Rosetta Inpharmatics Initial Submission regarding the Gene Expression RFP

Rosetta Inpharmatics

OMG Document lifesci/2000-11-13

Table of Contents

List of Figures

1 Preface

This document describes a proposal for standards in response to RFP, Gene Expression, Object Management Group (OMG) document number lifesci/00-03-09.

1.1 Submission Contact Points

The primary editor and contact point for this submission is

Michael Miller
Rosetta Inpharmatics
12040 115th Avenue NE
Kirkland, WA 98034
USA
(+1) 425 636 6352
mmiller@rii.com
www.rii.com

1.2 Supporting Organizations

The following organization has been involved in the process of developing, prototyping, and/or reviewing this submission. The submitter of this response thanks the company for participating and giving their valuable input.

Agilent Technologies, Inc., Palo Alto, California

1.3 Acknowledgements

The author of this document wishes to express his appreciation to those listed below (in alphabetic order) for their contributions of ideas and experience. Ultimately, the ideas expressed in this document are those of the author and do not necessarily reflect the views or ideas of these individuals, nor does the inclusion of their names imply an endorsement of the final product.

Doug Bassett	dbassett@rii.com
Mark Boguski	mboguski@rii.com
Herb Cattell	herb_cattell@agilent.com
Glenda Delenstarr	glenda_delenstarr@agilent.com
Katherine Farber	kfarber@rii.com
Stephen Friend	sfriend@rii.com
John Gardner	jgardner@rii.com
Brandon Hunt	bhunt@rii.com
Matthew Kidd	mkidd@rii.com
Robert Kincaid	robert_kincaid@labs.agilent.com
Jeff King	jsking@rii.com
Emily Schultz	eschultz@rii.com
Roland Stoughton	rstoughton@rii.com
Charles Troup	charles_troup@agilent.com
Peter Webb	peter_webb@labs.agilent.com

1.4 Proof of Concept

Rosetta Inpharmatics and Agilent Technologies have been using the GEML™ 1.0 format as part of internal pipelines for the past year. Rosetta has been continuously loading XML files on the order of thirteen megabytes into the Rosetta Resolver™ system, an enterprise expression data analysis product. We recently used internal tools to export the more than one thousand profiles, assigned annotations, and supporting patterns that constituted the data for the article, *Functional Discovery via a Compendium of Expression Profiles*, that appeared in the July 7, 2000 issue of *Cell*. The total size of the export, when compressed, was a little over a half of gigabyte of data. That data was then imported by Harvard into their Rosetta Resolver system.

We have not, as of yet, implemented the interfaces contained in this proposal but given that the size of the compressed XML files has proven no technical obstacle, we see no technical problems in implementing the interfaces.

Rosetta has developed the freeware GEML Conductor™ tools for visualization of GEML formatted data and for conversion of gene expression data in other formats into GEML.

2 Introduction

2.1 General Remarks

This document contains a proposal for a standard that addresses the representation of gene expression data and relevant annotations, as well as mechanisms for exchanging these data.

The field of gene expression experiments has several distinct technologies that a standard must include. These include single vs. dual channel experiments, cDNA vs. oligonucleotides. Because of these different technologies and different types of gene expression experiments, it is not expected that all aspects of the standard will be used by all organizations.

Given the massive amount of data associated with a single set of experiments, we feel that Extensible Markup Language (XML) is the best way to describe the data. The use of a Document Type Definition (DTD) allows a well-defined tag set, a vocabulary, to describe the domain of gene expression experiments. It also has the virtue of compressing very well so that files in an XML format compress to ten percent of their original size. XML is now widely accepted as a data exchange format across multiple platforms.

Organizations that request these XML streams can use freely available implementations of either of the W3C recommended DOM or the XML-DEV SAX parsing interfaces to create import applications. These import applications can be tailored for the specific needs of the organization without the need to burden the vocabulary of the XML with specifics of any organization's schema requirements.

With the coming acceptance of XML Metadata Interchange as an OMG standard and the support already available in modeling tools, it is now possible to specify a representation of an XML vocabulary as a model in a similar fashion as that used for IDL so that the specification can be interchanged between organizations. The new RFP template (ab/00-05-02), which came out after the Gene Expression RFP, contains an additional section (5.2.5) under Mandatory Requirements that sets the requirement for inclusion of Standardized Metadata in the form of XMI.

UML diagrams will be used for illustrative purposes. Typically, previous LSR proposals have used the IDL valuetype to represent the structure of the data, with certain restrictions. With the acceptance of XML as a means of representing data, the author has made the choice to vary from this practice. The interfaces in the IDL module that return data will return a stream formatted as XML and conforming to the Domain Model. The **bibliographic_ref** (section 4.1.3) element provides an example mapping of the valuetypes from Bibliographic Query Service (lifesci/00-09-04) to XML elements while the mapping of XML elements and attributes to valuetypes is similarly straight-forward.

Given the above, the Domain Model will be described by its representation in the Document Type Definition. Although not described in the Object Management Architecture Guide, we feel that this representation is appropriate for this proposal and is in the spirit of Section 4.9.1 of the Gene Expression RFP,

> "The models and terminology in the *Object Management Architecture Guide* and *CORBA* should be used in your submission. Where you believe this is not appropriate, describe and provide a rationale for the models and

terminology you believe OMG should use."

The preceding paragraphs are our rational.

A few design principles and patterns that we have used are outlined first.

2.1.1 Representing Relationships

There is a need in the XML vocabulary that describes the gene expression data to link two elements together, for instance, a Reporter to its BioSequence or a Hybridization to its Labeled Prep. There are three standard ways that are generally used in XML.

1. Using ID and IDREF or IDREFS attributes, where the IDREF or IDREFS values of an element 'point' to the ID value of one or more related elements within the document.
2. Using an XPointer to refer to an element or attribute in the current document or in another.
3. Using defined relationships that are implemented by applications.

We have chosen the third method for two reasons:

1. In general, both the physical items of gene expression data (the chip, the labeled prep, solvents, etc.) and the virtual (array pattern, biosequence cluster, etc.) are assigned unique identifiers within an organization or department of an organization. These identifiers can be used as the reference value.
2. By defining reference elements in the vocabulary, the reference, itself, can serve as a place holder for what it references and the actual data that is referenced does not need to be in the document. This allows the most freedom of inter- and intra-document references.

Where the first isn't true, the element and its reference can be provided in the same document and their reference resolved within the document. An implementation is free to take as much advantage of the second point to provide efficiency on download as desired.

There will be three reference elements specified, **algorithm_ref**, **external_ref** and **biliographic_ref,** whose reference is not expected to be provided by an exporting source. The definition of **algorithm_ref** is outside the scope of this document. The second, **external_ref**, is used when what is being exported was itself imported from another source and is to allow the ability to track the original source. The third, **bibliographic_ref**, is to allow a lookup of appropriate literature from a third party source.

The following naming convention is used:

The element **foo** with unambiguous identifier attribute **bar** would have a reference element named **foo_ref** with an attribute **bar**. In the case of multiple attributes defining an unambiguous identifier, the number of attributes would simply be extended.

2.1.2 *Use of **CosNaming::StringName***

The **pattern** and **profile type** attributes in the DTD, that name the particular gene expression technology type, can be represented as a qualified string. Even when two organizations use the same technology to process an array, their interpretation and methodology may be sufficiently different that the values from hybridizations cannot be compared directly. The **AttributePath** IDL attribute, used to uniquely name an attribute of a DTD element, can also be represented as a path.

We have found that using a convention similar to that described in the Interoperable Naming service, with the additional constraints below, provides a good naming system for the above. The following is from the Bibliographic Query Service (lifesci/00-09-04) which we will suggest to form **type** and **AttributePath** names.

> "To allow strings, yet mitigate their potential for abuse, this standard (and indeed some other LSR standards [BSA], [MAPS]) uses the syntax convention of **CosNaming::StringName** as described in the Interoperable Naming service [INS]. This convention is mainly a syntactical one; in no way is the use of a naming service implementation required or implied (but it is not precluded either).
>
> A brief description of **CosNaming::StringName** is as follows. **CosNaming::Name** is a list of **struct NameComponent**s. For the purpose of illustration, a **NameComponent** can be likened to a directory or filename, whereas **CosNaming::Name** constitutes a full path-name. The **struct NameComponent** has string members **id** and **kind**. To transform a **CosNaming::Name** into a string, all its **NameComponent**s are represented as strings " id.kind". If the **kind**-field is empty, this becomes simply " id"; if the **id**-field is empty, this becomes ".kind"; finally, the Naming service also allows both the **id**- and **kind**-fields to be empty, which is represented as ".". The full *stringified* **CosNaming::Name** is then obtained by concatenating all the **NameComponent**s using "/" as a separator character. The character "\" is designated as an escape character; if it precedes any of the special characters ".", "/" and "\", these special characters are taken as literal characters."

For **type**, typically the first name component can be the organization name that is using the technology, then any inner name components can have as **ids** the departments within the organization, if needed, and the final name component **id** would be the technology, such as ArrayVision™ (Imaging Research Inc.) for scanning or UniGEM™ (Incyte Genomics, Inc.) for array patterns.

For **AttributePath**, the first **id** is one of the sub-vocabulary elements (see below) then subsequenct **id** fields are the contained elements to the predicate's attribute element where the XML attribute is the **kind** field. If it is unambiguous, the attribute's element and the attribute are sufficient.

For example, if a predicate is on a **biosequence_ref's identifier**, **AttributePath** equals "pattern/reporter/biosequence_ref.identifier" or, simply "biosequence_ref.identifier".

2.1.3 *Extensibility*

Two mechanisms are used that allow additional information to be included for flexibility and extensibility.

The **other** element allows arbitrary name and value pairs to be specified. It is the first contained element of every element, allowing information to be included that can then be used in the processing of the next contained elements.

Occasional use of DTD ANY type elements allows, for those elements, arbitrary elements to be nested. This approach is used for **species_data**, **channel_data** and for elements nested in **statistics**. This allows organizations to define elements tailored to their needs, whose description can be obtained through the **GeneExpUtilities::dtdExtensions** IDL attribute.

2.1.4 Dates

There are several attributes that represent dates. The suggested encoding is carried over from the Bibliographic Query Service as and is as defined in a W3C NOTE “Date and Time Formats”. This NOTE defines a profile of ISO8601 standard. The NOTE reduces the scope and restricts the supported formats to a small number. The profile offers a number of options from which this specification permits the following ones:

- Year

  YYYY (e.g. 2000)

- Year and month

  YYYY-MM (e.g. 2000-12)

- Complete date

  YYYY-MM-DD (e.g. 2000-12-31)

- Complete date plus hours, minutes and seconds

  YYYY-MM-DDThh:mm:ssZ (e.g. 2000-12-31T23:59:59Z)

Where

YYYY	four-digit year
MM	two-digit month (01=January, etc.)
DD	two-digit day of month (01 through 31)
hh	two digits of hour (00 through 23)
mm	two digits of minute (00 through 59)
ss	two digits of second (00 through 59)

Exactly the components shown here must be present, with exactly this punctuation. Note that the "T" appears literally in the string, to indicate the beginning of the time element, as specified in ISO 8601.

Times are expressed in UTC (Coordinated Universal Time), with a special UTC designator ("Z"), again as specified in ISO 8601.

2.1.5 *Conventions*

We follow the OMG IDL Style Guide (ab/98-06-03) for the description of interfaces.

For the *Modules and Interfaces*, interface, valuetype, exception, and module names appear using this font when used inline: **InterfaceName**. Individual interface descriptions use the following pattern.

InterfaceName

Interface description.

```
    interface InterfaceName
    {
        ResultType operation_name(arguments) raises(ExceptionName);
    };
```

```
ResultType operation_name(arguments) raises(ExceptionName);
```

Description

Operation description.

Return value

Returns a **ResultType**.

Exceptions

Raises an **ExceptionName**.

We follow the following style for the description of elements of Document Type Definitions.

For the *Domain Model*, elements, attributes, processing instructions, and entities appear using this font when used inline: **ElementName**. Individual element descriptions use the following pattern.

ElementName

Element Description.

```
    <!ELEMENT elementName
        ((element1 |
          element2*)+,
         element3?)
    >
    <!ATTLIST elementName
        attribute1 CDATA  #REQUIRED
        attribute2 CDATA  #IMPLIED
        attribute3 CDATA  'default'
        attribute4 (enum1 | enum2 | enum3) 'enum2'
    >
```

Attribute	**Description**
attribute1	Description of attribute1
attribute2	Description of attribute2
attribute3	Description of attribute3
attribute4	Description of attribute4

2.2 Overview

The module and domain model below addresses the area of querying a source of gene expression data.

2.2.1 Module DsLSRGeneExpQuery

The DsLSRGeneExpQuery module provides the means to interrogate a gene expression data source for available experiment sets and to obtain information on data subsets of gene expression experiment sets through the **find**() method of the **GeneExpQuery** interface and information on the controlled vocabularies of the data source through the **VocabularyFinder voc_finder**.

The **export**() method of the **GeneExpQuery** interface is used to obtain selected small experiment sets directly, while the **export_to_location**() method allows arbitrarily large experiment sets to be exported to a well-known location.

Attributes and Properties of the data source are obtained from the **GeneExpUtilities** interface.

2.2.2 Domain Model DsLSR_GEML

The domain model is expressed in XML. With the evolution of modeling tools and specifications, it is possible to define a specific XML vocabulary which is XMI compliant, facilitating specification interchange between organizations, and can also be expressed as a Document Type Definition that an XMLParser can use to validate XML Documents. The vocabulary for gene expression data described in this document will be called DsLSR_GEML. The top-level element of the XML vocabulary, **project**, contains the following sub-vocabularies:

- **biosequence**--describes a nucleotide or amino acid sequence
- **pattern**--describes the physical layout of an array (DNA Microarray, Nylon Filter, Protein Chip, etc.) , and its physical characteristics.
- **printing**--describes the characteristics of a chip printing run.
- **sample**--describes the characteristics of the biological material used in a hybridization.
- **compound**--refers to a substance formed by chemical union of two or more ingredients in proportion by weight that is used in a sample preparation.
- **solvent**--refers to a substance for dissolving or dispersing one or more other

substances that is used in a sample preparation.

- **prep**--refers to a sample preparation consisting of **treatment** steps.
- **hyb**--refers to a hybridization consisting of labeled sample preparations.
- **combine**--collection of **hyb_refs** whose **hybs** represent replicates that can be statistically combined together to form a weighted-average group.
- **profile**--the results of the statistical analysis of a scan or scans of a hybridization.
- **biosequence_cluster**--defines the groupings of sequences that are believed to be derived from the same gene..
- **biosequence_set**--ordered sets of sequences that are useful to view and/or analyze as a group.
- **experiment_set**--orders sets of combined **hybs** along experimental dimensions, such as a Dosage or Time Series. **Experiment_sets** can also be used to pass hierarchical data structures such as the results of an experiment cluster analysis.

2.3 Response to RFP Requirements

2.3.1 Mandatory RFP Requirements

The Gene Expression Proposal <u>shall</u>:

- define Object models for areas of gene expression listed in Section 6.1 [of the RFP] (Problem Statement)

The domain model in Section 4 addresses the issues of array design, experimental conditions and data representation.

The means of retrieving this data is specified in the module of Section 3.

- define a set of interfaces to provide retrieval functionality for array patterns and gene expression data independent of underlying implementation or storage format

The domain model in Section 4 is independent of any one technology or type of gene expression experiment. The definition allows extensions for values not explicitly defined. The interfaces of the module defined in Section 3 are neutral to the underlying data source implementation.

- these interfaces shall specifically focus on array based hybridization events

Our data model captures a number of parameters critical to array based hybridization events such as x, y coordinates for features, the ability to track cDNA clone well position and plates described in section 4.1.5, the **pattern**

- the interfaces shall provide, at a minimum, information listed in 6.5.3 [of the RFP]:

- Array patterns, including position and content of features

These are described in section 4.1.5, the **Pattern** element

- Identities of reporters associated with a given feature, and the gene which that reporter is designed to report on
  These are described in section 4.1.4, the **biosequence_element** and 4.1.5, the **reporter** element in **pattern**
- Probe information and conditions (mobile phase)
  See sections 4.1.7 through 4.1.10 which define the elements that describe the **sample** and **treatments** that produce the **prep.**
- Hybridization conditions
  These are described in section 4.1.11, the **Hyb** element
- Expression levels for specific genes in a specific hybridization
  These are described in section 4.1.13, the **Profile** element, specifically, the **feature_data**¸ **reporter_data**, and **biosequence_data**.
- Normalization
  Normalization algorithms can be specified in the **error_model**, section 4.1.13, and **normalization**, section 4.1.12, elements. Also the **feature** and **reporter** elements can be specified as normalization controls through the **control_type** attribute in section 4.1.5, in **pattern**.
- Gene expression clusters
  These are described in section 4.1.15, **biosequence_sets**
- Array technology/design (focusing on immobile phase, not experimental protocol)
  This is described in section 4.1.5, **pattern** and section 4.1.6, **printing**
- Detection technology
  This is described in section 4.1.13, **profile's scanner** attribute.
- Replicate hybridizations
  This is described in section 4.1.12, **combine**
- Experimental paradigm (e.g., collection of hybridization events and relevant associations, such as time-series ordering)
  This is described in section 4.1.16, **experiment_set**
- Experimental annotations
  The **annotation** element, section 4.1.3, is contained in the top-level elements allowing annotation at several steps of the gene expression experiment as well as explicit attributes in **hyb**, **prep**, **treatment** and **sample**. **Bibliographic_ref** also provides document references for several elements.

- the interfaces shall also provide the ability to add links from expression data to other relevant data.

The **biosequence** element, section explicitly through the **accession** and **alias** elements provide links to third party sequence databases. The **annotation** and **bibliographic** elements, section 4.1.3, can also provide links to relevant data.

- include a glossary defining terms and concepts used in the response

See section 6 for the glossary.

2.3.2 Optional RFP Requirements

the proposal includes UML diagrams

UML diagrams are used to describe the interfaces and domain model

the proposed interface supplies additional information such as that listed in 6.6.2 [of the RFP]

- Raw experimental data

described in the **signal** and **background** elements in section 4.1.13

- Data collection and processing parameters
  described in section 4.1.11, the **hyb**, and 4.1.13, the **profile** itself, **channel_info**, and the elements in **statistics**
- Pointers to external references / annotation
  The **annotation** element, section 4.1.3, is contained in the top-level elements allowing annotation at several steps of the gene expression experiment as well as explicit attributes in **hyb**, **prep**, **treatment** and **sample**. **Bibliographic_ref** also provides document references for several elements.
- Image data and intensity information
  Through the **image_file**, **image_stats**, and **feature_data** in section 4.1.13.
- Statistical summaries of experiments
  Through the **summary_data** element in section 4.1.13

defines one or more XMI compliant Document Type Definitions (DTD's) intended for use as self-describing data structures for encapsulation of hybridization, expression, and cluster data.

The DsLSR_GEML.dtd is defined in the domain model, section 4

portions of the analysis machinery from LSR Biomolecular Sequence Analysis (lifesci/99-12-01) is used.

The **biosequence** element , through its **accession** and **alias** elements can be used to provide parameters to the interfaces described in LSR Biomolecular Sequence Analysis.

2.3.3 Issues To Be Discussed

The proposal should discuss design decisions relating to the overall performance and efficiency of the interface.

The DsLSRGeneExpQuery interface was made as lightweight and stateless as possible. Because of the size of data associated with gene expression experiments, XML was chosen to describe the data. This allows the exchange of data to be discontinuous, that is, the data source can create XML files and place them at a well known location which a client can then download locally at some later time.

As appropriate, the proposal should discuss any interface design decisions that would significantly affect a distributed system implementation.

The DsLSRGeneExpQuery interface remains stateless by providing the client with the state needed to make subsequent queries. This is accomplished by allowing the client to ask for attributes and properties and by returning XML streams that contain the necessary information for further refinement of queries. When **export_to_location** is invoked, the client is provided with an object that is returned to the data source that provides information back to the data source in order to check on the progress of the export.

Submitters should discuss applicability of their proposed approach to other expression technologies (proteomics/2D gels, SAGE).

See section 5 for the discussion.

2.3.4 *Relationship to existing or emerging OMG Specifications*

- XML Metadata Interchange
  The Domain Model DsLSR_GEML is expressed as a DTD and was imported into Rational Rose to create a UML model that was XMI-compliant.

- Biomolecular Sequence Analysis
  Attributes of the **Gene** and **Reporter** can be used in queries to data sources that support the interfaces in this specification.

- Bibliographic Query Service
  Attributes of the **bibliographic_ref** can be used in queries to data sources that support the interfaces in this specification.

- Objects by Value
  As explained above, we have chosen an XML representation of the data.

- Naming and Trader Services
  Specifying how gene experiment data sources or repositories are found is considered out of scope for this RFP.

- Event Service
  Potentially, the progress of an export could be monitored through an Event Service. We have decided not to use this service in favor of a polling method. This decision was made due to the potential length (hours) that an export might take when the data was not stored in the form of XML or was archived.

- Life Cycle Service
  Life cycle behavior was considered important for the **Query** interface. It was also designed, though, so that a new instance of that interface could be used to continue a subsequent query.

- Relationship Service
  Because the interfaces were designed to be simple and stateless, the Relationship Service was not used.

- Property Service
  The Property Service is used to provide additional information about a gene expression data source through the **GeneExpUtilities** interface.

- Security Service
  The Security Service was not considered. We judged it to be outside the scope of this RFP. Data sources for gene expression data may consider implementing verification and encryption services in addition to the interfaces below.

- Collection Service
  The Collection Service was not used. The W3C DOM and XML-DEV SAX parsers provide similar capabilities for XML data as the Collection Service for IDL data. The DOM parser is ideal for smaller XML data and provides full navigation backwards and forwards. The SAX parser provides a forward only traversal of the data, which is ideal for parsing large XML documents.

- Notification Service
  Because there was felt no need for the Event Service, the Notification Service was also not used.

2.4 Compliance Points

The **GeneExpQuery** interface derives from **CosQuery::QueryEvaluator**. A full implementation of the **evaluate** is not required.

3 Modules and Interfaces

3.1 Module DsLSRGeneExpQuery

Exchanging and storing gene expression experiment data has evolved into many different formats and schemas. With the explosion of companies and technologies within the field of gene expression there is a desire to be able to standardize on a data exchange format that is independent of any proprietary format and underlying database schema. This standardization will also allow organizations to be able to act as repositories of experiment sets which can be queried for by third parties and returned in this format. In addition, this will allow researchers in the field to readily exchange gene expression data and annotations and analyze them using a variety of tools.

The **find** method allows a client to interrogate the repository to discover what experiment sets are available based on attribute values of an experiment set and its referenced elements. The derivation for **CosQuery::QueryEvaluator** allows more sophisticated queries, although providers are not required to implement the **evaluate** method as more than throwing a **CosQuery::QueryInvalid** error.

One problem an implementation of these interfaces must face is the amount of data that is potentially available for a single experiment set. This can easily be in the range of gigabytes (even compressed) so a mechanism that is stateless must be provided to allow the data to be placed in an well-known location and in discrete files and then imported through ftp or a similar mechanism to the client. Depending on the implementation, the export and import of files can be minutes, if the data is already in XML, or take hours, if the data source must convert from another format and if the time for the client to import the data into their database system is considered. The **export** method allows small amounts of data to be directly streamed to a client. The **export_to_location** method allows a stateless exchange of the data with **export_completed** allowing the client to check on the status of the export.

Information about the provider is available through **properties** in the **GeneExpUtilities** interface. The **properties** provide information on what sub-vocabularies of the DTD that **find** will return, the maximum of data that can be streamed directly, locations that data can be exported to, descriptions, in DTD format, of what the source provides as elements contained by DTD-ANY elements, and the sequence database per species that provides the unique identifiers for **biosequences**. The **queryProperties** attribute allows the provider to specify provider-specific properties.

3.1.1 General

```
//File: DsLSRGeneExpQuery.idl
//

#ifndef _DS_LSR_GENE_EXP_QUERY_IDL_
#define _DS_LSR_GENE_EXP_QUERY_IDL_

#pragma prefix "omg.org"

#include <CosQuery.idl>
#include <CosLifeCycle.idl>
#include <CosPropertyService.idl>
```

```
#include <DsLSRControlledVocabularies.idl>

module DsLSRGeneExpQuery
{
   //
}
#endif // _DS_LSR_GENE_EXP_QUERY_IDL_
```

#pragma prefix "omg.org"

To prevent name space pollution and name clashing of IDL types, this module uses the pragma prefix that is the OMG's DNS name.

#include <CosQuery.idl>

GeneExpQuery inherits from **QueryEvaluator**.

#include <CosLifeCycle.idl>

GeneExpQuery also inherits from **LifeCycleObject**.

#include <CosPropertyService.idl>

GeneExpUtilities uses **Property** and **Properties**.

#include <DsLSRControlledVocabularies.idl>

GeneExpQuery uses **VocabularyFinder**.

3.1.2 Typedefs and Structures

The following typedefs and structures clarify, by name, the meaning of different parameters, return types, and attributes.

```
   typedef string XMLString;
   typedef string URL;
   typedef sequence<URL> URLList;

   typedef string ExtendedElement;
   typedef string DTDString;
   struct ExtensionDTD
   {
      ExtendedElement extElement;
      DTDString       extDTD;
   };
   typedef sequence<ExtensionDTD> ExtensionDTDList;

   typedef string SpeciesString;
   typedef string SequenceSource;
   struct SpeciesSource
   {
      SpeciesString  species;
      SequenceSource seqSource;
   };
   typedef sequence<SpeciesSource> SpeciesSourceList;

   typedef string ElementName;
   typedef sequence<ElementName> ElementNameList;

   typedef string AttributePath;
   typedef string AttributeValue;
```

```
    struct GE_NVPair
    {
        AttributePath  attrPath;
        AttributeValue attrValue;
    };
    typedef sequence<GE_NVPair> GE_Predicates;

    // shorthands for imported types for controlled vocabularies
     typedef DsLSRControlledVocabularies::VocabularyList       VocabularyList;
     typedef DsLSRControlledVocabularies::VocabularyString     VocabularyString;
     typedef DsLSRControlledVocabularies::VocabularyStringList VocabularyStringList;
```

XMLString

Description

Used as the return value of an XML formatted stream.

URL

Description

Used to pass a Universal Resource Locator.

URLList

Description

Used to define a list of URLs. Type of an **GeneExpUtilities** attribute.

ExtendedElement

Description

Used to specify one of the five DTD elements that are of type ANY in the **ExtensionDTD** struct.

DTDString

Description

Used to specify a DTD formatted stream in the **ExtensionDTD** struct.

ExtensionDTD

Description

A structure that associates an **ExtendedElement** with a **DTDString**. Type of an **GeneExpUtilities** attribute.

ExtensionDTDList

Description

A list of **ExtensionDTDs**, type of an **GeneExpUtilities** attribute.

SpeciesString

Description

Used in the **SpeciesSource** struct and represents the name of a species. "Default" can be specified to represent all species that are not otherwise listed in the **SpeciesSource** struct.

SequenceSource

Description

Name of a sequence source database for **biosequence** element.

SpeciesSource

Description

A struct that associates a species with a sequence source database, type of an **GeneExpUtilities** attribute.

ElementName

Description

A string that represents a sub-vocabulary element.

ElementNameList

Description

A list of ElementNames, type of an **GeneExpUtilities** attribute.

AttributePath

Description

A path, recommended to follow the **CosNaming** convention (see section 2.1.3), specifying a predicate attribute name. Part of the **GE_NVPair** struct.

AttributeValue

Description

The value to associate with the **AttributePath**. Part of the **GE_NVPair** struct.

GE_NVPair

Description

A structure that associates a DTD attribute with a value. Specified as a predicate for the **find**, **export** and **export_to_location** in the **GE_Predicates** list.

GE_Predicates

Description

List of **GE_NVPair**, used in **find**, **export** and **export_to_location**.

Vocabulary, VocabularyString, and VocabularyStringList

Description

Convenience typedefs for the strings a client is likely to use in conjunction with the **VocabularyFinder**.

3.1.3 Exceptions

Exceptions defined in the **DsLSRGeneExpQuery** module.

```
    exception LimitExceeded { long maxLength; };
    exception InvalidLocation {};
```

LimitExceeded

Description

This exception is raised when the length of the returned XMLString from **find** or **export** exceeds the value of **GeneExpUtilities::xmlStringLimit**.

Return Value

a long whose value equals **GeneExpUtilities::xmlStringLimit**.

InvalidLocation

Description

This exception is raised when the location used in **export_to_location** is not a valid location.

3.1.4 GeneExpQuery

The **GeneExpQuery** interface allows the gene expression data source to be interrogated for the top level sub-vocabularies that are listed by **GeneExpUtilities::permittedFindElements**. The size of most of the elements and their contained elements, except for **pattern** and **profile**, should allow them to be returned as an **XMLString** immediately. To facilitate even smaller downloads, the **detail** parameter can be passed in to **find** as false when multiple elements will be returned. This stops the nesting at the first level of contained elements, rather than returning the full XML for a sub-vocabulary element. The clients can then use a DOM parser to examine the attributes, particularly **annotation** elements and **description** attributes, to further refine their search through specifying additional predicates.

The **GeneExpQuery** is stateless, that is, the instances of the interfaces can be freed and a new instance used for a subsequent query, request for export or check on the status of an export.

In order to handle large experiment sets, the **export_to_location** method is provided. The client provides one of the locations provided in the **GeneExpUtilities::locations** list then periodical checks **export_complete** to know when the export is complete. By use of the return value of **export_to_location** in the call to **export_complete**, the data source can track which **export** is referenced. The data source can, optionally, return a **progress** indication in the out parameter.

Many of the attributes will have controlled vocabularies. The set of controlled vocabularies is likely to vary from data source to data source, and values for the same attribute will vary also. The **voc_finder** attribute can be used to find out the controlled vocabularies for the data source the client is using.

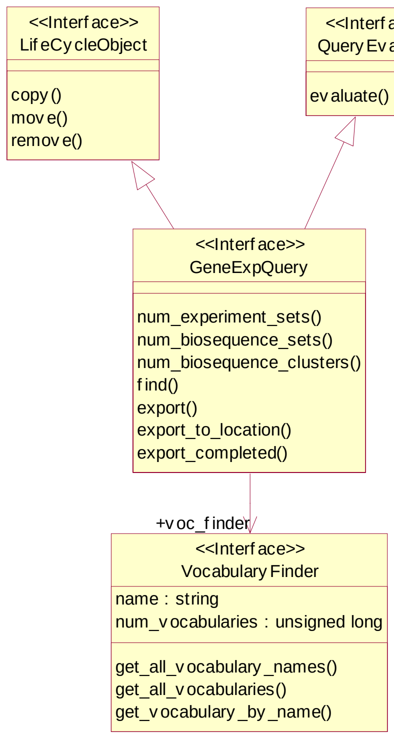

Figure 1: GeneExpQuery interface.

```
    interface GeneExpQuery : CosQuery::QueryEvaluator, CosLifeCycle::LifeCycleObject
    {
        unsigned long num_experiment_sets();
        unsigned long num_biosequence_sets();
        unsigned long num_biosequence_clusters();

        readonly attribute DsLSRControlledVocabularies::VocabularyFinder voc_finder;

        XMLString find( in ElementName element, in GE_Predicates include, in GE_Predicates
exclude, in boolean detail)
                  raises(CosQuery::QueryInvalid,LimitExceeded);

        XMLString export( in ElementName element, in ElementNameList includeElements)
                  raises(CosQuery::QueryInvalid,LimitExceeded);

        any export_to_location( in ElementName element, in ElementNameList includeElements, in
URL location, in boolean separateFiles)
                   raises(CosQuery::QueryInvalid,InvalidLocation);

       CosQuery::QueryStatus export_completed(in any cookie, out float progress)
                   raises(CosQuery::QueryProcessingError);
    };
```

GeneExpQuery

The **GeneExpQuery** inherits from **CosQuery::QueryEvaluator** in order to provide advanced query ability. A data source is not required to do more than provide a vacuous implementation of **evaluate** but implementations may provide additional query capabilities. The **GeneExpQuery** also inherits from **CosLifeCycle::LifeCycleObject** in order to control management of its lifetime.

```
unsigned long num_experiment_sets();
```

Description

Allows the client to obtain the total number of experiment sets in the data source.

Return value

Returns an unsigned long that is the count of experiment sets.

```
unsigned long num_biosequence_sets();
```

Description

Allows the client to obtain the total number of biosequence sets in the data source.

Return value

Returns an unsigned long that is the count of biosequence sets.

```
unsigned long num_biosequence_clusters();
```

Description

Allows the client to obtain the total number of biosequence clusters in the data source.

Return value

Returns an unsigned long that is the count of biosequence clusters.

```
readonly attribute DsLSRControlledVocabularies::VocabularyFinder voc_finder;
```

Description

Allows the client to obtain the controlled vocabularies associated with the data source. The controlled vocabulary names will reference attributes as an **AttributePath**, in all other respects it will follow the conventions of the **DsLSRControlledVocabularies** module.

```
XMLString find( in ElementName element, in GE_Predicates include, in GE_Predicates exclude,
                in boolean detail)
```

```
        raises(CosQuery::QueryInvalid,LimitExceeded);
```

Description

The **find** method is used by the client to interrogate the data source for information in the repository. The **element** specifies the sub-vocabulary that is being queried. The **include** and **exclude** inputs provide predicates to refine the query. The **include** inputs specify any criteria that must be met, the **exclude,** any criteria that must not be met. These predicates can be in other sub-vocabularies that are referenced by elements in **element**.

The **detail** input specifies whether the entire information for a match should be provided for any element that matches the criteria, when true, or whether only the element itself, and its immediately contained elements be returned, when false, in the XMLString. For instance, when querying for **experiment_sets**, if **detail** is true, then all **experiment_set_members** would be returned. If **detail** is false, only the top **experiment_set_members** are returned with other immediately contained elements. If **detail** is false, the XMLString will be well-formed but may not be valid if nested elements that aren't include are marked as mandatory in the DTD.

Return value

Returns an XMLString that will have one sub-vocabulary **element** entry for each match.

Exceptions

Raises **CosQuery::QueryInvalid** if the **element** is not on the **permittedFindElements** list or if one of the **inputs** or **outputs** is ill-formed.

Raises **LimitExceeded** if the length of the returned XMLString is greater than **GeneExpUtilities::xmlStringLimit**.

```
XMLString export( in ElementName element, in ElementNameList includeElements,
                  in GE_Predicates include, in GE_Predicates exclude)
        raises(CosQuery::QueryInvalid,LimitExceeded);
```

Description

The **export** method will return all the data for the sub-vocabulary of **element** that matches the **include** and **exclude** predicates. In addition, any of the sub-vocabulary elements specified by **includeElements** will also be include if they are referenced directly or indirectly by a match in an **element**.

The predicates can be as simple as a specification of the alternate keys for the desired **elements**.

Return value

Returns an XMLString that includes the data for the valid matches.

Exceptions

Raises **CosQuery::QueryInvalid** if the **element** is not on the **permittedFindElements** list or if one of the **inputs** or **outputs** is ill-formed. It is also thrown if an element specified in **includeElements** is not the name of a sub-vocabulary.

Raises **LimitExceeded** if the length of the returned XMLString is greater than **GeneExpUtilities::xmlStringLimit**.

```
any export_to_location( in ElementName element, in ElementNameList includeElements,
                        in GE_Predicates include, in GE_Predicates exclude,
                         in URL location, in boolean separateFiles)
       raises(CosQuery::QueryInvalid,InvalidLocation);
```

Description

The **export_to_location** allows an asynchronous export and import of the data to a well-known **location**. The **location** must be on the **GeneExpUtilities::locations** list. The **separateFiles** indicates whether each of the sub-vocabulary is exported as a separate file. Further, the data source may decide to place each match in a sub-vocabulary in its own file. The client should be prepared for these files to be compressed in some well-known format, such as zip or gz.

The other parameters have the same meanings as in the previous methods.

Return value

Returns an **any** object which can be used in calls to **export_completed**. The object must be persistable or be a well-known type so that the client can easily save its value between sessions with the data source

Exceptions

Raises **CosQuery::QueryInvalid** if the **element** is not on the **permittedFindElements** list or if one of the **inputs** or **outputs** is ill-formed. It is also thrown if an element specified in **includeElements** is not the name of a sub-vocabulary.

Raises **InvalidLocation** if the **location** is not on the **GeneExpUtilities::locations** list.

```
CosQuery::QueryStatus export_completed(in any cookie, out float progress)
       raises(CosQuery::QueryProcessingError);
```

Description

The **export_completed** method is used to query the data source on the status of an export that returned the **cookie** input. The out parameter **progress** will have a value between zero and a one hundred, indicating the progress, or minus one, if the data source does not support or cannot determine the progress.

Return value

Returns a **CosQuery::QueryStatus**, indicating the status.

Exceptions

Raises **CosQuery::QueryProcessingError** if an error occurred exporting the data.

3.1.5 GeneExpUtilities

The **GeneExpUtilities** interface contains the constants, attributes and properties that describe the data source. These allow a client to discover legal values for the inputs to the **GeneExpQuery** methods. It also allows for specifying what elements might be contained in those elements of type ANY in the **DsLSR_GEML.dtd**.

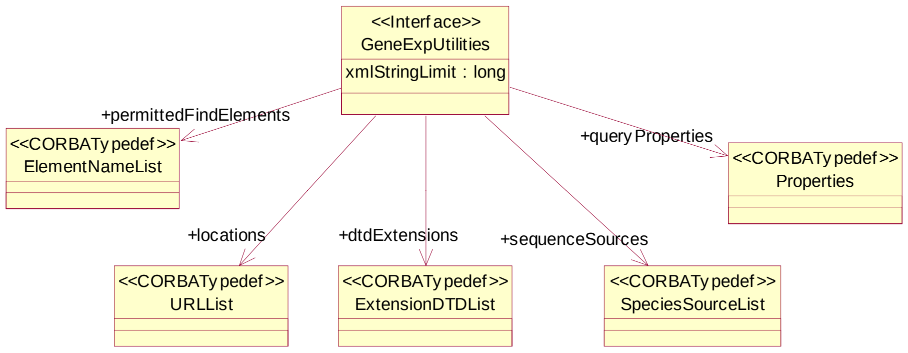

Figure 2: GeneExpUtilities interface.

```
interface GeneExpUtilities
{
    // constants for the top DTD sub-vocabularies
    const ElementName BIOSEQUENCE    = "biosequence";
    const ElementName SAMPLE         = "sample";
    const ElementName PATTERN        = "pattern";
    const ElementName PRINTING       = "printing";
    const ElementName COMPOUND       = "compound";
    const ElementName SOLVENT        = "solvent";
    const ElementName PREP           = "prep";
    const ElementName HYB            = "hyb";
    const ElementName COMBINE        = "combine";
    const ElementName PROFILE        = "profile";
    const ElementName EXPERIMENT_SET = "experiment_set";
    const ElementName BIOSEQUENCE_CLUSTER = "biosequence_cluster";

    // constants for DTD-ANY type elements that allow extensible elements
    const ExtendedElement SPECIES_DATA  = "species_data";
    const ExtendedElement REPORTER_DESC = "reporter_desc";
    const ExtendedElement CHANNEL_DATA  = "channel_data";
    const ExtendedElement SCANNER_STATS = "scanner_stats";
    const ExtendedElement IMAGE_STATS   = "image_stats";
    const ExtendedElement HYB_STATS     = "hyb_stats";
    const ExtendedElement COUNTS        = "counts";

    readonly attribute ElementNameList   permittedFindElements;
    readonly attribute long              xmlStringLimit;
    readonly attribute URLList           locations;
    readonly attribute ExtensionDTDList  dtdExtensions;
    readonly attribute SpeciesSourceList sequenceSources;

    readonly attribute CosPropertyService::Properties queryProperties;
};
```

Constants

Description

Two sets of constants are defined. The **ElementName** constants define names for each of the sub-vocabularies in **DsLSR_GEML.dtd**. The **ExtendedElement** constants are the element names that have been specified as DTD type ANY and which can contain arbitrary elements in the XML which will not be validated against the DTD.

Attributes

```
readonly attribute ElementNameList permittedFindElements;
```

Description

A list element names that can be used as input to the **GeneExpQuery::find** method.

Return value

Returns the list of element names.

```
readonly attribute long xmlStringLimit;
```

Description

The limit in length of the XMLString returned from the **GeneExpQuery::find** and **GeneExpQuery::find** methods.

Return value

Returns a long that is the maximum length allowed for a returned XMLString.

```
readonly attribute URLList locations;
```

Description

A list of locations that the data source can export XML to from the **GeneExpQuery::export_to_location** method.

Return value

Returns a string list of valid URLs.

```
readonly attribute ExtensionDTDList dtdExtensions;
```

Description

A list of elements of type ANY for which the data source has additional information on what the contained elements will be, paired with a DTD description of those elements and their hierarchy.

Return value

Returns the list of **ExtensionDTDs** that pairs the element name with a DTD.

```
readonly attribute SpeciesSourceList sequenceSources;
```

Description

A list of pairs of values, the first being a string which specifies a species, the second a string that specifies the biological sequence source for **biosequences**.

Return value

Returns a list of **SpeciesSource**.

```
readonly attribute CosPropertyService::Properties queryProperties;
```

Description

Properties specific to the data source.

Return value

Returns a list of CosPropertyService::Property.

3.1.6 Sequence Diagram

The following interaction diagram illustrates a possible communication between a client and the **DsLSRGeneExpQuery** module. The breaks in the clients interaction with **GeneExpQuery** indicate different instances of that interface.

Figure 3: Stateless invocation of GeneExpQuery interface

4 Domain Model

4.1 Domain Model DsLSR_GEML

The DsLSR_GEML domain model defines the elements for supporting gene expression data.

Because the exchange of gene expression data can be abstracted from the source from which it was obtained, it can be represented by XML files, which are both human readable and machine readable. This facilitates an independence between the export and the import of the gene expression data as illustrated below.

An organization can act as a repository of gene experiment data and allow access through an implementation of the **DsLSRGeneExpQuery** interfaces to the data as XML streams. The implementation of these interfaces allow sophisticated interactive queries that can first produce a list of experiments sets with brief descriptions then further refine the selection through subsequent queries to only obtain the data of interest.

It is also possible for an organization to not implement the DsLSRGeneExpQuery interfaces and use a custom application to export their gene expression data to an XML file in order to collaborate with another organization.

The import of these streams or files is then independent of the manner of export. In addition, the parsing of the document allows access to the elements and attributes in such a way as to facilitate additional annotation. For instance, as a **biosequence** or **biosequence_ref** element is parsed, the information from the **accession** element can be used to obtain additional annotation from annotation sources such as GenBank, Saccharomyces Genome Database™, etc.

The DTD file can be found in Section 9.1.

4.1.1 General

The vocabulary of DsLSR_GEML is organized into thirteen sub-vocabularies in such a way that the sub-vocabularies are independent of each other. That is, a valid XML document can contain the data from an individual sub-vocabulary, such as **sample** or **pattern**, or it can contain any combination of these sub-vocabularies, such as all the **hyb** and **profile** data for an experiment set. Implementations may impose additional ordering, such as **patterns** before their **profiles,** or they may require that they be exported to separate files.

Throughout the document will be attributes documented as identifiers within a scope. 'Within the source' means that the identifier is unambiguous within the scope of the originating organization, 'within the document' means it is unambiguous within the XML. Other scopes will be within a particular element, for instance, **feature.number** in **pattern**, which, combined with the pattern's unique scope within the source, makes it possible to identify the feature within the source as well.

The independence of sub-vocabularies is possible through the use of reference elements to link data from two sub-vocabularies within a document where the reference elements attributes are identifier attributes of the referenced element. That is, a **biosequence_data** element can link the data for a **biosequence** by using the attributes of its **biosequence_ref** to link to the **biosequence** element. Further, an import implementation is free to use the attributes of a **ref** element as an alternate key in its database and, for instance, either find that the biological sequence already exists in its database, or, optionally create a place holder for it. An implementation is also free to consider it an error if a referenced element doesn't exist in the document or doesn't exist in its database.

Export implementations, including those that implement the DsLSRGeneExpQuery interfaces, can decide to what extent that they export references without providing the referenced element. Export implementations, including those that implement the DsLSRGeneExpQuery interfaces, can choose to allow exports which contain unresolved reference elements, to allow a subset of reference elements to be unresolved, or to not allow any unresolved references in a document.

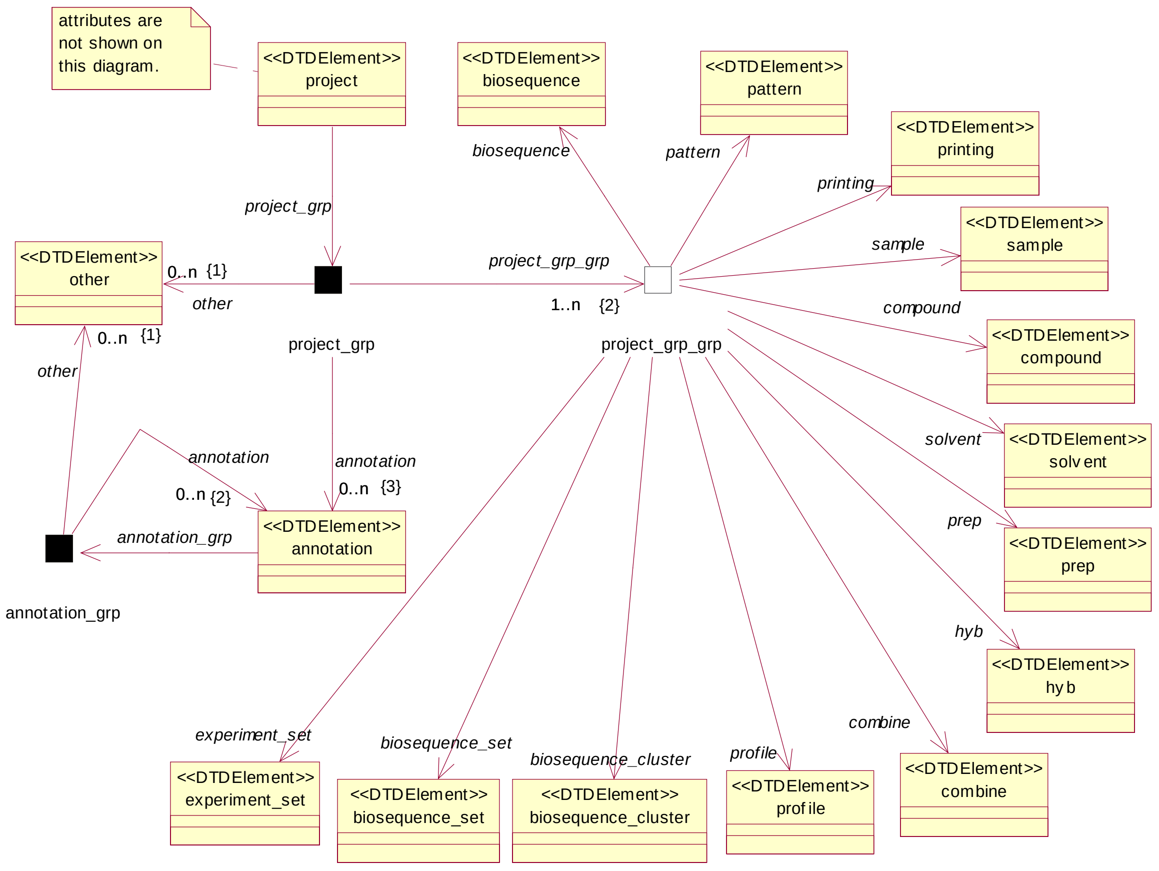

Figure 4: project and its contained elements

4.1.2 *Project*

The **Project** element groups together gene expression data through its child elements. Besides the sub-vocabulary elements, a project can also have multiple **annotation** elements and multiple **other** elements.

Patterns represent the physical layout of a gene expression experiment. The **pattern** contain **reporters** which reference **biosequences.** Each reporter primarily picks the expressed product of the **biosequence**. The **feature** elements represent the physical instances or locations of each reporter. The actual production and associated tracking information of a physical array with a layout corresponding to a given pattern is described by the **printing** element.

The **prep** element describes the biological cellular extract, a nucleotide or protein preparation, used in one or more hybridizations. This includes information on the cell lines and/or tissues (described by **sample**) and the treatments used (each treatment is described by the **compound** and **solvent** elements).

The hybridization of an array is described by the **hyb** element which contains one or more **labeled_preps**, the combination of a preparation and a label.

The **profile** element contains quantitated signals at the **feature_data**, **reporter_data**, and/or the **biosequence_data** analysis level from, in general, a scan per **labeled_prep.** It accommodates both an unlimited number of absolute, intensity-based results and one (optional) ratio-based result. Attributes exist for recording error bars and pvalues. The **profile** also provides tracking information, such as the scanner used, the person who performed the scan, various quality control information, etc.

The **combine** element contains one or more hybridizations that will be analyzed as a group. This includes aggregation to work around the limitation of the number of features per **profile** and data repeated to achieve higher statistical significance.

An **experiment_set** is a collection of experiments that is useful to view and/or analyze as a group. An **experiment_set** could, for example represent a group of experiments that represents a drug titration series.

A **biosequence_set** is a collection of sequences, potentially hierarchical, that is useful to view and/or analyze as a group. If the grouping is based on an **experiment_set**, an optional **experiment_set_ref** allows the values to be backtracked through their experimental history.

The **biosequence_cluster** allows biosequences (by **biosequence_ref**) believed to derive from the same gene, to be grouped together.

```
<!ELEMENT project
   (other*,
    (biosequence |
     pattern     |
     printing    |
     sample      |
     compound    |
     solvent     |
     prep        |
     hyb         |
     combine     |
     profile     |
     biosequence_cluster |
     experiment_set)+,
    annotation*)
>
<!ATTLIST project
   name         CDATA  #IMPLIED
   id           CDATA  #IMPLIED
   date         CDATA  #IMPLIED
   by           CDATA  #IMPLIED
```

```
    organization CDATA  #IMPLIED
>
```

Attribute	Description
name	project name
id	unambiguous identifier, within the source, for the project
date	date the xml file is generated
by	contact for project information
organization	origin of the data contained in the XML

4.1.3 Miscellaneous Elements

The following elements are used in many of the sub-vocabularies and are documented here. The **other** element is contained in every element (including itself) to allow extensibility. To keep the UML diagrams readable, that association is generally shown only for the top level element of the diagram.

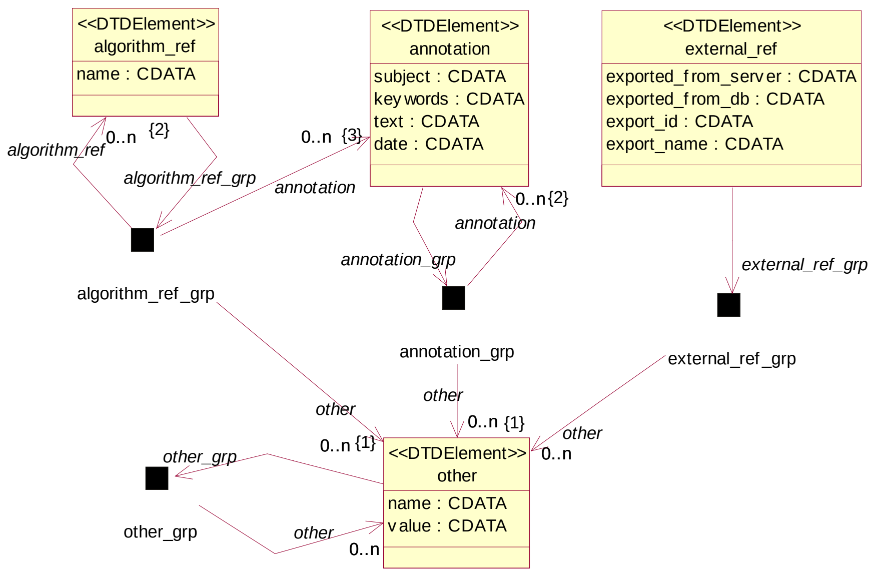

Figure 5: miscellaneous elements.

other

The other element allows for undocumented or proprietary attributes to be included as name/value pairs. This allows additional extensibility and flexibility to accommodate an organization's special needs.

```
<!ELEMENT other
   (other*)
>
<!ATTLIST other
   name CDATA #REQUIRED
   value CDATA #REQUIRED
>
```

Attribute	Description
name	name of the undocumented or proprietary attribute
value	value

annotation

An annotation or freehand description. Although all the elements are **#IMPLIED**, in practice, one or more would certainly be used.

```
<!ELEMENT annotation
   (other*,
    annotation*)
>
<!ATTLIST annotation
   subject  CDATA  #IMPLIED
   keywords CDATA  #IMPLIED
   text     CDATA  #IMPLIED
   date     CDATA  #IMPLIED
>
```

Attribute	Description
subject	brief description of the annotation
keywords	a list of one or more keywords for this annotation to facilitate searching. Keywords are separated by space. Multiple words that constitute a keyword will use double quotes to group them.
text	the annotation or description, itself
date	date annotation was created

annotation example

An example annotation for a biosequence with accession number of U49845 might look like the following (taken from NCBI GenBank example at http://www.ncbi.nlm.nih.gov/Sitemap/samplerecord.html#Top):

```
<annotation subject = 'GenBank Annotation' keywords = 'U49845 GenBank biosequence'>
   <annotation keyword = 'LOCUS' text = 'SCU49845 5028 bp DNA PLN 21-JUN-1999'/>
   <annotation keyword = 'DEFINITION' text = 'Saccharomyces cerevisiae TCP1-beta gene,
partial cds, and Axl2p (AXL2) and Rev7p (REV7) genes, complete cds.'/>
                                            .   .   .
   <annotation keyword = 'FEATURES' text = 'Location/Qualifiers'>
       <annotation keyword = 'source' text = '1..5028'>
           <annotation keyword = '/organism' text = '"Saccharomyces cerevisiae"'/>
           <annotation keyword = '/db_xref' text = '"taxon:4932"'/>
                                            .   .   .
```

```
        </annotation >
                                                  .   .   .
    </annotation >
</annotation>
```

external_ref

Description of the original source for the containing element, if the data was imported from another source. This allows for the tracking of the originating source for the data. Potentially a **pattern** for a **hyb**, for instance, could come from a different organization than that which performed the hybridization.

```
    <!ELEMENT external_ref
        (other*)
    >
    <!ATTLIST external_ref
        exported_from_server CDATA #IMPLIED
        exported_from_db     CDATA #IMPLIED
        export_id            CDATA #IMPLIED
        export_name          CDATA #IMPLIED
    >
```

Attribute	**Description**
exported_from_server	name of server (ip address, network name, etc.) that identifies the server. In most cases this should be fully qualified.
exported_from_db	name of source (database, file, etc.) of the data
export_id	unambiguous identifier, within the export source, for the data
export_name	unambiguous name, within the export source, for the data

bibliographic_ref

Allows elements, such as **experiment_set**, **biosequence**, etc., that contain **bibliographic_ref** to reference literature relevant to the data. For instance, if an article, based on a set of experiments, appears in a journal, information to obtain the article can be included in the XML data.

The elements and attributes are based on a subset of the LSR Bibliographic Query Service (lifesci/2000-10-01) (BQS) IDL specification with two differences. One, many attributes are removed which are not as relevant for straight lookup from a bibliographic source. The url attribute has been moved from web_resource to bibliographic_ref. These two changes have simplified what needs to be carried over from the BQS IDL without losing the ability to find the reference.

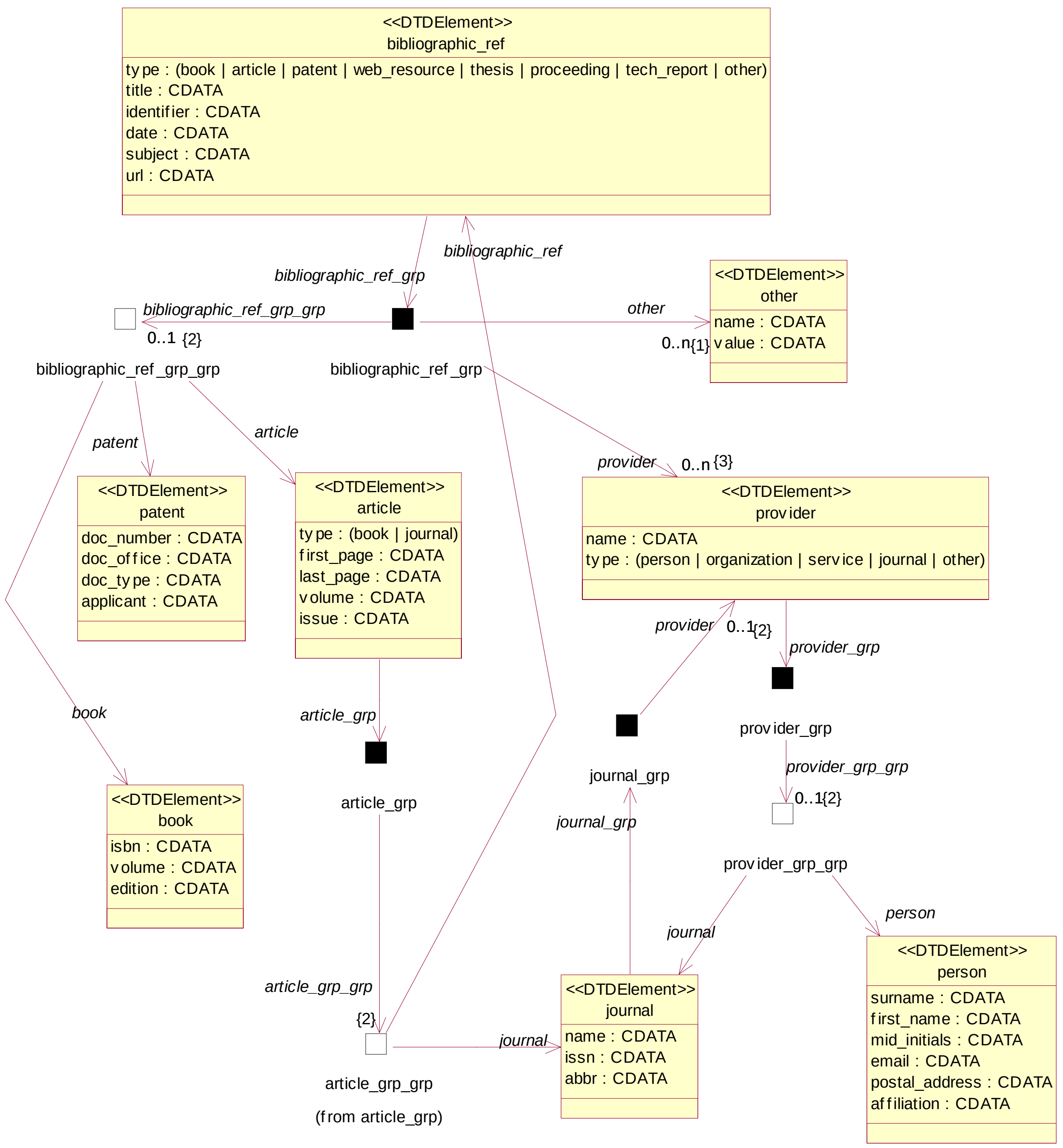

Figure 6: bibliographic_ref and contained elements.

```
<!ELEMENT bibliographic_ref
    (other*,
     (book    |
      article |
      patent)?,
     provider*)
>
<!ATTLIST bibliographic_ref
    type (book | article | patent | web_resource | thesis |
         proceeding | tech_report | other) #REQUIRED
    title      CDATA #IMPLIED
    identifier CDATA #IMPLIED
    date       CDATA #IMPLIED
    subject    CDATA #IMPLIED
    url        CDATA #IMPLIED
>
<!ELEMENT person
    (other*)
>
<!ATTLIST person
    surname        CDATA #IMPLIED
    first_name     CDATA #IMPLIED
    mid_initials   CDATA #IMPLIED
    email          CDATA #IMPLIED
    postal_address CDATA #IMPLIED
    affiliation    CDATA #IMPLIED
>

<!ELEMENT journal
    (other*,
     provider?)
>
<!ATTLIST journal
    name CDATA #IMPLIED
    issn CDATA #IMPLIED
    abbr CDATA #IMPLIED
>

<!ELEMENT provider
    (other*,
     (person |
      journal)?)
>
<!ATTLIST provider
    name CDATA #IMPLIED
    type (person | organization | service | journal | other) #REQUIRED
>

<!ELEMENT book (other*)
>
<!ATTLIST book
    isbn    CDATA #IMPLIED
    volume  CDATA #IMPLIED
    edition CDATA #IMPLIED
>

<!ELEMENT article
   (other*,
    (bibliographic_ref |
     journal))
>
<!ATTLIST article
   type (book | journal) #REQUIRED
    first_page CDATA #IMPLIED
    last_page  CDATA #IMPLIED
    volume     CDATA #IMPLIED
    issue      CDATA #IMPLIED
>

<!ELEMENT patent (other*)
>
<!ATTLIST patent
    doc_number CDATA #IMPLIED
    doc_office CDATA #IMPLIED
    doc_type   CDATA #IMPLIED
    applicant  CDATA #IMPLIED
>
```

4.1.4 Biosequence

The Biomolecular Sequence Analysis (lifesci/99-10-01) uses the following definition.

> "A **BioSequence** is an abstraction of a biological sequence, such as the ordered nucleotides of a DNA chain or the ordered amino acid residues of a protein molecule."

We use that definition here and note that the physical representation of the **biosequence** on the chip is described by the **reporter** element.

The attributes of **biosequence** form the basic information on the underlying biosequence. The **accession** element provides both a primary identifier, based on the match between **biosequence.sequenceDB** and **accession.database** and also alternative identifiers in other sequence databases to use to look up annotations. The **alias** elements can provide alternate biosequence names.

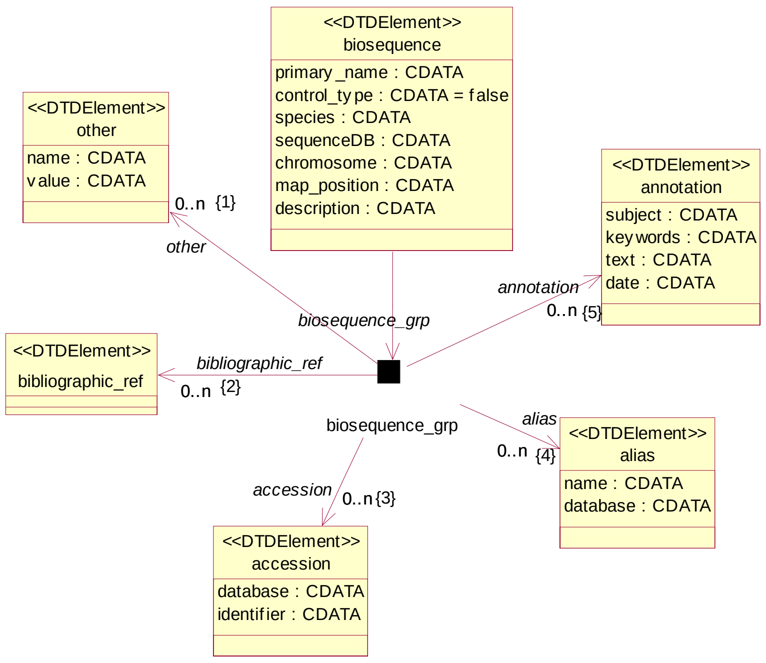

Figure 7: biosequence and contained elements

```
<!ELEMENT biosequence
   (other*,
    bibliographic_ref*,
    accession*,
    alias*,
    annotation*)
>
<!ATTLIST biosequence
   primary_name     CDATA #REQUIRED
   control_type     CDATA 'false'
   species          CDATA #REQUIRED
   sequenceDB       CDATA #REQUIRED
   chromosome       CDATA #IMPLIED
   map_position     CDATA #IMPLIED
   description      CDATA #IMPLIED
>

<!ELEMENT accession
   (other*)
>
<!ATTLIST accession
   database   CDATA #REQUIRED
   identifier CDATA #REQUIRED
>

<!ELEMENT alias   (other*)>
<!ATTLIST alias
   name CDATA #REQUIRED
>
```

Attribute	Description
primary_name	Most commonly accepted name, within the source, for this biosequence. This may vary from organization to organization for the same **biosequence** depending on their default annotation source
control_type	List of one or more names from a control name list, such as DELETION, NEGATIVE, etc.
species	Species which the **biosequence** is associated with, based on NCBI's Taxonomy
sequenceDB	The sequence database used by the source to identify the **biosequence**.
chromosome	The chromosome the **biosequence** appears on
map_position	Indication of where on the chromosome the **biosequence** appears
description	Freehand description of the **biosequence**

accession

Identifier in a particular annotation source. This allows for several annotation sources to be referenced. Generally, within an organization, each species will have an associated database as a primary identifier, for instance the SGD™ database for yeast or GenBank for human. The primary database can either be obtained from the attribute **GeneExpUtilities::sequenceSources** or from the containing **biosequence.sequenceDB**. If there are multiple **accession** elements that specify that **database**, then the first **accession** element should be considered the primary identifier.

Attribute	Description
database	Source (such as GenBank or Swiss-Prot) of the id
identifier	An unambiguous identifier, within the database, of the **biosequence** associated with the biosequence element

alias

An alternative name for the biosequence.

Attribute	Description
database	A source (such as GenBank or Swiss-Prot) of the alternative name
name	The name obtained from the database

biosequence_ref

A reference element to the biosequence. Used by **reporter** and **biosequence_data** to identify the biosequence associated with those elements. The three attributes identify, within the source, the particular **biosequence** and its context.

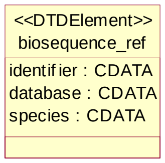

Figure 8: biosequence_ref.

```
<!ELEMENT biosequence_ref
   (other*)
>
<!ATTLIST biosequence_ref
   identifier   CDATA #REQUIRED
   database     CDATA #IMPLIED
   species      CDATA #REQUIRED
>
```

Attribute	Description
identifier	the primary biosequence identifier, either a systematic name for fully mapped species or an accession code within the database for species not yet mapped, that maps, within the source, to a biosequence
database	the database where the identifier is unique. If not provided, the default for the source of the data is assumed
species	species which the **biosequence** is associated with, based on NCBI's Taxonomy

4.1.5 Pattern

Describes an array pattern that can be printed and then hybridized. A **pattern** consists of several **features** (also called spots) in which **reporter** sequences are placed. The nature of the **biosequence** placed on a spot will depend on the technology. Two well-known technologies differ significantly—spotter arrays draw material from a well and place a spot on the array whereas oligo arrays are created through the synthesis of many, short (~20-100mer) nucleotide sequences onto the **features**.

Each **reporter** represents the biosequence it references. Generally, the physical reporter sequence is complementary to some region of the physical sequence of the biosequence. Moreover, the physical reporter sequence is typically chosen to simultaneously minimize binding with other biosequences.

For scientific investigation, quality control, and/or background determination, a sequence may be chosen which is not strictly complimentary, but, rather, deviates at a certain number of positions. For example, Affymetrix employs **features** that are mismatch controls for their **reporter**—negative controls containing single-base mismatches relative to their target sequence. Agilent Technologies employs **reporters** as deletion controls which lack one or more nucleotides relative to their target sequence.

One or more **features** are associated with a reporter in the pattern.

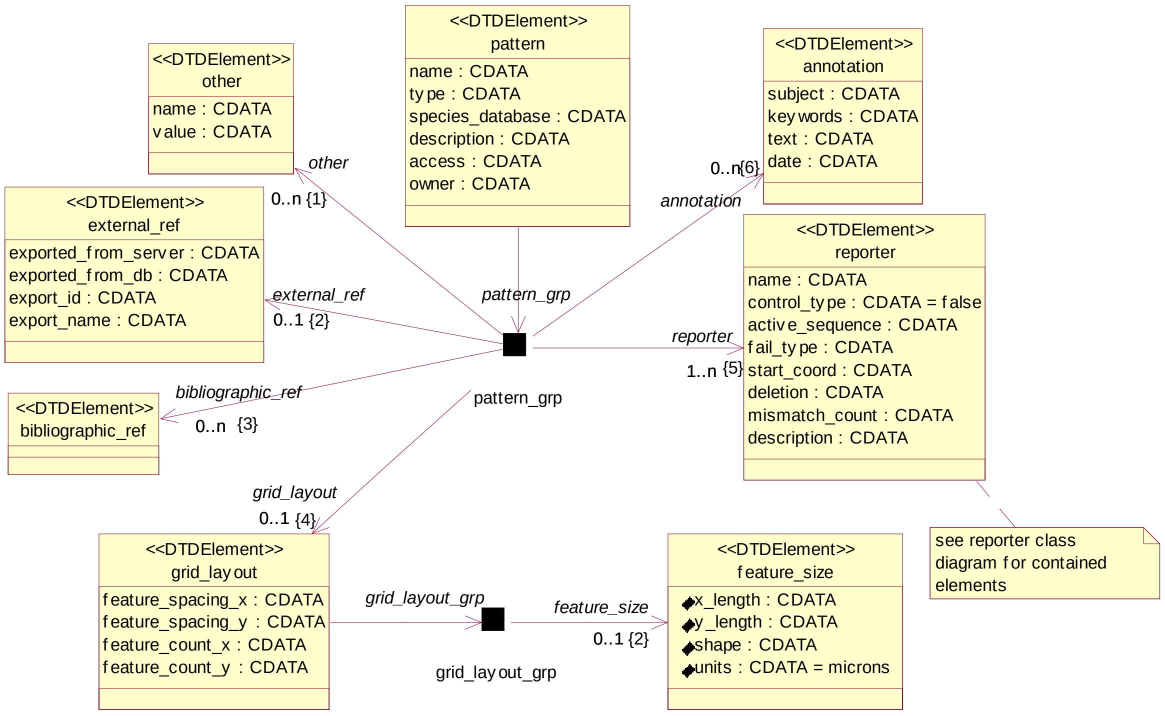

Figure 9: pattern and contained elements

```
<!ELEMENT pattern
   (other*,
    external_ref?,
    bibliographic_ref*,
    grid_layout?,
    reporter+,
    annotation*)
>
<!ATTLIST pattern
   name             CDATA #REQUIRED
   type             CDATA #REQUIRED
   species_database CDATA #REQUIRED
   description      CDATA #IMPLIED
   access           CDATA #IMPLIED
   owner            CDATA #IMPLIED
>
```

```
<!ELEMENT grid_layout
   (other*,
    feature_size?)
>
<!ATTLIST grid_layout
   feature_spacing_x CDATA #IMPLIED
   feature_spacing_y CDATA #IMPLIED
   feature_count_x   CDATA #IMPLIED
   feature_count_y   CDATA #IMPLIED
>

<!ELEMENT feature_size
   (other*)
>
<!ATTLIST feature_size
   x_length CDATA #REQUIRED
   y_length CDATA #REQUIRED
   shape    CDATA #IMPLIED
   units    CDATA 'microns'
>
```

Attribute	**Description**
name	unambiguous name of the pattern within the source
type	derived name (see section 2.1.2) to describe the type of array
species_database	unambiguous identifier, within the source, of the species, a combination of species, or a subset of a species which is represented by the reporters. By allowing this designation, an organization can logically separate experiments that contain the same biosequences and/or reporters from being analyzed inappropriately together
description	freehand description of the pattern
access	comma-delimited access group that lists what groups have permission to view the data
owner	the person, group or organization, within the source, that has ownership permission on this pattern

grid_layout

The grid layout describes a rectangular array pattern. In practice, the vast majority of arrays are rectangular but in those cases that they are not, then the **grid_layout** element will not be included and the **feature.x** and **feature.y** attributes either physically or logically place the feature on the array.

Attribute	**Description**
feature_spacing_x	distance between the center of the features on the x-axis
feature_spacing_y	distance between the center of the features on the y-axis
feature_count_x	number of features along the x-axis
feature_count_y	number of features along the y-axis

feature_size

Describes the area of the feature that is of interest for the statistical analysis.

Attribute	**Description**
x_length	length along the x-axis
y_length	length along the y-axis
shape	the shape, such as circular, oval, etc., of the feature
units	the units of the length attributes

reporter

Description of the material that is placed on a feature to represent a biological sequence. A reporter, in some technologies, can vary from the biosequence it represents. This information is captured in the **mismatch_info** and the **deletion_info** elements and/or summarized in the **deletion** and **mismatch_count** attributes.

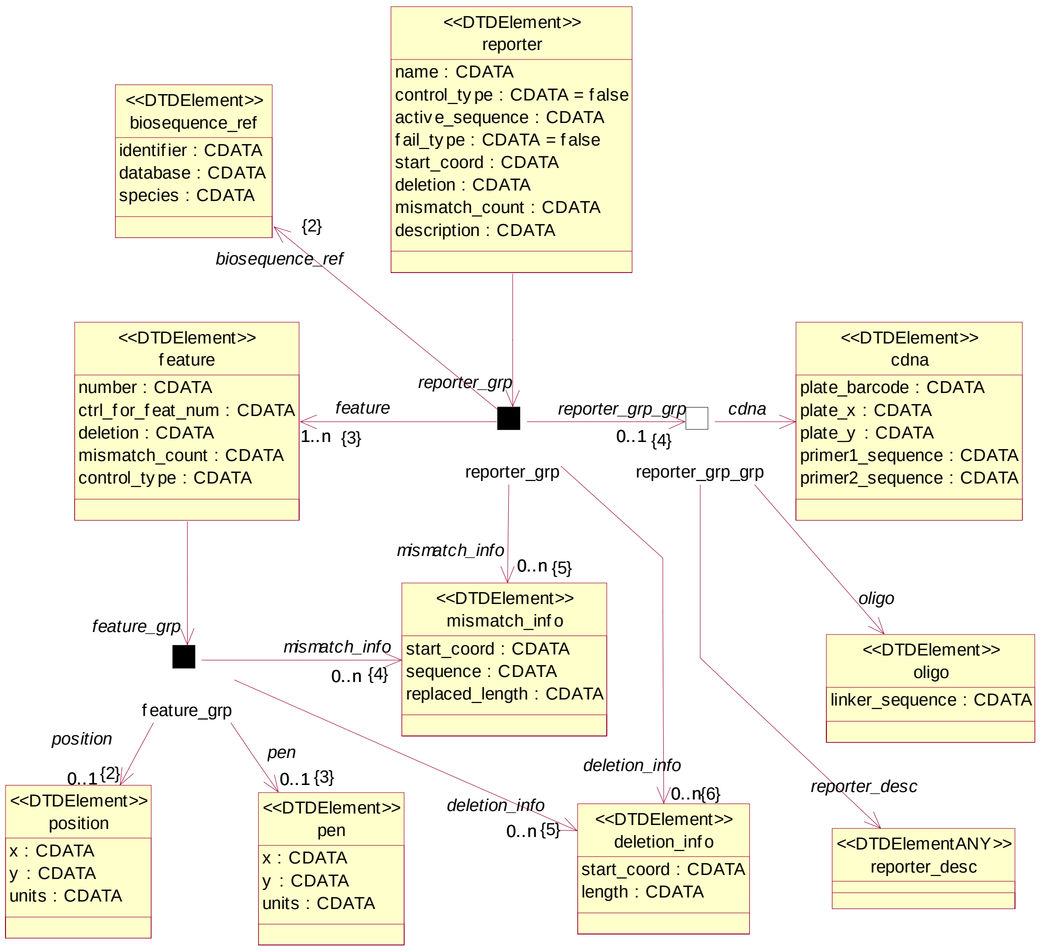

Figure 10: reporter and contained elements.

```
<!ELEMENT reporter
   (other*,
    feature+,
    biosequence_ref,
    (oligo |
     cdna  |
     reporter_desc)?,
    mismatch_info*,
    deletion_info*)
>
<!ATTLIST reporter
   name              CDATA #REQUIRED
   control_type      CDATA "false"
   fail_type         CDATA "false"
   active_sequence   CDATA #IMPLIED
   start_coord       CDATA #IMPLIED
   deletion          CDATA #IMPLIED
   mismatch_count    CDATA #IMPLIED
   description       CDATA #IMPLIED
>

<!ELEMENT oligo
   (other*)
```

```
>
<!ATTLIST oligo
   linker_sequence CDATA #IMPLIED
>

<!ELEMENT cdna
   (other*)
>
<!ATTLIST cdna
   plate_barcode    CDATA #IMPLIED
   plate_x          CDATA #IMPLIED
   plate_y          CDATA #IMPLIED
   primer1_sequence CDATA #IMPLIED
   primer2_sequence CDATA #IMPLIED
>

<!ELEMENT reporter_desc
   ANY
>

<!ELEMENT feature
   (other*,
    position?,
    pen?,
    mismatch_info*,
    deletion_info*)
>
<!ATTLIST feature
   number            CDATA #IMPLIED
   ctrl_for_feat_num CDATA #IMPLIED
   deletion          CDATA #IMPLIED
   mismatch_count    CDATA #IMPLIED
   control_flag      CDATA #IMPLIED
>

<!ELEMENT position
   (other*)
>
<!ATTLIST position
   x     CDATA #REQUIRED
   y     CDATA #REQUIRED
   units CDATA #IMPLIED
>

<!ELEMENT pen
   (other*)
>
<!ATTLIST pen
   x     CDATA #REQUIRED
   y     CDATA #REQUIRED
   units CDATA #IMPLIED
>

<!ELEMENT mismatch_info
   (other*)
>
<!ATTLIST mismatch_info
   start_coord     CDATA #REQUIRED
   sequence        CDATA #REQUIRED
   replaced_length CDATA #REQUIRED
>

<!ELEMENT deletion_info
   (other*)
>
<!ATTLIST deletion_info
   start_coord CDATA #REQUIRED
   length      CDATA #REQUIRED
>
```

Attribute	Description
name	the name for this reporter
control_type	list of one (usually) or more from a control name type enumeration, for instance: NEGATIVE_CONTROL, POSITIVE_CONTROL, DELETION_CONTROL, etc.
fail_type	a comma-delimited list from a fail enumeration indicating that the reporter has been determined to be bad. This can happen, for instance, if an oligo thought representative of a biosequence is shown not to be
active_sequence	the nucleotide sequence used on the array
start_coord	at what position in the underlying biosequence the reporter begins
deletion	true, if there is a deletion in the reporter sequence compared to the biosequence it reports on
mismatch_count	count of how many mismatches are present in the reporter sequence compared to the biosequence it reports on
description	freehand description of the reporter

feature

the position of a reporter on a pattern.

Attribute	Description
number	an unambiguous identifier, within the pattern, for this feature. This provides a reliable way for the profile elements, feature_data, to identify a particular feature and through the feature, to identify the reporter and the biosequence
ctrl_for_feat_num	if this feature is associated as a control for another feature, then this will be the **feature.number** of that feature. This is useful to identify feature(probe) sets of a reporter
mismatch_count	count of how many mismatches are present in the reporter sequence compared to the biosequence it reports on
deletion	true, if there is a deletion in the reporter sequence compared to the biosequence it reports on
control_type	list of one (usually) or more from a control type enumeration

position

if the pattern isn't associated with a **grid_layout** element, the physical or logical position of the feature can be defined by the **x**, **y** attributes of **position**.

pen

location of the pen in the grid of pens. Relevant for cDNA.

mismatch_info

describes where a feature differs from its reporter's sequence or where a reporter differs from its biosequence.

Insertions can be specified by a **replaced_length** of zero and a **sequence** of length one or more.

Attribute	**Description**
start_coord	position in the sequence where the mismatch begins
sequence	the replacement sequence
replaced_length	how many nucleotides have been replaced

deletion_info

similar to **mismatch_info**, only describing a pure deletion from either the **reporter**, for **feature**, or **biosequence**, for **reporter**.

oligo

represents the oligo placed onto a feature.

Attribute	**Description**
linker_sequence	an optional 'foundation' sequence used to link a biologically active oligonucleotide to the surface of an array and elevate it above the surface.

cdna

represents the cDNA placed on a feature.

Attribute	**Description**
plate_barcode	identifies the plate from which the cDNA was drawn from to be placed on the array pattern
plate_x	identifies the well from which the cDNA was drawn from to be placed on the array pattern
plate_y	identifies the well from which the cDNA was drawn from to be placed on the array pattern

primer1_sequence	sequences used in the PCR replication of the cDNA
primer2_sequence	sequences used in the PCR replication of the cDNA

reporter_desc

placeholder for **reporter** elements that describe another type of reporter other than oligo or cDNA.

pattern_ref

references a pattern, within the source, by **name**.

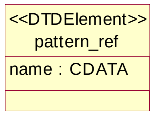

Figure 11: pattern_ref.

```
<!ELEMENT pattern_ref
   (other*)
>
<!ATTLIST pattern_ref
   name            CDATA #REQUIRED
>
```

feature_ref

an unambiguous reference to a feature within a pattern. This is either through the **number** attribute or through the **position** element.

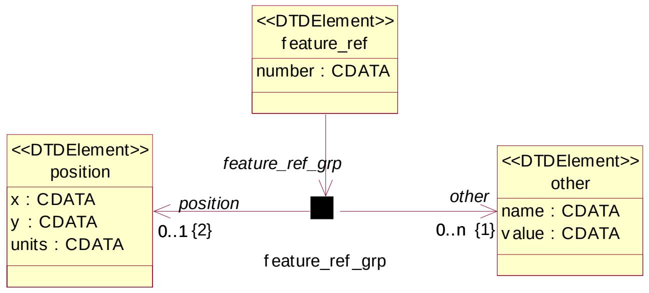

Figure 12: feature_ref

```
<!ELEMENT feature_ref
   (other*,
    position?)
>
<!ATTLIST feature_ref
   number           CDATA #IMPLIED
>
```

4.1.6 *Printing*

Describes the printing of physical arrays that conform to a particular pattern. The **printing** element allows the tracking of the production of the physical arrays, 'chips', and the assignment of a unique identifier, the **barcode**, for each array. It also allows, through the **printing_rep** element, the recording of any errors on a per feature level.

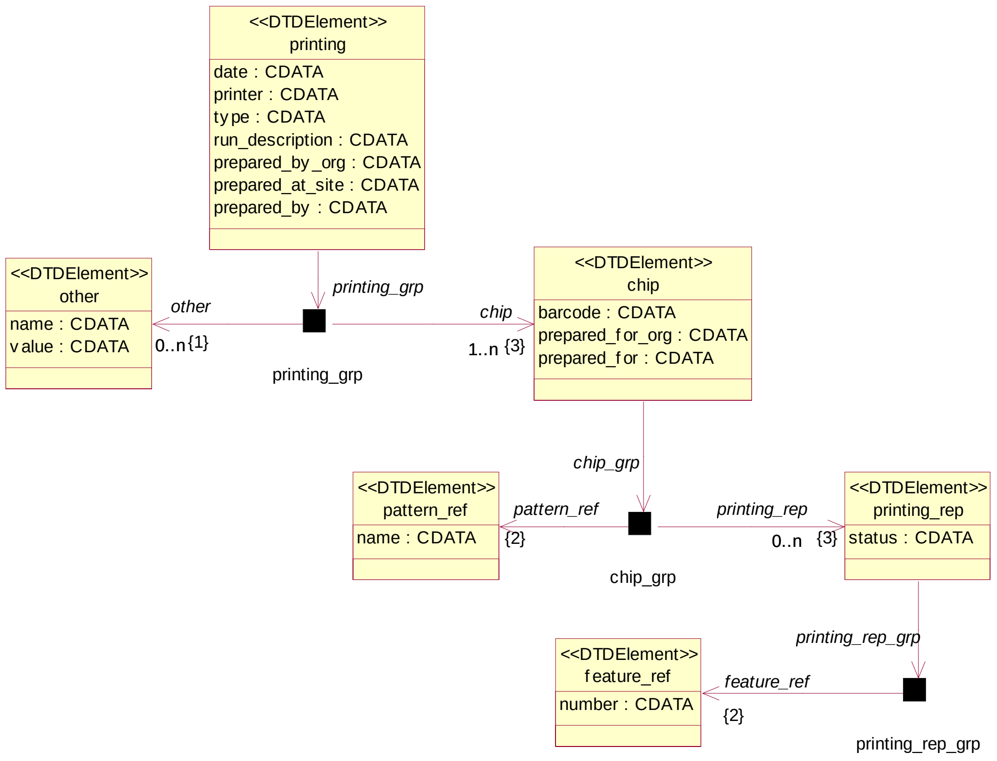

Figure 13: printing and contained elements.

```
<!ELEMENT printing
   (other*,
    chip+)
>
<!ATTLIST printing
   date             CDATA #IMPLIED
   printer          CDATA #IMPLIED
   type             CDATA #IMPLIED
   run_description  CDATA #IMPLIED
   prepared_by_org  CDATA #IMPLIED
   prepared_at_site CDATA #IMPLIED
   prepared_by      CDATA #IMPLIED
>

<!ELEMENT chip
   (other*,
    pattern_ref,
    printing_rep*)
>
<!ATTLIST chip
   barcode          CDATA #REQUIRED
   prepared_for_org CDATA #IMPLIED
   prepared_for     CDATA #IMPLIED
>

<!ELEMENT printing_rep
   (other*,
    feature_ref)
>
<!ATTLIST printing_rep
   status CDATA #REQUIRED
>
```

Attribute	**Description**
date	date of the printing run
printer	the unambiguous name, within the source, of the printer used for the run
type	the type of the printer
run_description	freehand description of the run
prepared_by_org	the organization that ran the printing
prepared_at_site	at which site of the organization the printing was run
prepared_by	the person who performed or oversaw the printing run

chip

the **chip** element identifies the actual physical chip that is produced as part of a run. The **barcode** identifier is the most important identifier in the array pattern process, since it links together the **pattern, profile, hyb** and **experiment_set** elements.

Attribute	**Description**
barcode	the unambiguous identifier, within the source, of the physical printing of the pattern. This is sometimes physically etched onto the chip

printing_rep

reports on a feature, for this chip, which had an anomaly in its printing.

Attribute	**Description**
status	a comma-delimited list from a STATUS ENUMERATION indicating errors

4.1.7 Sample

Specifies the characteristics of the biological source used in a hybridization. The contained element, **species_data**, is an ANY element type which allows for arbitrary elements to be nested with out validation. As mentioned above (section 3.1.5), the **ExtensionDTDList** may contain an entry for **species_data** that would be paired with a DTD that describes what nested elements can be expected.

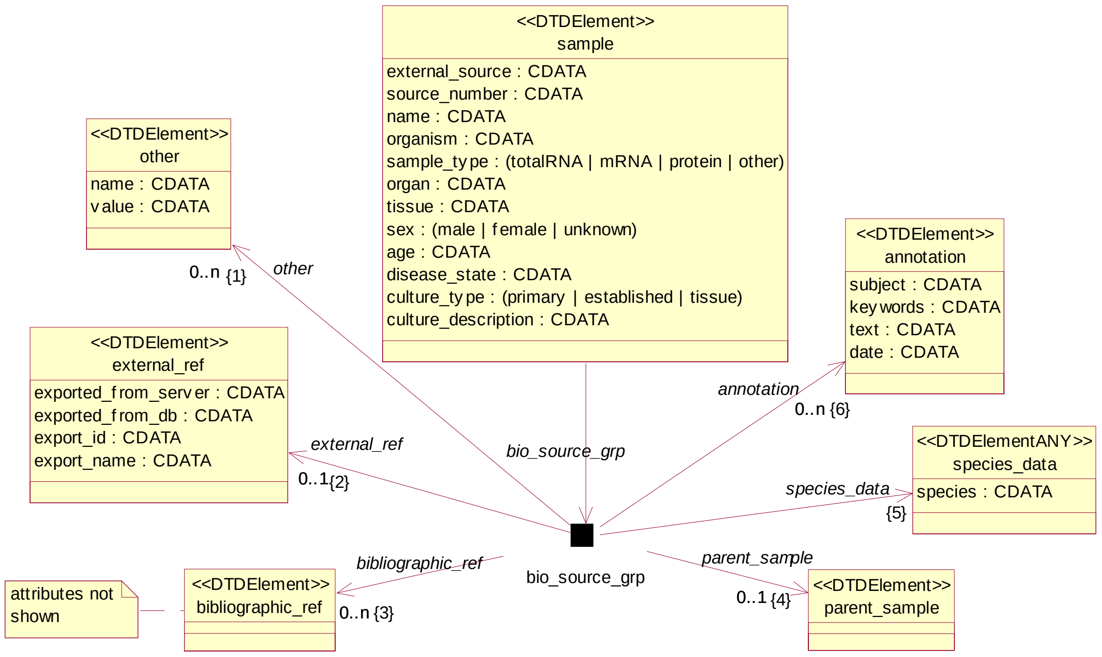

Figure 14: sample and contained elements.

```
<!ELEMENT sample
   (other*,
    external_ref?,
    bibliographic_ref*,
    parent_sample?,
    species_data,
    annotation*)
>
<!ATTLIST sample
   external_source    CDATA  #IMPLIED
   source_number CDATA  #IMPLIED
   name          CDATA  #REQUIRED
   organism      CDATA  #REQUIRED
   sample_type (totalRNA | mRNA | protein | other) #REQUIRED
   organ        CDATA  #IMPLIED
   tissue      CDATA  #IMPLIED
   sex (male | female | unknown) #IMPLIED
   age        CDATA  #IMPLIED
   disease_state CDATA  #IMPLIED
   culture_type (primary | established | tissue)  #IMPLIED
   culture_description CDATA  #IMPLIED
>

<!ELEMENT parent_sample
   (other*,
    sample_ref)
>

<!ELEMENT species_data
   ANY
>
<!ATTLIST species_data
   species CDATA #REQUIRED
```

```
>
```

Attribute	Description
external_source	the organization, such as ATCC, from which the sample was obtained
source_number	an unambiguous designation, within the source, for the sample
name	descriptive designation for the sample
organism	high-level category for the origin of the cell line, such as 'microorganism', 'mammals', etc.
bio_type	biological type used, one of total RNA, mRNA, protein or other.
organ	name of the organ (lung, heart, etc.)
tissue	name of the tissue (bronchus, ventricle, etc.)
sex	one of 'male', 'female' or 'unknown'
age	age, if relevant, of the source of the **sample**
disease_state	information on the state of the disease, if relevant
culture_type	designation of source type, one of 'primary', 'established', or 'tissue'
culture_description	freehand description of the culture

sample_ref

unambiguous reference, within the source, to a **sample** by the **source_number** reference.

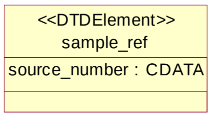

Figure 15: sample_ref

```
<!ELEMENT sample_ref
   (other*)
>
<!ATTLIST sample_ref
   source_number CDATA  #REQUIRED
>
```

parent_sample

References the sample this sample was created and/or derived from.

species_data

An element whose **species** attribute is the common species designation, such as 'A. Thaliana' or 'E. coli', from which the sample was obtained, based on NCBI's Taxonomy. **Species_data** is of element type ANY, to allow nested elements, specific to organizations, to further describe attributes of the species.

Example extended DTD

Below is an example of how an extended DTD might be specified for **species_data**. Here the attributes are explicitly called out for two species. An alternative might be an ontological-based set of elements.

```
<!ELEMENT h_sapiens
   (other*)
>
<!ATTLIST h_sapiens
   morphology  CDATA #IMPLIED
   karyotype   CDATA #IMPLIED
   growth      CDATA #IMPLIED
   propagation CDATA #IMPLIED
   deletion    CDATA #IMPLIED
   transfected CDATA #IMPLIED
   originator  CDATA #IMPLIED
   ethnicity   CDATA #IMPLIED
>

<!ELEMENT s_cerevisiae
   (other*)
>
<!ATTLIST s_cerevisiae
   genotype      CDATA  #IMPLIED
   phenotype     CDATA  #IMPLIED
   orf           CDATA  #IMPLIED
   ploidy_number CDATA  #IMPLIED
   ploidy_name   CDATA  #IMPLIED
   strain_name   CDATA  #IMPLIED
>
```

4.1.8 Compound

Compound refers to a substance added to a treatment.

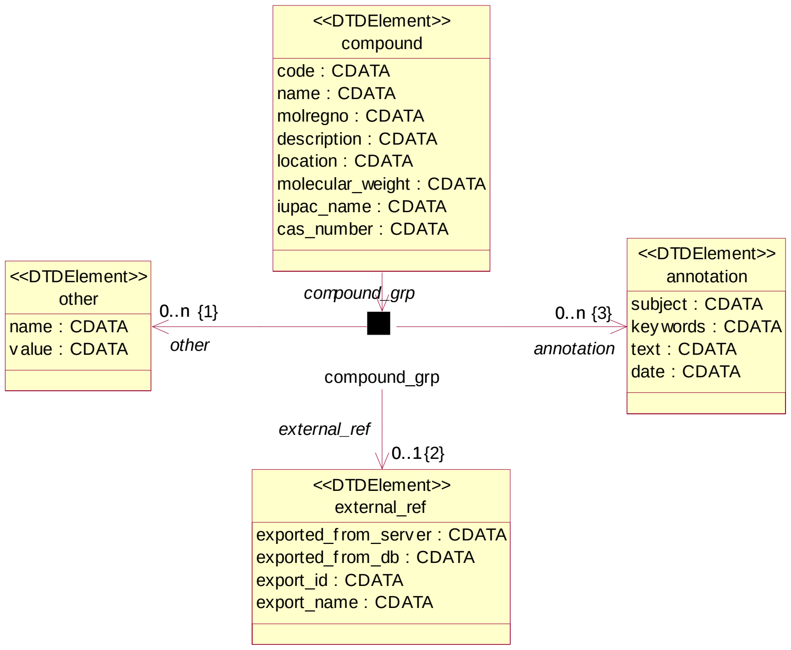

Figure 16: compound and contained elements

```
<!ELEMENT compound
   (other*,
    external_ref?,
    annotation*)
>
<!ATTLIST compound
   code                      CDATA  #REQUIRED
   name                      CDATA  #REQUIRED
   molregno                  CDATA  #IMPLIED
   description               CDATA  #IMPLIED
   location                  CDATA  #IMPLIED
   molecular_weight          CDATA  #IMPLIED
   iupac_name                CDATA  #IMPLIED
   cas_number                CDATA  #IMPLIED
>
```

Attribute	**Description**
code	unambiguous identifier, within the source, for this compound
name	common name for this compound
molregno	a unique compound identifier within a proprietary chemical database from MDL Information Systems
description	a freehand description of the compound
location	storage location of the compound within the laboratory
molecular_weight	sum of the masses (in atomic mass units) of all atoms in the compound

iupac_name	compound name in nomenclature system developed by International Union of Pure and Applied Chemistry
cas_number	Chemical Abstracts Service number (unique to the compound)

compound_ref

References a compound through the **code** attribute.

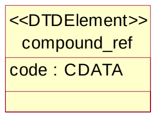

Figure 17: compound_ref

```
<!ELEMENT compound_ref
   (other*)
>
<!ATTLIST compound_ref
   code CDATA  #REQUIRED
>
```

4.1.9 Solvent

A substance for dissolving or dispersing one or more other substances, typically a compound(s), used in a treatment.

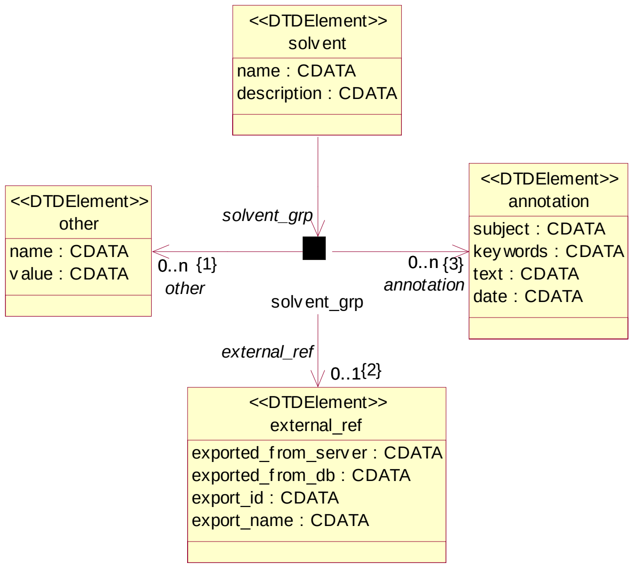

Figure 18: solvent and contained elements

```
<!ELEMENT solvent
   (other*,
    external_ref?)
>
<!ATTLIST solvent
```

```
   name        CDATA  #REQUIRED
   description CDATA  #IMPLIED
>
```

Attribute	Description
name	unambiguous name, within the source, of the solvent
description	freehand description of the solvent

solvent_ref

References a solvent through the **name** attribute.

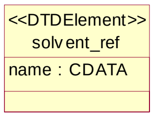

Figure 19: solvent_ref

```
<!ELEMENT solvent_ref
   (other*)
>
<!ATTLIST solvent_ref
   name CDATA  #REQUIRED
>
```

4.1.10 Prep

A cellular extract and its preparation described by its treatment steps. Each treatment step is described by a **treatment** element. The treatments will begin with a cell line or tissue treatment which is then subjected to a serial or concurrent set of treatments. Two of the most common treatments are a dose titration series or a single treatment measured across several time points.

There are cases, though, where two sets of treatments can be combined after they have gone through one or more steps. There is also the possibility that after a treatment step, the result is split into a set of treatments then recombined at a later step. The containment of **treatment** in itself is to allow this possibility.

There is also tracking information and quality control on the preparation

It is worth noting that the DTD specification allows a valid XML document without specifying a **sample**. The convolutions to the DTD to enforce this constraint is seen as unnecessary, since, in practice, treatments for gene expression hybridizations must begin with a biological source.

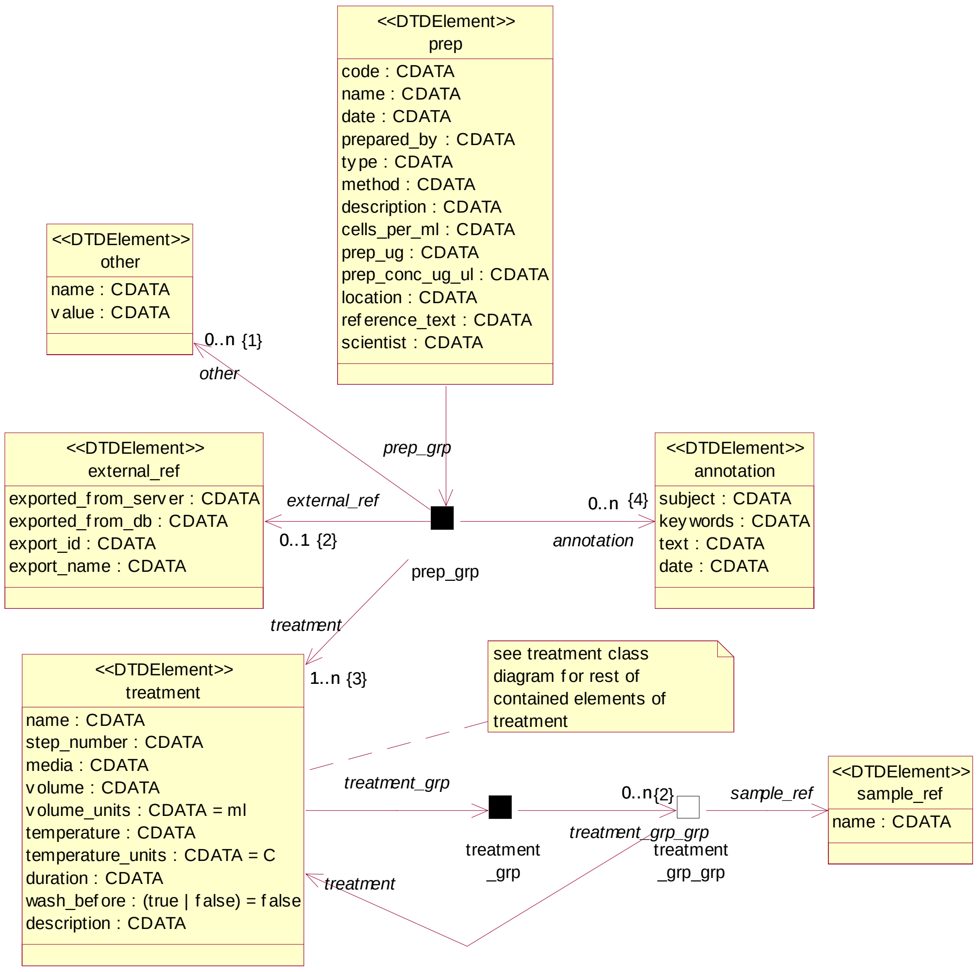

Figure 20: prep and contained elements.

```
<!ELEMENT prep
   (other*,
    external_ref?,
    treatment+,
    annotation*)
>
<!ATTLIST prep
   code                    CDATA  #REQUIRED
   name                    CDATA  #IMPLIED
   date                    CDATA  #IMPLIED
   prepared_by             CDATA  #IMPLIED
   type                    CDATA  #IMPLIED
   method                  CDATA  #IMPLIED
   description             CDATA  #IMPLIED
   cells_per_ml            CDATA  #IMPLIED
   prep_ug                 CDATA  #IMPLIED
   prep_conc_ug_ul         CDATA  #IMPLIED
   location                CDATA  #IMPLIED
   reference_text          CDATA  #IMPLIED
   scientist               CDATA  #IMPLIED
>
```

Attribute	Description
code	unambiguous identifier within the source for this preparation
name	common name for the preparation
date	date the preparation was created
prepared_by	the person who extracts the preparation
type	keyword referring to type of preparation
method	keyword referring to the method of preparation
description	freehand description of the preparation
cells_per_ml	concentration of cells per unit milliliter in the sample the preparation was extracted from
prep_ug	mass of the preparation (in micrograms)
prep_conc_ug_ul	concentration of the preparation in microgram per unit microliter
location	storage location of the preparation in the laboratory, e.g. a freezer and box number
reference_text	laboratory reference (sometimes a laboratory notebook reference) for this preparation
scientist	name of the scientist who designed or requested this preparation

treatment

Describes a step in the process of producing a preparation. The first step will start generally with either one or more cell lines or one or more tissues which receives one or more treatments. The treatment generally consists of a combination of compounds and solvents that is applied to a previous cell or tissue treatment, or an intermediate treatment, such as wash or duration.

The inclusion of **treatments** or **sample_refs** within a **treatment** element allows a **treatment** to not only begin with **sample** but to also take sets of separate **treatments** and combine them together for the next treatment step.

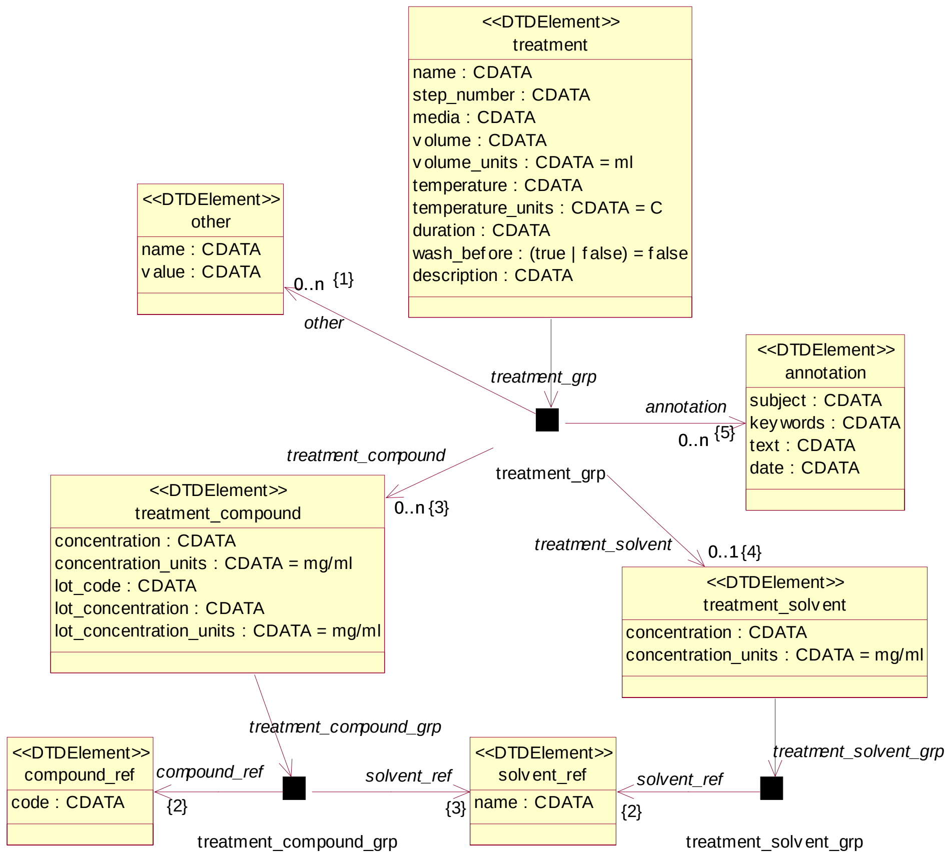

Figure 21: treatment and contained elements.

```
<!ELEMENT treatment
   (other*,
    (treatment |
     sample_ref)*,
    treatment_compound*,
    treatment_solvent?,
    annotation*)
>
<!ATTLIST treatment
   name              CDATA  #REQUIRED
   step_number       CDATA  #REQUIRED
   media             CDATA  #IMPLIED
   volume            CDATA  #IMPLIED
   volume_units      CDATA  'ml'
   temperature       CDATA  #IMPLIED
   temperature_units CDATA  'C'
   duration          CDATA  #IMPLIED
   wash_before       (true | false) 'false'
   description       CDATA #IMPLIED
>

<!ELEMENT treatment_compound
   (other*,
    compound_ref,
    solvent_ref)
>
```

```
<!ATTLIST treatment_compound
   concentration           CDATA  #REQUIRED
   concentration_units     CDATA  'mg/ml'
   lot_code                CDATA  #IMPLIED
   lot_concentration       CDATA  #IMPLIED
   lot_concentration_units CDATA  'mg/ml'
>

<!ELEMENT treatment_solvent
   (other*,
    solvent_ref)
>
<!ATTLIST treatment_solvent
   concentration           CDATA  #REQUIRED
   concentration_units     CDATA  'mg/ml'
>
```

Attribute	**Description**
name	an unambiguous identifier, within the source, for this treatment
step_number	indicates, in the preparation, where this treatment takes place in a series of treatments
media	the liquid the treatment is part of
volume	total amount of the treatment
volume_units	the units of the volume
temperature	the temperature the treatment occurs at
temperature_units	the units of the temperature
duration	the length of the treatment
wash_before	whether previous treatment removed
description	a freehand description of the treatment

treatment_compound

Describes a substance created by combining a compound and a solvent for creating a preparation.

Attribute	**Description**
concentration	concentration of the compound in the solvent
concentration_units	the units of concentration
lot_code	unambiguous identifier, within the source, of the lot of the compound
lot_concentration	concentration of the substance in the lot
lot_concentration_units	the units of concentration for the lot

treatment_solvent

Describes a treated solvent for use in creating a prep.

Attribute	**Description**
concentration	concentration of the solvent
concentration_units	the units of the concentration

prep_ref

References a preparation by its **code** attribute.

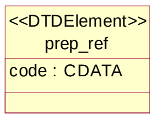

Figure 22: prep_ref.

```
<!ELEMENT prep_ref (other*)>
<!ATTLIST prep_ref
   code CDATA  #REQUIRED
>
```

4.1.11 Hyb

For microarray experiments, this is an array and labeled preparations.   The hybridized physical array will then be scanned to provided the raw data for the experiment profile. Information is also recorded how and where the hybridization was performed. The **reference_text** and **description** attributes and the **annotation** element allow a full explanation of what the experiment was intended to demonstrate and why.

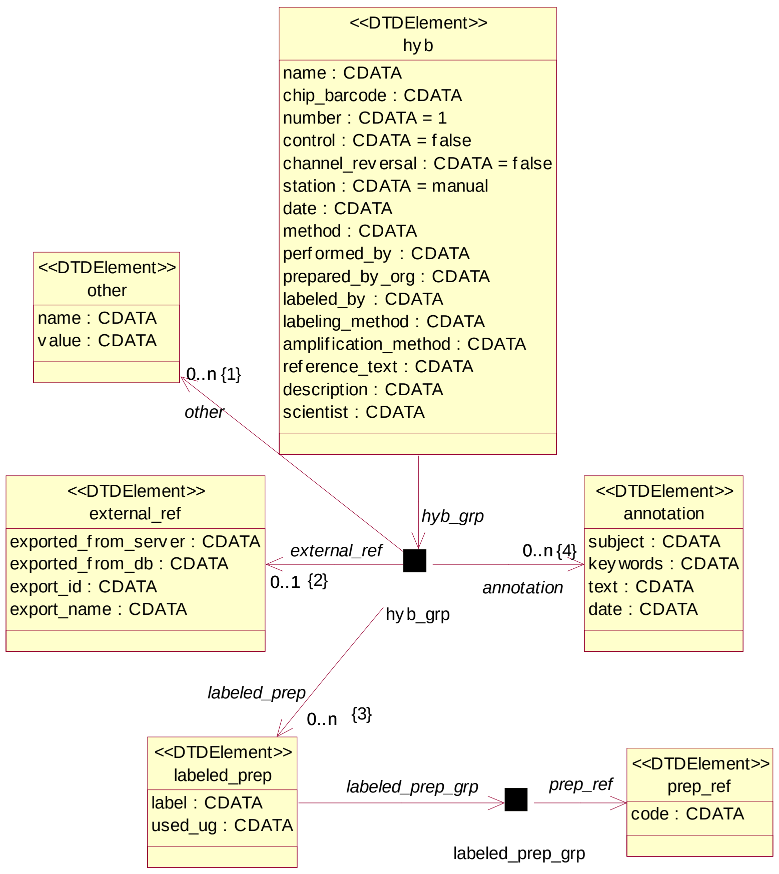

Figure 23: hyb and contained elements.

```
<!ELEMENT hyb
   (other*,
    external_ref?,
    labeled_prep*,
    annotation*)
>
<!ATTLIST hyb
   name                 CDATA  #REQUIRED
   chip_barcode         CDATA  #REQUIRED
   number               CDATA  "1"
   control              CDATA  'false'
   channel_reversal     CDATA  'false'
   station              CDATA  'manual'
   date                 CDATA  #IMPLIED
   method               CDATA  #IMPLIED
   performed_by         CDATA  #IMPLIED
   performed_by_org     CDATA  #IMPLIED
   labeled_by           CDATA  #IMPLIED
   labeling_method      CDATA  #IMPLIED
   amplification_method CDATA  #IMPLIED
   reference_text       CDATA  #IMPLIED
   description          CDATA  #IMPLIED
   scientist            CDATA  #IMPLIED
>
```

```
<!ELEMENT labeled_prep
   (prep_ref)
>
<!ATTLIST labeled_prep
   label     CDATA  #IMPLIED
   used_ug   CDATA  #IMPLIED
>
```

Attribute	Description
name	name, within the source, for this hybridization
chip_barcode	unambiguous reference within the source to the physical array or chip that was hybridized
number	If an array can be stripped and re-used, this will be the number of times the array has been used, inclusive
control	whether this hybridization is considered a control
channel_reversal	for ratio-based experiments, set to true if the first channel is experimental and the second is baseline. Set to false if the first channel is baseline and the second is experimental.
station	what station the hybridization was preformed at
date	the date and time the hybridization was performed
method	keyword describing how the hybridization was performed
performed_by	person who performed the hybridization
performed_by_org	organization where the hybridization was performed
labeling_method	keyword describing the labeling method
labeled_by	person who performed the labeling
amplification_method	keyword describing the amplification method used
reference_text	laboratory reference (sometimes a laboratory notebook reference) for this hybridization
description	freehand description of the hybridization
scientist	name of the scientist who designed the experiment

labeled_prep

Describes what the sample and its preparation was labeled with and with how much.

Attribute	**Description**
label	the name of the label (Cy3, Cy5, etc.)
used_ug	the amount of the label used (in micrograms)

hyb_ref

References an hybridization by the **chip_barcode** and which, by **number**, of a series of uses, this hybridization refers to. The **number** attribute is necessary since a physical chip can be washed and reused for new hybridizations.

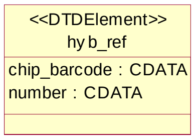

Figure 24: hyb_ref.

```
<!ELEMENT hyb_ref
   (other*)
>
<!ATTLIST hyb_ref
   chip_barcode CDATA  #REQUIRED
   number       CDATA  #IMPLIED
>
```

4.1.12 Combine

The **combine** element contains one or more hybridizations that will be analyzed as a group. This includes aggregation of non-redundant datasets from multiple arrays to work around the limitation of the maximum number of features per **array** and data repeated to achieve higher statistical significance.

It is convenient to group together hybridizations which should be analyzed together as a whole. This is true of hybridizations which are channel_reversals of each other and are replicates of each other. It also allows aggregation to work around the limitation of the number of features per profile. The **combine** element is a reference element that groups these hybridizations together, for instance, as a stand alone vocabulary in DsLSR_GEML or in the **experiment_set** element.

Hybridizations are grouped together, rather than profiles, for two reasons. A profile may be scanned multiple times and only the best may be selected to be associated with the hybridization. The second, it allows knowledge of how many times a chip array has been re-used recorded by the **number** attribute of **hyb**.

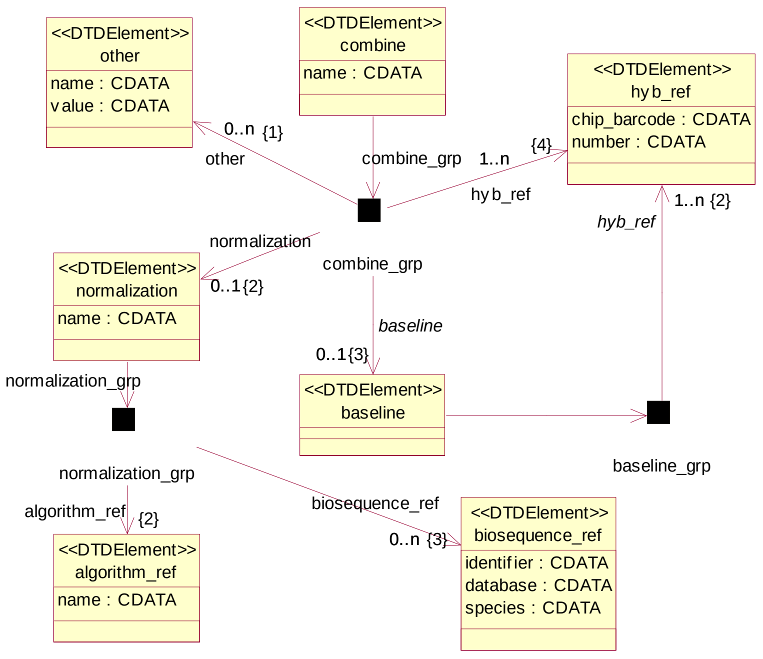

Figure 25: combine and contained elements

```
<!ELEMENT combine
   (other*,
    normalization?,
    baseline?,
    hyb_ref+)
>
<!ATTLIST combine
   name CDATA #IMPLIED
>

<!ELEMENT normalization
   (other*,
    algorithm_ref,
    biosequence_ref*)
>
<!ATTLIST normalization
   name CDATA #REQUIRED
>

<!ELEMENT baseline
    (other*,
     hyb_ref+)
>
```

normalization

Normalization is needed when the hybridizations grouped together are intensity-based and vary in their range of intensities. The most common technique is to use some **biosequences** as normalization controls, in which case normalization can take place at the **profile** level. Some of these algorithms will take a list of **biosequences** as a parameter.

The **normalization** element references the name of the underlying algorithm used through the **algorithm_ref**.

baseline

Container to group together hybridizations to use as a baseline in the statistical analysis of the **combine**.

4.1.13 Profile

The results of the analysis of a scan or scans of a hybridization. The **error_model** references the algorithm (which might be a combination of algorithms) used to analysis the data, or, in the case the analysis is to take place post hoc, to analysis it on loading. The **error_model** can, typically, perform normalization, detrending and/or averaging of duplicates within the data set. For example, if the same **reporter** exists on multiple **features**, it could be averaged into one **reporter_data** element.

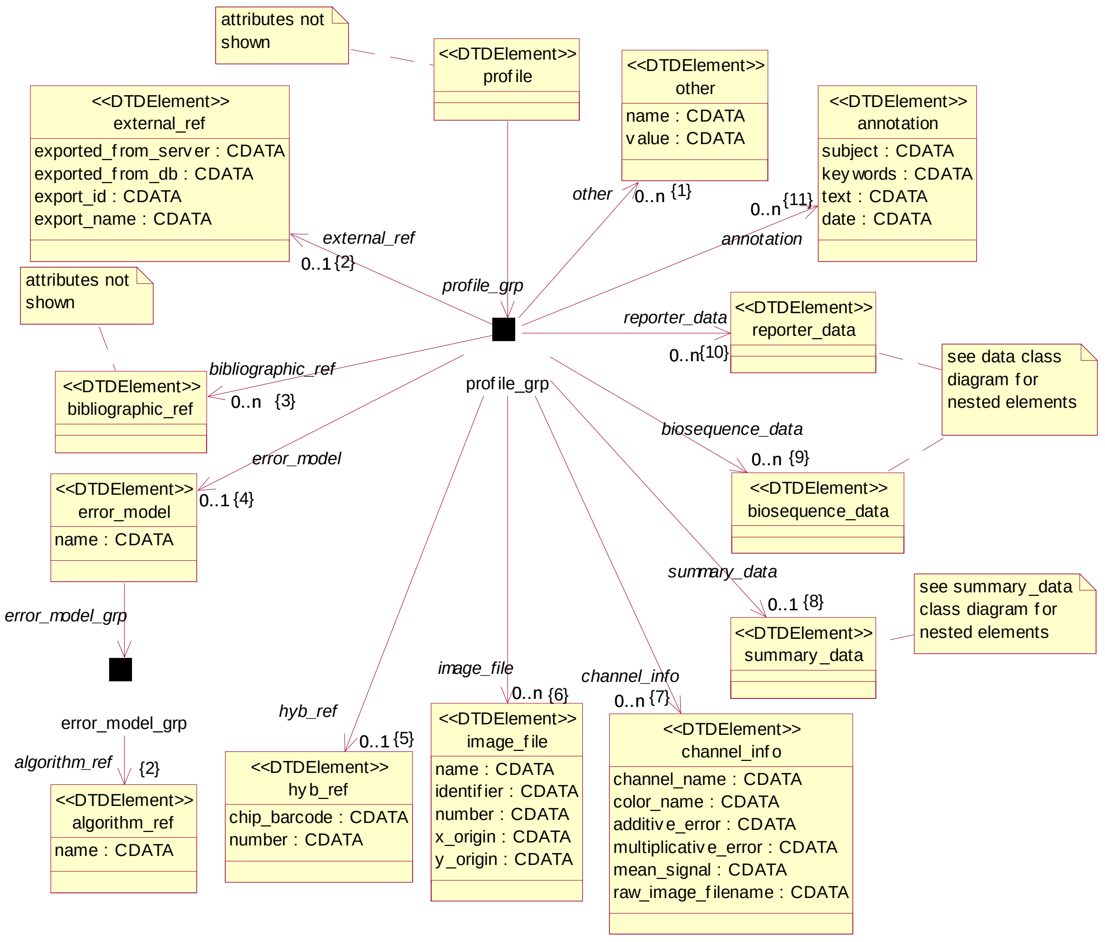

Figure 26: profile and contained elements.

```
<!ELEMENT profile
   (other*,
     external_ref?,
     bibliographic_ref*,
     error_model?,
     hyb_ref?,
     image_file*,
     channel_info*,
     summary_data?,
     reporter_data*,
     biosequence_data*,
     annotation*)
>
<!ATTLIST profile
   name             CDATA  #IMPLIED
   type             CDATA  #IMPLIED
   barcode          CDATA  #IMPLIED
   access           CDATA  #IMPLIED
   owner            CDATA  #IMPLIED
   scanner          CDATA  #IMPLIED
   number           CDATA  #IMPLIED
   performed_date   CDATA  #IMPLIED
   performed_by     CDATA  #IMPLIED
   analyzed_date    CDATA  #IMPLIED
   analyzed_by      CDATA  #IMPLIED
   fail_type        CDATA  #IMPLIED
   control_flag     CDATA  'false'
   algorithm_state (COMPLETE | CALCULATE | NONE) 'COMPLETE'
   profile_quality  CDATA  'Pending QC'
   qc_by            CDATA  #IMPLIED
>

<!ELEMENT error_model
   (other*,
    algorithm_ref)
>
<!ATTLIST error_model
   name CDATA #REQUIRED
>

<!ELEMENT image_file
   (other*)
>
<!ATTLIST image_file
   name       CDATA  #REQUIRED
   identifier CDATA  #IMPLIED
   number     CDATA  #IMPLIED
   x_origin   CDATA  #IMPLIED
   y_origin   CDATA  #IMPLIED
>

<!ELEMENT channel_info  (other*)>
<!ATTLIST channel_info
   channel_name         CDATA  #REQUIRED
   color_name           CDATA  #REQUIRED
   additive_error       CDATA  #IMPLIED
   multiplicative_error CDATA  #IMPLIED
   mean_signal          CDATA  #IMPLIED
   raw_image_filename   CDATA  #IMPLIED
>
```

Attribute	**Description**
name	name given to this profile
type	derived name (see section 2.1.2) to describe the type of profile
barcode	the unambiguous identifier, within the source, of the physical array that was used to generate this profile
access	a comma-delimited list of access groups who are entitled to access this document

owner	the person, group, or organization, within the source, that has ownership permission on this profile
scanner	the name of the scanner, within the source, that was used
scan_number	if multiple profiles were derived from the same hybridization, this number differentiates them
performed_date	date the scan occurred
performed_by	the person who performed the scan or generated the profile
analyzed_date	date the scan was analyzed
analyzed_by	the person who performed the analysis
fail_type	a comma-delimited list from a FAIL ENUMERATION indicating errors
control_flag	indicates whether this profile was used as a control
algorithm_state	used to indicate if a statistical algorithm has been applied, should be applied, or has not been applied
profile_quality	the quality of the scan, such as 'Pending QC', 'Passed QC', 'Failed QC', etc.
qc_by	the person who performed the quality control

error_model

In order to average repeated experiments with appropriate relative weights, a model for the uncertainties in individual array profiles is required. The error model must also consider those cases when a weakly expressed reporter is negative after adjustment for background intensities. The best error model can vary by technology and, even within the same technology, there are differences of opinion at how best to compute these values. The **error_model** references the name of the underlying algorithm used through the **algorithm_ref**.

image_file

The basis of all the values for a profile is from the scan of a hybridized chip. These provide the **raw_value** and **pixels** of the **signal** and **value** and **pixels** of the **backround** elements. Practice has shown that there can be multiple image files and/or multiple images in a file for a profile (especially in two channel profiles). These images can also be offset in relation to each other. The **image_file** attributes describe the relationship of an image with the file it is contained in and with the scans recorded in other files.

Attribute	**Description**
name	unambiguous name, typically a URI, URL or filename, that identifies where the file can be located
identifier	if there are multiple images within one file, an identifier specific to the file format to identify the image

number	if there are multiple images in one or more files, where this image belongs in the series of images
x_origin	if there are multiple images, describes the offset of its x-axis relative to the other images
y_origin	if there are multiple images, describes the offset of its y-axis relative to the other images

channel_info

Describes the data that is common to all the feature_data for a particular image scan.

Attribute	**Description**
channel_name	an unambiguous name, within the **profile**, for this channel
channel_color	the color associated with the label, if relevant
additive_error	a measure of the uncertainties due to background subtraction
multiplicative_error	a measure of errors that could come from hybridization non-uniformities, fluctuations in the dye incorporation efficiency, scanner gain fluctuations, etc.
mean_signal	mean_signal over all the feature_data for this channel
raw_image_filename	for archive purposes, the location of the raw image file or files (the image_file provided with its profile is sometimes compressed, such as a jpg, from the original file)

reporter_data

The data for the profile can be expressed on three levels: the raw, feature level through the **feature_data** elements, **reporter_data** elements (combined from its features), and on the biosequence level through the **biosequence_data** elements. Any or all three might be in an export of a profile, depending on the source and depending on what data was requested.

The error model determines a number of values from the raw intensities. It can normalize across the scan and determine an error measure on a channel across the **features**. These values can then analyzed to produce values for the **channel_data** for the **reporter_data** and/or **biosequence_data**. For ratio-based data, a ratio at any level of data can be computed.

The **channel_data** is of type ANY. Currently there are distinct ways that values are calculated for averaged intensity-based **channel_data**, depending on the technology and the **error_model**. We feel it is better to allow the freedom for an organization to describe additional data through a nested element whose XML attributes can be discovered through the **GeneExpUtilities::dtdExtensions** IDL attribute.

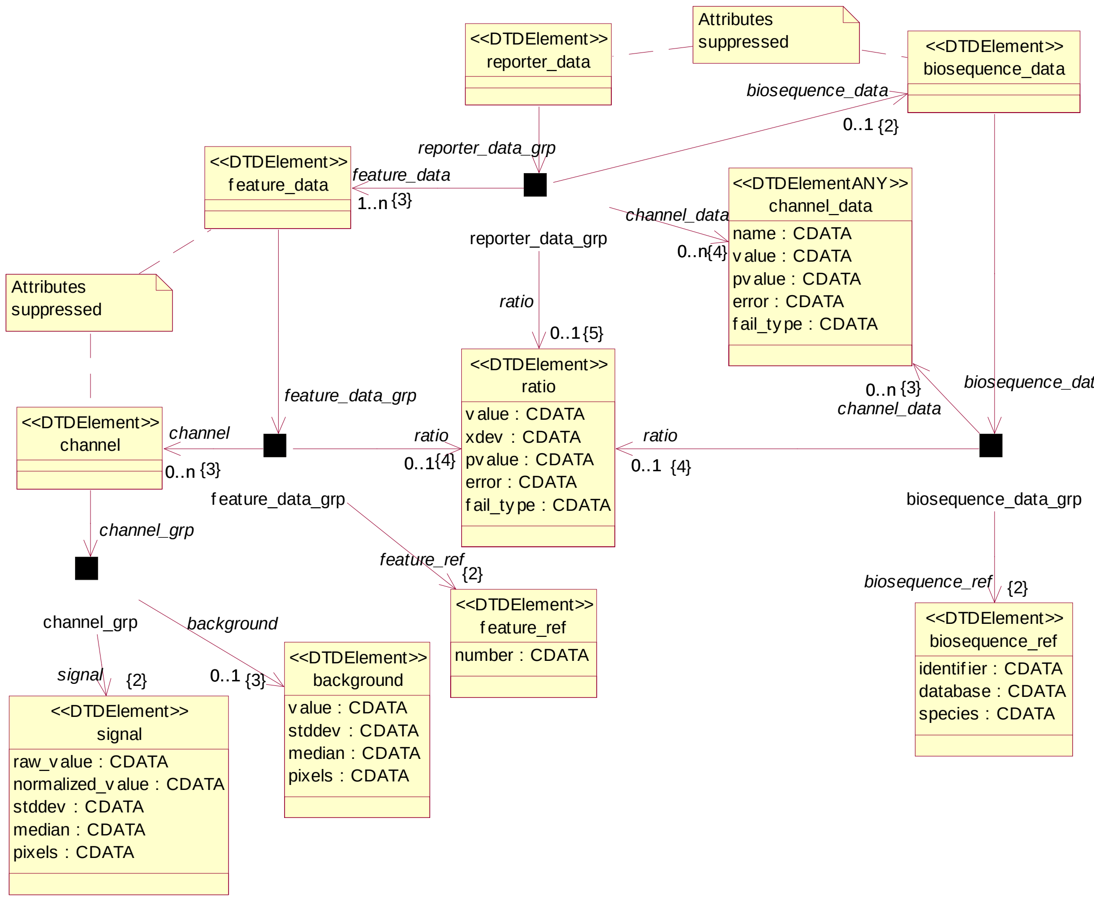

Figure 27: reporter_data and contained elements

```
<!ELEMENT reporter_data
   (other*,
    biosequence_data?,
    feature_data+,
    channel_data*,
    ratio?)
>
<!ATTLIST reporter_data
   channel_data_type ( LINEAR | LN | LOG2 |LOG10 | OTHER ) 'LINEAR'
   ratio_type ( LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER ) 'LOG10'
>

<!ELEMENT biosequence_data
   (other*,
    biosequence_ref,
    channel_data*,
    ratio?)
>
<!ATTLIST biosequence_data
   channel_data_type ( LINEAR | LN | LOG2 |LOG10 | OTHER ) 'LINEAR'
   ratio_type ( LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER ) 'LOG10'
>

<!ELEMENT channel_data
   ANY
>
```

```
<!ATTLIST channel_data
   name      CDATA #REQUIRED
   value     CDATA #IMPLIED
   pvalue    CDATA #IMPLIED
   error     CDATA #IMPLIED
   fail_type CDATA #IMPLIED
>

<!ELEMENT feature_data
   (other*,
    feature_ref,
    channel*,
    ratio?)
>
<!ATTLIST feature_data
   fail_type   CDATA #IMPLIED
   status_type CDATA #IMPLIED
   ratio_type ( LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER ) 'LOG10'

>

<!ELEMENT channel
   (other*,
    signal,
    background?)
>
<!ATTLIST channel
   name      CDATA  #REQUIRED
   fail_type CDATA  #IMPLIED
   data_type ( LINEAR | LN | LOG2 |LOG10 | OTHER ) 'LINEAR'
>

<!ELEMENT signal
   (other*)
>
<!ATTLIST signal
   raw_value        CDATA  #REQUIRED
   normalized_value CDATA  #IMPLIED
   stddev           CDATA  #IMPLIED
   median           CDATA  #IMPLIED
   pixels           CDATA  #IMPLIED
>

<!ELEMENT background
   (other*)
>
<!ATTLIST background
   value  CDATA  #REQUIRED
   stddev CDATA  #IMPLIED
   median CDATA  #IMPLIED
   pixels CDATA  #IMPLIED
>

<!ELEMENT ratio
   (other*)
>
<!ATTLIST ratio
   value     CDATA  #REQUIRED
   xdev      CDATA  #IMPLIED
   pvalue    CDATA  #IMPLIED
   error     CDATA  #IMPLIED
   fail_type CDATA  #IMPLIED
>
```

Attribute	Description
channel_data_type	specifies how the values of channel_data is to be interpreted: {(LINEAR | LN | LOG2 |LOG10 | OTHER }
ratio_type	specifies how the value for the ratio is to be interpreted: {LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER}

biosequence_data

Container of the **biosequence_ref** for the reporter and, optionally, for the statistical summarized data across all the reporters for this experimental profile. For single channel data, **channel_data** will be used, and, if there are two channels for the hybridization, **ratio** can be used.

Attribute	Description
channel_data_type	specifies how the values of channel_data is to be interpreted: {(LINEAR | LN | LOG2 |LOG10 | OTHER }
ratio_type	specifies how the value for the ratio is to be interpreted: {LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER}

channel_data

The single channel summary data for **biosequence** is calculated from the **channel_data** of its **reporters** or **features' channel** while the **reporter's** is calculated from its **features' channel** data.

The **error_model** determines if, or how, these values are calculated.

Attribute	Description
name	name that matches the appropriate summarized **channel_info** for this **channel_data**
value	the value obtained from the analysis of data from the **features** the **reporter** was place or the data from the **reporters** reporting on the same **biosequence**
pvalue	the probability that the value is not significantly different than zero
error	measure of error on the value
fail_type	a comma-delimited list from a FAIL ENUMERATION indicating errors

feature_data

The **feature_data** contains the elements that describe the data obtained from the scan and the statistical analysis of that data based on the error model used. It contains a reference back to the **pattern** information to its location and to the relevant **reporter** for the data. There will be zero to many **channels** for each feature and zero to one **ratio** in the profile. Each channel will contain the data for a scan and its **name** element will match the corresponding **channel_info** of the profile. It is expected, even though it is not required by the DTD, that implementers will provide a **channel** element with a given **name** attribute for every **feature_data** if there is one **feature_data** with a **channel** element with that **name**. The **fail_type** and **status_type** can be implemented as bitmask fields in a schema, with a lookup table that matches a human-readable enumeration value with its appropriate bit to set.

Attribute	Description
fail_type	a comma-delimited list from a FAIL ENUMERATION indicating errors
status_type	a comma-delimited list from a STATUS ENUMERATION indicating warnings or information
ratio_type	specifies how the value for the ratio is to be interpreted: {LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER}

channel

The container for the data from the scan and optional values derived by the **error_model** for the **feature**.

Attribute	Description
name	name that matches the appropriate summarized channel_info for this channel
fail_type	a comma-delimited list from a FAIL ENUMERATION indicating errors
data_type	specifies how the values of signal and backround are to be interpreted: {(LINEAR | LN | LOG2 |LOG10 | OTHER }

signal

The measurement of the intensity value at a feature location.

Attribute	Description
raw_value	intensity value as read by the scanner
normalized_value	raw_value adjusted by a normalization factor of the error model
stddev	standard deviation of the pixels' value
median	median of the pixels' value
pixels	number of pixels used for the integrated signal

background

The measurement of the intensity value at the background around the feature location.

Attribute	Description
value	intensity value as read by the scanner
stddev	standard deviation of the pixels' values
median	median of the pixels' values
pixels	number of pixels used for the integrated signal

ratio

Values and error values across two channels for a **feature**, for **features** across a **reporter**, or a set of **reporters** on the same **biosequence**.

At each level of calculation, there are different factors that may be used to obtain the values and errors for the set. For instance, across the two channels for **feature**, non-biological biases indicated by array-wide averages, positive controls and/or negative controls may be factored out.

Attribute	Description
value	normalized ratio of the channels
xdev	the number of standard deviations that the absolute value of value is away from zero
pvalue	the probability that the ratio (value) is not significantly different than zero
error	the ratio (value) divided by xdev
fail_type	a comma-delimited list from a FAIL ENUMERATION indicating errors

summary_data

Provides description of the information on the over all **profile**. This includes description of the image scan and statistics on the hybridization. Organizations vary considerably in the kind of statistical tracking of the hybridization process they perform so that many of the elements are of type ANY to provide an extensible mechanism.

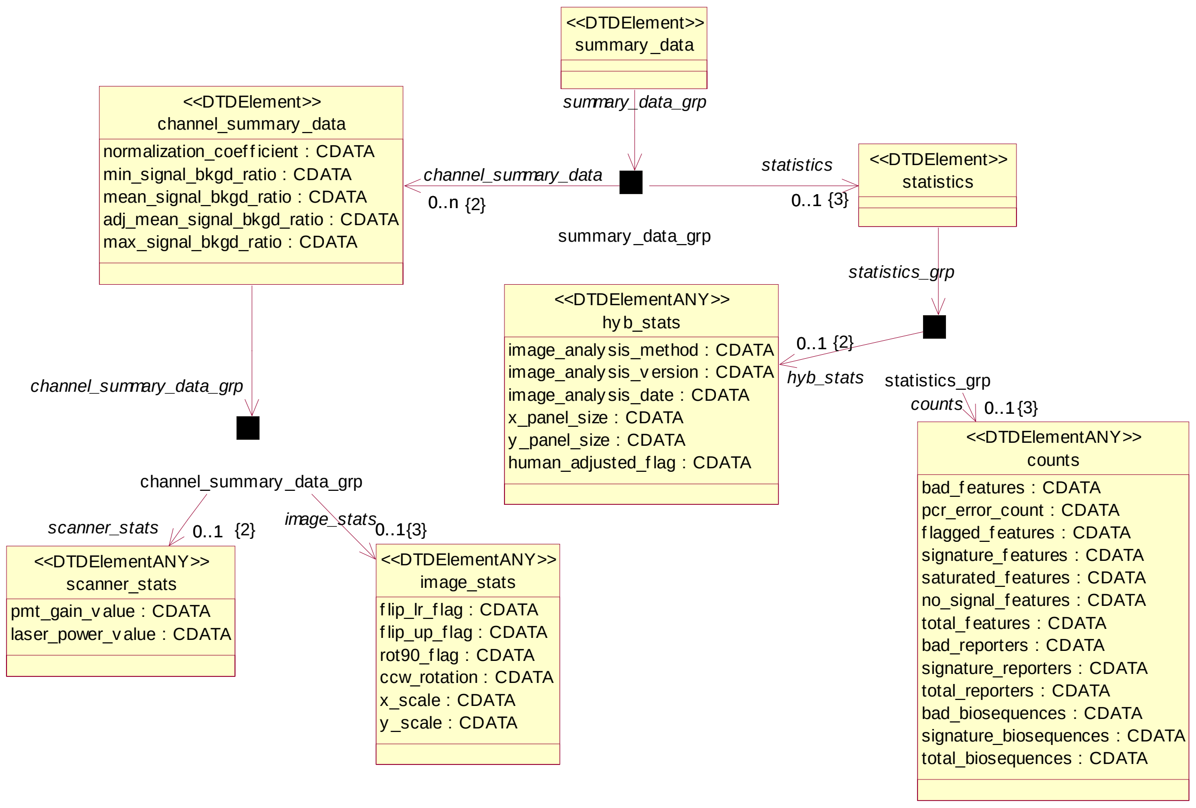

Figure 28: summary_data and its contained elements.

```
<!ELEMENT summary_data
   (other*,
    channel_summary_data*,
    statistics?)
>

<!ELEMENT channel_summary_data
   (other*,
    scanner_stats?,
    image_stats?)
>
<!ATTLIST channel_summary_data
   normalization_coefficient  CDATA #IMPLIED
   min_signal_bkgd_ratio      CDATA #IMPLIED
   mean_signal_bkgd_ratio     CDATA #IMPLIED
   adj_mean_signal_bkgd_ratio CDATA #IMPLIED
   max_signal_bkgd_ratio      CDATA #IMPLIED
>

<!ELEMENT scanner_stats
   ANY
>
<!ATTLIST scanner_stats
   pmt_gain_value    CDATA  #IMPLIED
   laser_power_value CDATA  #IMPLIED
>

<!ELEMENT image_stats
   ANY
>
```

```
<!ATTLIST image_stats
   flip_lr_flag CDATA  #IMPLIED
   flip_up_flag CDATA  #IMPLIED
   rot90_flag   CDATA  #IMPLIED
   ccw_rotation CDATA  #IMPLIED
   x_scale      CDATA  #IMPLIED
   y_scale      CDATA  #IMPLIED
>

<!ELEMENT statistics
   (other*,
    hyb_stats?,
    counts?)
>

<!ELEMENT hyb_stats
   ANY
>
<!ATTLIST hyb_stats
   image_analysis_method  CDATA  #IMPLIED
   image_analysis_version CDATA  #IMPLIED
   image_analysis_date    CDATA  #IMPLIED
   x_panel_size           CDATA  #IMPLIED
   y_panel_size           CDATA  #IMPLIED
   human_adjusted_flag    CDATA  #IMPLIED
>

<!ELEMENT counts
   ANY
>
<!ATTLIST counts
   bad_features        CDATA  #IMPLIED
   pcr_error_count     CDATA  #IMPLIED
   flagged_features    CDATA  #IMPLIED
   signature_features  CDATA  #IMPLIED
   saturated_features  CDATA  #IMPLIED
   no_signal_features  CDATA  #IMPLIED
   total_features      CDATA  #IMPLIED
   bad_reporters       CDATA  #IMPLIED
   signature_reporters CDATA  #IMPLIED
   total_reporters     CDATA  #IMPLIED
   bad_biosequences       CDATA  #IMPLIED
   signature_biosequences CDATA  #IMPLIED
   total_biosequences     CDATA  #IMPLIED
>
```

channel_summary_data

Provides summary information on a channel of the hybridization.

Attribute	**Description**
normalization_coefficient	value applied to the intensities to normalize them for statistical comparison
min_signal_bkgd_ratio	the minimum backround ratio of all the backround areas calculated
mean_signal_bkgd_ratio	the mean backround ratio of all the backround areas calculated
adj_mean_signal_bkgd_ratio	the adjusted mean backround ratio of all the backround areas calculated
max_signal_bkgd_ratio	the maximum backround ratio of all the backround areas calculated

scanner_stats

Provides information on the actual scan of the array.

Attribute	Description
pmt_gain_value	the adjusted gain to the photomultiplier tubes for the scan
laser_power_value	the power value of the laser for the scan

image_stats

Provides information on how the array and image were manipulated for the scan.

Attribute	Description
flip_lr_flag	indicates whether the image was flipped on the x-axis
flip_up_flag	indicates whether the image was flipped on its y-axis
rot90_flag	indicates whether the image was flipped on the diagonal axis
ccw_rotation	the amount, if any, the image was rotated counter-clockwise
x_scale	the amount the image was scaled on the x-axis
y_scale	the amount the image was scaled on the y-axis

statistics

Container for how the hybridization was performed and for the overall quality of the scan.

hyb_stats

Describes the methods used to analysis the hybridization.

Attribute	Description
image_analysis_method	image_analysis_method
image_analysis_version	the version of the analysis method
image_analysis_date	the date of the analysis
x_panel_size	the length along the x-axis that defines the granularity of background statistical analysis
y_panel_size	the length along the y-axis that defines the granularity of background statistical analysis
human_adjusted_flag	whether the statistical analysis has been adjusted by hand

counts

Counts on **features**, **reporters** and **biosequences**.

Attribute	Description
bad_features	number of features that failed (some bit set for fail_type)
pcr_error_count	an error in the production of the sequence was detected
signature_features	the number of features whose values are considered valid by the statistical analysis
saturated_features	number of features scanned whose intensity was above a threshold
no_signal_features	number of features scanned whose intensity was below a threshold
total_features	total number of features on the array
bad_reporters	number of considered failed
signature_reporters	the number of reporters whose values are considered valid
total_reporters	total number of reporters on the array
bad_biosequences	number of biosequences considered failed
signature_biosequences	the number of biosequences whose values are considered valid
total_biosequences	total number of biosequences on the array

4.1.14 Biosequence_cluster

Biosequence_cluster defines a group of biological sequences that are believed to be derived from the same gene. Expressed Sequence Tag (EST) sequences that derive from the same clone, partial overlapping cDNA sequences, and a full-length cDNA sequence, for example, might all be grouped together under a **biosequence_cluster** if all of these sequences are believed to represent the same gene or transcript. Resources such as UniGene (http://www.ncbi.nlm.nih.gov/UniGene) provide information on sequence accession numbers that are believed to be derived from the same gene. These biosequence_clusters are updated periodically as additional information becomes available from EST projects, genomic DNA sequencing projects, gene prediction efforts, and full-length cDNA sequencing projects.

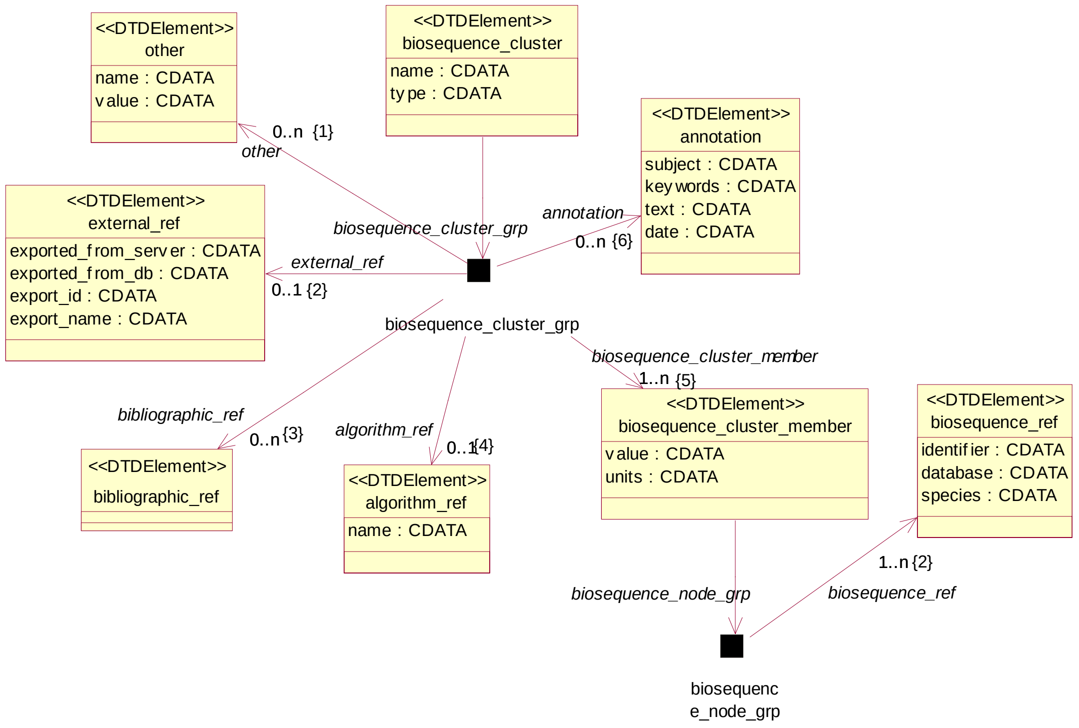

Figure 29: biosequence_cluster and contained elements.

```
<!ELEMENT biosequence_cluster
   (other*,
    external_ref?,
    bibliographic_ref*,
    algorithm_ref?,
    biosequence_cluster_member+,
    annotation*)
>
<!ATTLIST biosequence_cluster
   name CDATA #REQUIRED
   type CDATA #IMPLIED
>

<!ELEMENT biosequence_cluster_member
   (other*,
    biosequence_ref+)
>
<!ATTLIST biosequence_cluster_member
   value CDATA #REQUIRED
   units CDATA #IMPLIED
>
```

Attribute	Description
name	unambiguous identifier, within the source, for the cluster.
type	type of the cluster, for example UniGene.

biosequence_cluster_member

A member of the cluster. The logical value of this node will vary, depending on the algorithm used. For a flat clustering, it may simple be an identifier assigned to group together biosequences believed to belong to the same exon, or, for instance, when based on experimental data, may be a measurement of some correlation.

Attribute	Description
value	the value for this node, often a simple identifier
units	the units of the value, if relevant

4.1.15 Biosequence_set

A biosequence_set is a collection of sequences that is useful to view and/or analyze as a group. A biosequence_set could, for example, represent a group of sequences that are involved in a particular biochemical pathway or cellular process. Biosequence sets can also be used to pass hierarchical data structures such as the results of a cluster analysis on sequence behavior across a number of experiments, or an annotation hierarchy such as Enzyme Commission classification number (http://prowl.rockefeller.edu/enzymes/enzymes.htm).

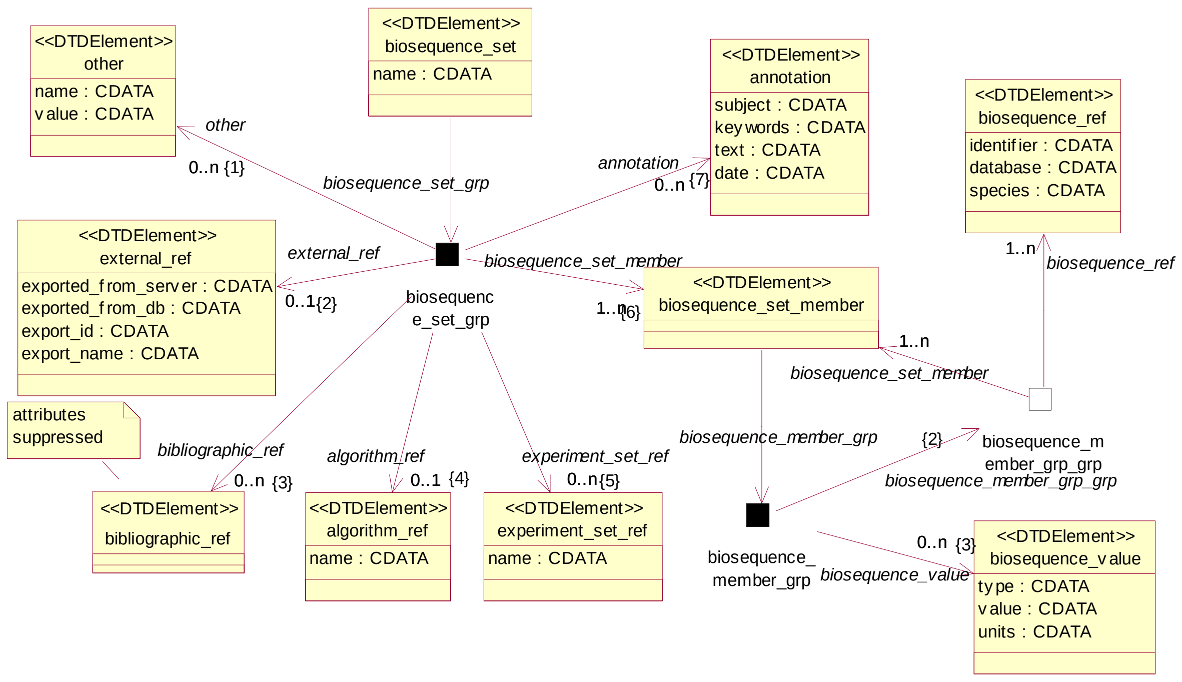

Figure 30: biosequence_set and contained elements.

```
<!ELEMENT biosequence_set
   (other*,
    external_ref?,
    bibliographic_ref*,
    algorithm_ref?,
    experiment_set_ref*,
    biosequence_set_member+,
    annotation*)
>
<!ATTLIST biosequence_set
   name CDATA #REQUIRED
>

<!ELEMENT biosequence_set_member
   (other*,
    (biosequence_set_member+ |
     biosequence_ref+),
    biosequence_value*)
>

<!ELEMENT biosequence_value
   (other*)
>
<!ATTLIST biosequence_member
   type  CDATA #REQUIRED
   value CDATA #REQUIRED
   units CDATA #IMPLIED
>
```

Attribute	**Description**
name	unambiguous name, within the source, for the **biosequence_set**

biosequence_set_member

Represents a node in the set. The **biosequence_set_member** then reference, if it is a leaf node, the **biosequences** that are grouped together in this node, by the criteria of the set. Otherwise a set of nested nodes further define the hierarchy of the set. The node can have zero or more **biosequence_values** associated with it.

biosequence_value

Represents one value associated with the containing **biosequence_set_member**.

Attribute	**Description**
type	the type of this node in the cluster
value	the value assigned to this node for the cluster
units	units of the value, if appropriate

4.1.16 Experiment_set

An experiment_set is a collection of experiments that is useful to view and/or analyze as a group. An experiment_set could, for example, represent a group of experiments that represents a drug titration series, for example, or a series of timepoints along a timecourse following a particular treatment. Experiment sets can also be used to pass hierarchical data structures such as the results of an experiment cluster analysis.

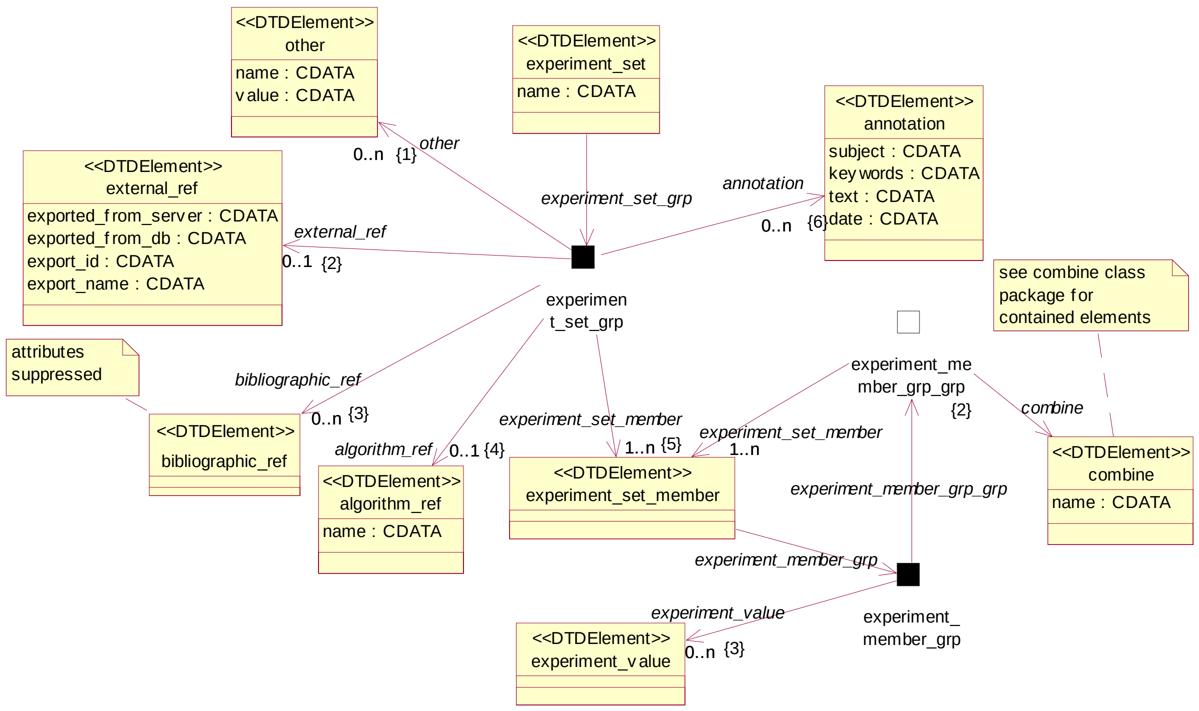

Figure 31: experiment_set and contained elements.

```
<!ELEMENT experiment_set
   (other*,
    external_ref?,
    bibliographic_ref*,
    algorithm_ref?,
    experiment_set_member+,
    annotation*)
>
<!ATTLIST experiment_set
   name CDATA #REQUIRED
>

<!ELEMENT experiment_set_member
   (other*,
    (experiment_set_member+ |
     combine),
    experiment_value*)
>

<!ELEMENT experiment_value
   (other*)
>
<!ATTLIST experiment_member
   type  CDATA #REQUIRED
   value CDATA #REQUIRED
   units CDATA #IMPLIED
>
```

Attribute	Description
name	unambiguous identifier, within the source, for the experiment set

experiment_set_member

Represents a node in the set. The **experiement_set_member** then contains, if it is a leaf node, a **combine** that groups together the **hybs** that belong together, by the criteria of the cluster. Otherwise, a set of nested nodes further define the hierarchy of the set. The node can have zero or more **experiment_values** associated with it.

experiment_value

Represents a value associated with the containing **experiment_set_member**.

Attribute	Description
type	the type of this node in the cluster
value	the value assigned to this node for the cluster
units	units of the value, if appropriate

experiment_set_ref

references an **experiment_set** by name.

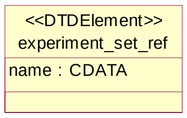

Figure 32: experiment_set_ref.

```
<!ELEMENT experiment_set_ref
   (other*)
>
<!ATTLIST experiment_set_ref
   name CDATA #REQUIRED
>
```

5 Applicability to Sage and Proteomics

5.1.1 Serial Analysis of Gene Expression (SAGE)

Gene expression analysis using Serial Analysis of Gene Expression (SAGE) technology involves the sequencing and subsequent counting of restriction enzyme fragments "SAGE Tags" that in most cases uniquely define a particular transcript. Data generated using this technology can be described using the GEML format. The signal/intensity element is used to hold counts of individual SAGE Tags sequenced in a given SAGE profile. The reporter element holds the sequence of each SAGE Tag. In most cases, there will be only one SAGE Tag per gene in a given SAGE profile. If multiple SAGE Tags are observed for a given gene, the biosequence_cluster element in GEML can be used to group together those SAGE Tags that are believed to all report upon the same gene.

5.1.2 Proteomics

Spots identified by 2-D gel analysis software such as Melanie3 (See www.genebio.com) represent distinct proteins and protein modifications present in a biological sample. These spots can be represented in the current version of GEML as biosequence entries. Both absolute (intensity-based) and relative (ratio-based) analyses of proteomics data from 2-D gel profiles are possible, the latter using software that allows users to select identical spots on two different 2-D gels and analyze ratios between signal intensities for spots that are in common between the two gels. These signal intensities and ratios can be accommodated within the present GEML under the signal/ratio and signal/intensity elements. The biosequence_cluster element can be used to group together spots that all represent different modifications of the same protein for downstream analysis.

5.1.3 Conclusion

While much of the information can be captured from these experiments, the actual performance of the experiment, for instance, is not captured well. Hybridization does not apply to proteomics, where the desire would be to describe the 2-D gel process.

There are also some newer technologies in proteomics, for instance, isotope-coded affinity tag peptide labelling and multi-dimensional protein identification technique, which vary further in that there is no true pattern. In these experiments, the results are obtained using mass spectrometry. Although there are many elements that can be still shared (**biosequence**, **preparation**, etc), DTDs, even with name-spacing, do not provide ease of sharing between DTDs.

With the evolution of specifying an XML format moving from DTD to the W3C XML Schema, it will become easier to share elements between multiple formats and define the new elements needed. Potentially, a future RFP could address this issue.

6 Glossary

Amino Acid
Any of a class of 20 small molecule building blocks that are combined to form proteins in living things (21 amino acids if selenocysteine is included). The sequence of amino acids in a protein and hence protein function are determined by the nucleotide sequence of its gene and the genetic code. The terms residue and amino acid are often used interchangeably.

Array
Array refers to the physical substrate to which biosequence reporters are attached to create features. In gene expression profiling experiments, arrays are hybridized with labeled sample and then scanned and analyzed to generate data.

Background
Background is the measured signal outside of a feature on an array. In many gene expression analysis methods, background subtraction is performed to correct measured signals for observed local and/or global background.

Biosequence
The ordered set of nucleotides in a DNA or RNA molecule, or the ordered set of amino acids in a protein.

Biosequence Cluster
A biosequence cluster refers to a set of biosequences that are believed to be derived from the same gene. Biosequences that are derived from the same cDNA clone or that have been "clustered" together by resources such as UniGene are grouped together into biosequence clusters for analysis. Biosequence cluster should not be confused with cluster analysis.

Biosequence Set
A biosequence_set is a collection of sequences that is useful to view and/or analyze as a group. A biosequence_set could, for example, represent a group of sequences that are involved in a particular biochemical pathway or cellular process. Biosequence sets can also be used to pass hierarchical data structures such as the results of a cluster analysis on sequence behavior across a number of experiments.

Cell Line
A cell line is a collection of cells propagated in culture. Cell lines are typically homogenous cell populations.

Channel
A channel is an intensity-based portion of an expression dataset that consists of the set of signal measurements across all features on an array for a particular labeled preparation used in a hybridization. In some cases, such as Cy3/Cy5 array hybridizations, multiple channels (one for each label used) may be combined in a single expression profile to create ratios.

Chromosome
One of the linear or sometimes circular DNA-containing bodies of viruses, prokaryotic organisms, and the cell nucleus of eukaryotic organisms that contain most or all of the genes of the individual

Cluster Analysis
Cluster Analysis is a multivariate analysis technique that seeks to organize information about variables so that relatively homogeneous groups, or "clusters," can be formed. The clusters formed with this family of methods should be highly

internally homogenous (members are similar to one another) and highly externally heterogenous (members are not like members of other clusters). In expression profiling, cluster analysis refers to the grouping of hybridizations, expression profiles, or combines across common sets of genes as well as grouping of features, reporters, biosequences, or biosequence clusters across sets of expression profiling experiments.

Compound
A Compound is a small molecule such as a drug. These small molecules can interact with cellular components to produce results that are therapeutic, toxic, or produce a desired experimental effect. Compounds are commonly used in Treatments of cellular samples in gene expression experiments.

Deletion
Refers to a sequence that varies from its target sequence by the removal of one or more nucleotides.

Detrending
Detrending refers to the removal or reduction of systematic biases from a data set.

DNA (deoxyribonucleic acid)
The molecule that encodes genetic information. DNA is a double-stranded polymer of nucleotides. The two strands are held together by hydrogen bonds between base pairs of nucleotides. The four nucleotides in DNA contain the bases: adenine (A), guanine (G), cytosine (C), and thymine (T). In nature, base pairs form only between A and T and between G and C; thus the base sequence of each single strand can be deduced from that of its partner.

Document Type Definition (DTD)
An instance of a Document Type Definition uses a formal grammar to specify the structure and permissible values of XML Documents that declare it as their DTD. It specifies what elements can follow each other and which can be nested. It also specifies the permissible attributes of an element.

Error Model
An error model is an algorithm that computes quality statistics such as P-values and error bars for each gene expression measurement. Error models are typically specific to a particular expression profiling technology.

Experiment Set
An experiment_set is a collection of experiments that is useful to view and/or analyze as a group. An experiment_set could, for example, represent a group of experiments that represents a drug titration series, for example, or a series of timepoints along a timecourse following a particular treatment. Experiment sets can also be used to pass hierarchical data structures such as the results of an experiment cluster analysis.

Expression
The conversion of the genetic instructions present in a DNA sequence into a unit of biological function in a living cell. Typically involves the process of transcription of a DNA sequence into an RNA sequence followed by translation of the RNA into protein. The RNA may be spliced before translation to remove introns.

Extensible Markup Language (XML)
The Extensible Markup Language (XML) is a Recommendation of the World Wide Web. XML is designed to be a universal format for structured documents and data on the Web that allows putting structured data in a text file.

Feature
A feature refers to a specific instance of a reporter positioned upon an array.

Gene
A gene refers to DNA which codes for a particular protein, or in certain cases a functional or structural RNA molecule. Genes may be inferred from the DNA sequence by way of a coding sequence.

Genome
The complete set of genetic information for a particular organism.

Label
Label refers to a reagent, system, or mechanism of differentiating one preparation used in a given expression profiling experiment from others used in the same experiment. Cy3 and Cy5 fluorescent labels, for example, are commonly used to distinguish baseline & experimental preparations in gene expression microarray hybridizations.

Hybridization
A hybridization is the act of treating an array with one or more labeled preparations under a specified set of conditions.

Mismatch
Refers to a sequence that varies from its target sequence by the replacement of one or more nucleotides.

Normalization
Mean signal intensity can vary dramatically among expression profiles or channels. Normalization is the procedure by which signal intensities from two or more expression profiles (or channels) are made directly comparable through application of an appropriate algorithm.

Nucleic Acid
A polymer of nucleotides. DNA and RNA are different classes of nucleic acids. May be double- or single-stranded.

Nucleotide
A subunit of DNA or RNA consisting of a nitrogenous base (adenine, guanine, thymine, or cytosine in DNA; adenine, guanine, uracil, or cytosine in RNA), a phosphate molecule, and a sugar molecule (deoxyribose in DNA and ribose in RNA).

Pattern
A pattern is the layout or blueprint of one or more Arrays.

Preparation
A preparation is an extract from a biological source or sample such as a tissue or cultured cells. It could consist of purified protein, RNA, DNA, or other cellular material. For gene expression experiments, mRNA or total RNA preparations are labeled and then hybridized to arrays.

Probe
In some organizations, probe is used as a synonym for Feature (see Feature above). Probe is also occasionally used to refer to the labeled sample used to hybridize to an array.

Profile
A profile, short for expression profile, is a data set extracted from an expression experiment.

Protein
A biological molecule which consists of many amino acids chained together by peptide bonds. The sequence of amino acids in a protein is determined by the sequence of nucleotides in a DNA molecule. Proteins perform most of the enzymatic and structural roles within living cells.

Ratio
A ratio refers to a normalized signal intensity generated from one feature (or reporter, biosequence, or biosequence cluster) in a given channel divided by a normalized signal intensity generated by the same feature (or reporter, biosequence, or biosequence cluster) in another channel. The channels compared are typically baseline versus experimental, e.g., normal vs. diseased or untreated vs. treated.

Reporter
Description of the material that is placed on a feature to represent a biological sequence. A reporter, in some technologies, can vary from the biosequence it represents. These variations are usually a mismatch or a deletion in the nucleotide sequence.

RNA (ribonucleic acid)
A class of nucleic acids that consist of nucleotides containing the bases: adenine (A), guanine (G), cytosine (C), and uracil (U). An RNA molecule is typically single-stranded and can pair with DNA (where U pairs with A) or with another RNA molecule. RNA nucleotides are chemically distinct from DNA nucleotides and enable RNA molecules to have more complex structural and functional roles within a living cell.

Sample
In this document sample refers to the biological source (typically a tissue, cell line, or whole organism) of one or more Preparations.

Sequence
The ordered set of nucleotides in a DNA or RNA molecule, or the ordered set of amino acids in a protein. Synonym for Biosequence.

Signal
Signal refers to an experimental measurement of expression level.

Solvent
A substance for dissolving or dispersing one or more other substances, typically a compound(s), used in a treatment.

Species
Common designation, such as 'A. Thaliana' or 'E. coli', from which the sample was obtained, based on NCBI's Taxonomy.

Spot
See feature.

Tissue
A tissue is a primary cellular sample isolated from a biological source organism that contains an enriched subset of one or more cell types from that organism, e.g. cardiac muscle or skeletal muscle.

Treatment
A treatment is the experimental manipulation of a sample such as a cell culture, tissue, or organism prior to extraction of a preparation.

XML Attribute
Attributes are name value pairs that are associated with an element type. They

follow the element type name inside the start tag. They can be thought of as the 'adjectives' of XML.

XML Element
The container for content in XML, delimited by a start tag and an end tag. That content may be character data, other elements, and/or other markup. They may be thought of as the 'nouns' of XML.

XML Tag
A start tag is an element type name enclosed in angle brackets which opens an element. An end tag ends the content of an element, comprised of an angle slash and then the element type name, all enclosed by angle brackets. Every start tag must have a corresponding end tag.

7 References

Object Management Group. 2000. Bibliographic Query Service RFP response. OMG Document lifesci/00-09-14.

Object Management Group. 1999. Biomolecular Sequence Analysis RFP response. OMG Document lifesci/99-10-01.

Object Management Group. 1998. The Common Object Request Broker: Architecture and Specification, v2.2. OMG Document formal/98-07-01.

Object Management Group. 1998. CORBAservices: Common Object Services Specification. OMG Document formal/98-12-09.

Object Management Group. 1998. CORBAservices: Common Object Services IDL. OMG Document formal/98-10-53.

Object Management Group. 1998. CORBA v2.3a - Core final revision. OMG PC Document ptc/98-12-04.

Object Management Group. 2000. Gene Expression RFP. OMG Document lifesci/00-03-09.

Object Management Group. 1998. Interoperable Naming Service. OMG Document orbos/98-10-11.

Object Management Group. 1998. Joint Revised Objects by Value Submission – with Errata. OMG TC Document orbos/98-01-18.

Object Management Group. 1998. OMG IDL Style Guide. OMG Document ab/98-06-03.

Anderson, Richard, et al. 2000. Professional XML. Wrox Press Ltd. ISBN: 1-861003-11-0.

Apparao, Victor, et al, eds. 1998. Document Object Model (DOM) Level 1 Specification, W3C Recommendation. 1 October 1998.

Bairoch, Amos, et al. 1997. The Swiss-Prot Protein Sequence Data Bank User Manual. Release 35; November 1997.

Daniel, Ron Jr., Steve Rose, and Eve Maler, eds. 2000. XML Pointer Language (XPointer) Version 1.0, W3C Candidate Recommendation. 7 June 2000.

Gamma, Erich, Richard Helm, Ralph Johnson, and John Vlissides. 1995. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. ISBN: 0-201-63361-2.

Haynes, Paul A., John R. Yeats III. 2000. Proteome profiling—pitfalls and progress. Yeast, Vol. 17, 81-87, 2000.

Hughes, Timothy R., et al. 2000. Functional Discovery via a Compendium of Expression Profiles. Cell, Vol. 102, 109-126. July 7, 2000.

International Organization for Standardization. 1988. International Standard for representation of dates and times. http://www.iso.ch/markete/8601.pdf

Megginson, David, Peter Murray-Rust, Tim Bray, XML-DEV community. 1998. SAX 1.0: The Simple API for XML. http://www.megginson.com/SAX/SAX1/

National Center for Biotechnology Information, et al. 1997. The DDJB/EMBL/GenBank Feature

Table: Definitions. Version 2.0. December 15, 1997.

NCBI Taxonomy database, http://www.ncbi.nlm.nih.gov/Taxonomy/tax.html.

Wolf , Misha, and Charles Wicksteed.   1997. Date and Time Formats. http://www.w3.org/TR/NOTE-datetime

World Wide Web Consortium. 2000. XML Schema. http://www.w3.org/XML/Schema.html

8 Complete IDL Specification

8.1 File: DsLSRGeneExpQuery.idl

```
///File: DsLSRGeneExpQuery.idl
//

#ifndef _DS_LSR_GENE_EXP_QUERY_IDL_
#define _DS_LSR_GENE_EXP_QUERY_IDL_

#pragma prefix "omg.org"

#include <CosQuery.idl>
#include <CosLifeCycle.idl>
#include <CosPropertyService.idl>
#include <DsLSRControlledVocabularies.idl>

module DsLSRGeneExpQuery
{

    // typedefs and struct for parameters, return types and properties.
   typedef string XMLString;
    typedef string URL;
    typedef sequence<URL> URLList;

   typedef string ExtendedElement;
   typedef string DTDString;
   struct ExtensionDTD
   {
      ExtendedElement extElement;
      DTDString       extDTD;
   };
   typedef sequence<ExtensionDTD> ExtensionDTDList;

   typedef string SpeciesString;
   typedef string SequenceSource;
   struct SpeciesSource
   {
      SpeciesString  species;
      SequenceSource seqSource;
   };
   typedef sequence<SpeciesSource> SpeciesSourceList;

   typedef string ElementName;
   typedef sequence<ElementName> ElementNameList;

   // AttributePath recommended to use CosNaming, so
pattern/reporter/biosequence_ref.identifier
   typedef string AttributePath;
   typedef string AttributeValue;
   struct GE_NVPair
   {
      AttributePath  attrPath;
      AttributeValue attrValue;
   };
   typedef sequence<GE_NVPair> GE_Predicates;

   // shorthands for imported types for controlled vocabularies
    typedef DsLSRControlledVocabularies::VocabularyList       VocabularyList;
    typedef DsLSRControlledVocabularies::VocabularyString     VocabularyString;
    typedef DsLSRControlledVocabularies::VocabularyStringList VocabularyStringList;

   exception LimitExceeded { long maxLength; };
   exception InvalidLocation {};

   interface GeneExpQuery : CosQuery::QueryEvaluator, CosLifeCycle::LifeCycleObject
   {
      unsigned long num_experiment_sets();
```

```
       unsigned long num_biosequence_sets();
       unsigned long num_biosequence_clusters();

       readonly attribute DsLSRControlledVocabularies::VocabularyFinder voc_finder;

       XMLString find( in ElementName element, in GE_Predicates include, in GE_Predicates
exclude,
                       in boolean detail)
                  raises(CosQuery::QueryInvalid,LimitExceeded);

       XMLString export( in ElementName element, in ElementNameList includeElements,
                         in GE_Predicates include, in GE_Predicates exclude)
                  raises(CosQuery::QueryInvalid,LimitExceeded);

       any export_to_location( in ElementName element, in ElementNameList includeElements,
                               in GE_Predicates include, in GE_Predicates exclude,
                              in URL location, in boolean separateFiles)
                  raises(CosQuery::QueryInvalid,InvalidLocation);

       CosQuery::QueryStatus export_completed(in any cookie, out float progress)
                  raises(CosQuery::QueryProcessingError);
   };

   interface GeneExpUtilities
   {
       // constants for the top DTD sub-vocabularies
       const ElementName BIOSEQUENCE    = "biosequence";
       const ElementName SAMPLE         = "sample";
       const ElementName PATTERN        = "pattern";
       const ElementName PRINTING       = "printing";
       const ElementName COMPOUND       = "compound";
       const ElementName SOLVENT        = "solvent";
       const ElementName PREP           = "prep";
       const ElementName HYB            = "hyb";
       const ElementName COMBINE        = "combine";
       const ElementName PROFILE        = "profile";
       const ElementName EXPERIMENT_SET = "experiment_set";
       const ElementName BIOSEQUENCE_CLUSTER = "biosequence_cluster";

       // constants for DTD-ANY type elements that allow extensible elements
       const ExtendedElement SPECIES_DATA  = "species_data";
       const ExtendedElement REPORTER_DESC = "reporter_desc";
       const ExtendedElement CHANNEL_DATA  = "channel_data";
       const ExtendedElement SCANNER_STATS = "scanner_stats";
       const ExtendedElement IMAGE_STATS   = "image_stats";
       const ExtendedElement HYB_STATS     = "hyb_stats";
       const ExtendedElement COUNTS        = "counts";

       readonly attribute ElementNameList   permittedFindElements;
       readonly attribute long              xmlStringLimit;
       readonly attribute URLList           locations;
       readonly attribute ExtensionDTDList  dtdExtensions;
       readonly attribute SpeciesSourceList sequenceSources;

       readonly attribute CosPropertyService::Properties queryProperties;
   };

};

#endif // _DS_LSR_GENE_EXP_QUERY_IDL_
```

9 Domain Model DTD

9.1 File: DsLSR_GEML.dtd

```
<!ELEMENT project
    (other*,
     (biosequence |
      pattern     |
      printing    |
      sample      |
      compound    |
      solvent     |
      prep        |
      hyb         |
      combine     |
      profile     |
      biosequence_cluster |
      biosequence_set     |
      experiment_set)+,
     annotation*)
>
<!ATTLIST project
    name         CDATA  #IMPLIED
    id           CDATA  #IMPLIED
    date         CDATA  #IMPLIED
    by           CDATA  #IMPLIED
    organization CDATA  #IMPLIED
>

<!ELEMENT other
    (other*)
>
<!ATTLIST other
    name CDATA #REQUIRED
    value CDATA #REQUIRED
>

<!ELEMENT annotation
    (other*,
     annotation*)
>
<!ATTLIST annotation
    subject  CDATA  #IMPLIED
    keywords CDATA  #IMPLIED
    text     CDATA  #IMPLIED
    date     CDATA  #IMPLIED
>

<!ELEMENT external_ref
    (other*)
>
<!ATTLIST external_ref
    exported_from_server CDATA #IMPLIED
    exported_from_db     CDATA #IMPLIED
    export_id            CDATA #IMPLIED
    export_name          CDATA #IMPLIED
>

<!ELEMENT algorithm_ref
    (other*,
     algorithm_ref*,
     annotation*)
>
<!ATTLIST algorithm_ref
    name CDATA #REQUIRED
>

<!ELEMENT bibliographic_ref
```

```
    (other*,
     (book    |
      article |
      patent)?,
     provider*)
>
<!ATTLIST bibliographic_ref
    type (book | article | patent | web_resource | thesis |
         proceeding | tech_report | other) #REQUIRED
    title      CDATA #IMPLIED
    identifier CDATA #IMPLIED
    date       CDATA #IMPLIED
    subject    CDATA #IMPLIED
    url        CDATA #IMPLIED
>

<!ELEMENT book (other*)
>
<!ATTLIST book
    isbn    CDATA #IMPLIED
    volume  CDATA #IMPLIED
    edition CDATA #IMPLIED
>

<!ELEMENT article
   (other*,
    (bibliographic_ref |
     journal))
>
<!ATTLIST article
   type (book | journal) #REQUIRED
    first_page CDATA #IMPLIED
    last_page  CDATA #IMPLIED
    volume     CDATA #IMPLIED
    issue      CDATA #IMPLIED
>

<!ELEMENT patent (other*)
>
<!ATTLIST patent
    doc_number CDATA #IMPLIED
    doc_office CDATA #IMPLIED
    doc_type   CDATA #IMPLIED
    applicant  CDATA #IMPLIED
>

<!ELEMENT provider
    (other*,
     (person |
      journal)?)
>
<!ATTLIST provider
    name CDATA #IMPLIED
    type (person | organization | service | journal | other) #REQUIRED
>

<!ELEMENT person
    (other*)
>
<!ATTLIST person
    surname        CDATA #IMPLIED
    first_name     CDATA #IMPLIED
    mid_initials   CDATA #IMPLIED
    email          CDATA #IMPLIED
    postal_address CDATA #IMPLIED
    affiliation    CDATA #IMPLIED
>

<!ELEMENT journal
    (other*,
     provider?)
>
```

```
<!ATTLIST journal
    name CDATA #IMPLIED
    issn CDATA #IMPLIED
    abbr CDATA #IMPLIED
>

<!ELEMENT biosequence
    (other*,
     bibliographic_ref*,
     accession*,
     alias*,
     annotation*)
>
<!ATTLIST biosequence
    primary_name CDATA #REQUIRED
    control_type CDATA 'false'
    species      CDATA #REQUIRED
    sequenceDB   CDATA #REQUIRED
    chromosome   CDATA #IMPLIED
    map_position CDATA #IMPLIED
    description  CDATA #IMPLIED
>

<!ELEMENT biosequence_ref
    (other*)
>
<!ATTLIST biosequence_ref
    identifier   CDATA #REQUIRED
    database     CDATA #IMPLIED
    species      CDATA #REQUIRED
>

<!ELEMENT accession
    (other*)
>
<!ATTLIST accession
    database   CDATA #REQUIRED
    identifier CDATA #REQUIRED
>

<!ELEMENT alias   (other*)>
<!ATTLIST alias
    name CDATA #REQUIRED
>

<!ELEMENT pattern
    (other*,
     external_ref?,
     bibliographic_ref*,
     grid_layout?,
     reporter+,
     annotation*)
>
<!ATTLIST pattern
    name             CDATA #REQUIRED
    type             CDATA #REQUIRED
    species_database CDATA #REQUIRED
    description      CDATA #IMPLIED
    access           CDATA #IMPLIED
    owner            CDATA #IMPLIED
>

<!ELEMENT pattern_ref
    (other*)
>
<!ATTLIST pattern_ref
    name             CDATA #REQUIRED
>

<!ELEMENT reporter
    (other*,
     biosequence_ref,
     feature+,
     (oligo |
      cdna  |
      reporter_desc)?,
     mismatch_info*,
     deletion_info*)
>
<!ATTLIST reporter
    name             CDATA #REQUIRED
    control_type     CDATA "false"
    fail_type        CDATA "false"
    active_sequence  CDATA #IMPLIED
    start_coord      CDATA #IMPLIED
    deletion         CDATA #IMPLIED
    mismatch_count   CDATA #IMPLIED
    description      CDATA #IMPLIED
>

<!ELEMENT oligo
    (other*)
>
<!ATTLIST oligo
    linker_sequence CDATA #IMPLIED
>

<!ELEMENT cdna
    (other*)
>
<!ATTLIST cdna
    plate_barcode    CDATA #IMPLIED
    plate_x          CDATA #IMPLIED
    plate_y          CDATA #IMPLIED
    primer1_sequence CDATA #IMPLIED
    primer2_sequence CDATA #IMPLIED
>

<!ELEMENT reporter_desc
    ANY
>

<!ELEMENT feature
    (other*,
     position?,
     pen?,
     mismatch_info*,
     deletion_info*)
>
<!ATTLIST feature
    number            CDATA #IMPLIED
    ctrl_for_feat_num CDATA #IMPLIED
    deletion          CDATA #IMPLIED
    mismatch_count    CDATA #IMPLIED
    control_type      CDATA #IMPLIED
>

<!ELEMENT feature_ref
    (other*,
     position?)
>
<!ATTLIST feature_ref
    number            CDATA #IMPLIED
>

<!ELEMENT position
    (other*)
>
<!ATTLIST position
    x     CDATA #REQUIRED
    y     CDATA #REQUIRED
    units CDATA #IMPLIED
>
```

```
<!ELEMENT pen
    (other*)
>
<!ATTLIST pen
    x     CDATA #REQUIRED
    y     CDATA #REQUIRED
    units CDATA #IMPLIED
>

<!ELEMENT mismatch_info
    (other*)
>
<!ATTLIST mismatch_info
    start_coord     CDATA #REQUIRED
    sequence        CDATA #REQUIRED
    replaced_length CDATA #REQUIRED
>

<!ELEMENT deletion_info
    (other*)
>
<!ATTLIST deletion_info
    start_coord CDATA #REQUIRED
    length      CDATA #REQUIRED
>

<!ELEMENT grid_layout
    (other*,
     feature_size?)
>
<!ATTLIST grid_layout
    feature_spacing_x CDATA #IMPLIED
    feature_spacing_y CDATA #IMPLIED
    feature_count_x   CDATA #IMPLIED
    feature_count_y   CDATA #IMPLIED
>

<!ELEMENT feature_size
    (other*)
>
<!ATTLIST feature_size
    x_length CDATA #REQUIRED
    y_length CDATA #REQUIRED
    shape    CDATA #IMPLIED
    units    CDATA 'microns'
>

<!ELEMENT printing
    (other*,
     chip+)
>
<!ATTLIST printing
    date             CDATA #IMPLIED
    printer          CDATA #IMPLIED
    type             CDATA #IMPLIED
    run_description  CDATA #IMPLIED
    prepared_by_org  CDATA #IMPLIED
    prepared_at_site CDATA #IMPLIED
    prepared_by      CDATA #IMPLIED
>

<!ELEMENT chip
    (other*,
     pattern_ref,
     printing_rep*)
>
<!ATTLIST chip
    barcode          CDATA #REQUIRED
    prepared_for_org CDATA #IMPLIED
    prepared_for     CDATA #IMPLIED
>
```

```
<!ELEMENT printing_rep
    (other*,
     feature_ref)
>
<!ATTLIST printing_rep
    status CDATA #REQUIRED
>

<!ELEMENT sample
    (other*,
     external_ref?,
     bibliographic_ref*,
     parent_sample?,
     species_data,
     annotation*)
>
<!ATTLIST sample
    external_source CDATA  #IMPLIED
    source_number   CDATA  #IMPLIED
    name            CDATA  #REQUIRED
    organism        CDATA  #REQUIRED
    sample_type (totalRNA | mRNA | protein | other) #REQUIRED
    organ           CDATA  #IMPLIED
    tissue          CDATA  #IMPLIED
    sex (male | female | unknown) #IMPLIED
    age             CDATA  #IMPLIED
    disease_state   CDATA  #IMPLIED
    culture_type (primary | established | tissue)  #IMPLIED
    culture_description CDATA  #IMPLIED
>

<!ELEMENT parent_sample
    (other*,
     sample_ref)
>

<!ELEMENT sample_ref
    (other*)
>
<!ATTLIST sample_ref
    source_number CDATA  #REQUIRED
>

<!ELEMENT species_data
    ANY
>
<!ATTLIST species_data
    species CDATA #REQUIRED
>

<!ELEMENT compound
    (other*,
     external_ref?,
     annotation*)
>
<!ATTLIST compound
    code                      CDATA  #REQUIRED
    name                      CDATA  #REQUIRED
    molregno                  CDATA  #IMPLIED
    description               CDATA  #IMPLIED
    location                  CDATA  #IMPLIED
    molecular_weight          CDATA  #IMPLIED
    iupac_name                CDATA  #IMPLIED
    cas_number                CDATA  #IMPLIED
>

<!ELEMENT compound_ref
    (other*)
>
<!ATTLIST compound_ref
    code                      CDATA  #REQUIRED
>
```

```
<!ELEMENT solvent
    (other*,
     external_ref?,
     annotation*)
>
<!ATTLIST solvent
    name                     CDATA  #REQUIRED
    description              CDATA  #IMPLIED
>

<!ELEMENT solvent_ref
    (other*)
>
<!ATTLIST solvent_ref
    name CDATA  #REQUIRED
>

<!ELEMENT prep
    (other*,
     external_ref?,
     treatment+,
     annotation*)
>
<!ATTLIST prep
    code                   CDATA  #REQUIRED
    name                   CDATA  #IMPLIED
    date                   CDATA  #IMPLIED
    prepared_by            CDATA  #IMPLIED
    type                   CDATA  #IMPLIED
    method                 CDATA  #IMPLIED
    description            CDATA  #IMPLIED
    cells_per_ml           CDATA  #IMPLIED
    prep_ug              CDATA  #IMPLIED
    prep_conc_ug_ul      CDATA  #IMPLIED
    location               CDATA  #IMPLIED
    reference_text         CDATA  #IMPLIED
    scientist              CDATA  #IMPLIED
>

<!ELEMENT prep_ref (other*)>
<!ATTLIST prep_ref
    code CDATA  #REQUIRED
>

<!ELEMENT treatment
    (other*,
     (treatment |
      sample_ref)*,
     treatment_compound*,
     treatment_solvent?,
     annotation*)
>
<!ATTLIST treatment
    name              CDATA  #REQUIRED
    step_number       CDATA  #REQUIRED
    media             CDATA  #IMPLIED
    volume            CDATA  #IMPLIED
    volume_units      CDATA  'ml'
    temperature       CDATA  #IMPLIED
    temperature_units CDATA  'C'
    duration          CDATA  #IMPLIED
    wash_before       (true | false) 'false'
    description       CDATA #IMPLIED
>

<!ELEMENT treatment_compound
    (other*,
     compound_ref,
     solvent_ref)
>
<!ATTLIST treatment_compound
```

```
    concentration           CDATA  #REQUIRED
    concentration_units     CDATA  'mg/ml'
    lot_code                CDATA  #IMPLIED
    lot_concentration       CDATA  #IMPLIED
    lot_concentration_units CDATA  'mg/ml'
>

<!ELEMENT treatment_solvent
    (other*,
     solvent_ref)
>
<!ATTLIST treatment_solvent
    concentration           CDATA  #REQUIRED
    concentration_units     CDATA  'mg/ml'
>

<!ELEMENT hyb
    (other*,
     external_ref?,
     labeled_prep*,
     annotation*)
>
<!ATTLIST hyb
    name                 CDATA  #REQUIRED
    chip_barcode         CDATA  #REQUIRED
    number               CDATA  "1"
    control              CDATA  'false'
    channel_reversal     CDATA  'false'
    station              CDATA  'manual'
    date                 CDATA  #IMPLIED
    method               CDATA  #IMPLIED
    performed_by         CDATA  #IMPLIED
    prepared_by_org      CDATA #IMPLIED
    labeled_by           CDATA  #IMPLIED
    labeling_method      CDATA  #IMPLIED
    amplification_method CDATA  #IMPLIED
    reference_text       CDATA  #IMPLIED
    description          CDATA  #IMPLIED
    scientist            CDATA  #IMPLIED
>

<!ELEMENT hyb_ref
    (other*)
>
<!ATTLIST hyb_ref
    chip_barcode CDATA  #REQUIRED
    number       CDATA  #IMPLIED
>

<!ELEMENT labeled_prep
    (prep_ref)
>
<!ATTLIST labeled_prep
    label     CDATA  #IMPLIED
    used_ug   CDATA  #IMPLIED
>

<!ELEMENT combine
    (other*,
     normalization?,
     baseline?,
     hyb_ref+)
>
<!ATTLIST combine
    name CDATA #IMPLIED
>

<!ELEMENT normalization
    (other*,
     algorithm_ref,
     biosequence_ref*)
>
```

```
<!ATTLIST normalization
    name CDATA #REQUIRED
>

<!ELEMENT baseline
    (other*,
     hyb_ref+)
>

<!ELEMENT profile
    (other*,
     external_ref?,
     bibliographic_ref*,
     error_model?,
     hyb_ref?,
     image_file*,
     channel_info*,
     summary_data?,
     reporter_data*,
     biosequence_data*,
     annotation*)
>
<!ATTLIST profile
    name            CDATA  #IMPLIED
    type            CDATA  #IMPLIED
    barcode         CDATA  #IMPLIED
    access          CDATA  #IMPLIED
    owner           CDATA  #IMPLIED
    scanner         CDATA  #IMPLIED
    number          CDATA  #IMPLIED
    performed_date  CDATA  #IMPLIED
    performed_by    CDATA  #IMPLIED
    analyzed_date   CDATA  #IMPLIED
    analyzed_by     CDATA  #IMPLIED
    fail_type       CDATA  #IMPLIED
    control_flag    CDATA  'false'
    algorithm_state (COMPLETE | CALCULATE | NONE) 'COMPLETE'
    profile_quality CDATA  'Pending QC'
    qc_by           CDATA  #IMPLIED
>

<!ELEMENT error_model
    (other*,
     algorithm_ref)
>
<!ATTLIST error_model
    name CDATA #REQUIRED
>

<!ELEMENT image_file
    (other*)
>
<!ATTLIST image_file
    name       CDATA  #REQUIRED
    identifier CDATA  #IMPLIED
    number     CDATA  #IMPLIED
    x_origin   CDATA  #IMPLIED
    y_origin   CDATA  #IMPLIED
>

<!ELEMENT channel_info  (other*)>
<!ATTLIST channel_info
    channel_name         CDATA  #REQUIRED
    color_name           CDATA  #REQUIRED
    additive_error       CDATA  #IMPLIED
    multiplicative_error CDATA  #IMPLIED
    mean_signal          CDATA  #IMPLIED
    raw_image_filename   CDATA  #IMPLIED
>

<!ELEMENT summary_data
    (other*,
     channel_summary_data*,
     statistics?)
>

<!ELEMENT channel_summary_data
    (other*,
     scanner_stats?,
     image_stats?)
>
<!ATTLIST channel_summary_data
    normalization_coefficient  CDATA #IMPLIED
    min_signal_bkgd_ratio      CDATA #IMPLIED
    mean_signal_bkgd_ratio     CDATA #IMPLIED
    adj_mean_signal_bkgd_ratio CDATA #IMPLIED
    max_signal_bkgd_ratio      CDATA #IMPLIED
>

<!ELEMENT scanner_stats
    ANY
>
<!ATTLIST scanner_stats
    pmt_gain_value    CDATA  #IMPLIED
    laser_power_value CDATA  #IMPLIED
>

<!ELEMENT image_stats
    ANY
>
<!ATTLIST image_stats
    flip_lr_flag CDATA  #IMPLIED
    flip_up_flag CDATA  #IMPLIED
    rot90_flag   CDATA  #IMPLIED
    ccw_rotation CDATA  #IMPLIED
    x_scale      CDATA  #IMPLIED
    y_scale      CDATA  #IMPLIED
>

<!ELEMENT statistics
    (other*,
     hyb_stats?,
     counts?)
>

<!ELEMENT hyb_stats
    ANY
>
<!ATTLIST hyb_stats
    image_analysis_method  CDATA  #IMPLIED
    image_analysis_version CDATA  #IMPLIED
    image_analysis_date    CDATA  #IMPLIED
    x_panel_size           CDATA  #IMPLIED
    y_panel_size           CDATA  #IMPLIED
    human_adjusted_flag    CDATA  #IMPLIED
>

<!ELEMENT counts
    ANY
>
<!ATTLIST counts
    bad_features        CDATA  #IMPLIED
    pcr_error_count     CDATA  #IMPLIED
    flagged_features    CDATA  #IMPLIED
    signature_features  CDATA  #IMPLIED
    saturated_features  CDATA  #IMPLIED
    no_signal_features  CDATA  #IMPLIED
    total_features      CDATA  #IMPLIED
    bad_reporters       CDATA  #IMPLIED
    signature_reporters CDATA  #IMPLIED
    total_reporters     CDATA  #IMPLIED
    bad_biosequences       CDATA  #IMPLIED
    signature_biosequences CDATA  #IMPLIED
    total_biosequences     CDATA  #IMPLIED
```

```
>

<!ELEMENT reporter_data
    (other*,
     biosequence_data?,
     feature_data+,
     channel_data*,
     ratio?)
>
<!ATTLIST reporter_data
    channel_data_type ( LINEAR | LN | LOG2 |LOG10 | OTHER ) 'LINEAR'
    ratio_type ( LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER ) 'LOG10'
>

<!ELEMENT biosequence_data
    (other*,
     biosequence_ref,
     channel_data*,
     ratio?)
>
<!ATTLIST biosequence_data
    channel_data_type ( LINEAR | LN | LOG2 |LOG10 | OTHER ) 'LINEAR'
    ratio_type ( LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER ) 'LOG10'
>

<!ELEMENT channel_data
    ANY
>
<!ATTLIST channel_data
    name      CDATA #REQUIRED
    value     CDATA #IMPLIED
    error    CDATA #IMPLIED
    pvalue    CDATA #IMPLIED
    fail_type CDATA #IMPLIED
>

<!ELEMENT feature_data
    (other*,
     feature_ref,
     channel*,
     ratio?)
>
<!ATTLIST feature_data
    fail_type   CDATA #IMPLIED
    status_type CDATA #IMPLIED
    ratio_type ( LINEAR | LN | LOG2 |LOG10 | FOLD_CHANGE | OTHER ) 'LOG10'
>

<!ELEMENT channel
    (other*,
     signal,
     background?)
>
<!ATTLIST channel
    name      CDATA  #REQUIRED
    fail_type CDATA  #IMPLIED
    data_type ( LINEAR | LN | LOG2 |LOG10 | OTHER ) 'LINEAR'
>

<!ELEMENT signal
    (other*)
>
<!ATTLIST signal
    raw_value        CDATA  #REQUIRED
    normalized_value CDATA  #IMPLIED
    stddev           CDATA  #IMPLIED
    median           CDATA  #IMPLIED
    pixels           CDATA  #IMPLIED
>

<!ELEMENT background
    (other*)
```

```
>
<!ATTLIST background
    value  CDATA  #REQUIRED
    stddev CDATA  #IMPLIED
    median CDATA  #IMPLIED
    pixels CDATA  #IMPLIED
>

<!ELEMENT ratio
    (other*)
>
<!ATTLIST ratio
    value     CDATA  #REQUIRED
    xdev      CDATA  #IMPLIED
    pvalue    CDATA  #IMPLIED
    error     CDATA  #IMPLIED
    fail_type CDATA  #IMPLIED
>

<!ELEMENT biosequence_cluster
   (other*,
    external_ref?,
    bibliographic_ref*,
    algorithm_ref?,
    biosequence_cluster_member+,
    annotation*)
>
<!ATTLIST biosequence_cluster
   name CDATA #REQUIRED
   type CDATA #IMPLIED
>

<!ELEMENT biosequence_cluster_member
   (other*,
    biosequence_ref+)
>
<!ATTLIST biosequence_cluster_member
   value CDATA #REQUIRED
   units CDATA #IMPLIED
>

<!ELEMENT biosequence_set
   (other*,
    external_ref?,
    bibliographic_ref*,
    algorithm_ref?,
    experiment_set_ref*,
    biosequence_set_member+,
    annotation*)
>
<!ATTLIST biosequence_set
   name CDATA #REQUIRED
>

<!ELEMENT biosequence_set_member
   (other*,
    (biosequence_set_member+ |
     biosequence_ref+),
    biosequence_value*)
>

<!ELEMENT biosequence_value
   (other*)
>
<!ATTLIST biosequence_value
   type  CDATA #REQUIRED
   value CDATA #REQUIRED
   units CDATA #IMPLIED
>

<!ELEMENT experiment_set
   (other*,
     external_ref?,
     bibliographic_ref*,
     algorithm_ref?,
     experiment_set_member+,
     annotation*)
>
<!ATTLIST experiment_set
   name CDATA #REQUIRED
>

<!ELEMENT experiment_set_ref
   (other*)
>
<!ATTLIST experiment_set_ref
   name CDATA #REQUIRED
>

<!ELEMENT experiment_set_member
   (other*,
    (experiment_set_member+ |
     combine),
    experiment_value*)
>

<!ELEMENT experiment_value
   (other*)
>
<!ATTLIST experiment_member
   type  CDATA #REQUIRED
   value CDATA #REQUIRED
   units CDATA #IMPLIED
>
```

9.2 Example XML

The following are brief examples of data spread across five files to illustrate the use of DsLSR_GEML and reference elements. Some of the references resolve within the file, some reference between the files and other references are presumed present in the client or the client will use as place holders, for instance the **sample_ref**, **compound_ref**, and **solvent_ref** in file3.

File One

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE project SYSTEM "DsLSR_GEML.dtd">
<project name="ABC and XYZ collaboration #2" id="ABC_XYZ_2" date="1999-11-02T11:01:09Z"
by="John Doe" organization="ABC">
   <biosequence primary_name="TNN_1" species="H. sapiens" sequenceDB="GenBank">
      <accession identifier="A000001" database="GeneBank"/>
      <accession identifier="XYZ_01" database="ABC"/>
   </biosequence>
   <biosequence primary_name="XN_7" species="H. sapiens" sequenceDB="GenBank">
      <accession identifier="A000002" database="GeneBank"/>
      <accession identifier="XYZ_03" database="ABC"/>
   </biosequence>
   <biosequence primary_name="TNN_6" species="H. sapiens" sequenceDB="GenBank">
      <accession identifier="A000003" database="GeneBank"/>
      <accession identifier="XYZ_05" database="ABC"/>
   </biosequence>
   <biosequence primary_name="IT_7" species="H. sapiens" sequenceDB="GenBank">
      <accession identifier="A000004" database="GeneBank"/>
      <accession identifier="XYZ_07" database="ABC"/>
   </biosequence>
   <biosequence primary_name="XN_19" species="H. sapiens" sequenceDB="GenBank">
      <accession identifier="A000005" database="GeneBank"/>
      <accession identifier="XYZ_09" database="ABC"/>
   </biosequence>
   <pattern name="XYZ HSAPIENS #11" type="ABC/XYZ_collab/inkjet" species_database="H. sapiens
XYZ" access="XYZ_collab" owner="John Doe">
```

```
       <grid_layout feature_spacing_x=".001" feature_spacing_y=".00015" feature_count_x="500"
feature_count_y="250">
           <feature_size x_length=".0005" y_length=".0001" shape="oval"/>
       </grid_layout>
       <reporter name="A000001_NORM" control_type="NORMALIZATION"
active_sequence="ACTGACTGACTG">
           <biosequence_ref identifier="A000001" database="GenBank" species="H. sapiens"/>
           <feature number="1"/>
           <feature number="2" ctrl_for_feat_num="1" control_type="true">
               <mismatch_info start_coord="4" sequence="c" replaced_length="1"/>
           </feature>
           <oligo/>
       </reporter>
       <reporter name="A000002" active_sequence="ATGCATGCATGC">
           <biosequence_ref identifier="A000002" database="GenBank" species="H. sapiens"/>
           <feature number="3"/>
           <feature number="4" ctrl_for_feat_num="4" control_type="true">
               <mismatch_info start_coord="5" sequence="T" replaced_length="1"/>
           </feature>
           <oligo/>
       </reporter>
       <reporter name="A000003" active_sequence="CGTACGTACGTA">
           <biosequence_ref identifier="A000003" database="GenBank" species="H. sapiens"/>
           <feature number="5"/>
           <feature number="6" ctrl_for_feat_num="5" control_type="true">
               <mismatch_info start_coord="6" sequence="C" replaced_length="1"/>
           </feature>
           <oligo/>
       </reporter>
       <reporter name="A000004" active_sequence="GTACGTACGTAC">
           <biosequence_ref identifier="A000004" database="GenBank" species="H. sapiens"/>
           <feature number="7"/>
           <feature number="8" ctrl_for_feat_num="7" control_type="true">
               <mismatch_info start_coord="7" sequence="T" replaced_length="1"/>
           </feature>
           <oligo/>
       </reporter>
       <reporter name="A000005" active_sequence="CATGCATGCATG">
           <biosequence_ref identifier="A000005" database="GenBank" species="H. sapiens"/>
           <feature number="9"/>
           <feature number="10" ctrl_for_feat_num="9" control_type="true">
               <mismatch_info start_coord="8" sequence="C" replaced_length="1"/>
           </feature>
           <oligo/>
       </reporter>
   </pattern>
</project>
```

File Two

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE project SYSTEM "DsLSR_GEML.dtd">
<project name = 'ABC and XYZ collaboration #2'
         id   = 'ABC_XYZ_2'
         date = '1999-11-02T11:01:09Z'
         by   = 'John Doe'
         organization = 'ABC'
>
    <printing date = '2000-04-01T12:01:03Z' printer = 'IJ_34' type = 'inkjet'
              prepared_by_org = 'ABC' prepared_at_site = 'Hometown' prepared_by = 'Sam Smith'>
        <other name = 'adjusted' value = 'x offset'>
            <other name = 'minus' value = '.0001'/>
        </other>
        <chip barcode = 'XYZ0000000AA' prepared_for_org = 'XYZ'>
            <pattern_ref name = 'XYZ HSAPIENS #11' />
        </chip>
        <chip barcode = 'XYZ0000000AB' prepared_for_org = 'XYZ'>
            <pattern_ref name = 'XYZ HSAPIENS #11' />
            <printing_rep status = 'MISFIRE'>
                <feature_ref number = '3'/>
            </printing_rep>
        </chip>
        <chip barcode = 'XYZ0000000AC' prepared_for_org = 'XYZ'>
            <pattern_ref name = 'XYZ HSAPIENS #11' />
        </chip>
        <chip barcode = 'XYZ0000000AD' prepared_for_org = 'XYZ'>
            <pattern_ref name = 'XYZ HSAPIENS #11' />
        </chip>
        <chip barcode = 'XYZ0000000AE' prepared_for_org = 'XYZ'>
            <pattern_ref name = 'XYZ HSAPIENS #11' />
        </chip>
    </printing>
</project>
```

File Three

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE project SYSTEM "DsLSR_GEML.dtd">
<project name = 'ABC and XYZ collaboration #2'
         id   = 'ABC_XYZ_2'
         date = '1999-11-02T11:01:09Z'
         by   = 'John Doe'
         organization = 'ABC'
>
    <prep code = 'P1' date = '2000-11-01T09:12:11Z'>
        <treatment name = 'T1' step_number = '1'>
            <sample_ref name = 'sample 1'/>
            <sample_ref name = 'sample 2' />
            <treatment_compound concentration = '20'>
                <compound_ref code = 'C1' />
                <solvent_ref name = 'S1' />
            </treatment_compound>
            <treatment_solvent concentration = '20'>
                <solvent_ref name = 'S2' />
            </treatment_solvent>
        </treatment>
        <treatment name = 'WASH' step_number = '2' />
        <treatment name = 'T2' step_number = '3' >
            <treatment_compound concentration = '40'>
                <compound_ref code = 'C1' />
                <solvent_ref name = 'S2' />
            </treatment_compound>
        </treatment>
        <treatment name = 'WASH' step_number = '4' />
    </prep>
</project>
```

File Four

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE project SYSTEM "DsLSR_GEML.dtd">
<project name = 'ABC and XYZ collaboration #2'
         id   = 'ABC_XYZ_2'
         date = '1999-11-02T11:01:09Z'
         by   = 'John Doe'
         organization = 'ABC'
>
    <hyb name = 'H1' chip_barcode = 'XYZ0000000AA' >
        <labeled_prep label = 'Cy5' used_ug = '20' >
            <prep_ref code = 'P1' />
        </labeled_prep >
    </hyb>

    <hyb name = 'H3' chip_barcode = 'XYZ0000000AC' >
        <labeled_prep label = 'Cy5' used_ug = '20' >
            <prep_ref code = 'P1' />
        </labeled_prep >
    </hyb>
</project>
```

File Five

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE project SYSTEM "DsLSR_GEML.dtd">
<project name = 'ABC and XYZ collaboration #2'
         id   = 'ABC_XYZ_2'
         date = '1999-11-02T11:01:09Z'
         by   = 'John Doe'
         organization = 'ABC'
>
    <profile barcode = 'XYZ0000000AA' access = 'XYZ_collab' owner = 'John Doe'>
        <image_file name = 'H1Scan.jpg' />
        <channel_info channel_name = 'Cy5' color_name = 'red' raw_image_filename =
'//server1/XYZ_collab2/H1Scan.tif' />
        <reporter_data>
            <feature_data>
                <feature_ref number ='1' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.02' normalized_value = '1.02' stddev = '.5' pixels
= '33'/>
                    <background value = '0.1' stddev = '.001' pixels = '41'/>
                </channel>
            </feature_data>
            <feature_data>
                <feature_ref number ='2' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.4' normalized_value = '1.03' stddev = '.1' pixels =
'34'/>
                    <background value = '0.2' stddev = '3.0' pixels = '22'/>
                </channel>
            </feature_data>
        </reporter_data>
        <reporter_data>
            <feature_data fail_type = 'SATURATED'>
                <feature_ref number ='3' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.3' normalized_value = '1.4' stddev = '.3' pixels =
'31'/>
                    <background value = '1.0' stddev = '.3' pixels = '31'/>
                </channel>
            </feature_data>
            <feature_data>
                <feature_ref number ='4' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.4' normalized_value = '1.03' stddev = '.1' pixels =
'34'/>
                    <background value = '0.2' stddev = '3.0' pixels = '22'/>
                </channel>
            </feature_data>
        </reporter_data>
        <reporter_data>
            <feature_data>
                <feature_ref number ='5' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.02' normalized_value = '1.02' stddev = '.5' pixels
= '33'/>
                    <background value = '0.1' stddev = '.001' pixels = '41'/>
                </channel>
            </feature_data>
            <feature_data fail_type = 'SATURATED'>
                <feature_ref number ='6' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.3' normalized_value = '1.4' stddev = '.3' pixels =
'31'/>
                    <background value = '1.0' stddev = '.3' pixels = '31'/>
                </channel>
            </feature_data>
        </reporter_data>
        <reporter_data>
            <feature_data>
                <feature_ref number ='7' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.02' normalized_value = '1.02' stddev = '.5' pixels
```

```
= '33'/>
                    <background value = '0.1' stddev = '.001' pixels = '41'/>
                </channel>
            </feature_data>
            <feature_data>
                <feature_ref number ='8' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.4' normalized_value = '1.03' stddev = '.1' pixels =
'34'/>
                    <background value = '0.2' stddev = '3.0' pixels = '22'/>
                </channel>
            </feature_data>
        </reporter_data>
        <reporter_data>
            <feature_data>
                <feature_ref number ='9' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.4' normalized_value = '1.03' stddev = '.1' pixels =
'34'/>
                    <background value = '0.2' stddev = '3.0' pixels = '22'/>
                </channel>
            </feature_data>
            <feature_data>
                <feature_ref number ='10' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.02' normalized_value = '1.02' stddev = '.5' pixels
= '33'/>
                    <background value = '0.1' stddev = '.001' pixels = '41'/>
                </channel>
            </feature_data>
        </reporter_data>
    </profile>

    <profile barcode = 'XYZ0000000AC' access = 'XYZ_collab' owner = 'John Doe'>
        <image_file name = 'H3Scan.jpg' />
        <channel_info channel_name = 'Cy5' color_name = 'red' raw_image_filename =
'//server1/XYZ_collab2/H3Scan.tif' />
        <reporter_data>
            <feature_data>
                <feature_ref number ='1' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.02' normalized_value = '1.02' stddev = '.5' pixels
= '33'/>
                    <background value = '0.1' stddev = '.001' pixels = '41'/>
                </channel>
            </feature_data>
            <feature_data>
                <feature_ref number ='2' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.4' normalized_value = '1.03' stddev = '.1' pixels =
'34'/>
                    <background value = '0.2' stddev = '3.0' pixels = '22'/>
                </channel>
            </feature_data>
        </reporter_data>
        <reporter_data>
            <feature_data fail_type = 'SATURATED'>
                <feature_ref number ='3' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.3' normalized_value = '1.4' stddev = '.3' pixels =
'31'/>
                    <background value = '1.0' stddev = '.3' pixels = '31'/>
                </channel>
            </feature_data>
            <feature_data>
                <feature_ref number ='4' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.4' normalized_value = '1.03' stddev = '.1' pixels =
'34'/>
                    <background value = '0.2' stddev = '3.0' pixels = '22'/>
                </channel>
            </feature_data>
        </reporter_data>
        <reporter_data>
            <feature_data>
                <feature_ref number ='5' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.02' normalized_value = '1.02' stddev = '.5' pixels
= '33'/>
                    <background value = '0.1' stddev = '.001' pixels = '41'/>
                </channel>
            </feature_data>
            <feature_data fail_type = 'SATURATED'>
                <feature_ref number ='6' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.3' normalized_value = '1.4' stddev = '.3' pixels =
'31'/>
                    <background value = '1.0' stddev = '.3' pixels = '31'/>
                </channel>
            </feature_data>
        </reporter_data>
        <reporter_data>
            <feature_data>
                <feature_ref number ='7' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.02' normalized_value = '1.02' stddev = '.5' pixels
= '33'/>
                    <background value = '0.1' stddev = '.001' pixels = '41'/>
                </channel>
            </feature_data>
            <feature_data>
                <feature_ref number ='8' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.4' normalized_value = '1.03' stddev = '.1' pixels =
'34'/>
                    <background value = '0.2' stddev = '3.0' pixels = '22'/>
                </channel>
            </feature_data>
        </reporter_data>
        <reporter_data>
            <feature_data>
                <feature_ref number ='9' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.4' normalized_value = '1.03' stddev = '.1' pixels =
'34'/>
                    <background value = '0.2' stddev = '3.0' pixels = '22'/>
                </channel>
            </feature_data>
            <feature_data>
                <feature_ref number ='10' />
                <channel name = 'Cy5'>
                    <signal raw_value = '1.02' normalized_value = '1.02' stddev = '.5' pixels
= '33'/>
                    <background value = '0.1' stddev = '.001' pixels = '41'/>
                </channel>
            </feature_data>
        </reporter_data>
    </profile>

    <combine>
        <hyb_ref chip_barcode = 'XYZ0000000AA' number = '1'/>
        <hyb_ref chip_barcode = 'XYZ0000000AC' number = '1'/>
    </combine>
</project>
```