node.

Data type: Elements

Multiplicity: Single, **eCollege Required**
Attributes: See Section 3.6 for details of this common data structure.
Elements: See Section 3.6 for details of this common data structure.

3.4.2. <grouptype> Elements



Description: Category information for the group.

eCollege Implementation: The `grouptype` element is required by the IMS Enterprise Schema and is used to determine which course or enrollable node the group relates to. The element `typevalue` indicates whether the group is a course or an enrollable node. If it is a course, `typevalue` must be *Call Number;* for an enrollable node, it must be *Enrollable Node.* Elements with other `typevalue` values will be discarded; however, including numerous extraneous groups can have a negative impact on processing time.

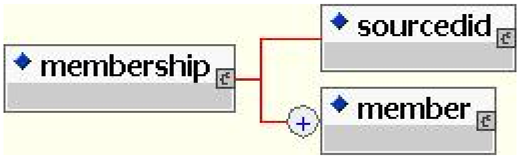
Data type: Elements
Multiplicity: Single, Required
Attributes: None.
Elements:

- `typevalue`

Example:

```
<grouptype>
    <typevalue>Call Number</typevalue>
</grouptype>
```

3.5. <membership> Elements



Description: The container for all of the information about the members (as defined in the `person` and/or `group` structures) of a particular Group. This structure is used to establish the membership relations between `Groups` and `Groups/Persons`.

eCollege Implementation: This is where an enrollment in a course or node (rarely a node) is established, by relating a `person` to a `group`. Because the API for SIS only handles enrollment information, an incoming document is meaningless without at least one `membership` element. The entities involved in the relationship are determined through the descendent `sourcedid` elements (see below for more detail).

Data type: Elements
Multiplicity: Many, **eCollege Required**
Attributes: None.
Elements:

- `sourcedid`
- `member`

Example: (Because it is important to see most of the membership element at once to understand the function of this key element, this example drills further into the child elements than others.)

```
<membership>
    <sourcedid>
        <source>Muppet University</source>
        <id>19791007</id>
    </sourcedid>
    <member>
        <sourcedid>
            <source>Muppet University</source>
            <id>KERM148</id>
        </sourcedid>
        <role>...</role>
    <member>
</membership>
```

Note that the membership/sourcedid exactly matches the sourcedid for the group example, while the membership/member/sourcedid exactly matches the sourcedid for the person example, enrolling Mr. Kermit The Frog in BUS 201 with the specified role.

3.5.1. <sourcedid>

Description: The identifier of the group as defined by the source system. The sourcedid must uniquely identify the group with the document so that it can be used as a referential key between one or more membership elements and the group. While the sourcedid must be unique within the context of all group elements, the same sourcedid may appear in multiple membership elements (although this would be unusual). The effect of having multiple membership elements with the same sourcedid is the same as including every child member element of each of the membership elements under a single membership element.

eCollege Implementation: No additional information.

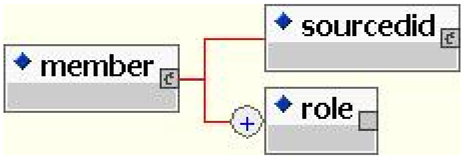
Data type: Elements

Multiplicity: Single, Required

Attributes: See Section 3.6 for details of this common data structure.

Elements: See Section 3.6 for details of this common data structure.

3.5.2. <member> Elements



Description: A member of the group defined by the sourcedid in the parent membership.

eCollege Implementation: No additional information.

Data type: Elements

Multiplicity: Many, Required

Attributes: None.

Elements:

- sourcedid
- role

Examples:

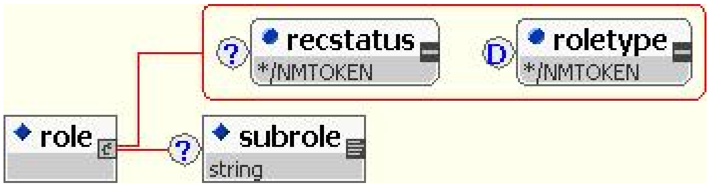
```
<member>
    <sourcedid>
        <source>Muppet University</source>
        <id>KERM148</id>
    </sourcedid>
    <role>...</role>
</member>
```

3.5.2.1. <sourcedid>

Description: The identifier of the person as defined by the source system. The sourcedid must uniquely identify the person within the document so that it can be used as a referential key between the member element and the person. While the sourcedid must be unique within the context of all person elements, the same sourcedid may appear in multiple membership/member elements (i.e., the person may be associated with more than one group in a single document).

eCollege Implementation: No additional information.
Data type: Elements
Multiplicity: Single, Required
Attributes: See Section 3.6 for details of this common data structure.
Elements: See Section 3.6 for details of this common data structure.

3.5.2.2. <role> Elements



Description: The role of the `member` in the `group`.
eCollege Implementation: While the IMS Enterprise specification allows a member to have multiple roles in a single group, the eCollege system does not support this. If more than one role is defined for a particular `membership/member` element, only one may have an active `status`.
Data type: Elements
Multiplicity: Many, Required
Attributes:

- `recstatus` (NMToken: 1=Add; 2=Update; 3=Delete) optional. This is ignored by eCollege since the required action is determined by comparing with any existing information. Delete is not supported by the eCollege API for SIS. Members must be given a Drop role instead.
- `roletype` (NMToken) – optional – the member's function with a Group. This value is too coarse for effective use within the eCollege system and is ignored.

Elements:

- `subrole`

Example:

```
<role>
     <subrole>2</subrole>
</role>
```

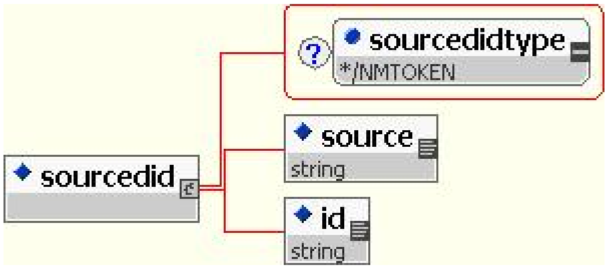
3.5.2.2.1. <subrole>

Description: Further qualifies the `member`'s `role` in the `group`.
eCollege Implementation: The eCollege role identifier for the person in the group. Because the eCollege system defines roles in a much more granular fashion, tailored to the needs of each EP, the `role/@roletype` attribute is too coarse to be effective in determining the role a person should be assigned within the eCollege system. A Client Services Consultant can provide specific information about the role identifiers for a specific EP.

Note: The IMS Specification defines `subrole` as a string; however, the eCollege system is looking for a numeric RoleID for this field. Refer to the *Use case and test data* sample in the Developer's Guide, under Sample Code for an example of an XML file.

Data type: string 32
Multiplicity: Single, Required
Attributes: None
Elements: None

3.6. Common <sourcedid> Elements



Description: The ID of the data object as defined by the source system. The `sourcedid` is key to relating data elements to one another in the `membership` data structure. The `sourcedid` must uniquely identify the `person` or `group` in the document.
eCollege Implementation: The `sourcedid` is key to relating elements of the XML document to one another; it is also the means used by the eCollege Sis API to identify entities that already exist in the eCollege system. For persons and groups that are courses, the source of the id should be the EP's Sis. For groups that are enrollable nodes, the source should be ECLG.
Data type: Elements
Multiplicity: Single, Required
Attributes:

- `sourcedidtype` (optional) – ignored by eCollege

Elements:

- `source`
- `id`

Example:

```
<sourcedid>
     <source>Muppet University</source>
     <id>19791007</id>
</sourcedid>
```

3.6.1. <source> Element

Description: The source system generating the identifier.
eCollege Implementation: No additional information.
Data type: string 32
Multiplicity: Single, Required
Attributes: None.
Elements: None.

3.6.2. <id> Element

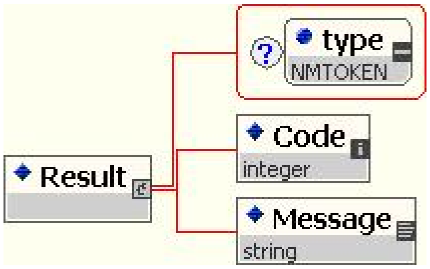
Description: The unique identifier of the data element (`person` or `group`) in the `source` system.
eCollege Implementation: eCollege does not retain this `id`.
Data type: string 256
Multiplicity: Single, Required
Attributes: None.
Elements: None.

4. eCollege Extensions

The eCollege API for SIS uses some custom extensions to the IMS Enterprise Schema 1.1 in log documents to provide feedback about processing of incoming documents (see `result` Elements). Consumption of these elements is entirely at the discretion of the EP's SIS system.

Extensions to the `person` element (see `personproperty` Elements) allow EP SIS systems to send additional information about a person in incoming documents. Outgoing documents provide EP SIS systems with additional information that may have been collected or updated through the eCollege system outside of the EP's SIS.

4.1. <result> Elements



Description: Generated in log files for incoming requests to indicate the result of processing the `person` or the `membership`. Results include errors, warnings, success and information messages.

The IMS schema accepts either the numeric or string value for type; however, due to how the eCollege system serializes the code, the result type will always return as a string value (Success, Warning, Error).

Document Type: log
Data type: Elements
Multiplicity: Many, Optional
IMS Parent: `properties, person, group, role`

Attributes:

- type (required) – NMToken
    - Success: Information about successful actions
    - Warning: Unusual, non-error condition
    - Error: General Error

Elements:

- `resultcode`
- `message`

Examples:

```
<person>
     <extension>
          <result type=”2”>
               <resultcode>0</resultcode>
               <message>User already exists, user not 
created</message>
          </result>
          <result type=”0”>
               <resultcode>0</resultcode>
               <message>User properties successfully 
updated.</message>
          </result>
     </extension>
</person>
```

4.1.1. <resultcode>

Description: A numeric code for the error encountered. If the result is successful, informational, or a warning, the value is 0. A list of known result codes where the source system may reasonably attempt to fix the problem is available from your Client Services Consultant.

The IMS schema accepts either the numeric or string value for type; however, due to how the eCollege system serializes the code, the result type will always return as a string value (Success, Warning, Error).

Datatype: integer
Multiplicity: Single, Required
Attributes: None
Elements: None

4.1.2. <message>

Description: A human-readable message describing the result.
Datatype: string 4096
Multiplicity: Single, Required
Attributes: None
Elements: None

4.2. <personproperty> Elements

Description: Additional information about a person which does not otherwise fit into the existing person elements. Specific `propertynames` are implemented on an EP by EP basis. A Client Services Consultant can provide the latest set of values, or help to establish new values appropriate for a specific EP.
Document Type: incoming, outgoing
Datatype: string 255
Multiplicity: Many, Optional
IMS Parent: `person`
Attributes:

- `propertyname` (required) – string 50

Elements: None

5. Schema Namespaces and URIs

5.1. IMS Enterprise v1.1

The IMS Global Learning Consortium has defined the IMS Enterprise v1.1 with a DTD only. There is an XSD for the IMS Enterprise v1.01, but because 1.1 documents do not comply with the 1.01 specification, and some specific additions, such as the password attribute on the <userid> element, are critical for the eCollege API for SIS, the v1.01 specification should *not* be referenced. eCollege provides an XSD of the v1.1 specification at:

http://schemas.ecollege.com/ims_epv1p1.xsd.

You can use XSD to define the default namespace for any document.

6. More about SourceIDs

This section address commonly asked questions about SourceIDs including what they are, how users are identified, and best practices in assigning IDs to users.

6.1. What is sourcedid used for within eCollege?

Within the eCollege system, the sourcedid is the mapping point between an entity in the eCollege system and your system. The person sourcedid maps to a single user. The course group sourcedid maps to a single course instance. eCollge uses sourcids as shared identifiers and mapping points. The node group sourcedid maps to a single node. eCollege may make other internal use of the id for the node (client sort string), but those do not effect the EP or the Sis Api, and we will always retain that value as an identifier for use by our EPs.

6.2. Is sourceid assigned for delimited files?

No. If you are using a delimited format, a sourcedid is not assigned. The login name in delimited files is the shared identifier for a student or faculty member.

6.3. Are there limitations with using both the batch and Web Services APIs concurrently?

Yes. First, there is the potential for race conditions. The single request web service is intended to provide immediate, synchronous access to the eCollege system. The batch options are asynchronous.

For example, if you submit a batch request to enroll Johnny in English 101 as a Waitlisted student, and the request is received at 10 AM, at 1 PM a synchronous web service request is made to update Johnny's enrollment in English 101 to a full Student. At this point, it is possible for that line in the batch request to not yet process. The web service will still enroll Johnny in English 101 as a Student and report success. However, the asynchronous process will later get to that line in the request and determine that Johnny should have a role of Waitlist instead of student, thus update the enrollment and report success. As a result, the end result is not correct.

The web service does not contain code to distinguish between a successful new enrollment and a successful updated enrollment. As such, it is best practice to either:

- Use a combination of asynchronous and synchronous requests where you have two different systems (that do not share users or courses) each work with one type of request
- Use asynchronous requests only during periods of very low activity, so that race conditions are unlikely, or synchronous requests on the EP side may reasonably be queued up until the asynchronous request is complete.

6.4. How is the sourceid mapped to a user in the eCollege system?

The eCollege uses the following logic to map a sourcedid to an existing/new user:

(1) Is the sourcedid already mapped to a user in the eCollege system?
  a. Yes – this must be the same user. Go to question 3
  b. No – go to question 2

(2) Does this login name exist in the eCollege system?
  a. Yes – this must be the same user. Map the sourcedid to this user. Done.
  b. No – this must be a new user. Go to question 5

(3) Is the login name the same as for the user mapped to the sorucedid?
  a. Yes – do nothing else. Done.
  b. No – we must need to update the user's login name. Go to question 4

(4) Can the login name be updated (configurable setting)
  a. Yes – go to question 5
  b. No – do nothing else. Done.

(5) Is this login name already in use in the eCollege system?
  a. Yes – cannot create/update the user. Log error. Done.
  b. No – create a new user with this sourcedid and login/update the user identified by this sourcedid to have a new login. Done.

7. Quick Reference

For your convenience, we added this quick reference section for you to use to easily locate user properties and required elements.

User Properties				
Section	**Property Description**	**eCollege Property Name**	**XPath to Property in IMS Enterprise Schema**	**Example of property element**
3.3.2	Login Name	cn	enterprise/person/userid	<userid>*IKant*</userid>
3.3.2	Password	userPassword	enterprise/person/userid/@password	<userid password=”*Think!*”>
3.3.3.2.2	First Name	FirstName	enterprise/person/name/n/given	<given>*Immanuel*</given>
3.3.3.2.1	Last Name	LastName	enterprise/person/name/n/family	<family>*Kant*</family>
3.3.3.2.3	Middle Name	Middlename	enterprise/person/name/n/partname[@partnametype=’Middlename’]	<partname **partnametype=”Middlename”>**_Alexander_</partname>
3..3.4.1	Gender	Gender	enterprise/person/demographics/gender	<gender>*M*</gender>
3.3.4.2	Birth Date	BirthDay	enterprise/person/demographics/bday	<bday>*April 22, 1724*</bday>
3.3.4.3	Disability	Disability	enterprise/person/demographics/disability	<disability>*hearing impaired*</disability>
3.3.5	Day-time phone number	TelephoneNumber	enterprise/person/tel*	<tel>*720.555.1212*</tel>
3.3.6	Email Address	Mail	enterprise/person/email	<email>*Imm.Kant@philosophers.net*</email>
3.3.7.3	Street Address	Street	enterprise/person/adr/street[position()=’1’]	<street>*123 Main Str*.</street>
3.3.7.3	Street Address 2nd line	Street2	enterprise/person/adr/street[position()=’2’]	<adr> <street>above the bakery</street> <street>*123 Maint St.*</street> </adr>
3.3.7.4	City/Town	City	enterprise/person/adr/locality	<locality>*Königsberg*</locality>
3.3.7.5	State/Province	State	enterprise/person/adr/region	<region>*Kalinigrad*</region>
3.3.7.6	Postal/Zip Code	PostalCode	enterprise/person/adr/pcode	<pcode>*99299*</pcode>
3.3.7.7	Country	Country	enterprise/person/adr/country	<country>*Russia*</country>
4.2	Any other user property	Exact name, including capitalization	enterprise/person/extension/personproperty	<personproperty **propertyname=”SomePropertyName”** >*Some Value*</personproperty>

User Properties

Section	Property Description	eCollege Property Name	XPath to Property in IMS Enterprise Schema	Example of property element
		provided by Client Services		

Legend

- *Italics - Example values are in italics*
- **Bold - Bold text indicates information that must be provided in addition to the value and the element name**
- Sans Serif - Additional elements to provide context are in a Sans Serif typeface

*See section 3.3.5 of the annotated guide for additional important information

Required Elements

3. MS elements consumed by the eCollege API for SIS

3.1 <enterprise> Elements

Description:	The enterprise element is the root of the document and the container for all the data objects of the IMS Enterprise Schema
eCollege Implementation:	The person, group, and membership elements are each optional under the IMS specification, but logically at least one of them must be present. For the purposes of eCollege, there must be at least one instance of each of the three elements. Because a person cannot exist within the eCollege system without being in a defined context beyond just the EP, those context(s) are defined in the group elements, and the relation of the person to the group is defined in a membership element. As a result, all three must be present in every IMS Enterprise document sent to eCollege, and are present in every document created by eCollege.
Data Type:	
Multiplicity:	Single, Required
Attributes:	xmlns (optional) – the enterprise element should reference the namespace for the document: http://schemas.ecollege.com/ims_epv1p1.xsd
Elements:	• properties • person • group • membership

Example:	`<enterprise xmlns=”http://schemas.ecollege.com/Common/2006/01/ims_epv1p1.xsd”>`
	`<properties>...</properties>`
	`<person>...</person>`
	`<group>...</group>`
	`<membership>...</membership>`
	`</enterprise>`

3.2 <properties> Elements

Description:	Information about the entire document and the data being exchanged between the SIS and the eCollege API
eCollege Implementation:	eCollege ignores all elements within properties.
Data Type:	Elements
Multiplicity:	Single, Required
Attributes:	xmlns (optional) – the enterprise element should reference the namespace for the document: http://schemas.ecollege.com/ims_epv1p1.xsd

3.3 <person> Elements

Description:	The container for information about a particular person relevant to the eCollege LMS.

Required Elements	
eCollege Implmentation:	The majority of the information provided in the IMS specification is ignored by eCollege (see the child elements list). Extensions to the person element are supported to allow the EP and eCollege to transfer specific information necessary for the EP and information agreed upon between the two organizations.This element is required in every document sent to the eCollege API for SIS so that eCollege can identify and/or create the user.
Data type:	Elements
Multiplicity:	Single, **eCollege Required**
Attributes:	`recstatus` (optional and ignored by eCollege)
Elements:	• `sourcedid` • `name` • `email` • `tel` • `adr`

Example:	`<person>`
	`    <sourcedid>`
	`        <source>Muppet University</source>`
	`        <id>KERM148</id>`
	`    </sourcedid>`
	`    <userid password=”BeingGreen”>Kfrog1</userid>`
	`    <name>...</name>`
	`    <email>frogk@Disney.com</email>`
	`    <tel teltype=”Voice”>303-555-1212</tel>`
	`    <adr>...</adr>`
	`    <extension>…</extension>`
	`</person>`

3.3.1 <sourceid>

Description:	The identifier of the person as defined by the source system. The sourcedid must uniquely identify the person within the document so that it can be used as a referential key between one or more member elements and the person.
eCollege Implementation:	The eCollege SIS API will permanently associate the provided sourcedid with the user in our datastore.
Data type:	Elements
Multiplicity:	Single, Required
Attributes:	See Section 3.6 of the *Annotated Guide to the IMS Spec* for details of this common data structure.
Elements:	See Section 3.6 of the *Annotated Guide to the IMS Spec* for details of this common data structure.

3.3.2 <userid>

Required Elements

Description:	The person's ID to access the eCollege LMS.
eCollege Implementation:	Also often called the "Login ID", the userid is the unique value that eCollege uses to identify a person within the context of an EP in the eCollege LMS. Because the IMS Enterprise Schema specifies this element as optional, the eCollege SIS for API will attempt to identify a person by the sourcedid/id value if, and only if, the userid is absent. *Also note that the string lengths for the userid and password are more restrictive that the IMS Enterprise Schema.* Because the userid is the account a person uses to access the eCollege LMS, and that account is what is associated with enrollments (through the membership structure), while it is possible for a physical person to have more than one login that they might use, it is **not** possible for a person element to have more than one userid. As a consequence, while the IMS specification for multiplicity is Many, Optional, the eCollege implementation is Single, Optional. Because we cannot guarantee that the first userid element will always be used, no more than one userid element should be included when sending data to eCollege.
Data type:	string 255 – invalid characters are: **%][+<>";'=:/|** and any white space.
Multiplicity:	Single, **eCollege Required**
Attributes:	eCollege currently supports only the password attribute. • password (optional) – the password used to validate the person when logging in to the eCollege LMS. The password may be left blank if the person does not need to be created in the eCollege LMS, or the EP has opted to have random passwords generated. The length can be up to 50 characters. Invalid characters are: **%][+<>";'=:/|_** and any white space.
Elements:	None

3.3.3 <name> Elements

Description:	Data element for the name of the person.
eCollege Implementation:	No additional information.
Data type:	Elements
Multiplicity:	Single, Required
Attributes:	None
Elements:	None

Example:	<name>
	    <fn>Mr. Kermit The Frog</fn>
	    <n>…</n>
	</name>

3.3.3.1 <fn>

Required Elements	
Description:	The complete formatted name of the person.
eCollege Implementation:	eCollege SIS for API does not make use of this value.
Data type:	string 256
Multiplicity:	Single, Required
Attributes:	None
Elements:	None
3.3.3.2 <n> Elements	
Description:	The name of the person broken into all of its distinct components.
eCollege Implementation:	The eCollege SIS for API uses the elements of the n element to obtain all of the naming information about a person. As a consequence, this element—while optional in the IMS Enterprise specification—is Required for the eCollege API for SIS.
Data type:	Elements
Multiplicity:	Single, **eCollege Required**
Attributes:	None
Elements:	None

Example:	`<n>`
	`    <family>Frog</family>`
	`    <given>Kermit</given>`
	`    <partname partnametype="Middlename">The</partname>`
	`</n>`

3.3.3.2.1 <family>	
Description:	The family name of the person. Because the IMS Enterprise specification is culture neutral, this is not necessarily the last name.
eCollege Implementation:	While optional in the IMS Enterprise specification, this element is Required for the eCollege API for SIS. The string length is also more restrictive than the IMS Enterprise Schema.
Data type:	string 40
Multiplicity:	Single, **eCollege Required**
Attributes:	None
Elements:	None
3.3.3.2.2 <given>	

Required Elements

Description:	The given name of the person. Because the IMS Enterprise specification is culture neutral, this is not necessarily the first name.
eCollege Implementation:	While optional in the IMS Enterprise specification, this element is Required for the eCollege API for SIS. The string length is also more restrictive than the IMS Enterprise Schema.
Data type:	string 40
Multiplicity:	Single, **eCollege Required**
Attributes:	None
Elements:	None
3.3.4 <email>	
Description:	Email address used to contact a person.
eCollege Implementation:	No additional information.
Data type:	String 256
Multiplicity:	Single, **eCollege Required**
Attributes:	None
Elements:	None
3.4 <group> Elements	
Description:	The container for all of the information about a group and its relationship to other groups. A group can be a collection of individuals, a set of curriculum definitions, or any other collection of relevant objects. The group structure is a convenient abstract container for any collection of common objects.
eCollege Implementation:	eCollege uses the `group` element to represent an area in which a user can be enrolled – typically this is a course, although less frequently it may be an Enrollable Node. (Nodes are a concept unique to eCollege, relating to hierarchical administrative structures. An eCollege Client Services Consultant can help determine when it is appropriate to enroll a user specifically in a node.) The actual enrollment is described by the `membership` elements. While the IMS Enterprise Schema considers the `group` element optional, in the context of a document for the eCollege API for SIS it is Required. Since the document describes enrollments (`membership`) of users (`person`) in courses (`group`), the data is meaningless without at least one instance of each.
Data type:	Elements
Multiplicity:	Many, **eCollege Required**
Attributes:	`recstatus` (optional) – because courses are not created using the API for SIS, this attribute is ignored.

Required Elements

Elements:	- `sourcedid` - `grouptype`

Example:	`<group>`
	`<sourcedid>`
	`<source>Muppet University</source>`
	`<id>19791007</id>`
	`</sourcedid>`
	`<grouptype>`
	`<typevalue>Call Number</typevalue>`
	`</grouptype>`
	</group>

3.4.1 <sourceid>

Description:	The identifier of the group as defined by the source system. The sourcedid must uniquely identify the group with the document so that it can be used as a referential key between one or more membership elements and the group
eCollege Implementation:	The sourcedid for a course is expected to have the Course Call Number as the id and the EP's SIS system as the source. Since Course Call Number is a value generated by the EP and stored in the eCollege system at the time of course creation, that value is already mapped to a course in the eCollege system. If it is necessary to provide Enrollable Node information, the sourcedid is the eCollege system, and the source should be ECLG while the id is the Client Sort String of the enrollable node.
Data type:	Elements
Multiplicity:	Single, **eCollege Required**
Attributes:	See Section 3.6 of the *Annotated Guide to the IMS Spec* for details of this common data structure.
Elements:	See Section 3.6 of the *Annotated Guide to the IMS Spec* for details of this common data structure.

3.4.2 <grouptype> Elements

Description:	Category information for the group.
eCollege Implementation:	The grouptype element is required by the IMS Enterprise Schema and is used to determine which course or enrollable node the group relates to. The element typevalue indicates whether the group is a course or an enrollable node. If it is a course, typevalue must be *Call Number;* for an enrollable node, it must be *Enrollable Node.* Elements with other typevalue values will be discarded; however, including numerous extraneous groups can have a negative impact on processing time.
Data type:	Elements
Multiplicity:	Single, Required
Attributes:	None
Elements:	`typevalue`

Required Elements

Example:	`<grouptype>`
	`    <typevalue>Call Number</typevalue>`
	`</grouptype>`

3.5 <membership> Elements	
Description:	The container for all of the information about the members (as defined in the `person` and/or `group` structures) of a particular Group. This structure is used to establish the membership relations between `Groups` and `Groups/Persons`.
eCollege Implementation:	This is where an enrollment in a course or node (rarely a node) is established, by relating a person to a group. Because the API for SIS only handles enrollment information, an incoming document is meaningless without at least one membership element. The entities involved in the relationship are determined through the descendent sourcedid elements (see below for more detail).
Data type:	Elements
Multiplicity:	Many, **eCollege Required**
Attributes:	None
Elements:	• `sourcedid` • `member`

Example:	(Because it is important to see most of the membership element at once to understand the function of this key element, this example drills further into the child elements than others.)
	
	`<membership>`
	`    <sourcedid>`
	`        <source>Muppet University</source>`
	`        <id>19791007</id>`
	`    </sourcedid>`
	`    <member>`
	`        <sourcedid>`
	`            <source>Muppet University</source>`
	`            <id>KERM148</id>`
	`        </sourcedid>`
	`        <role>...</role>`
	`    <member>`
	`</membership>`
	
	Note that the `membership/sourcedid` exactly matches the `sourcedid` for the `group` example, while the `membership/member/sourcedid` exactly matches the `sourcedid` for the `person` example, enrolling Mr. Kermit The Frog in BUS 201 with the specified `role`.

Required Elements

3.5.1 <sourceid>

Description:	The identifier of the group as defined by the source system. The sourcedid must uniquely identify the group with the document so that it can be used as a referential key between one or more membership elements and the group. While the sourcedid must be unique within the context of all group elements, the same sourcedid may appear in multiple membership elements (although this would be unusual). The effect of having multiple membership elements with the same sourcedid is the same as including every child member element of each of the membership elements under a single membership element.
eCollege Implementation:	No additional information.
Data type:	Elements
Multiplicity:	Single, Required
Attributes:	See Section 3.6 of the *Annotated Guide to the IMS Spec* for details of this common data structure.
Elements:	See Section 3.6 of the *Annotated Guide to the IMS Spec* for details of this common data structure.

3.5.2 <member> Elements

Description:	A member of the group defined by the sourcedid in the parent membership
eCollege Implementation:	No additional information
Data type:	Elements
Multiplicity:	Many, Required
Attributes:	None
Elements:	• sourcedid • role

Examples:

```
<member>
    <sourcedid>
        <source>Muppet University</source>
        <id>KERM148</id>
    </sourcedid>
    <role>...</role>
</member>
```

3.5.2.1 <sourcedid>

Description:	The identifier of the person as defined by the source system. The sourcedid must uniquely identify the person within the document so that it can be used as a referential key between the member element and the person. While the sourcedid must be unique within the context of all person elements, the same sourcedid may appear in multiple membership/member elements (i.e., the person may be associated with more than one group in a single document).

Required Elements

eCollege Implementation:	No additional information.
Data type:	Elements
Multiplicity:	Single, Required
Attributes:	See Section 3.6 of the *Annotated Guide to the IMS Spec* for details of this common data structure.
Elements:	See Section 3.6 of the *Annotated Guide to the IMS Spec* for details of this common data structure.

3.5.2.2 <role> Elements

Description:	The role of the member in the group.
eCollege Implementation:	While the IMS Enterprise specification allows a member to have multiple roles in a single group, the eCollege system does not support this. If more than one role is defined for a particular membership/member element, only one may have an active status.
Data type:	Elements.
Multiplicity:	Many, Required
Attributes:	• `recstatus` (NMToken: 1=Add; 2=Update; 3=Delete) optional. This is ignored by eCollege since the required action is determined by comparing with any existing information. Delete is not supported by the eCollege API for SIS. Members must be given a Drop role instead. • `roletype` (NMToken) – optional – the member's function with a Group. This value is too coarse for effective use within the eCollege system and is ignored.
Elements:	Subrole

Example:	`<role>`
	`<subrole>2</subrole>`
	`</role`

3.5.2.2.1 <subrole>

Description:	Further qualifies the member's role in the group.
eCollege Implementation:	The eCollege role identifier for the person in the group. Because the eCollege system defines roles in a much more granular fashion, tailored to the needs of each EP, the `role/@roletype` attribute is too coarse to be effective in determining the role a person should be assigned within the eCollege system. A Client Services Consultant can provide specific information about the role identifiers for a specific EP. **Note:** The IMS Specification defines `subrole` as a string; however, the eCollege system is looking for a numeric RoleID for this field. Refer to the *Use case and test data* sample in the Developer's Guide, under Sample Code for an example of an XML file.
Data type:	string 32
Multiplicity:	Single, Required
Attributes:	None

Required Elements	
Elements:	None
3.6 Common <sourcedid> Elements	
Description:	The ID of the data object as defined by the source system. The sourcedid is key to relating data elements to one another in the membership data structure. The sourcedid must uniquely identify the person or group in the document.
eCollege Implementation:	The sourcedid is key to relating elements of the XML document to one another; it is also the means used by the eCollege Sis API to identify entities that already exist in the eCollege system. For persons and groups that are courses, the source of the id should be the EP's Sis. For groups that are enrollable nodes, the source should be ECLG.
Data type:	`Elements`
Multiplicity:	Single, Required
Attributes:	`sourcedidtype` (optional) – ignored by eCollege
Elements:	• `source` • `id`

Example:	`<sourcedid>`
	`<source>Muppet University</source>`
	`<id>19791007</id>`
	`</sourcedid>`

3.6.1 <source> Element	
Description:	The source system generating the identifier
eCollege Implementation:	The source system generating the identifier
Data type:	`string 32`
Multiplicity:	Single, Required
Attributes:	None.
Elements:	None.
3.6.2 <id> Element	
Description:	The unique identifier of the data element (`person` or `group`) in the `source` system.
eCollege Implementation:	eCollege does not retain this id.
Data type:	`string 256`

Required Elements	
Multiplicity:	Single, Required
Attributes:	None
Elements:	None
4. eCollege Extensions	
4.1.1 <resultcode>	
Description:	A numeric code for the error encountered. If the result is successful, informational, or a warning, the value is 0. A list of known result codes where the source system may reasonably attempt to fix the problem is available from your Client Services Consultant. The IMS schema accepts either the numeric or string value for type; however, due to how the eCollege system serializes the code, the result type will always return as a string value (Success, Warning, Error).
Data type:	integer
Multiplicity:	Single, Required
Attributes:	None
Elements:	None
4.1.2 <message>	
Description:	A human-readable message describing the result.
Data type:	string 4096
Multiplicity:	Single, Required
Attributes:	None
Elements:	None