XCEDE 2.0 - A Manual

XCEDE 2.0 - A Manual

Table of Contents

List of Figures

List of Tables

Introduction

This is a manual for version 2.0 of XCEDE (XML-based Clinical and Experimental Data Exchange). The target audience for this manual is anyone who is interested in using or learning more about XCEDE. This manual will serve as both a tutorial and as a reference.

XCEDE is an extensible schema designed to store scientific data and metadata. XCEDE has its origins in various XML schemas developed for collaborative neuroinformatics projects, and was developed to enable the transfer and storage of several types of data including (but not limited to) clinical, demographic, behavioral, physiological and image data.

Chapter 1. The XCEDE Experiment Hierarchy

This is where the hierarchy text goes.

Overview

Examples

Reference

Chapter 2. Data Resources

This is where the data resource text goes.

Overview

Examples

Reference

Chapter 3. Catalogs

This is where the catalog text goes.

Overview

Examples

Reference

Chapter 4. Provenance

This is where the provenance text goes.

Overview

Examples

Reference

Chapter 5. Events

Overview

Events in XCEDE are merely time intervals annotated with arbitrary metadata. This component can be used to represent several types of behavioral data, statistics calculated on time series data, or any other metadata whose proper interpretation requires that it be associated with a particular time interval.

An XCEDE event consists of the following:

onset	The onset (in seconds) of the time interval.
duration	The duration (in seconds) of the time interval.
type	Usage of this field is user-specified
name	Usage of this field is user-specified
units	The units of the onset and duration fields. This field is optional, and it is recommended that users of the schema prescribe an implicit unit of measurement and use it consistently. In that case, this field may be considered informational only.
values	A *value* adds named metadata to this event.

The following instance shows how each of these fields may be populated.

```
<event type="visual" name="event#1" units="secs">
  <onset>0</onset>
  <duration>2</duration>
  <value name="shape">square</value>
  <value name="shapecolor">red</value>
</event>
```

Event elements are stored within the `<data>` element of an `<acquisition>`. The `<data>` element should be of type `events_t` (using `xsi:type` — see examples below).

All onsets are relative to an arbitrary time reference. Typically, time 0 (zero) could mean the start of data acquisition. An event list may be interpreted as concurrent with data in other `<acquisition>` elements (which could be other event lists). If so, the same time reference should be used in all concurrent acquisition data.

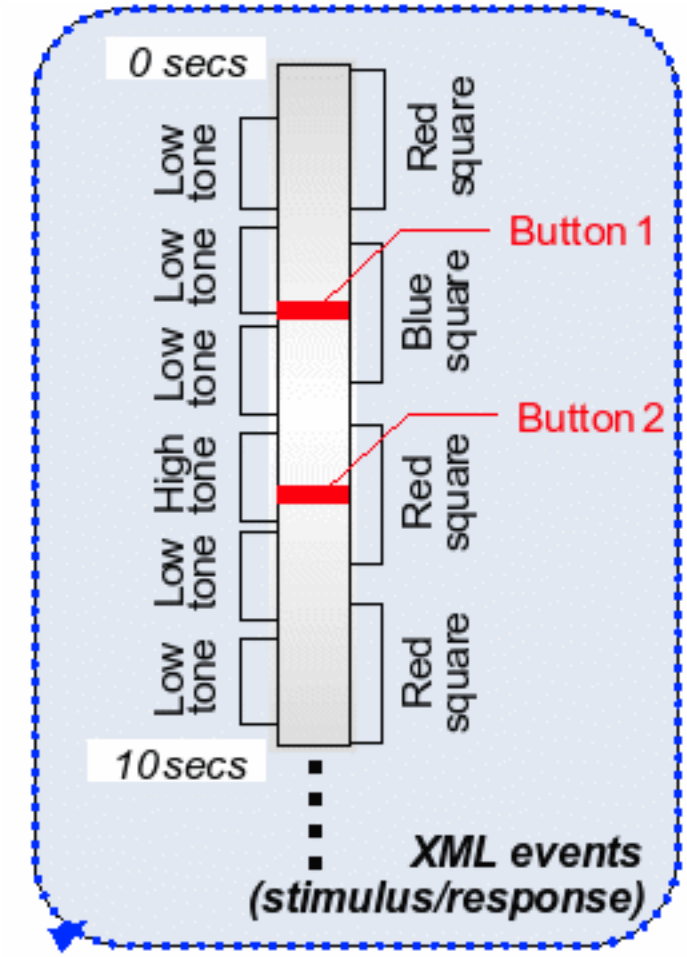
There is no ordering constraint on events in a list. Applications should depend on using the `<onset>` elements to order the events chronologically if they so desire.

An optional `<params>` element may precede the first event in a list, and this element stores arbitrary metadata (using the same `<value>` element used above) that apply to all events in the list.

Examples

> This example represents stimuli and responses in a neuroimaging study as XCEDE events. However, this is not the only type of data that can be represented using XCEDE events. A different example will be shown later in the chapter.

Figure 5.1. An event timeline



Consider the timeline shown in Figure 5.1. We show in Figure 5.2 how the first 5 seconds worth of the events might be represented in XCEDE.

Figure 5.2. XCEDE Events example

```
<XCEDE xmlns="http://www.xcede.org/xcede-2"
       xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'>
  <acquisition ID="my_events">
    <data xsi:type="events_t">
      <event type="visual">
        <onset>0</onset>
        <duration>2</duration>
        <value name="shape">square</value>
        <value name="shapecolor">red</value>
      </event>
      <event type="visual">
        <onset>2.5</onset>
        <duration>2</duration>
        <value name="shape">square</value>
        <value name="shapecolor">blue</value>
      </event>
      <event type="audio">
        <onset>0.3</onset>
        <duration>1.4</onset>
        <value name="frequency">low</value>
      </event>
      <event type="audio">
        <onset>2.0</onset>
        <duration>1.4</onset>
        <value name="frequency">low</value>
      </event>
      <event type="audio">
        <onset>3.5</onset>
        <duration>1.4</onset>
        <value name="frequency">low</value>
      </event>
      <event type="response">
        <onset>3.4</onset>
        <value name="button">1</value>
      </event>
    </data>
  </acquisition>
</XCEDE>
```

Each stimulus and each response are stored as separate event elements. Note that all the visual events appear first in the XCEDE file, then the audio events, and then the response event. This ordering is arbitrary, and the events could easily have been presented in chronological (or random!) order. The semantic interpretation of the events within an event list must not depend on their document order.

Reference

Table 5.1. Type: `event_t` (derives from `abstract_data_t`)

Table 5.2.

Attribute	Type
None	

Table 5.3.

Children	Type
params	eventParams_t
event*	event_t
description	xsd:string
annotation	textAnnotation_t

Chapter 6. Assessments

This is where the assessment text goes.

Overview

Examples

Reference

Chapter 7. Protocols

This is where the protocol text goes.

Overview

Examples

Reference

Appendix A. Schema

```
    <?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v2004 rel. 3 U (http://www.xmlspy.com) by dbk (UNIV CA IRVINE)
 -->
<xs:schema xmlns="http://www.xcede.org/xcede-2"
 xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xlink="http://www.w3.org/1999/xlink"
 targetNamespace="http://www.xcede.org/xcede-2" elementFormDefault="qualified"
 attributeFormDefault="unqualified">
 <xs:element name="XCEDE">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="annotationList" minOccurs="0">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="annotation" type="textAnnotation_t" minOccurs="0"
 maxOccurs="unbounded"/>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="revisionList" minOccurs="0">
     <xs:annotation>
      <xs:documentation>container for document revision history</xs:documentation>
     </xs:annotation>
     <xs:complexType>
      <xs:sequence>
       <xs:element name="revision" type="revision_t" minOccurs="0"
 maxOccurs="unbounded"/>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="project" type="project_t" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="subject" type="subject_t" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="visit" type="visit_t" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="study" type="study_t" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="episode" type="episode_t" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="acquisition" type="acquisition_t" minOccurs="0"
 maxOccurs="unbounded"/>
    <xs:element name="catalog" type="catalog_t" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="analysis" type="analysis_t" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="resource" type="abstract_resource_t" minOccurs="0"
 maxOccurs="unbounded"/>
    <xs:element name="protocol" type="protocol_t" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="version" type="xs:string"/>
  </xs:complexType>
 </xs:element>
 <!--*********            Top-level containers            *********-->
 <xs:complexType name="project_t">
  <xs:complexContent>
   <xs:extension base="abstract_container_t">
    <xs:sequence>
     <xs:element name="projectInfo" type="projectInfo_t" minOccurs="0"/>
     <xs:element name="contributorList" minOccurs="0">
      <xs:complexType>
       <xs:sequence>
        <xs:element name="contributor" type="person_t" minOccurs="0"
 maxOccurs="unbounded"/>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
     <xs:element name="subjectList" minOccurs="0">
      <xs:complexType>
       <xs:choice minOccurs="0" maxOccurs="unbounded">
```

```
       <xs:element name="subject" type="subject_t"/>
       <xs:element name="subjectRef" type="ref_t">
        <xs:annotation>
         <xs:documentation>This should be an xlink reference</xs:documentation>
        </xs:annotation>
       </xs:element>
      </xs:choice>
     </xs:complexType>
    </xs:element>
    <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="projectInfo_t">
 <xs:complexContent>
  <xs:extension base="abstract_info_t">
   <xs:sequence>
    <xs:element name="exptDesignList" minOccurs="0">
     <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
       <xs:element name="exptDesign"/>
       <xs:element name="exptDesignRef" type="ref_t"/>
      </xs:choice>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="subject_t">
 <xs:complexContent>
  <xs:extension base="abstract_container_t">
   <xs:sequence>
    <xs:element name="projectList" minOccurs="0">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="projectRef" type="ref_t" minOccurs="0" maxOccurs="unbounded">
        <xs:annotation>
         <xs:documentation>This should include participation information, including
per project ID and group.</xs:documentation>
        </xs:annotation>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="subjectInfo" type="subjectInfo_t" minOccurs="0"/>
    <xs:element name="visitList" minOccurs="0">
     <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
       <xs:element name="visit" type="visit_t"/>
       <xs:element name="visitRef" type="ref_t"/>
      </xs:choice>
     </xs:complexType>
    </xs:element>
    <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="assessment" type="assessment_t" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="subjectInfo_t">
 <xs:complexContent>
  <xs:extension base="abstract_info_t">
   <xs:sequence>
    <xs:element name="sex" type="terminologyString_t" minOccurs="0"/>
```

```
    <xs:element name="species" type="terminologyString_t" minOccurs="0"/>
    <xs:element name="birthdate" type="terminologyString_t" minOccurs="0"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="visit_t">
 <xs:complexContent>
  <xs:extension base="abstract_container_t">
   <xs:sequence>
    <xs:element name="projectRef" type="ref_t" minOccurs="0"/>
    <xs:element name="subjectRef" type="ref_t" minOccurs="0"/>
    <xs:element name="visitInfo" type="visitInfo_t" minOccurs="0"/>
    <xs:element name="studyList" minOccurs="0">
     <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
       <xs:element name="study" type="study_t"/>
       <xs:element name="studyRef" type="ref_t"/>
      </xs:choice>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="visitInfo_t">
 <xs:complexContent>
  <xs:extension base="abstract_info_t"/>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="study_t">
 <xs:complexContent>
  <xs:extension base="abstract_container_t">
   <xs:sequence>
    <xs:element name="projectRef" type="ref_t" minOccurs="0"/>
    <xs:element name="subjectRef" type="ref_t" minOccurs="0"/>
    <xs:element name="visitRef" type="ref_t" minOccurs="0"/>
    <xs:element name="studyInfo" type="studyInfo_t" minOccurs="0"/>
    <xs:element name="episodeList" minOccurs="0">
     <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
       <xs:element name="episode" type="episode_t"/>
       <xs:element name="episodeRef" type="ref_t"/>
      </xs:choice>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="studyInfo_t">
 <xs:complexContent>
  <xs:extension base="abstract_info_t"/>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="episode_t">
 <xs:complexContent>
  <xs:extension base="abstract_container_t">
   <xs:sequence>
    <xs:element name="projectRef" type="ref_t" minOccurs="0"/>
    <xs:element name="subjectRef" type="ref_t" minOccurs="0"/>
    <xs:element name="visitRef" type="ref_t" minOccurs="0"/>
    <xs:element name="studyRef" type="ref_t" minOccurs="0"/>
    <xs:element name="episodeInfo" type="episodeInfo_t" minOccurs="0"/>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
     <xs:element name="acquisition" type="acquisition_t">
      <xs:annotation>
```

```
       <xs:documentation>These represent the actual protocols and data obtained during
an episode.  Multiple acquistions can occur simulataneously during an episode.  For
example, MR, heart rate, and button presses.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="acquisitionRef" type="ref_t"/>
    </xs:choice>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="episodeInfo_t">
 <xs:complexContent>
  <xs:extension base="abstract_info_t"/>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="acquisition_t">
 <xs:complexContent>
  <xs:extension base="abstract_container_t">
   <xs:sequence>
    <xs:element name="acquisitionInfo" type="acquisitionInfo_t" minOccurs="0"/>
    <xs:choice minOccurs="0">
     <xs:element name="dataResource" type="abstract_resource_t">
      <xs:annotation>
       <xs:documentation>A resource that contains acquistion data (i.e. a URL to a
document that contains the data) . The resource could be any format: text file, a
binary file, or an XCEDE xml document, etc</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="dataResourceRef" type="ref_t">
      <xs:annotation>
       <xs:documentation>A reference to a resource as described above.  The resource
could be part of a catalog, a root level resource, etc.</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="data" type="abstract_data_t">
      <xs:annotation>
       <xs:documentation>A container that the actual acquisition data can go into (as
opposed to being in an external reousrce)</xs:documentation>
      </xs:annotation>
     </xs:element>
    </xs:choice>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="acquisitionInfo_t">
 <xs:complexContent>
  <xs:extension base="abstract_info_t"/>
 </xs:complexContent>
</xs:complexType>
<!--*********            Abstract types            *********-->
<xs:complexType name="abstract_container_t" abstract="true">
 <xs:sequence>
  <xs:element name="commentList" minOccurs="0">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="comment" type="authoredText_t" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="annotationList" minOccurs="0">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="annotation" type="abstract_annotation_t" minOccurs="0"
maxOccurs="unbounded"/>
```

```
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="resourceList" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Informational resources related to the
container</xs:documentation>
   </xs:annotation>
   <xs:complexType>
    <xs:sequence>
     <xs:element name="resource" type="abstract_resource_t" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="provenance" type="provenance_t" minOccurs="0"/>
  <xs:element name="analysisList" minOccurs="0">
   <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
     <xs:element name="analysis" type="analysis_t">
      <xs:annotation>
       <xs:documentation>This should be an abstract analysis_t that is extended to
capture derived data</xs:documentation>
      </xs:annotation>
     </xs:element>
     <xs:element name="analysisRef" type="ref_t"/>
    </xs:choice>
   </xs:complexType>
  </xs:element>
 </xs:sequence>
 <xs:attribute name="ID" type="xs:string"/>
 <xs:attribute name="rev" type="xs:string">
  <xs:annotation>
   <xs:documentation>Revision number, should correspond with an appropriate revision
ID in the XCEDE/history element</xs:documentation>
  </xs:annotation>
 </xs:attribute>
 <xs:attribute name="type" type="xs:string">
  <xs:annotation>
   <xs:documentation>Attribute for creating categories within a container set. For
example, within study, types might include 'PET' or 'MR'.  One could also create
sub-classes using colon notation: "MR:STRUCT"</xs:documentation>
  </xs:annotation>
 </xs:attribute>
 <xs:attributeGroup ref="terminology_ag"/>
</xs:complexType>
<xs:complexType name="abstract_info_t">
 <xs:annotation>
  <xs:documentation>info elements are present in each of the hierarchy levels.
these can be extended to capture instance specific content (following recommendation
5)</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="description" type="xs:string" minOccurs="0"/>
 </xs:sequence>
</xs:complexType>
<xs:complexType name="abstract_data_t"/>
<xs:complexType name="abstract_protocol_t" abstract="true">
 <xs:sequence>
  <xs:element name="protocolOffset" type="protocolOffset_t" minOccurs="0"
maxOccurs="unbounded"/>
 </xs:sequence>
 <xs:attributeGroup ref="ID_name_description"/>
 <xs:attributeGroup ref="terminology_ag"/>
 <xs:attribute name="level" type="levelDescriptor">
  <xs:annotation>
```

```
   <xs:documentation>Describes the level of the XCEDE hierarchy that this protocol
instance should be validated against</xs:documentation>
  </xs:annotation>
 </xs:attribute>
 <xs:attribute name="required" type="xs:boolean"/>
 <xs:attribute name="minOccurences" type="xs:integer"/>
 <xs:attribute name="maxOccurences" type="xs:integer">
  <xs:annotation>
   <xs:documentation>Are these occurences within a step (i.e. during a single time
point)?  How to refer to repeats across steps?</xs:documentation>
  </xs:annotation>
 </xs:attribute>
 <xs:attribute name="minTimeFromStart" type="xs:string">
  <xs:annotation>
   <xs:documentation>Absolute time from start of overall protocol</xs:documentation>
  </xs:annotation>
 </xs:attribute>
 <xs:attribute name="maxTimeFromStart" type="xs:string"/>
</xs:complexType>
<!--*********            Misc. types            *********-->
<xs:complexType name="ref_t">
 <xs:simpleContent>
  <xs:extension base="xs:string"/>
 </xs:simpleContent>
</xs:complexType>
<xs:complexType name="assessment_t">
 <xs:complexContent>
  <xs:extension base="abstract_data_t">
   <xs:sequence>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="dataInstance" minOccurs="0" maxOccurs="unbounded">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="assessmentInfo" type="assessmentInfo_t" minOccurs="0">
        <xs:annotation>
         <xs:documentation>Block for describing things like informant, clinical rater,
assessment date, etc</xs:documentation>
        </xs:annotation>
       </xs:element>
       <xs:element name="assessmentValue" type="assessmentValue_t" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="validated" type="xs:boolean" default="false">
       <xs:annotation>
        <xs:documentation>Indicates whether the instance has been validated (e.g. by
reconciling double-entry instances).  There should be only one validated instance per
assessment.</xs:documentation>
       </xs:annotation>
      </xs:attribute>
     </xs:complexType>
    </xs:element>
    <xs:element name="annotation" type="textAnnotation_t" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="assessmentInfo_t">
 <xs:complexContent>
  <xs:extension base="acquisitionInfo_t"/>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="assessmentValue_t">
 <xs:sequence>
  <xs:element name="itemText" minOccurs="0">
   <xs:annotation>
```

```
    <xs:documentation>'itemText' describes the text preceeding and following an
assessment form item. It is different then 'itemName' which stores the shortened item
identifier</xs:documentation>
   </xs:annotation>
   <xs:complexType>
    <xs:sequence>
     <xs:element name="leadText" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
     <xs:element name="trailText" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="itemName" type="xs:string">
   <xs:annotation>
    <xs:documentation>'itemName' is a short identifier for the assessment item.
'itemText' should be used for the actual question text</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="valueStatus" type="xs:string">
   <xs:annotation>
    <xs:documentation>Information on the status of a value (e.g. subject refused to
answer)</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="itemValue" type="value_t">
   <xs:annotation>
    <xs:documentation>Actual value of the assessment item as recorded on the
form</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="itemNormValue" type="value_t" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Normalized or scaled value of the assessment
item</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="reconciliationNote" type="textAnnotation_t" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Normalized or scaled value of the assessment
item</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="annotation" type="textAnnotation_t" minOccurs="0"/>
 </xs:sequence>
 <xs:attributeGroup ref="terminology_ag"/>
</xs:complexType>
<xs:complexType name="analysis_t"/>
<xs:complexType name="investigator_t">
 <xs:complexContent>
  <xs:extension base="person_t"/>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="authoredText_t">
 <xs:simpleContent>
  <xs:extension base="xs:string">
   <xs:attributeGroup ref="authoredText_ag"/>
  </xs:extension>
 </xs:simpleContent>
</xs:complexType>
<xs:complexType name="abstract_annotation_t" abstract="true">
 <xs:attributeGroup ref="authoredText_ag"/>
</xs:complexType>
<xs:complexType name="textAnnotation_t">
 <xs:complexContent>
  <xs:extension base="abstract_annotation_t">
   <xs:sequence>
```

```
    <xs:element name="text" type="xs:string" minOccurs="0"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="nsTermAnnotation_t">
 <xs:complexContent>
  <xs:extension base="abstract_annotation_t">
   <xs:sequence>
    <xs:element name="ontologyClass" type="xs:string" maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="nsOntologyAnnotation_t">
 <xs:complexContent>
  <xs:extension base="abstract_annotation_t">
   <xs:sequence>
    <xs:element name="term" type="xs:string" maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="generator_t">
 <xs:sequence>
  <xs:element name="application" type="versionedEntity_t">
   <xs:annotation>
    <xs:documentation>Program used to generate document</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="invocation" type="xs:string">
   <xs:annotation>
    <xs:documentation>Application input required to generate this document. Should be
explicit such that this document can be re-generated from this info</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="dataSource" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Description of data source with version numbers and/or
timestamp of data</xs:documentation>
   </xs:annotation>
  </xs:element>
 </xs:sequence>
</xs:complexType>
<xs:attributeGroup name="authoredText_ag">
 <xs:attribute name="author" type="xs:string"/>
 <xs:attribute name="timestamp" type="xs:dateTime"/>
</xs:attributeGroup>
<xs:attributeGroup name="terminology_ag">
 <xs:attribute name="termID" type="xs:string">
  <xs:annotation>
   <xs:documentation>Applications will likely want to constrain what are
valid IDs within the context of their application (for example, allowing only
LSID's)</xs:documentation>
  </xs:annotation>
 </xs:attribute>
 <xs:attribute name="termPath" type="xs:string"/>
</xs:attributeGroup>
<xs:simpleType name="uniqueID_t">
 <xs:annotation>
  <xs:documentation> Having a distinct unique ID type is a convenience for building
referential
   links. The reason we are not using the native XML Schema ID attribute is that
enforces
   document-wide uniqueness, whereas there may be instances of this bioterm schema
that contain
```

```
   multiple namespace-qualified term or ontology class sets where IDs are unique
within their
   namespace but not across the entire document. </xs:documentation>
 </xs:annotation>
 <xs:restriction base="xs:string">
  <xs:pattern value="[A-Za-z0-9\-:_.]+"/>
 </xs:restriction>
</xs:simpleType>
<xs:complexType name="person_t">
 <xs:annotation>
  <xs:documentation>Add additional fields (address, email, etc)</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="salutation" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>e.g. Dr., Mr., Mrs.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="givenName" type="xs:string" minOccurs="0"/>
  <xs:element name="middleName" type="xs:string" minOccurs="0"/>
  <xs:element name="surname" type="xs:string" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Used for last name or only name (e.g. Prince)</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="academicTitles" type="xs:string" minOccurs="0"/>
  <xs:element name="institution" type="xs:string" minOccurs="0"/>
  <xs:element name="department" type="xs:string" minOccurs="0"/>
 </xs:sequence>
 <xs:attribute name="ID" type="xs:string"/>
 <xs:attribute name="role" type="xs:string"/>
</xs:complexType>
<xs:complexType name="mixedType_t" mixed="true"/>
<xs:complexType name="versionedEntity_t">
 <xs:simpleContent>
  <xs:extension base="xs:string">
   <xs:attribute name="version" type="xs:string"/>
  </xs:extension>
 </xs:simpleContent>
</xs:complexType>
<xs:simpleType name="levelDescriptor">
 <xs:restriction base="xs:string">
  <xs:enumeration value="project"/>
  <xs:enumeration value="subject"/>
  <xs:enumeration value="visit"/>
  <xs:enumeration value="study"/>
  <xs:enumeration value="episode"/>
  <xs:enumeration value="acquisition"/>
 </xs:restriction>
</xs:simpleType>
<xs:complexType name="protocolOffset_t">
 <xs:sequence>
  <xs:element name="protocolTimeRef" type="xs:string" minOccurs="0"/>
  <xs:element name="preferedTimeOffset" type="xs:string" minOccurs="0"/>
  <xs:element name="minTimeOffset" type="xs:string" minOccurs="0"/>
  <xs:element name="maxTimeOffset" type="xs:string" minOccurs="0"/>
 </xs:sequence>
</xs:complexType>
<xs:complexType name="protocolMeasure_t">
 <xs:attribute name="name"/>
 <xs:attribute name="required"/>
 <xs:attribute name="minValue"/>
 <xs:attribute name="maxValue"/>
</xs:complexType>
<xs:complexType name="protocol_t">
 <xs:complexContent>
  <xs:extension base="abstract_protocol_t">
```

```
   <xs:sequence>
    <xs:element name="steps" minOccurs="0">
     <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
       <xs:element name="step" type="abstract_protocol_t"/>
       <xs:element name="stepRef" type="ref_t"/>
      </xs:choice>
     </xs:complexType>
    </xs:element>
    <xs:element name="measures" minOccurs="0">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="measure" type="protocolMeasure_t" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:attributeGroup name="ID_name_description">
 <xs:attribute name="ID" type="xs:string"/>
 <xs:attribute name="name" type="xs:string"/>
 <xs:attribute name="description" type="xs:string"/>
</xs:attributeGroup>
<xs:complexType name="terminologyString_t">
 <xs:simpleContent>
  <xs:extension base="xs:string">
   <xs:attributeGroup ref="terminology_ag"/>
  </xs:extension>
 </xs:simpleContent>
</xs:complexType>
<xs:complexType name="admin">
 <xs:simpleContent>
  <xs:extension base="xs:string"/>
 </xs:simpleContent>
</xs:complexType>
<xs:complexType name="revision_t">
 <xs:sequence>
  <xs:element name="timestamp" type="xs:dateTime" minOccurs="0"/>
  <xs:element name="generator" type="generator_t" minOccurs="0"/>
  <xs:element name="annotation" type="textAnnotation_t" minOccurs="0"/>
 </xs:sequence>
 <xs:attribute name="ID" type="xs:string"/>
</xs:complexType>
<xs:complexType name="value_t">
 <xs:sequence>
  <xs:element name="actualValue" type="xs:string"/>
  <xs:element name="dataClassification" type="valueTypes_t" minOccurs="0"/>
  <xs:element name="unitRef" minOccurs="0">
   <xs:complexType>
    <xs:annotation>
     <xs:documentation>'unitDefID' is a reference to a unitDef block defining
units</xs:documentation>
    </xs:annotation>
    <xs:attribute name="unitDefID" type="xs:string"/>
   </xs:complexType>
  </xs:element>
  <xs:element name="codeRef" minOccurs="0">
   <xs:complexType>
    <xs:annotation>
     <xs:documentation>'codeDefID' is a reference to a codeDef block defining coded
values</xs:documentation>
    </xs:annotation>
    <xs:attribute name="codeDefID" type="xs:string"/>
   </xs:complexType>
```

```
  </xs:element>
 </xs:sequence>
</xs:complexType>
<xs:simpleType name="valueTypes_t">
 <xs:restriction base="xs:string">
  <xs:enumeration value="float"/>
  <xs:enumeration value="boolean"/>
  <xs:enumeration value="varchar"/>
  <xs:enumeration value="integer"/>
  <xs:enumeration value="URI"/>
 </xs:restriction>
</xs:simpleType>
<!--*********            Provenance types            *********-->
<xs:complexType name="processStep_t">
 <xs:sequence>
  <xs:element name="package" type="versionedEntity_t" minOccurs="0"/>
  <xs:element name="program" type="versionedEntity_t"/>
  <xs:element name="programInvocation" type="mixedType_t"/>
  <xs:element name="timeStamp" type="mixedType_t"/>
  <xs:element name="cvs" type="mixedType_t" minOccurs="0"/>
  <xs:element name="user" type="mixedType_t" minOccurs="0"/>
  <xs:element name="machine" type="mixedType_t" minOccurs="0"/>
  <xs:element name="platform" type="versionedEntity_t" minOccurs="0"/>
  <xs:element name="compiler" type="versionedEntity_t" minOccurs="0"/>
  <xs:element name="library" type="versionedEntity_t" minOccurs="0"
maxOccurs="unbounded"/>
 </xs:sequence>
</xs:complexType>
<xs:complexType name="provenance_t">
 <xs:annotation>
  <xs:documentation>Note: sourceData should be included along with application
parameters and configuration values</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="processStep" type="processStep_t" maxOccurs="unbounded"/>
 </xs:sequence>
 <xs:attribute name="ID" type="xs:string"/>
</xs:complexType>
<!--*********            Resource types            *********-->
<xs:complexType name="abstract_resource_t" abstract="true">
 <xs:annotation>
  <xs:documentation>A resource is something that we haven't agreed on
yet.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="provenance" type="provenance_t" minOccurs="0"/>
  <xs:element name="metaFields" minOccurs="0">
   <xs:complexType>
    <xs:sequence minOccurs="0">
     <xs:element name="metaField" minOccurs="0" maxOccurs="unbounded">
      <xs:complexType>
       <xs:simpleContent>
        <xs:extension base="xs:string">
         <xs:attribute name="name" type="xs:string"/>
        </xs:extension>
       </xs:simpleContent>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="uri" type="frag_uri_t" minOccurs="0" maxOccurs="unbounded"/>
 </xs:sequence>
 <xs:attributeGroup ref="ID_name_description"/>
 <xs:attribute name="format" type="xs:string" use="optional">
  <xs:annotation>
   <xs:documentation>Format of file. E.g. DICOM, Analyze, 4dfp</xs:documentation>
```

```
  </xs:annotation>
 </xs:attribute>
 <xs:attribute name="content" type="xs:string" use="optional">
  <xs:annotation>
   <xs:documentation>Code indicating the contents of the image. E.g. GFC,
T88</xs:documentation>
  </xs:annotation>
 </xs:attribute>
 <xs:attribute name="cachePath" use="optional">
  <xs:simpleType>
   <xs:restriction base="xs:string">
    <xs:maxLength value="255"/>
   </xs:restriction>
  </xs:simpleType>
 </xs:attribute>
</xs:complexType>
<xs:complexType name="informationResource_t">
 <xs:complexContent>
  <xs:extension base="abstract_resource_t"/>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="binaryDataResource_t">
 <xs:complexContent>
  <xs:extension base="abstract_resource_t">
   <xs:sequence>
    <xs:element name="elementType" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element describes the type of individual data elements in
the data record.  For numeric data types, this indicates whether the element type is
a signed integer ("int"), unsigned integer ("uint"), or floating-point ("float"), as
well as the number of bits allocated to each element.</xs:documentation>
     </xs:annotation>
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:enumeration value="int8"/>
       <xs:enumeration value="uint8"/>
       <xs:enumeration value="int16"/>
       <xs:enumeration value="uint16"/>
       <xs:enumeration value="int32"/>
       <xs:enumeration value="uint32"/>
       <xs:enumeration value="int64"/>
       <xs:enumeration value="uint64"/>
       <xs:enumeration value="float32"/>
       <xs:enumeration value="float64"/>
       <xs:enumeration value="ascii"/>
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
    <xs:element name="byteOrder" minOccurs="0">
     <xs:annotation>
      <xs:documentation>This element describes whether the individual data elements
in the data record are stored with the most-significant-byte first (msbfirst)
or least-significant-byte first (lsbfirst).  This element is required if the
the data type given by the "elementtype" element has a size larger than one
byte.</xs:documentation>
     </xs:annotation>
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:enumeration value="lsbfirst"/>
       <xs:enumeration value="msbfirst"/>
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
```

```
<xs:complexType name="dimensionedBinaryDataResource_t">
 <xs:complexContent>
  <xs:extension base="binaryDataResource_t">
   <xs:sequence>
    <xs:element name="dimension" type="binaryDataDimension_t" maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="mappedBinaryDataResource_t">
 <xs:complexContent>
  <xs:extension base="binaryDataResource_t">
   <xs:sequence>
    <xs:element name="dimension" type="mappedBinaryDataDimension_t"
maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="binaryDataDimension_t">
 <xs:annotation>
  <xs:documentation xml:lang="en">This element stores information about one of the N
dimensions in the data record.  Multiple instances of this element are ordered from
fastest-moving to slowest-moving.  These elements provide information to describe the
size (in data elements) of the N-dimensional bounding box for the data, and in some
cases to describe the mapping of indexes within this bounding box to 'real-world'
coordinates.</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="size" type="xs:int" minOccurs="0">
   <xs:annotation>
    <xs:documentation xml:lang="en">The number of elements in the data along one
traversal of this dimension.</xs:documentation>
   </xs:annotation>
  </xs:element>
 </xs:sequence>
 <xs:attribute name="label" type="xs:string">
  <xs:annotation>
   <xs:documentation xml:lang="en">This is a label for the dimension.  The first three
spatial dimensions (or however many exist) must be labeled, in order, 'x', 'y', and
'z'.  The first temporal dimension must be labeled 't'.
       The data record allows the writer to specify some measure of data
permutation/dimension merging when data is read and presented to a higher application
level.  This is useful, for example, if the data is stored in Siemens' Mosaic DICOM
format, where slices of a 3-D volume are arranged to look like they are tiled onto a
square 2-D area.  In this case, what would normally be called the 'z' dimension has
two forks, one that occurs before the 'y' dimension (the first row in the data covers
the first row of several slices), and one that occurs after the 'y' dimension.  If,
as in this case, there are two or more dimensions that should be merged into one,
they should be labeled 'DIMTYPE-splitRANK' where DIMTYPE is the name of the resultant
merged dimension, and RANK specifies the order in which all 'split' dimensions of this
type will be merged.  After merging, the resultant dimension should have the values
of the highest-ranked DIMTYPE split dimension, except for the 'size' element, which
will be the product of the sizes of all DIMTYPE split dimensions.  The position of
the resultant dimension should be the position of the highest-ranked DIMTYPE split
dimension.  The data itself should also be reordered to reflect the new dimension
structure.</xs:documentation>
  </xs:annotation>
 </xs:attribute>
 <xs:attribute name="outputselect" type="xs:string" use="optional">
  <xs:annotation>
   <xs:documentation xml:lang="en">In the same way that the 'label' attribute allows
you to specify dimensions that should be merged before presenting the data to an
application, this attribute specifies a data filter along this dimension.  If this
attribute exists, it should contain a whitespace-separated list of indices (indexed
starting at 0).  Only data points along this dimension that occur in the index list
should be presented to the application.  Likewise, the 'size' of the dimension, after
```

```
selection, should be updated to reflect the new size of this dimension (which should
be the number of indices in the content of this attribute).</xs:documentation>
  </xs:annotation>
 </xs:attribute>
</xs:complexType>
<xs:complexType name="mappedBinaryDataDimension_t">
 <xs:complexContent>
  <xs:extension base="binaryDataDimension_t">
   <xs:sequence>
    <xs:element name="origin" type="xs:float" minOccurs="0">
     <xs:annotation>
      <xs:documentation xml:lang="en">A value assigned to the first data element
along this dimension.  For example, if this dimension corresponds to "time",
this element could store the time corresponding to the first data element.
If this is a two-dimensional projection of the surface of the Earth, and this
dimension takes you around the Earth parallel to the equator, this value could
be the degrees longitude.  For MRI data, this is the single coordinate on the
Left-to-Right, Posterior-to-Anterior, or Inferior-to-Superior axis to which this
dimension most closely matches (see 'direction' element and 'rasOrigin' element in
'mrImageDataResource_t').</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="spacing" type="xs:float" minOccurs="0">
     <xs:annotation>
      <xs:documentation xml:lang="en">This is the average distance between consecutive
data elements in this dimension.  If the spacing is not regular, then it may be
possible to calculate the actual distance between any two data elements in this
dimension using the 'datapoints' element.</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="gap" type="xs:float" minOccurs="0">
     <xs:annotation>
      <xs:documentation xml:lang="en">This is the length of the unsampled space
between consecutive data elements in this dimension, i.e. the distance between the end
of one data element and the beginning of the next.  For MRI data, this can be used to
specify the gap between two collected slices &#x02015; the actual width of each slice
can be calculated as 'spacing' minus 'gap'.</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="datapoints" type="xs:string" minOccurs="0">
     <xs:annotation>
      <xs:documentation xml:lang="en">The content of this element is either (1) a
whitespace-separated list of values, or (2) a list of 'value' elements, that can
be used as a label for each data point along this dimension.  The values can be
numbers representing points on an axis (this is the typical case), text strings,
coordinate tuples, etc.  Any datapoint label that includes whitespace (coordinate
tuples included) must be encapsulated within a child 'value' element.  If this element
is missing, it is assumed that labels can be calculated using information in other
fields (such as 'origin', 'spacing', etc.).  This element is particularly useful for
dimensions with irregular spacing.</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="direction" type="listoffloats_t" minOccurs="0">
     <xs:annotation>
      <xs:documentation xml:lang="en">This element contains a vector (represented as
a whitespace-separated list of floating-point values in the appropriate coordinate
system) that is parallel to this dimension's edge of the bounding box.  The vector
starts at the first element in the data and points towards subsequent elements along
this dimension.  For MRI data, this should be a unit vector in (R,A,S) coordinates
(positive values are Right, Anterior, or Superior respectively) &#x02015; for 'x' and
'y' dimensions, this corresponds to the two vectors in the ImagePatientOrientation
field in DICOM.</xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="units" type="xs:string" minOccurs="0">
     <xs:annotation>
```

```
      <xs:documentation xml:lang="en">This stores the units used for all numeric
values in this dimension element.  In MRI data, this should be 'ms' for all spatial
dimensions ('x', 'y', 'z') and 'ms' for the temporal dimension 't'.</xs:documentation>
     </xs:annotation>
    </xs:element>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="frag_uri_t">
 <xs:annotation>
  <xs:documentation>The external data pointed to by this uri is a "fragment", where a
"fragment" is defined as a stream of data contiguously stored in the same file offset
by 'offset' bytes and of 'size' bytes.</xs:documentation>
 </xs:annotation>
 <xs:simpleContent>
  <xs:extension base="xs:anyURI">
   <xs:attribute name="offset" type="xs:unsignedLong">
    <xs:annotation>
     <xs:documentation>The data for this fragment will start at this byte position in
the resource specified by the 'uri' element.  If this attribute does not exist or is
empty, it is assumed to be zero.</xs:documentation>
    </xs:annotation>
   </xs:attribute>
   <xs:attribute name="size" type="xs:unsignedLong">
    <xs:annotation>
     <xs:documentation>This specifies the size(s) of this block in the resource
specified by the 'uri' element.  If this attribute does not exist or is empty, it is
calculated using the dimension and elementtype element.</xs:documentation>
    </xs:annotation>
   </xs:attribute>
  </xs:extension>
 </xs:simpleContent>
</xs:complexType>
<xs:simpleType name="listoffloats_t">
 <xs:list itemType="xs:float"/>
</xs:simpleType>
<xs:complexType name="catalog_t">
 <xs:sequence minOccurs="0">
  <xs:element name="metaDataRef" type="ref_t" minOccurs="0">
   <xs:annotation>
    <xs:documentation>Reference to the meta-date element that this catalog is related
to.  Should point to one of the core xcede hierarchy components: project, subject,
visit, study, acquisition.</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="catalogList" minOccurs="0">
   <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
     <xs:element name="catalog" type="catalog_t"/>
     <xs:element name="catalogRef" type="ref_t"/>
    </xs:choice>
   </xs:complexType>
  </xs:element>
  <xs:element name="entryList" minOccurs="0">
   <xs:complexType>
    <xs:choice minOccurs="0" maxOccurs="unbounded">
     <xs:element name="entry" type="abstract_resource_t"/>
     <xs:element name="entryRef" type="ref_t"/>
    </xs:choice>
   </xs:complexType>
  </xs:element>
 </xs:sequence>
 <xs:attribute name="ID" type="xs:string" use="optional"/>
</xs:complexType>
<xs:complexType name="format_t">
 <xs:annotation>
```

```
  <xs:documentation>Container for describing imaging formats and file name extensions
(currently underimplemented)</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="description" type="xs:string" minOccurs="0"/>
  <xs:element name="documentationList" minOccurs="0">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="documentation" type="informationResource_t" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
  <xs:element name="extensionList" minOccurs="0">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="extension" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>
 </xs:sequence>
 <xs:attribute name="name"/>
</xs:complexType>
<xs:complexType name="catalog_t_expt">
 <xs:complexContent>
  <xs:extension base="abstract_resource_t"/>
 </xs:complexContent>
</xs:complexType>
<!--*********            Event types            *********-->
<xs:complexType name="events_t">
 <xs:complexContent>
  <xs:extension base="abstract_data_t">
   <xs:sequence>
    <xs:element name="params" type="eventParams_t" minOccurs="0"/>
    <xs:element name="event" type="event_t" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="description" type="xs:string" minOccurs="0"/>
    <xs:element name="annotation" type="textAnnotation_t" minOccurs="0"
maxOccurs="unbounded"/>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>
<xs:complexType name="event_t">
 <xs:sequence>
  <xs:element name="onset" type="xs:float" minOccurs="0"/>
  <xs:element name="duration" type="xs:float" minOccurs="0"/>
  <xs:element name="value" type="eventValue_t" minOccurs="0" maxOccurs="unbounded"/>
  <xs:element name="annotation" type="textAnnotation_t" minOccurs="0"
maxOccurs="unbounded"/>
 </xs:sequence>
 <xs:attribute name="type" type="xs:string" use="optional"/>
 <xs:attribute name="units" type="xs:string" use="optional">
  <xs:annotation>
   <xs:documentation>This attribute is optional, but an group using this schema should
agree on, use, and enforce the same measurement units consistently, to avoid the need
for unit conversion in an application.</xs:documentation>
  </xs:annotation>
              </xs:attribute>
 <xs:attribute name="name" type="xs:string" use="optional"/>
</xs:complexType>
<xs:complexType name="eventValue_t">
 <xs:simpleContent>
  <xs:extension base="xs:string">
   <xs:attribute name="name" type="xs:string"/>
   <xs:anyAttribute processContents="lax"/>
  </xs:extension>
```

```
  </xs:simpleContent>
 </xs:complexType>
 <xs:complexType name="eventParams_t">
  <xs:sequence>
   <xs:element name="value" type="eventValue_t" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
 </xs:complexType>
</xs:schema>
```