MAGE-TAB Specification
Version 1.1

May 31, 2011

Abstract

This is a specification for a microarray data acquisition and communication standard MAGE-TAB, considerably simpler than MAGE-ML, but powerful enough to encode any microarray investigation and all information required for MIAME compliance. It is proposed that MAGE-TAB becomes a part of MAGEv2, defining the 'simple layer' in MAGEv2. The main guiding principles in designing MAGE-TAB are:

1. The format should be simple, but should also provide an explicit, structured representation of the details required by the MIAME standard (see http://www.mged.org/Workgroups/MIAME/miame_checklist.html).
2. The format should support concise description of the most frequently used experimental designs in a fashion familiar to biologists.
3. It should be possible to easily create, read, understand and edit documents in this format using only commonly available tools, and requiring no special training in bioinformatics or computer programming.
4. The format should have a formal definition, it should be machine-readable to the level of granularity defined by the MIAME structure, and it should be usable for communicating microarray data between different databases, data analysis tools and other software packages.
5. The formal definition should be based on the MAGE object model and for documents that can be expressed in MAGE-TAB there is a unique mapping to and from MAGE-ML. At the same time no general MAGE knowledge should be needed to use MAGE-TAB format.

The proposed format is based on coding investigation designs, array descriptions, and normalized data as tab-delimited files, while leaving raw data in native formats, and protocols largely as free texts. The most important concept at the basis of this format is the notion of an *investigation design graph* (IDG) — a directed acyclic graph that shows the relationships between different 'biomaterials' or data objects, for instance, showing which sample goes on which array using which label and producing which data-file.

Contents

1 Introduction

A MAGE-TAB document consists of four different types of files:

1. **Investigation Description Format (IDF)** — a plain-text, tab-delimited file providing general information about the investigation, including its name, a brief description, the investigator's contact details, bibliographic references, and free text descriptions of the protocols used in the investigation.
2. **Sample and Data Relationship Format (SDRF)** — a plain-text, tab-delimited file (or files) describing the relationships between samples, arrays, data, and other objects used or produced in the investigation, and providing all MIAME information that is not provided elsewhere. This is often the least trivial part of the experiment description due to the complex relationships which are possible between samples and their respective assays; however, for simple experimental designs, constructing the SDRF file is straightforward, and even complex loop designs can be expressed in this format.
3. **Array Design Format (ADF)** — a plain-text, tab-delimited file defining each array type used. An ADF file describes the design of an array, e.g., what sequence is located at each position on an array and what the annotation of this sequence is. If the investigation uses arrays for which a description has been previously provided, such as a standard commercial array, cross-references to entries in a public repository (e.g., an Array Express accession number) can be included instead of explicit array descriptions.
4. **Data files —**
    i. **Raw data files** — binary files containing experimental data, typically in native formats defined by the provider of the technology used (e.g. CEL format files)
    ii. **Processed data files** — plain text, tab-delimited files generated by some processing operation, usually on a raw data file. This includes both custom processed file formats and the MAGE-TAB specific "data matrix" format, described below.

The main weight of the investigation description is in the SDRF. The most important concept behind the SDRF is the **investigation design graph**, which is a directed acyclic graph (DAG), where nodes correspond to *biomaterials* (*e.g.*, samples, RNA extracts, labeled cDNA, etc.) or data objects (*e.g.*, raw or normalized data files), and arcs correspond to the relationships between these objects. Biomaterials have properties, some of which can be *experimental factors*. Attributes can be attached to nodes and to arcs to describe biomaterial or data properties, *e.g.*, sample descriptions attached to sample nodes, protocol references attached to edges, raw data-files attached to assays. Attributes can be pointers to some longer descriptions or external objects, *e.g.*, protocols described in the IDF file.

The investigation design graph could be encoded in various ways, for instance using the graph mark-up language GML. Here we use a tabular format for the following reasons:

1. The observation that large investigation designs typically have a regular structure, i.e., the same subgraph is repeated many times (possibly with well defined modifications); moreover, the replicated structure is simple. This observation was supported by analysis of the structure of over 1,000 different investigations in the ArrayExpress database.
2. The degree of nodes in these graphs (i.e., the number of incoming and outgoing edges for a node), is small (most often 1 to 4), except for a few specific nodes which are related 'reference' samples or extracts (e.g., 'Reference LE' in Figure 24), or common source nodes (e.g., Figure 37).
3. The observation that DAGs which correspond to commonly used investigation designs have a property that their nodes can be grouped in consecutive layers, i.e., the source nodes (the nodes in the DAG which do not have entering edges) are in layer 1, the nodes that are connected to source nodes by an edge are in layer 2, etc. Furthermore, the grouping can be done so that each layer only contains objects of the same type, e.g., for the graph in Figure 8(a), we have source layer 1, sample layer 2, extract layer 3, labeled extract layer 4 and assay layer 5.
4. Similar tabular formats have been used successfully in the biosciences and are familiar to many practitioners.

Once a DAG of a regular structure has been represented in such a layered fashion, it is natural to encode it as a tab-delimited file (a 'spreadsheet' in the broad sense of this word). Each column in the spreadsheet corresponds to a layer in the DAG, while each row corresponds to a path in the graph from one of the source nodes, to one of the 'sink' nodes.

2 Specification of MAGE-TAB

Unlike the full MAGE and FuGE models, we will not start here by developing the object model explicitly. Instead, we will discuss MAGE-TAB with the *investigation design graphs* as its basis. These graphs contain some extra information that is difficult to encode in an object model, in particular:

1. the fact that investigation design cannot have loops, *i.e.*, it is a DAG as oppose to an arbitrary oriented graph (this property is not automatically following from the FuGE or MAGE object model);
2. the notion of *layers* in the graph, each in effect representing a stage in the experimental process;
3. investigation designs typically have a regular structure.

2.1 Overview

The following problems have to be solved on the document level:

- how to encode MIAME-required investigation level information;
- how to create packages of MAGE-TAB files;
- how to identify objects in MAGE-TAB;
- how to split definition of an object between investigation-level (IDF) and sample/assay level (SDRF) documents;
- how to reference objects in MAGE-TAB that are defined elsewhere (in the IDF file or in another MAGE-TAB file);
- how to describe MAGE-TAB syntax elements.

2.1.1 Describing "top-level" MIAME information

The investigation design graph will not contain everything necessary to describe investigations according to MIAME. Overall information about investigations, protocols, and contact information does not have high volume, nor any internal regularity. Therefore, the graph will be encoded in one file, or set of files (the SDRF), and the associated metadata will be kept in a separate file (the IDF).

2.1.2 Packaging IDF and SDRF

The IDF file should contain a reference to the SDRFs which comprise the investigation description. This is done via the tag "SDRF File", which may be used to include all the SDRF files required. A typical investigation will only use one SDRF, but it is possible to reference multiple SDRFs if the situation demands it (for example, in mixed array-based/sequencing experiments). Data files and ADF files are referenced from the SDRF table directly. It is recommended that all data files be in a single directory or archive with no sub-directory structure.

File names must be treated in a case-sensitive manner, to ensure cross-platform compatibility. It is recommended that files be given filename extensions of ".idf.txt", ".sdrf.txt" and ".adf.txt". This is used to help software-based parsers infer the file types as a starting point. Blank lines containing zero or more spaces or tabs are permitted in any of these files. Lines starting with the "#" symbol are interpreted as comments.

2.1.3 Referencing files, external databases and ontology sources

File references (the SDRF File tag in the IDF, data file references in the SDRF etc.) can contain relative or absolute paths. Relative paths may contain two dots ("..") to refer to parent directories. For file references containing relative paths, the referenced file must be present on the same file system. Absolute path references can refer to a URL, a local Windows file system or local unix file system and should therefore start with a protocol ("http://domain/path"), drive letter (e.g. "C:\Folder\Subfolder\") or a slash ("/home/user/") respectively. It is recommended that all MAGE-TAB documents use either relative path references or absolute URL references, avoiding system-specific file references wherever possible.

All of the MAGE-TAB components (IDF, ADF, SDRF and data matrices) allow for referencing ontology terms or database accessions from external sources. In each case the source of the term(s) is indicated by a separate "Term Source REF" entry. See Table 10 and Figure 38 for examples. These tags are defined in the IDF using the "Term Source Name" tags shown in Table 1.

2.1.4 Identifying objects in MAGE-TAB

In general, the values specified for the Name columns in the SDRF will have the scope of the experiment documented by the IDF and SDRF files. That is, they are not expected to refer to objects outside of the experiment. At times, especially for "Source

Name" and "Sample Name" values, a value may have a scope greater than the current experiment. For example a sample may have been aliquoted and used in more than one experiment.

In the case a value has a greater scope than the experiment, in order to facilitate cross-referencing , one can use an LSID construction of the name: <authority>:[<namespace>]:<object>[:<revision>] (see http://xml.coverpages.org/lsid.html for further information).

2.1.5 Referencing objects in MAGE-TAB

Similarly as for defining the objects we need a uniform way of referencing objects that are defined in the IDF (*e.g.*, Protocols). Columns should be named ending with "REF", *e.g.*, "Protocol REF". In those columns only object identifiers defined in the IDF should be used.

2.1.6 MAGE-TAB field delimiters

IDF, SDRF and ADF documents will contain data divided into columns and rows. Columns are separated by tab characters, while lines are separated by newlines and/or carriage returns. Fields within columns may be escaped by surrounding them with double quotes, indicating that any tab or newline characters contained therein are not to be interpreted as a field delimiter, *i.e.*, that such characters are part of the content encoded by the document and not part of the document structure. Quote characters within fields must be escaped with a backslash, like so: \". Note that column headers are also permitted to be enclosed in double quotes, but no characters other than spaces are permitted between the multiple keywords that comprise a column header.

2.1.7 MAGE-TAB Headers

MAGE-TAB headers (the first element of a row or a column, depending on the file type) are case-insensitive and whitespace-insensitive (obviously excluding the tab-character, which acts as the header delimiter). Although headers will normally be written in a human-friendly way ("Sample Name") the capitalization of the headers and any space characters in this header should be ignored by parsers when reading the document. Therefore, the following values are all equivalent headers:

Sample Name ~ SampleName ~ SAMPLENAME ~ samplename ~ samPLeNaMe ~ SAM PL EN A ME

Note that this only applies to the defined headers themselves. Values inside square brackets or parentheses are user-supplied and should be treated verbosely, in a case-sensitive and spacing sensitive manner. They should not be altered or interpreted at all when reading the file. So:

Factor Value [ORGANISM_PART] ~ factorvalue[ORGANISM_PART] !~ Factor Value [Organism Part].

Tab characters are permitted inside square brackets, but only when the header is appropriately escaped using quotes as described in 2.1.6.

2.1.8 MAGE-TAB encoding

IDF, SDRF, ADF and processed data files are all plain-text files. To avoid encoding errors when reading or writing MAGE-TAB documents, and to avoid limiting the range of characters available for use in MAGE-TAB, all MAGE-TAB files should be encoding using UTF-8 unless otherwise specified in the IDF.

2.2 Investigation Design Format (IDF)

2.2.1 IDF Specification

	Value Type	Cardinality
MAGE-TAB Version	Text	1
Investigation Title	Text	0..1
Investigation Accession	Accession	0..1
Investigation Accession Term Source REF	Term Source Name	0..1
Experimental Design	Ontology term	0..*
Experimental Design Term Source REF	Term Source Name	0..*
Experimental Design Term Accession Number	Term Accession Number	0..*
Experimental Factor Name	Text	0..*
Experimental Factor Type	Ontology term	0..*
Experimental Factor Term Source REF	Term Source Name	0..*
Experimental Factor Term Accession Number	Term Accession Number	0..*
Person Last Name	Text	0..*
Person First Name	Text	0..*
Person Mid Initials	Text	0..*
Person Email	Text	0..*
Person Phone	Text	0..*
Person Fax	Text	0..*
Person Address	Text	0..*
Person Affiliation	Text	0..*
Person Roles	Ontology term (semicolon-delimited list)	0..*
Person Roles Term Source REF	Term Source Name	0..*
Person Roles Term Accession Number	Term Accession Number	0..*
Quality Control Type	Ontology term	0..*
Quality Control Term Source REF	Term Source Name	0..*
Quality Control Term Accession Number	Term Accession Number	0..*
Replicate Type	Ontology term	0..*
Replicate Term Source REF	Term Source Name	0..*
Replicate Term Accession Number	Term Accession Number	0..*
Normalization Type	Ontology term	0..*
Normalization Term Source REF	Term Source Name	0..*
Normalization Term Accession Number	Term Accession Number	0..*
Date of Experiment	Date (YYYY-MM-DD)	0..1
Public Release Date	Date (YYYY-MM-DD)	0..1
PubMed ID	ID	0..*
Publication DOI	DOI	0..*
Publication Author List	Text	0..*
Publication Title	Text	0..*
Publication Status	Ontology term	0..*
Publication Status Term Source REF	Term Source Name	0..*
Publication Status Term Accession Number	Term Accession Number	0..*
Experiment Description	Text	0..1
Protocol Name	ID	0..*
Protocol Type	Ontology term	0..*
Protocol Term Source REF	Term Source Name	0..*
Protocol Term Accession Number	Term Accession Number	0..*
Protocol Description	Text	0..*
Protocol Parameters	Text (semicolon-delimited list)	0..*
Protocol Hardware	Text	0..*
Protocol Software	Text	0..*
Protocol Contact	Text	0..*
SDRF File	File ref	0..*
Term Source Name	Text tag as used in SDRF	0..*
Term Source File	File ref	0..*
Term Source Version	Text	0..*
Comment[]	Text	0..*

Table 1: IDF: Allowed fields, typing and cardinalities

2.2.2 Notes on the IDF

The IDF is arranged as a tab-delimited plain-text file. The first element on each line of the file represents the tag, and subsequent tab-separated elements represent the values associated with this tag.

In Table 1, the first column indicates the text tag that should be used in the IDF file, and the second column indicates the type of entry expected for each row. The third column shows the cardinalities allowed for each elements – some fields can have unlimited values, others are allowed a maximum of one value. For example, one can only supply one value for "Date of Experiment" but one should use as many "Person Last Name" columns as there are contacts for the investigation. In cases where multiple terms need to be entered into a single column, they should be separated by semicolons (*e.g.*, "Protocol Parameters", "Person Roles"). All such semicolon-separated roles must be from one ontology.

Additional comments on Table 1:

1. All row types are optional, except for "MAGE-TAB Version", which is required from MAGE-TAB version 1.1 onwards. (If the MAGE-TAB Version does not exist, it is assumed that it is a MAGE-TAB version 1.0 IDF file.) All row types allow multiple values (columns), except for the rows highlighted in blue, which do not allow multiple values to be specified. Note that fields which contain ontology individual terms should indicate the origin of those terms using the relevant "Term Source REF" tag. Dates should be supplied in the ISO format "YYYY-MM-DD". See Table 10 for an example IDF.
2. The general "Comment []" field name is included as a basic extensibility mechanism for local implementations, analogous to the use of NameValueTypes in MAGEv1. The name associated with the comment is included in square brackets in the row name, and the value entered in the body of the IDF. Types are not currently supported. Example use-cases for the IDF are "Comment[Goal]" to describe the specific hypothesis being tested by the experiment, or "Comment[AnnotationFile]" to include extra annotation files (e.g. CDISC or MAGE-ML descriptions of source materials). It is anticipated that ArrayExpress will include one or both of these fields in their own local implementation.
3. The Comment[Character Encoding] field should be used to describe the character encoding, if an encoding other that UTF-8 is used. If omitted, UTF-8 encoding should be assumed.
4. To specify bibliographic references accompanying the experiment, it is sufficient to enter just the PubMed ID for each citation into the IDF. Where a given article is not yet published, the available information should be given using the IDF tags shown.
5. All values can be entered in plain text; the typing indicates those fields that can be validated against other criteria. A typing of "accession" indicates that a special value is attached to the value supplied – it provides a unique identifier. Ontology Term, Term Source Name and Term Accession Number are used to indicate a value is supplied that is defined in an external resource (usually an ontology), and that it should be possible to locate the term in the given external resource using this information. File refs are used to indicate the supplied value references a file on the

2.3 Sample and Data Relationship Format (SDRF)

2.3.1 SDRF Specification

Node/Edge	Associated attributes	Cardinality
Source Name	Characteristics, Provider, Material Type, Description, Comment	0..1
Sample Name	Characteristics, Material Type, Description, Comment	0..*
Extract Name	Characteristics, Material Type, Description, Comment	0..*
Labeled Extract Name	Characteristics, Material Type, Description, Label, Comment	0..1
Assay Name	Technology Type, Array Design File / REF, Comment	0..1
Scan Name	Comment	0..*
Normalization Name	Comment	0..*
Array Data File	Comment	0..*
Derived Array Data File	Comment	0..*
Array Data Matrix File	Comment	0..*
Derived Array Data Matrix File	Comment	0..*
Image File	Comment	0..*
Protocol REF	Term Source REF, Parameter, Performer, Date, Comment	0..*

Table 2: SDRF: Association of labels to identifiers (Node and edge columns)

Attribute	Associated attributes	Cardinality
Characteristics [ ]	Unit, Term Source REF	0..*
Provider	Comment	0..1
Material Type	Term Source REF	0..1
Array Design File/REF	Term Source REF, Comment	0..1
Technology Type	Term Source REF	0..1
Label	Term Source REF	0..1
Factor Value [ ] ( )	Unit, Term Source REF	0..*
Performer	Comment	0..1
Date		0..1
Parameter Value [ ]	Unit, Comment, Term Source REF	0..*
Unit [ ]	Term Source REF	0..1
Description		0..1
Term Source REF	Term Accession Number	0..1
Term Accession Number		0..1
Comment [ ]		0..*

Table 3: SDRF: Association of labels to identifiers (Attribute columns)

2.3.2 Notes on the SDRF

The SDRF is arranged as a tab-delimited, plain-text file. It is laid out as a typical tab-delimited spreadsheet, where the first line contains column headings separated with tab characters and each subsequent line contains values associated to the headings.

The order of columns in the SDRF explicitly follows the order of the node layers in the IDG as described in Section 2.3.5. The "* Name" and "* File" **node** columns are linked by "Protocol REF" columns which represent the graph **edges**. (Protocol REF is the only type of **edge** possible.) Furthermore, each node and edge column may be associated with one or more **attribute** columns containing annotation, *e.g.*, "Source Name" may be associated with "Provider"; "Parameter Value []" with "Unit". In each case the attribute column follows immediately after the respective node or edge column. Similarly, where ontology terms are used a "Term Source REF" column should follow immediately to the right of the column containing the actual ontology terms (see *e.g.*, Figure 40). Table 2 and 3 describe the column headers allowed in the SDRF spreadsheet. Examples of valid SDRF files,for a range of typical hypothetical experiments and taken from real experiments can be found in section 3.2 and 3.3.

Additional comments on tables 2 and 3:

1. “Name” columns indicate that the column contains an identifier, and “REF” columns indicate that the column references an identifier defined elsewhere in the document.
2. Where ontology terms are used, a second column, “Term Source REF” should be used to indicate the ontology source database or file. These ontology sources are defined in the IDF (see Section 2.2). If no ontology term source is provided then the text is assumed to be user-defined. Also, a third column “Term Accession Number” can be used to indicate the accession number of the term in that term source.
3. The “REF” tag is used in columns which reference objects defined in the IDF or in other MAGE documents. The “REF” tag may be appended with a namespace tag (*e.g.*, “REF:ebi.ac.uk:MIAMExpress:E-MEXP-438”), where the reference is to an object external to the whole document. In the absence of a namespace tag, it is assumed that the column references an object in the document namespace.
4. Each identifier column can be used as many times in the table as desired, so that for example a “Sample Name” column can be followed by as many other “Sample Name” columns as are necessary to fully describe the manipulation of materials in the investigation. See Figure 44 for an example of this.
5. Multiple “Protocol REF” columns may be used between “Name” graph node columns to indicate an ordered set of protocols. See Figure 3 for an example of this.
6. The following columns can use the “REF” suffix to indicate that they reference identifiers defined elsewhere: “Protocol”, “Array Design”, “Term Source”.
7. The columns “Array Design REF” and “Protocol REF” may contain identifiers defined elsewhere in the document (in the ADF and IDF respectively). Alternatively, these columns may reference external identifiers such as accessions from a repository database (e.g., ArrayExpress or GEO). In such cases the database should be defined in the IDF as a Term Source, and referenced in the SDRF as “Term Source REF” associated with these columns. In cases where the identifier is not defined in the document and no external Term Source is provided, it may be assumed that the identifier is local to the context in which the spreadsheet is used (e.g., an ArrayExpress accession number where submitting data to the ArrayExpress database).
8. “Characteristics” column headings should contain an ontology property term in square brackets. The source database or file for ontology terms in these columns may be given in an adjacent “Term Source” column immediately to the right of the “Characteristics” column. In the absence of a “Term Source” column the value is assumed to be user defined. Multiple Characteristic columns of the same category (e.g., “Characteristics[OrganismPart]”) are allowed. Typically the usage implies whole to part from left to right.
9. The "Technology Type" column would typically have the values' microarray' or 'sequencing assay', with a "Term Source Ref" of OBI.
10. “Parameter Value” columns should indicate which Parameter is described by including the parameter name declared in the IDF using square brackets (*e.g.*, “Parameter Value [growth temp]”).
11. “Factor Value” columns should indicate which experimental factor it represents by including the relevant “Experimental Factor Name”, defined in the IDF, in square brackets. An optional term may be appended to this in parentheses where the factor category in the SDRF is more specific than the “Experimental Factor Type” given in the IDF (*e.g.*, “age” *vs.* “time”). The “Factor Value” columns should occur after all element nodes and the attributes of those element nodes. A “Term Source REF” column may be used here in the same way as for “Characteristics”, discussed above. Note that the “Experimental Factor Name must be unique within the documents (IDF and SDRF). Note that biomaterial characteristics and protocol parameters can be factor values, but there is no requirement or ability in the MAGE-TAB specification to enforce consistency between these columns.
12. “Unit” columns must include an ontology property term describing the unit class in square brackets. An example of such a class term would be “TimeUnit” from the MGED Ontology.
13. In general, where square-bracketed values are associated with column headings in Table 3 (*e.g.*, “Characteristics[]”, “Factor Value[]”), these values are compulsory. Values in parentheses are optional.
14. Array designs may be referenced in the spreadsheet by identifier (using “Array Design REF”) or by using “Array Design File” to point to an included ADF file.
15. Where multiple Providers or Performers can meaningfully be attached to a Source or Protocol, respectively, these should be included in a single column and delimited by semicolons.
16. Empty fields should simply be left blank. Fields that say “null” are not the same as empty fields.
17. SDRF files may be split into an arbitrary number of sub-files on any “Name” column, such that the leftmost and rightmost columns of one file will correspond to columns in a second spreadsheet.

18. The “Comment” columns are included as a basic extensibility mechanism for local implementations, analogous to the use of NameValueTypes in MAGEv1. The name associated with the comment is included in square brackets in the column heading, and the value(s) entered in the body of the column. Types are not currently supported. (Note that Comment is permitted as a row type in both the IDF and the ADF header.)

19. Comment columns could be used in various ways - to provide references to supplementary files like PowerPoint presentations; to include identifiers of objects in external systems; to qualify the type of Protocol REF (e.g., growth protocol).

20. There is no MAGE-TAB Version explicitly specified in the SDRF file; it is only specified in the IDF file. The SDRF version is required to be the same as the version specified in the corresponding IDF file.

21. Differentially dimensioned arrays, and multi-technology investigations are allowed in the same SDRF.

2.3.3 Ordering and Cardinality

Element column headers in the SDRF, except for Protocol REF, must occur in the order, and must occur with the cardinalities that are specified in Table 2. The attributes of an element or of another attribute must follow the attributed element or attribute without any intervening element or attribute. When an element or attribute has more than one attribute, there is no ordering defined for that set, except:

- Factor Value: must occur after all element nodes and the attributes of those element nodes.
- Comment: must immediately follow either the element or attribute node for which it is a Comment, or another such Comment. This permits an unambiguous association of a Comment with the element or attribute for which it is a comment.
- Term Source REF: must immediately follow the ontology term for which it provides the source reference. This permits an unambiguous association of the Term Source REF to the ontology term.
- Protocol REF should come directly before the Name column it is the transformation for.

2.3.4 Investigation Design Graph

Two basic notions we use in describing investigations are *biomaterial* and *data object*. The first intuitively represents a physical material such as a sample, RNA extract, array, or hybridized array. A protocol, when applied to a biomaterial, can generate a new biomaterial as its result. Biomaterials can also be split or pooled. For instance, one can take two samples, apply an RNA extraction/labeling protocol to each of them, labeling with Cy3 in the first case and with Cy5 in the second case, mix them and hybridize them on the array:

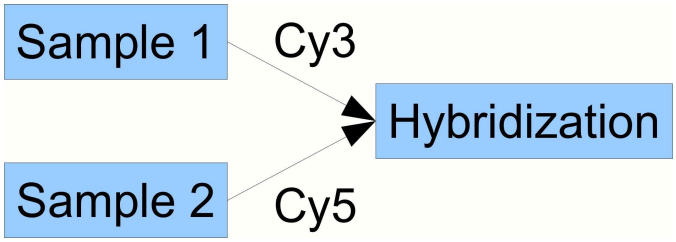

Figure 1: Two samples pooling to one hybridization

Data objects can be created from biomaterials by applying a ‘measurement’ protocol, for example, by scanning a hybridized array to obtain feature intensities. Data objects can be transformed into new data objects by applying a data transformation protocol; for precise definitions of these objects MAGE-TAB will refer to the Functional Genomics Experiment (FuGE) object model that provides a higher-level class model for extension by technology-specific models such as MAGEv2.

The *investigation design graph* (IDG) is a general concept applicable to any investigation description, and not restricted to microarray investigations. Effectively, the IDG represents the workflow of the investigation. The IDG is a labeled DAG, where each node represents either a biomaterial or a data object. Each node in the IDG has an identifier plus an ordered list of labels, each of which has a type. For instance, a node corresponding to a sample will have the sample ID and the sample properties, *e.g.*, species, tissue type, extraction protocol. A label can be either a character string, or a pointer to an external object (including ontology entries). For instance, ‘species’ will be normally described by an external ontology (NCBI taxonomy), ‘tissue type’ can be either a character string or an ontology entry, while ‘protocol’ would normally be a pointer either to an external object (*e.g.*, a

protocol accession number in a database), or to a protocol defined in the accompanying IDF document. Each node in the graph will have a *type*, *e.g.*, sample, extract, assay etc.

A question arises: How granular should the graph be? For instance, should one represent samples, extracts and labeled extracts within the same node, or using three different nodes? In practice, the degree of granularity used in the IDG largely does not matter, unless one of the 'intermediate' objects is being split or pooled. Nodes in the graph that have only one incoming and one outgoing edge can be contracted into their predecessor nodes, by adding extra labels. Thus, unless extracts are pooled or split, it is sufficient to show which sample is hybridized to which array.

To encode an investigation, biologists will normally use the spreadsheet representation directly, without ever drawing the graph explicitly. However, for more complex investigation designs, thinking of them as a graph may be helpful, even if it is not explicitly represented on paper. The graph representation is even more important if one is developing software allowing for data export/import from one's own database or tool.

Next we will describe how to encode the structure of the IDG in a 'spreadsheet' format called *Sample and Data Relationship Format* (SDRF). First we will ignore labels and only consider the IDs of the graph nodes. The labels can be added later simply by introducing extra columns in the spreadsheet, one column per label.

2.3.5 DAG Layers

One of the essential ideas behind the proposed encoding is based on the notion of a *layer* — each node in the DAG will be assigned a layer numbered by 0, 1, 2, . .. , n, in a way such that if there is an arc from a node *v* to node *w*, then the node *w* is in a layer higher than the node *v*. For instance, the layer structure of a simple DAG is shown in Figure 2.

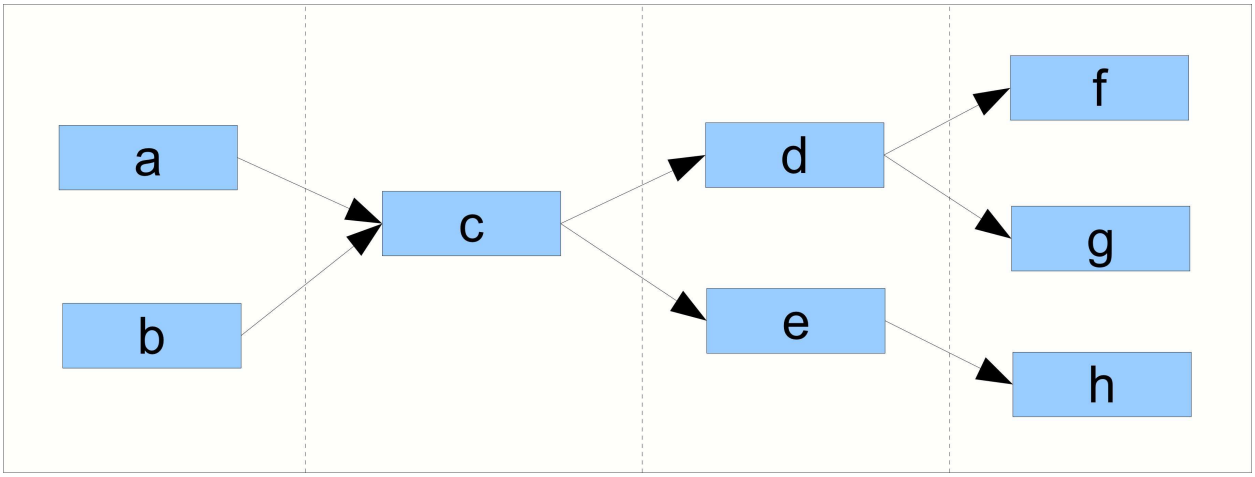

Figure 2: Layered structure of a simple DAG

Formally, to assign layers to nodes in a DAG, we have to solve the following problem: given a DAG *G*, with the set of nodes *V*, assign to every node $v \in V$, an integer l=layer(v), such that for every two nodes *v* and *w*,

if there is an arc *v* -> *w* in *G*, then layer(*v*) < layer(*w*) (x)

Additionally, we want to minimize the total number of layers used. In fact the minimum number of layers equals the length of the longest path in *G*.

Next we will consider typed DAGs, where each node in *V* belong to one of *k* predefined types, *i.e.*, let type(v) $\in$ {1, .. . *k*}. In this case the problem is to assign layers to the nodes so that in addition to the property (x) above, also the following holds:

if layer(*v*) = layer(*w*), then type(*v*) = type(*w*) (y)

Again, we want to keep the total number of layers minimal. In this case the exact minimization problem is difficult, but we will introduce a simple heuristic, which works well for the popular investigation designs.

Let us begin with some basic definitions. The nodes in a DAG that do not have any entering arcs are called *source nodes* , the nodes which do not have any exiting arcs are called *sink nodes*.

Algorithm to assign layers to the nodes of a DAG *G* ignoring the types:

- Find all the paths from any of the source nodes to any of the sink nodes in *G*. Let they be P_1, P_2, ..., P_t in the order of non-ascending length, *i.e.*, the length of P_{i+1} is not longer than the length of P_i.
- Take the longest path v_0, v_1, ..., v_p and assign to its nodes the layers 0, 1, ..., *p* respectively, *i.e.*, layer(v_0) := 0, layer(v_1) := 1, ..., layer(v_p) = *p*.
- Iterate:
    - Take the longest path P_i, such that P_i contains at least one node with unassigned layer;
    - Iterate:
        - Take the first node *v* on this path P_i, such that *v* has not been assigned a layer
        - If *v* is a source node, then assign layer(*v*):=0;
        - Otherwise, find all nodes u_1, ..., u_k, such that u_i -> *v*, and u_i has assigned layer;
        - assign layer(*v*):= max{layer(u_i)}+1

One can prove that for un-typed DAGs the result satisfies the property (x) above, and that the total number of layers assigned is *p* +1. (For the proof the only tricky bit is to prove that in the last step, the nodes among u_1, ..., u_k, that do not have a layer assigned at the time when we are assigning layer(*v*):=max{layer(u_i)}+1, will not be assigned layers higher than max{layer(u_i)}, but this follows from the fact that we have chosen the longest path containing nodes with unassigned layers.)

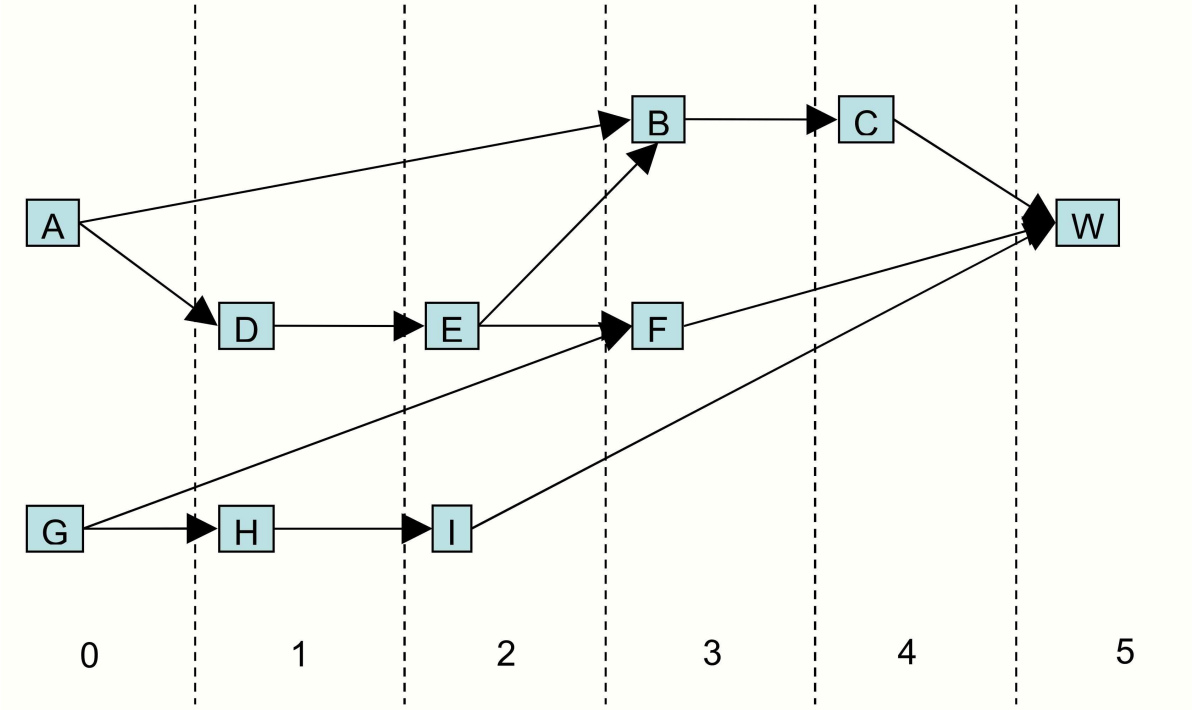

Figure 3: A more complicated example of assigning layers to nodes in a DAG

There are two source nodes A and G, and one sink node W. The longest path is A, D, E, B, C, W, therefore these nodes get layers 0, 1, …, 5 assigned first according to our algorithm as described below. The next longest is A, D, E, F, W, the only node that has not been assigned a layer on this path is F, which gets assigned layer(E)+1 = 3. Finally, the nodes G, H and I get layers 0, 1, 2. Note that, although node G gets a layer assigned after F, there is no conflict (as we always do the longer path first).

Next, let us consider *typed, ordered* DAGs. We want to rearrange our layers, if necessary so, that all the nodes in each layer has the same type (*i.e.*, for the property (y) above to be satisfied). We also want to impose an ordering in which the layers can occur, and at the same time minimize the number of layers. Instead of solving the exact minimization problem, we will apply the following heuristics. If two nodes in layer *'n'* are of different types, and this cannot be resolved by moving differently-typed nodes to a layer containing nodes of the same type, then we insert an additional layer *'n+1'*, obeying the required ordering of layers. We then assign the node of the lower ranked type to the new layer, leaving the node of the higher rank where it was. For instance, in Figure 3, suppose the nodes *d* and *e* in layer 2 are of different types, then we introduce an additional layer 2a (later

we can renumber the layers so that they are numbered consecutively by integers), and assign the node *e* to layer 2a. In the general case, we can have a layer which contains nodes of *n* types, each type containing several nodes. In this case we will introduce *n-1* additional layer, and put in each nodes of the respective type.

2.3.6 Coding a layered typed DAG in a spreadsheet

Once we have assigned layers to all nodes in the way that each layer contains only nodes of the same type, we can encode this graph by a spreadsheet in a simple way. First we mark each column in the spreadsheet by the number of the layer (in an increasing way) and with the particular type of the node in that layer. For this we need to take every pair of a source node *v* and sink node *w*, and find the path from *v* to *w*.

In an arbitrary DAG there may be an exponential explosion of the number of possible paths, but in DAGs corresponding to the popular investigation designs, the number of paths are roughly proportional to the number of samples used in the investigation, making this encoding very practical.

Note that in the popular investigation designs (*e.g.*, the ones given in the examples in the previous section), each node in the graph belongs to only very few paths, thus meaning that the coding of the graph by a spreadsheet is compact and does not exceed the size of the graph itself more than a few times. In fact, often the size of the coding is not larger that the size of the original graph. Moreover, we only need to represent each arc in the graph once in the SDRF, therefore at the point when this has been done, we stop adding new rows.

2.3.7 Experimental factors

Experimental factors are material properties and protocol parameters; *i.e.*, values from any Characteristics or Parameter Value column in the SDRF can be annotated as experimental factors, as described below.

The experimental factors are the principal variables in the investigation, for instance “time” in time series investigations, “dose” in dose response investigations, “compound” in compound treatment investigations, or “disease state” (normal or otherwise) in disease studies. The same investigation may have several experimental factors; for example, compound, dose and time may all be experimental factors in a dose response investigation in which several compounds are added to the samples over a time course.

Experimental factors and their values can be taken from any column in the SDRF file in (*e.g.*, Figure 39), and are annotated as such by also being listed in a separate “Factor Value[]” column, which in turn references an “Experimental Factor Name” defined in the IDF. For example, the IDF linked to the SDRF in Figure 39 would include an “Experimental Factor Name” of “Behavior” in its list of experimental factors, linked to the ontology term “innate behavior” as its “Experimental Factor Type” (see Section 2.2). Where the values in a “Factor Value[]” column are from a more specific subcategory of the factor described in the IDF, parentheses may be used to include the subcategory (*e.g.*, “Factor Value [Growth condition] (media)”). While this arrangement adds some redundancy to the specification of experimental factor values within the SDRF, it allows for such cases where more complex experimental factors may have been used.

Biological replicates are represented by distinct biological sources, grouped together by common experimental factor values. In contrast, technical replicates are represented by branching of the investigation design graph at intermediate steps of the experimental processing.

The experimental factor values are the values of the respective experimental factors in a particular sample. For instance, in a time series the values are the time points at which each measurement was taken.

Experimental factor values provide a means of annotating investigations concisely — the most important experimental variables are clearly and accessibly defined. Moreover, one can easily represent biological replicates: these are samples which have different sources, but exactly the same values for all experimental factors. By propagating the factor values down to data columns in the processed data, one can annotate data concisely. For instance, if we have two experimental factors compound and dose, each of which have two possible values, *e.g.*, compounds c1 and c2, and low dose and high dose, then the data columns will be annotated by combinations of these values: (c1, low), (c2, low), (c1, high), (c2, high). Where the array design itself is an experimental factor, this should be included as a Parameter Value associated with the Assay protocol and also included in a separate “Factor Value[]” column.

SDRF parsers should associate Factor Value annotations as specifically as possible. Two main strategies are available to accomplish this. Firstly, factor value annotations can always be associated with the relevant assay, but typed by channel for multi-channel array-based experiments to ensure annotations are result-specific. Alternatively, factor value attributes can be associated with a node in the graph upstream of the assay, but before any pooling events take place. In practice these strategies yield the same results, but annotations occur in different places.

Ideally, factor value columns are placed as the very last column in an SDRF spreadsheet.

2.4 Array Design Format (ADF)

2.4.1 ADF Specification

	Value Type	Cardinality
Array Design Name	Text	0..1
Version	Text	0..1
Provider	Text	0..1
Printing Protocol	Text	0..1
Technology Type	Ontology term	0..*
Technology Type Term Source REF	Term Source Name	0..*
Technology Type Term Accession Number	Term Accession Number	0..*
Surface Type	Ontology term	0..*
Surface Type Term Source REF	Term Source Name	0..*
Surface Type Term Accession Number	Term Accession Number	0..*
Substrate Type	Ontology term	0..*
Substrate Type Term Source REF	Term Source Name	0..*
Substrate Type Term Accession Number	Term Accession Number	0..*
Sequence Polymer Type	Ontology term	0..*
Sequence Polymer Type Term Source REF	Term Source Name	0..*
Sequence Polymer Type Term Accession Number	Term Accession Number	0..*
Term Source Name	Text tag as used in main ADF table	0..*
Term Source File	File ref	0..*
Term Source Version	Text	0..*
Comment []	Text	0..*

Table 4: ADF header: Allowed fields, typing and cardinalities.

Node/Attribute	Associated attributes	Cardinality
(Feature)	Block Column, Block Row, Column, Row	0..1
Reporter Name	Reporter Database Entry, Reporter Sequence, Reporter Group, Control Type	0..1
Composite Element Name	Composite Element Database Entry, Composite Element Comment	0..1
Block Column		0..1
Block Row		0..1
Column		0..1
Row		0..1
Reporter Sequence		0..1
Reporter Group [ ]	Reporter Group Term Source REF	0..*
Control Type	Control Type Term Source REF	0..1
Reporter Database Entry [ ]		0..*
Composite Element Database Entry [ ]		0..*
Composite Element Comment		0..*
Reporter Group Term Source REF	Reporter Group Term Accession Number	0..1
Control Type Term Source REF	Control Type Term Accession Number	0..1
Reporter Group Term Accession Number		0..1
Control Type Term Accession Number		0..1

Table 5: ADF table: Association of labels to identifiers (Node and attribute columns)

Node/Attribute	Associated attributes	Cardinality
Composite Element Name	Map2Reporters, Composite Element Database Entry, Composite Element Comment	0..1
Map2Reporters		0..*
Composite Element Database Entry [ ]		0..*
Composite Element Comment		0..*

Table 6: ADF extended table: Association of labels to identifiers (Node and attribute columns)

2.4.2 Notes on the ADF

The ADF consists of a single plain-text, tab-delimited file arranged into three possible sections, described in tables 4-6:

1. An optional header section,
2. The main ADF table itself. This table should be preceded by a "[main]" header (section delimiter) which is case-insensitive.
3. An optional extended ADF table, allowing for the representation of complex many-to-many Reporter – Composite Element mappings. If present, this table should be preceded by a "[mapping]" header (section delimiter) which is case insensitive.

The header component is formatted in a similar manner to IDF files, whereas the table and extended table components are laid out as a spreadsheet-encoded DAG, resembling SDRF files.

Each ADF file may start with an optional header section providing some top-level information about the array design. Note that this header contains MIAME-required information, and, as such, any ADF files lacking a header are unlikely to be MIAME-compliant. As described for the IDF in Section 2.2, optional "Comment[]" rows may be used to provide extra information needed by local implementations. Additional rows providing Term Source information are included in the ADF header to allow the full encoding of array design information in the absence of any investigation-level detail. These Term Source rows are treated in the same way as for the IDF (Section 2.2), and are used to indicate the source databases or files used for sequence database accessions and ontology terms. As many Term Sources may be used as needed, listed horizontally in columns as for the IDF. See Table 4 for a list of ADF header row types. All tags are optional, and a tag can have at most one value. The tags (rows) can appear in any order, except that associated attributes must immediately follow the object they are associated with.

The ADF table section must contain columns in the order shown in Table 5. Feature nodes are unusual in that they do not require an identifier – the position itself is unique per array. For this reason, there is no "Feature Name" column – hence the reason Feature is shown in brackets in Table 5. Features always start with a "Block Column" attribute column.

In some array designs, it may be necessary for all Composite Element information to be split into a second table, and a new column, "Map2Reporters" is used to list the Reporters to which each Composite Element is related. The Reporters are expressed as a semicolon-delimited list (see Figure 52 for an example).

The "Reporter Group" ADF heading may be used to describe a variety of different group types; typical examples would be "role" (with values "experimental" and "control") or "species" for multi-species arrays. The types ("role" and "species") are free-text.

2.4.3 Spot Location: The concept of Feature

A spot location uniquely identifies a physical location in the two-dimensional space of the microarray surface. it is defined by its coordinates in the ADF coordinates system: "Block Column", "Block Row", "Column", "Row" (see Table 7).

Block Column	Block Row	Column	Row
1	1	1	1
1	1	1	2
1	1	1	3

Table 7: ADF: Feature coordinate columns

2.4.4 Spot Content / Spot Sequence: The concept of Reporter

Synthetic sequences, used as proxies for genomic entities, can be deposited in one or more spot locations and array designs. These elements correspond to Reporter objects in MAGE terms (*i.e.*, a subclass of DesignElement), and it is a MIAME requirement to publish the actual sequences physically present on the array. Therefore, a Reporter is uniquely defined by its ID and its sequence. Additional information is also required by the model, such as the role (experimental or control), and, where appropriate, the kind of control it represents.

Reporter Name	Reporter Sequence	Reporter Group [role]	Control Type
R1	ATGGTTGGTTACGTGT	Experimental	
R2	CCGCGTTGCCCCGCC	Experimental	
R3	TCCCTTCCGTTGTCCT	Control	control spike calibration

Table 8: ADF: General case for oligonucleotide based microarrays

Reporter Name	Reporter Database Entry [flybase]	Reporter Group [role]	Control Type
R1	Fb2353	Experimental	
R2	Fb2354	Experimental	
R3	Fb2345	Control	control spike calibration

Table 9: ADF: General case for PCR based microarrays

2.4.5 Genomic Entities of interest: The concept of Composite Element

This section addresses the description of the biological sequence of interest which is interrogated by the synthetic probe (Reporter) sequences. For simple microarray designs, spot location, spot sequence and genomic sequences are directly associated in a one-to-one relationship. Interpretation is straightforward: one location, one probe, one gene or biological entity. For these cases, all layers can be combined in a single spreadsheet, and the ADF can be considered completely and unequivocally represented (see Figure 50 for example). However, with advances in microarray technology enabling high density printing, more elaborate array layouts are possible. Complex association patterns between spot sequences and surveyed genomic sequences are possible. Hybridization signals observed from series of spot sequences can be combined to provide measure estimates about surveyed genomic sequences. The format proposed here is designed to encode simple cases where there is a one-to-one or many-to-one mapping from Reporters (probe sequences) to Composite Elements (biologically relevant sequences). In cases where a many-to-many mapping has been used, a fuller ADF specification will usually be required, using two separate spreadsheets as presented in Section 2.4.6 below (see also Figure 51 and 52 for example).

2.4.6 ADF classification

There are two major classes of array design that can be identified and encoded as ADFs , “simple” and “complex”, as described below:

Simple design:

1. Absence of technical replicates, direct association between representative sequences and genomic sequences:

Figure 4: Simple one-to-one array design graph

2. Technical replicates, and direct association between representative sequences and genomic sequences: In such situations only one spreadsheet is needed, as annotation can be collapsed down a level; description of Composite Element is not required, and the relevant Composite Element columns may be omitted from the ADF.

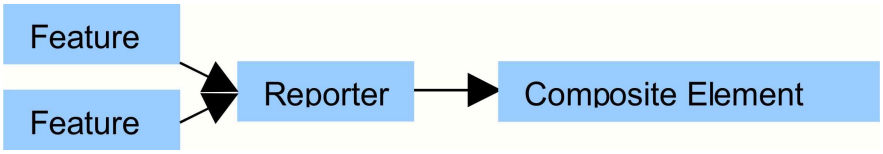

Figure 5: Simple array design graph containing technical replicates

3. Absence of technical replicates, and any genomic sequence being represented by more than one representative sequence. This use-case requires extra columns to describe the Composite Elements, and is only supported for cases where many Reporters map to one Composite Element:

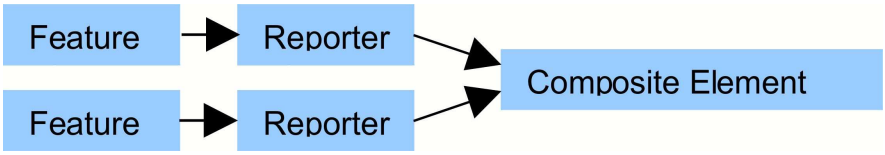

Figure 6: Simple array design with genomic sequences being represented by more than one representative sequence

Complex design:

Presence of technical replicates, and any genomic sequence is represented by more than one synthetic representative sequence:

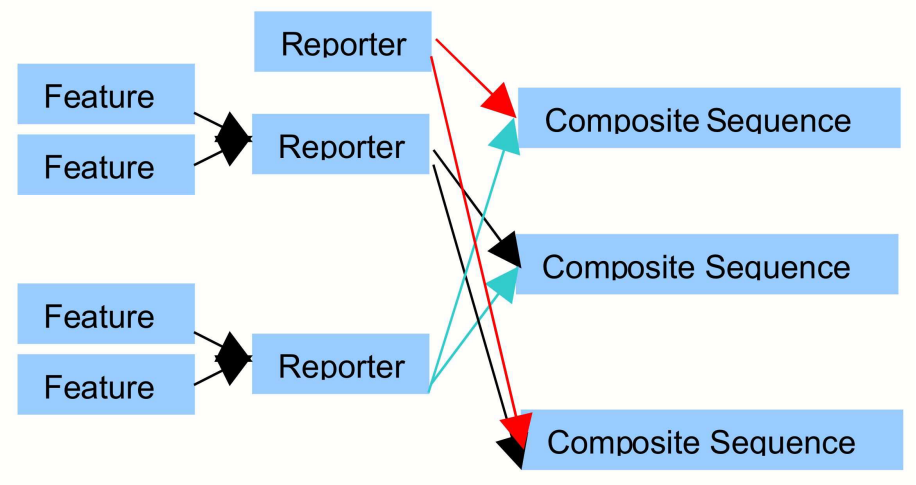

Figure 7: Complex array design graph

This many-to-many relationship of Reporters to Composite Elements requires either the use of a supplementary table to hold these mappings (see Section 3.4, and Figure 51 and 52), or alternatively the array design may be described using the full MAGEv2 object model. Note that this requirement applies to only a very small fraction of array designs currently held in public repositories such as ArrayExpress, and is included here as a future-proofing measure to allow the representation of next-generation array designs. It is anticipated that a single-spreadsheet ADF will suffice for the majority of array designs in use today.

2.5 Protocols

Protocols are encoded as a part of the IDF — see Section 2.2. Protocols are described as free text, with optional fields for hardware, software and a free-text field for contacts. Protocols may have parameters.

2.6 Data files

The MAGE-TAB specification requires that raw data files are provided as binary or ASCII files in their native formats, such as Affymetrix CEL files, Agilent TXT files, or GenePix GPR files, whereas processed data files may be communicated in tab-delimited text format as *data matrix* files (see Section 3.3.7).

2.6.1 Raw data

Raw data files may be binary or ASCII, in native formats as documented by the software manufacturers. These files should be self-describing, *i.e.*, it should be possible to determine the dimensions of the data from the files themselves. For some software types it may be preferable to supply raw data as an Array Data Matrix file (Section 3.3.7). An example of such a use-case would be probe-level data exported from Illumina BeadStudio.

2.6.2 Processed data

Processed data may be supplied in native format, as specified for the raw data. Alternatively, these derived data may be supplied as a Derived Array Data Matrix file (Section 3.3.7). The SDRF columns “Assay Name”, “Scan Name” and “Normalization Name” may be used with tProtocol REF columns to express complex mappings from assays to normalization to individual columns within a data matrix.

2.6.3 Data Matrices

Normally, a MAGE-TAB document will have one data matrix where rows typically represent genes (though they may also represent other biological entities, such as exons or genomic locations), and columns typically represent samples or experimental

conditions. One can think of such a matrix as containing the data that are typically published as supplementary information for a given paper and on which the author would perform analyses such as clustering.

The main feature of data matrices that distinguishes them from arbitrary data files is that columns in such matrices have references to Name objects in SDRF files, for instance to particular raw data files or particular samples. This enables mapping from biomaterials and their characteristics (especially experimental factor values) to individual processed data columns by following the edges in the investigation design graph.

Data matrix files accompanying an SDRF are annotated as such using the SDRF columns “Array Data Matrix File” and “Derived Array Data Matrix File”. The formats of both types of data matrix are the same, and the only distinction between them is the type of data contained therein (unprocessed (raw) and normalized, respectively).

Syntactically, each data matrix file has two header rows, as shown in Figure 10. The first header row contains references to “Name” objects in an SDRF file. All the Names should come from one particular column in the SDRF. That is, each column in the data matrix is marked by unique Names from a particular column in the SDRF. The “Array Data File” and “Derived Array Data File” columns may also be used for this purpose. The second row contains the names of the quantitation types, such as ‘signal’, ‘p-value’, or ‘log ratio(Cy3/Cy5)’ (from the MAGE-TAB perspective, these are simply labels that do not have to have a particular meaning, but normally should be defined in the data processing protocol). The left-most field on the second header row indicates the nature of the identifiers used in the first column, and may be one of the following:

1. “Reporter REF” or “Composite Element REF”, indicating that each row maps to a DesignElement of the given class. It is anticipated that this will be the most common use for these data matrices.
2. A Term Source tag, expressed as “Term Source REF:<tag>” (*e.g.*, “Term Source REF:embl”, where “embl” is the Term Source Name), as defined in the IDF; this is used, for example, to map rows to gene annotation in public databases.
3. A genome build: “Coordinates REF:<version >” where the version build is defined in the same way as other Term Sources in the IDF (*e.g.*, “Coordinates REF:ncbi34”). This heading is used to link row-level data to chromosome coordinates in the absence of gene-level annotation.

Where the row-level annotation is not taken from the array design described by an ADF, MAGE-TAB implementations may create virtual array designs to hold this information.

Using this mapping each column in the summary data matrix can be automatically and concisely annotated by the most important characteristics, such as experimental factor values. An example SDRF is shown in Figure 8, with the corresponding data matrix in Figures 9 and 10.

Sample Name	Characteristics [Organism]	Characteristics [OrganismPart]	Protocol REF	Hybridization Name	ArrayDesign REF	Array Data Matrix File	Protocol REF	Derived Array Data Matrix File
liver 1	Homo sapiens	liver	P-XMPL-1	hyb 1	HG_U95A	CELdata.txt	P-XMPL-2	FGDM.txt
kidney 1	Homo sapiens	kidney	P-XMPL-1	hyb 2	HG_U95A	CELdata.txt	P-XMPL-2	FGDM.txt
brain 1	Homo sapiens	brain	P-XMPL-1	hyb 3	HG_U95A	CELdata.txt	P-XMPL-2	FGDM.txt

Figure 8: Data Matrix example: Measured data on a per-assay basis; derived data in a common file (“FGDM.txt”)

Hybridization REF	hyb 1	hyb 1	hyb 2	hyb 2	hyb 3	hyb 3
Reporter REF	CELIntensity	CELIntensityStdev	CELIntensity	CELIntensityStdev	CELIntensity	CELIntensityStdev
Gene 1	i11	sd11	i21	sd21	i31	sd31
Gene 2	i12	sd12	i22	sd22	i32	sd32
Gene 3	i13	sd13	i23	sd23	i33	sd33
…	…	…	…	…	…	…
Gene n	i1n	sd1n	i2n	sd2n	i3n	sd3n

Figure 9: Data Matrix example: Common “Array Data Matrix File” example linked to the investigation in Figure 8

Array Data REF	Data1.cel	Data1.cel	Data2.cel	Data2.cel	Data3.cel	Data3.cel
Reporter REF	signal	p-value	signal	p-value	signal	p-value
Gene 1	x11	p11	x21	p21	x31	p31
Gene 2	x12	p12	x22	p22	x32	p32
Gene 3	x13	p13	x23	p23	x33	p33
…	…	…	…	…	…	…
Gene n	x1n	p1n	x2n	p2n	x3n	p3n

Figure 10: Data Matrix example: Common “Derived Array Data Matrix File” example linked to the investigation in Figure 8

3 Examples and use-cases

3.1 Investigation Design Format

Top-level information concerning an investigation is included in a single tab-delimited format, an example of which is given below in Table 10. A more complete specification of this format is given in Section 2.2.

MAGE-TAB Version	1.1	
Investigation Title	University of Heidelberg H sapiens TK6	
Investigation Accession	E-MEXP-12	
Investigation Accession Term Source REF	ArrayExpress	
Experimental Design	genetic_modification_design	time_series_design
Experimental Factor Name	Genetic Modification	Incubation Time
Experimental Factor Type	genetic_modification	timepoint
Experimental Factor Term Source REF	MGED Ontology	MGED Ontology
Experimental Factor Term Accession Number	MO_927	MO_738
Person Last Name	Maier	Fleckenstein
Person First Name	Patrick	Katharina
Person Email	patrick.maier@radonk.ma.uni-heidelberg.de	
Person Phone	+496213833773	
Person Address	Theodor-Kutzer-Ufer 1-3	
Person Affiliation	Department of Radiation Oncology, University of Heidelberg	
Person Roles	submitter; investigator	investigator
Person Roles Term Source REF	MGED Ontology	MGED Ontology
Quality Control Type	biological_replicate	
Quality Control Term Source REF	MGED Ontology	
Replicate Type	biological_replicate	
Replicate Term Source REF	MGED Ontology	
Date of Experiment	2005-02-28	
Public Release Date	2006-01-03	
PubMed ID	12345678	
Publication Author List	Patrick Meyer; Katharina Fleckenstein, Li Li; Stephanie Laufs; Jens Zeller; Stefan Fruehauf; Carsten Herskind; Frederik Wenz	
Publication Status	submitted	
Experiment Description	Gene expression of TK6 cells transduced with an oncoretrovirus expressing MDR1 (TK6MDR1) was compared to untransduced TK6 cells and to TK6 cells transduced with an oncoretrovirus expressing the Neomycin resistance gene (TK6neo).	
Protocol Name	GROWTHPRTCL 10653	EXTPRTCL 10654
Protocol Type	grow	nucleic_acid_extraction
Protocol Description	TK6 cells were grown in suspension cultures in RPMI 1640 medium supplemented with…	Approximately 10^6 cells were lysed in RLT buffer (Qiagen). Total RNA was extracted…
Protocol Parameters	media; time	Extracted Product; Amplification
Protocol Term Source REF	MGED Ontology	MGED Ontology
SDRF File	e-mexp-428_tab.txt	
Term Source Name	MGED Ontology	NCI Thesaurus
Term Source File	http://mged.sourceforge.net/ontologies/MGEDontology.php	http://nciterms.nci.nih.gov/NCIBrowser/Dictionary.do
Term Source Version	1.3.0.1	

Table 10: IDF File example

3.2 Conceptual examples for Investigation Design descriptions

In the examples that follow, each complete path through an investigation design graph has been represented by a row in the corresponding table. Column headings in blue denote graph node identifier columns.

3.2.1 Example: Simple Iterated Design

For extremely simple applications such as the example in Figure 11, a table may be as simple as shown in Figure 12, in which the protocol referenced by the identifier P-XMPL-10 should include all the processing needed to get from the source sample to the final assay.

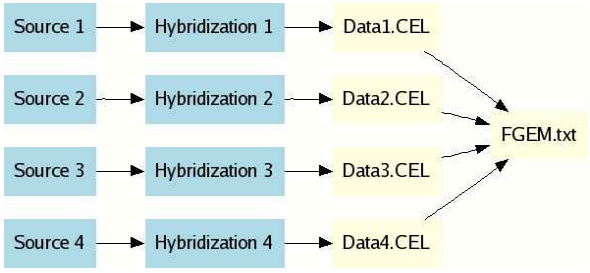

Figure 11: Investigation design graph for a simple iterated design

Source Name	Protocol REF	Hybridization Name	Array Data File	Derived Array Data Matrix File
Source 1	P-XMPL-10	Hybridization 1	Data1.CEL	FGEM.txt
Source 2	P-XMPL-10	Hybridization 2	Data2.CEL	FGEM.txt
Source 3	P-XMPL-10	Hybridization 3	Data3.CEL	FGEM.txt
Source 4	P-XMPL-10	Hybridization 4	Data4.CEL	FGEM.txt

Figure 12: Simple, unstructured representation of sample-assay relationships in Figure 11

Source Name	Protocol REF	Protocol REF	Protocol REF	Protocol REF	Hybridization Name
Source 1	P-XMPL-5	P-XMPL-2	P-XMPL-4	P-XMPL-3	Hybridization 1
Source 2	P-XMPL-5	P-XMPL-2	P-XMPL-4	P-XMPL-3	Hybridization 2
Source 3	P-XMPL-5	P-XMPL-2	P-XMPL-4	P-XMPL-3	Hybridization 3
Source 4	P-XMPL-5	P-XMPL-2	P-XMPL-4	P-XMPL-3	Hybridization 4

Figure 13: Use of repeated protocol columns showing how protocols can be added to a simple iterated design

This very coarse level of granularity in describing experimental procedures may be unwieldy, but this captures the MIAME-required mapping between source and assay, and the procedures used. We regard this as an important use case, for example in the description of simple Affymetrix chip-based investigations. For a higher degree of granularity, where multiple protocols have been used in the processing of Sources through to Assays, it is proposed that these protocols be given in order using repeated "**Protocol REF**" columns as shown in Figure 13.

This corresponds to the MAGE version 1 coding of a BioSource object being referenced directly by a Assay object, via a Treatment having four ordered ProtocolApplications.

A more complex alternative for such simple investigations is to explicitly indicate the materials used and created during the investigation as nodes in the investigation design graph, as shown in Figure 14. Figure 15 describes a similar investigation to that given in Figure 12. The "**Protocol REF**" columns have been omitted for clarity; see below for further examples.

Use of technical replicates may be indicated by branching within the graph, as shown in Figure 16. To indicate that biological replicates were used in an investigation, the investigation design graph will typically be constructed as shown in Figure 11 or Figure 14, with additional sample annotation indicating the relationships between replicate samples. The example given in Section 3.3.3 illustrates how to use experimental factor value annotation for this purpose.

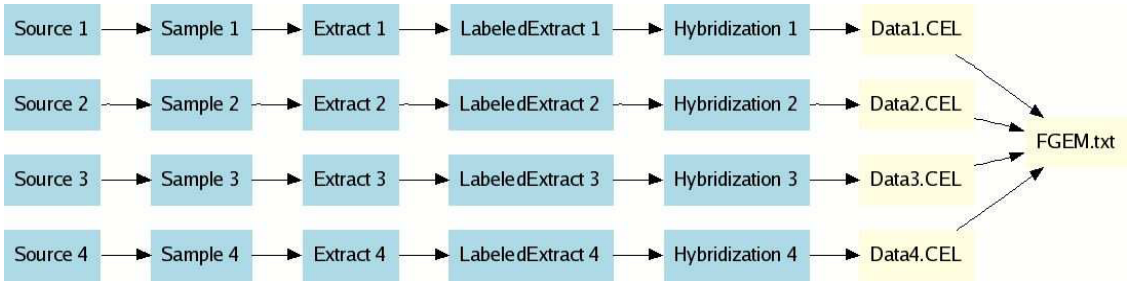

Figure 14: A more complete investigation design graph for a simple iterated design

Source Name	Sample Name	Extract Name	Labeled Extract Name	Hybridization Name	Array Data File	Derived Array Data Matrix File
Source 1	Sample 1	Extract 1	LabeledExtract 1	Hybridization 1	Data1.CEL	FGEM.txt
Source 2	Sample 2	Extract 2	LabeledExtract 2	Hybridization 2	Data2.CEL	FGEM.txt
Source 3	Sample 3	Extract 3	LabeledExtract 3	Hybridization 3	Data3.CEL	FGEM.txt
Source 4	Sample 4	Extract 4	LabeledExtract 4	Hybridization 4	Data4.CEL	FGEM.txt

Figure 15: The design graph in Figure 14, encoded as an SDRF table

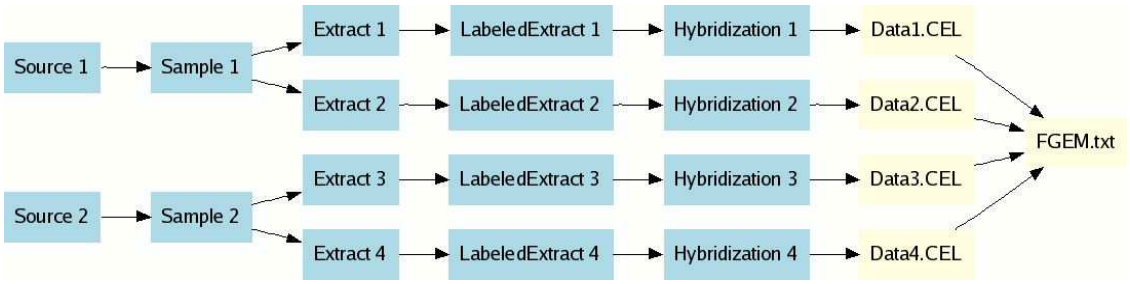

Figure 16: Investigation design graph showing an iterated design incorporating technical replicates

Source Name	Sample Name	Extract Name	Labeled Extract Name	Hybridization Name	Array Data File	Derived Array Data Matrix File
Source 1	Sample 1	Extract 1	LabeledExtract 1	Hybridization 1	Data1.CEL	FGEM.txt
Source 1	Sample 1	Extract 2	LabeledExtract 2	Hybridization 2	Data2.CEL	FGEM.txt
Source 2	Sample 2	Extract 3	LabeledExtract 3	Hybridization 3	Data3.CEL	FGEM.txt
Source 2	Sample 2	Extract 4	LabeledExtract 4	Hybridization 4	Data4.CEL	FGEM.txt

Figure 17: The design graph in Figure 16, encoded as an SDRF table

Further examples are given below of the way in which spreadsheets should be constructed to represent a variety of investigation design graphs. Graph nodes referring to the data files ("**Array Data File**" and "**Derived Array Data File**") have been omitted in all subsequent examples for the sake of clarity.

3.2.2 Example: Iterated Design single channel with sample pooling

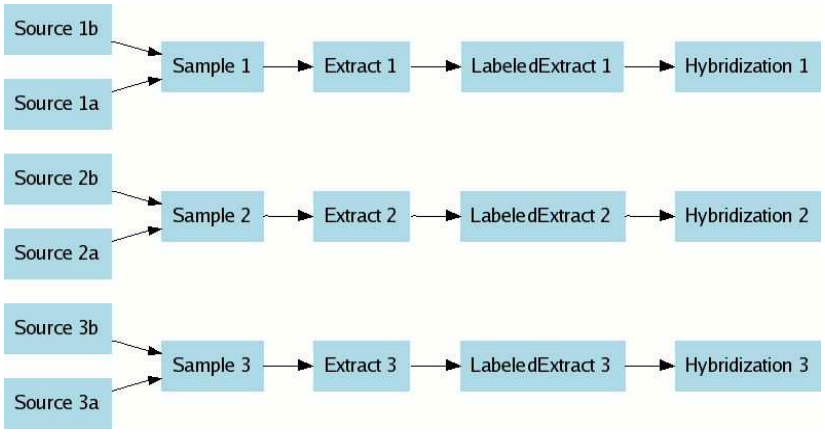

Figure 18: Investigation design graph showing an iterated design, single channel, sample pooling

Source Name	Sample Name	Extract Name	Labeled Extract Name	Hybridization Name
Source 1a	Sample 1	Extract 1	LabeledExtract 1	Hybridization 1
Source 1b	Sample 1	Extract 1	LabeledExtract 1	Hybridization 1
Source 2a	Sample 2	Extract 2	LabeledExtract 2	Hybridization 2
Source 2b	Sample 2	Extract 2	LabeledExtract 2	Hybridization 2
Source 3a	Sample 3	Extract 3	LabeledExtract 3	Hybridization 3
Source 3b	Sample 3	Extract 3	LabeledExtract 3	Hybridization 3

Figure 19: The design graph in Figure 18, encoded as an SDRF table

3.2.3 Example: Iterated Design dual channel

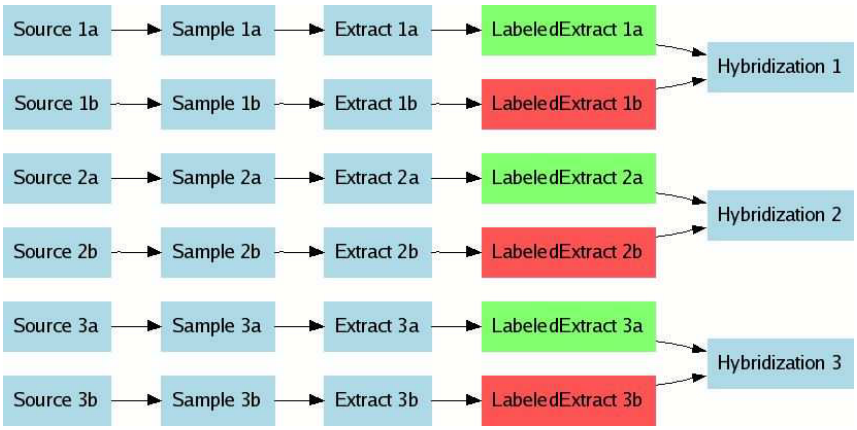

Figure 20: Investigation design graph showing an iterated design, dual channel. LabeledExtract-Dye associations can be added as a separate "Label" column

Source Name	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
Source 1a	Sample 1a	Extract 1a	LabeledExtract 1a	Cy3	Hybridization 1
Source 1b	Sample 1b	Extract 1b	LabeledExtract 1b	Cy5	Hybridization 1
Source 2a	Sample 2a	Extract 2a	LabeledExtract 2a	Cy3	Hybridization 2
Source 2b	Sample 2b	Extract 2b	LabeledExtract 2b	Cy5	Hybridization 2
Source 3a	Sample 3a	Extract 3a	LabeledExtract 3a	Cy3	Hybridization 3
Source 3b	Sample 3b	Extract 3b	LabeledExtract 3b	Cy5	Hybridization 3

Figure 21: The design graph in Figure 20, encoded as an SDRF table

3.2.4 Example: Iterated Design, dual channel with dye swap

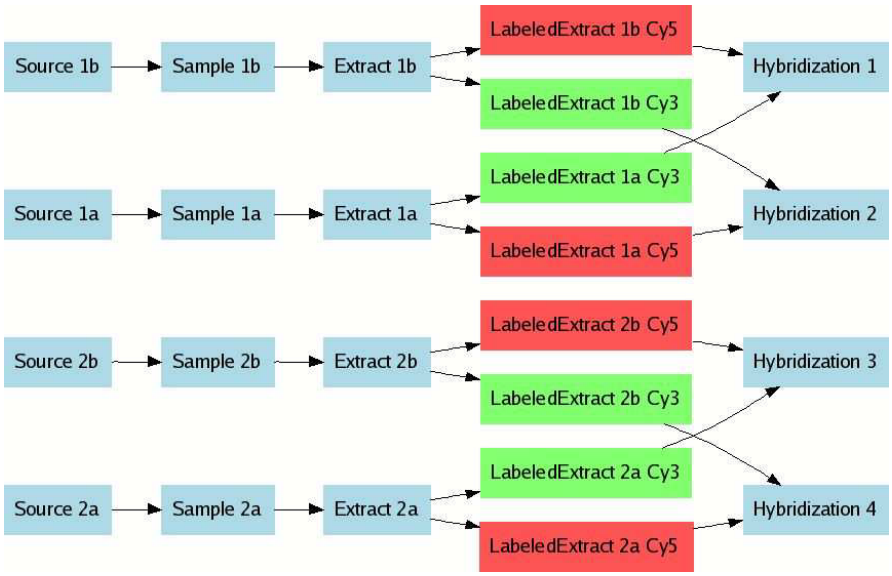

Figure 22: Investigation design graph for an iterated design, dual channel with dye swap

Source Name	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
Source 1a	Sample 1a	Extract 1a	LabeledExtract 1a Cy3	Cy3	Hybridization 1
Source 1b	Sample 1b	Extract 1b	LabeledExtract 1b Cy5	Cy5	Hybridization 1
Source 1a	Sample 1a	Extract 1a	LabeledExtract 1a Cy5	Cy5	Hybridization 2
Source 1b	Sample 1b	Extract 1b	LabeledExtract 1b Cy3	Cy3	Hybridization 2
Source 2a	Sample 2a	Extract 2a	LabeledExtract 2a Cy3	Cy3	Hybridization 3
Source 2b	Sample 2b	Extract 2b	LabeledExtract 2b Cy5	Cy5	Hybridization 3
Source 2a	Sample 2a	Extract 2a	LabeledExtract 2a Cy5	Cy5	Hybridization 4
Source 2b	Sample 2b	Extract 2b	LabeledExtract 2b Cy3	Cy3	Hybridization 4

Figure 23: The design graph in Figure 22, encoded as an SDRF table

3.2.5 Example: Iterated Design with reference

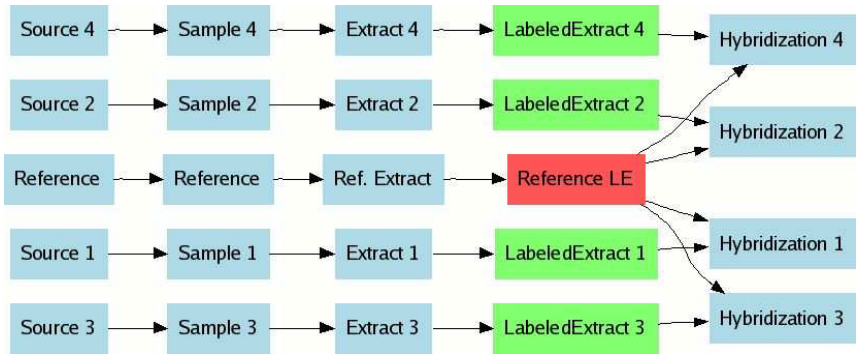

Figure 24: Investigation design graph showing an iterated design with reference

Source Name	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
Source 1	Sample 1	Extract 1	LabeledExtract 1	Cy3	Hybridization 1
Source 2	Sample 2	Extract 2	LabeledExtract 2	Cy3	Hybridization 2
Source 3	Sample 3	Extract 3	LabeledExtract 3	Cy3	Hybridization 3
Source 4	Sample 4	Extract 4	LabeledExtract 4	Cy3	Hybridization 4
Reference	Reference	Ref. Extract	Reference LE	Cy5	Hybridization 1
Reference	Reference	Ref. Extract	Reference LE	Cy5	Hybridization 2
Reference	Reference	Ref. Extract	Reference LE	Cy5	Hybridization 3
Reference	Reference	Ref. Extract	Reference LE	Cy5	Hybridization 4

Figure 25: The design graph in Figure 24, encoded as an SDRF table

3.2.6 Example: Iterated Design with a reference and dye swap

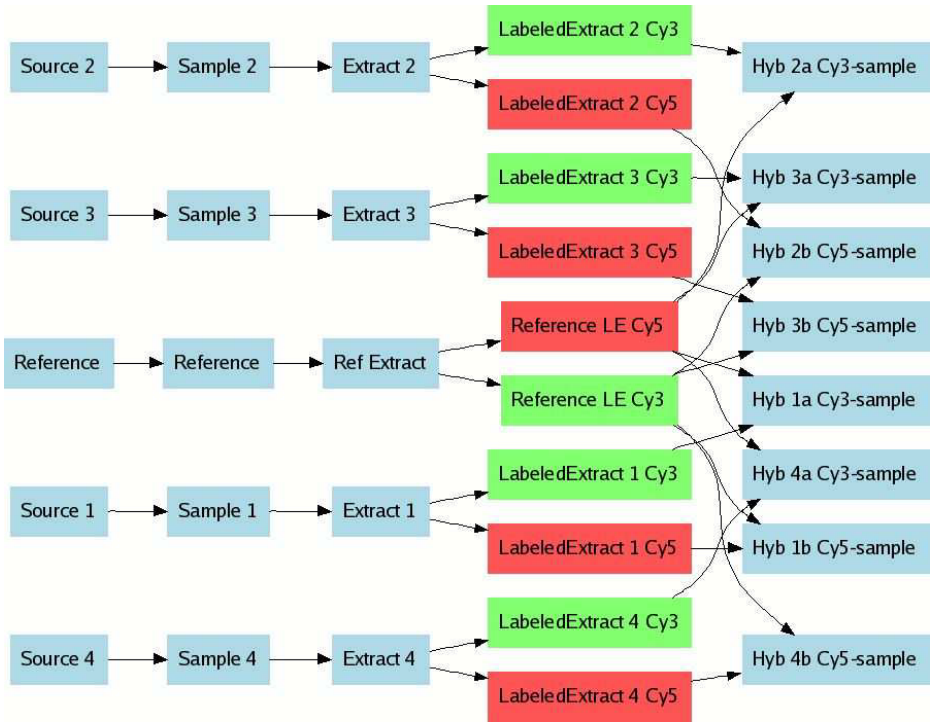

Figure 26: Investigation design graph showing iterated design with reference and dye swap

Source Name	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
Source 1	Sample 1	Extract 1	LabeledExtract 1 Cy3	Cy3	Hyb 1a Cy3-sample
Source 1	Sample 1	Extract 1	LabeledExtract 1 Cy5	Cy5	Hyb 1b Cy5-sample
Source 2	Sample 2	Extract 2	LabeledExtract 2 Cy3	Cy3	Hyb 2a Cy3-sample
Source 2	Sample 2	Extract 2	LabeledExtract 2 Cy5	Cy5	Hyb 2b Cy5-sample
Source 3	Sample 3	Extract 3	LabeledExtract 3 Cy3	Cy3	Hyb 3a Cy3-sample
Source 3	Sample 3	Extract 3	LabeledExtract 3 Cy5	Cy5	Hyb 3b Cy5-sample
Source 4	Sample 4	Extract 4	LabeledExtract 4 Cy3	Cy3	Hyb 4a Cy3-sample
Source 4	Sample 4	Extract 4	LabeledExtract 4 Cy5	Cy5	Hyb 4b Cy5-sample
Reference	Reference	Ref Extract	Reference LE Cy5	Cy5	Hyb 1a Cy3-sample
Reference	Reference	Ref Extract	Reference LE Cy5	Cy5	Hyb 2a Cy3-sample
Reference	Reference	Ref Extract	Reference LE Cy5	Cy5	Hyb 3a Cy3-sample
Reference	Reference	Ref Extract	Reference LE Cy5	Cy5	Hyb 4a Cy3-sample
Reference	Reference	Ref Extract	Reference LE Cy3	Cy3	Hyb 1b Cy5-sample
Reference	Reference	Ref Extract	Reference LE Cy3	Cy3	Hyb 2b Cy5-sample
Reference	Reference	Ref Extract	Reference LE Cy3	Cy3	Hyb 3b Cy5-sample
Reference	Reference	Ref Extract	Reference LE Cy3	Cy3	Hyb 4b Cy5-sample

Figure 27: The design graph in Figure 26, encoded as an SDRF table

3.2.7 Example: Iterated Design with pooled reference

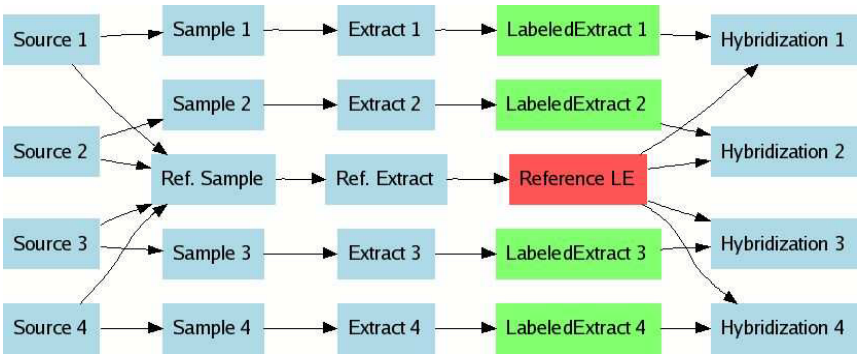

Figure 28: Investigation design graph showing iterated design with pooled reference

Source Name	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
Source 1	Sample 1	Extract 1	LabeledExtract 1	Cy3	Hybridization 1
Source 2	Sample 2	Extract 2	LabeledExtract 2	Cy3	Hybridization 2
Source 3	Sample 3	Extract 3	LabeledExtract 3	Cy3	Hybridization 3
Source 4	Sample 4	Extract 4	LabeledExtract 4	Cy3	Hybridization 4
Source 1	Ref. Sample	Ref. Extract	Reference LE	Cy5	Hybridization 1
Source 2	Ref. Sample	Ref. Extract	Reference LE	Cy5	Hybridization 2
Source 3	Ref. Sample	Ref. Extract	Reference LE	Cy5	Hybridization 3
Source 4	Ref. Sample	Ref. Extract	Reference LE	Cy5	Hybridization 4

Figure 29: The design graph in Figure 28, encoded as an SDRF table

3.2.8 Example: Loop Design

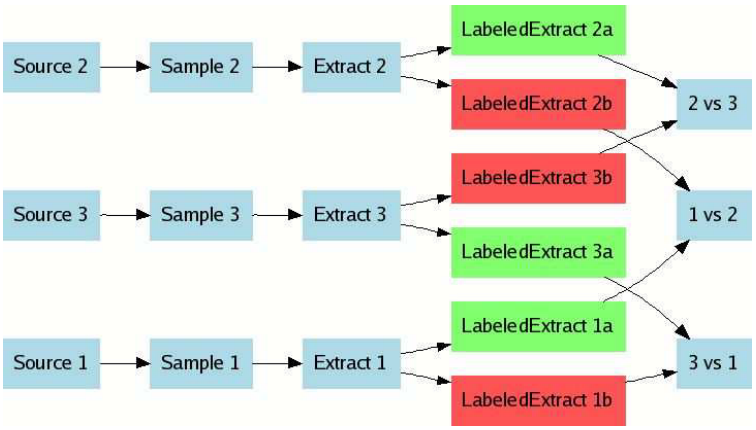

Figure 30: Investigation design graph showing loop design

Source Name	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
Source 1	Sample 1	Extract 1	LabeledExtract 1a	Cy3	1 vs 2
Source 2	Sample 2	Extract 2	LabeledExtract 2b	Cy5	1 vs 2
Source 2	Sample 2	Extract 2	LabeledExtract 2a	Cy3	2 vs 3
Source 3	Sample 3	Extract 3	LabeledExtract 3b	Cy5	2 vs 3
Source 1	Sample 1	Extract 1	LabeledExtract 1b	Cy5	3 vs 1
Source 3	Sample 3	Extract 3	LabeledExtract 3a	Cy3	3 vs 1

Figure 31: The design graph in Figure 30, encoded as an SDRF table

3.2.9 Example: Loop Design with dye swap

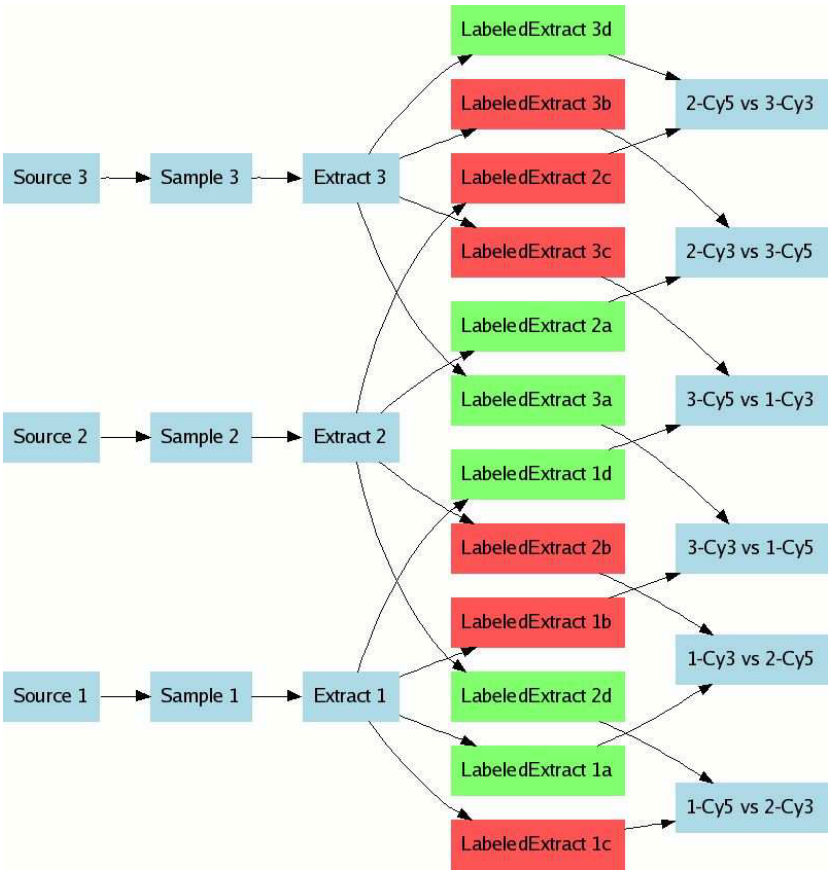

Figure 32: Investigation design graph showing loop design with dye swap

Source Name	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
Source 1	Sample 1	Extract 1	LabeledExtract 1a	Cy3	1-Cy3 vs 2-Cy5
Source 2	Sample 2	Extract 2	LabeledExtract 2b	Cy5	1-Cy3 vs 2-Cy5
Source 1	Sample 1	Extract 1	LabeledExtract 1c	Cy5	1-Cy5 vs 2-Cy3
Source 2	Sample 2	Extract 2	LabeledExtract 2d	Cy3	1-Cy5 vs 2-Cy3
Source 2	Sample 2	Extract 2	LabeledExtract 2a	Cy3	2-Cy3 vs 3-Cy5
Source 3	Sample 3	Extract 3	LabeledExtract 3b	Cy5	2-Cy3 vs 3-Cy5
Source 2	Sample 2	Extract 2	LabeledExtract 2c	Cy5	2-Cy5 vs 3-Cy3
Source 3	Sample 3	Extract 3	LabeledExtract 3d	Cy3	2-Cy5 vs 3-Cy3
Source 1	Sample 1	Extract 1	LabeledExtract 1b	Cy5	3-Cy3 vs 1-Cy5
Source 3	Sample 3	Extract 3	LabeledExtract 3a	Cy3	3-Cy3 vs 1-Cy5
Source 1	Sample 1	Extract 1	LabeledExtract 1d	Cy3	3-Cy5 vs 1-Cy3
Source 3	Sample 3	Extract 3	LabeledExtract 3c	Cy5	3-Cy5 vs 1-Cy3

Figure 33: The design graph shown in Figure 16, encoded as an SDRF table

3.2.10 Example: Complex Time Series

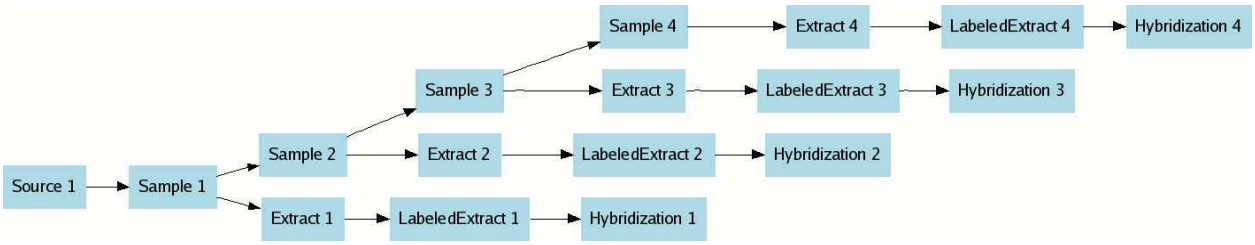

Figure 34: Investigation design graph showing complex time series

Source Name	Protocol REF	Sample Name	Extract Name	Labeled Extract Name	Hybridization Name	Factor Value [Incubation Time]	Unit [TimeUnit]	Term Source REF
Source 1	P-XMPL-1	Sample 1	Extract 1	LabeledExtract 1	Hybridization 1	0	hours	MGED Ontology
Source 1	P-XMPL-1	Sample 2	Extract 2	LabeledExtract 2	Hybridization 2	1	hours	MGED Ontology
Source 1	P-XMPL-1	Sample 3	Extract 3	LabeledExtract 3	Hybridization 3	2	hours	MGED Ontology
Source 1	P-XMPL-1	Sample 4	Extract 4	LabeledExtract 4	Hybridization 4	3	hours	MGED Ontology

Figure 35: The cascading time series design graph in Figure 34, shown as a flattened SDRF representation

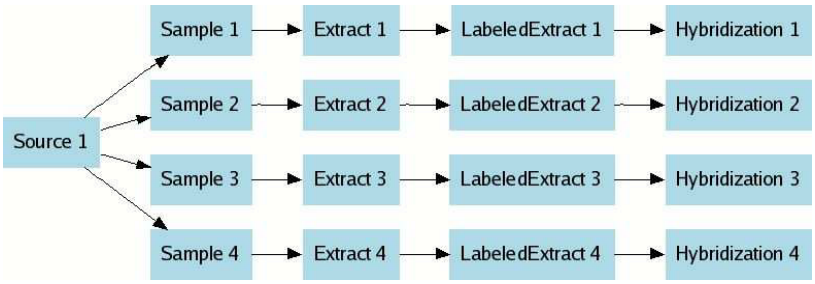

Figure 36: A simpler representation of the investigation design graph in Figure 34

For simplicity, we can collapse the cascading graph into a flatter structure, using Time as a "Factor Value" (Figure 35). This would translate into the simplified investigation design graph in Figure 36. See Section 3.3, covering real examples for further discussion of "Protocol REF" columns.

3.2.11 Example: Real-world example (ArrayExpress experiment E-MIMR-12)

In the real world, the conceptual examples scale up to structures such as the following. The graph for E-MIMR-12 can be coded as shown in Figure 37. In this example, Sources are split into Samples, which are then pooled into Extracts. The grey shading indicates the materials linked to a single assay.

Source Name	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
BS_TKAC_13	BSM_TKAC_01m	BSM_TKAC_22p	BSM_TKAC_22p	biotin	HFB2002012101A
BS_TKAC_13	BSM_TKAC_02m	BSM_TKAC_23p	BSM_TKAC_23p	biotin	HFB2002012102A
BS_TKAC_13	BSM_TKAC_03m	BSM_TKAC_24p	BSM_TKAC_24p	biotin	HFB2002012103A
BS_TKAC_13	BSM_TKAC_04m	BSM_TKAC_25p	BSM_TKAC_25p	biotin	HFB2002012104A
BS_TKAC_13	BSM_TKAC_05m	BSM_TKAC_26p	BSM_TKAC_26p	biotin	HFB2002012105A
BS_TKAC_13	BSM_TKAC_06m	BSM_TKAC_27p	BSM_TKAC_27p	biotin	HFB2002012106A
BS_TKAC_13	BSM_TKAC_07m	BSM_TKAC_28p	BSM_TKAC_28p	biotin	HFB2002012107A
BS_TKAC_13	BSM_TKAC_08m	BSM_TKAC_29p	BSM_TKAC_29p	biotin	HFB2002012108A
BS_TKAC_13	BSM_TKAC_09m	BSM_TKAC_30p	BSM_TKAC_30p	biotin	HFB2002012109A
BS_TKAC_13	BSM_TKAC_10m	BSM_TKAC_31p	BSM_TKAC_31p	biotin	HFB2002012110A
BS_TKAC_14	BSM_TKAC_01n	BSM_TKAC_22p	BSM_TKAC_22p	biotin	HFB2002012101A
BS_TKAC_14	BSM_TKAC_02n	BSM_TKAC_23p	BSM_TKAC_23p	biotin	HFB2002012102A
BS_TKAC_14	BSM_TKAC_03n	BSM_TKAC_24p	BSM_TKAC_24p	biotin	HFB2002012103A
BS_TKAC_14	BSM_TKAC_04n	BSM_TKAC_25p	BSM_TKAC_25p	biotin	HFB2002012104A
BS_TKAC_14	BSM_TKAC_05n	BSM_TKAC_26p	BSM_TKAC_26p	biotin	HFB2002012105A
BS_TKAC_14	BSM_TKAC_06n	BSM_TKAC_27p	BSM_TKAC_27p	biotin	HFB2002012106A
BS_TKAC_14	BSM_TKAC_07n	BSM_TKAC_28p	BSM_TKAC_28p	biotin	HFB2002012107A
BS_TKAC_14	BSM_TKAC_08n	BSM_TKAC_29p	BSM_TKAC_29p	biotin	HFB2002012108A
BS_TKAC_14	BSM_TKAC_09n	BSM_TKAC_30p	BSM_TKAC_30p	biotin	HFB2002012109A
BS_TKAC_14	BSM_TKAC_10n	BSM_TKAC_31p	BSM_TKAC_31p	biotin	HFB2002012110A
BS_TKAC_15	BSM_TKAC_01o	BSM_TKAC_22p	BSM_TKAC_22p	biotin	HFB2002012101A
BS_TKAC_15	BSM_TKAC_02o	BSM_TKAC_23p	BSM_TKAC_23p	biotin	HFB2002012102A
BS_TKAC_15	BSM_TKAC_03o	BSM_TKAC_24p	BSM_TKAC_24p	biotin	HFB2002012103A
BS_TKAC_15	BSM_TKAC_04o	BSM_TKAC_25p	BSM_TKAC_25p	biotin	HFB2002012104A
BS_TKAC_15	BSM_TKAC_05o	BSM_TKAC_26p	BSM_TKAC_26p	biotin	HFB2002012105A
BS_TKAC_15	BSM_TKAC_06o	BSM_TKAC_27p	BSM_TKAC_27p	biotin	HFB2002012106A
BS_TKAC_15	BSM_TKAC_07o	BSM_TKAC_28p	BSM_TKAC_28p	biotin	HFB2002012107A
BS_TKAC_15	BSM_TKAC_08o	BSM_TKAC_29p	BSM_TKAC_29p	biotin	HFB2002012108A
BS_TKAC_15	BSM_TKAC_09o	BSM_TKAC_30p	BSM_TKAC_30p	biotin	HFB2002012109A
BS_TKAC_15	BSM_TKAC_10o	BSM_TKAC_31p	BSM_TKAC_31p	biotin	HFB2002012110A
BS_TKAC_16	BSM_TKAC_01q	BSM_TKAC_22p	BSM_TKAC_22p	biotin	HFB2002012101A
BS_TKAC_16	BSM_TKAC_02q	BSM_TKAC_23p	BSM_TKAC_23p	biotin	HFB2002012102A
BS_TKAC_16	BSM_TKAC_03q	BSM_TKAC_24p	BSM_TKAC_24p	biotin	HFB2002012103A
BS_TKAC_16	BSM_TKAC_04q	BSM_TKAC_25p	BSM_TKAC_25p	biotin	HFB2002012104A
BS_TKAC_16	BSM_TKAC_05q	BSM_TKAC_26p	BSM_TKAC_26p	biotin	HFB2002012105A
BS_TKAC_16	BSM_TKAC_06q	BSM_TKAC_27p	BSM_TKAC_27p	biotin	HFB2002012106A
BS_TKAC_16	BSM_TKAC_07q	BSM_TKAC_28p	BSM_TKAC_28p	biotin	HFB2002012107A
BS_TKAC_16	BSM_TKAC_08q	BSM_TKAC_29p	BSM_TKAC_29p	biotin	HFB2002012108A
BS_TKAC_16	BSM_TKAC_09q	BSM_TKAC_30p	BSM_TKAC_30p	biotin	HFB2002012109A
BS_TKAC_16	BSM_TKAC_10q	BSM_TKAC_31p	BSM_TKAC_31p	biotin	HFB2002012110A

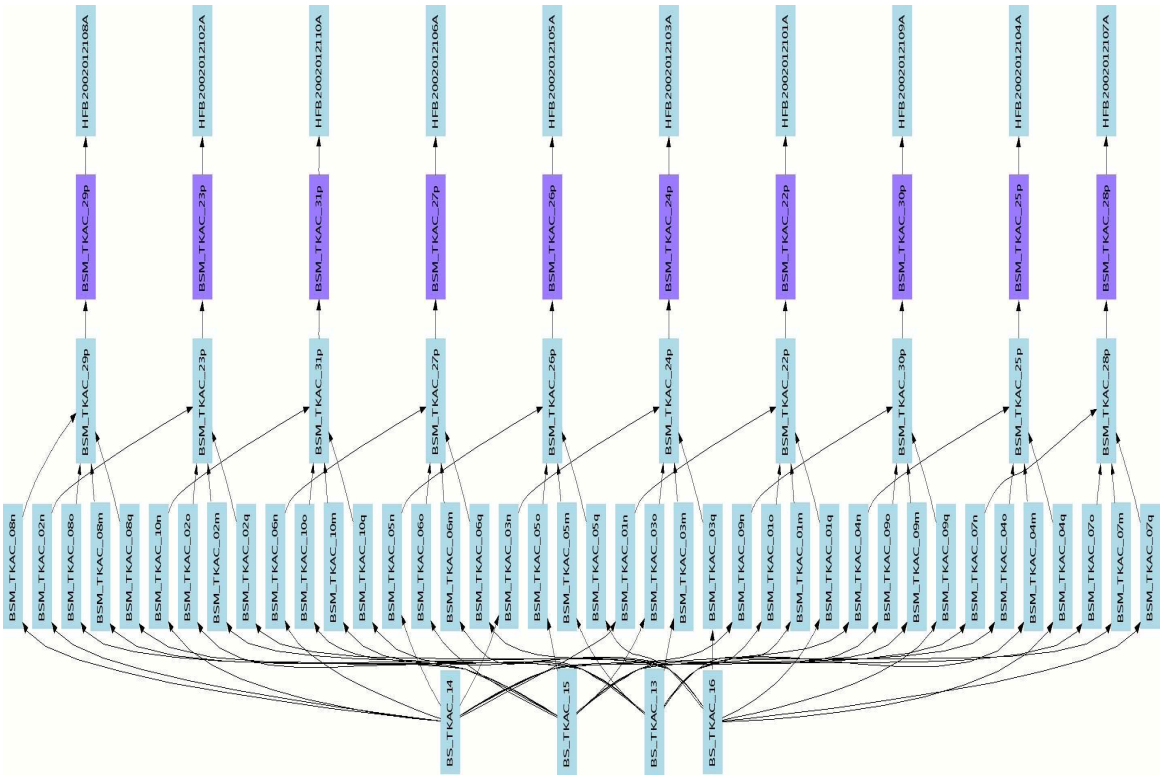

Figure 37: ArrayExpress experiment E-MIMR-12

3.3 Fully encoded examples of investigation design

3.3.1 Example: Real-world example (ArrayExpress experiment E-TABM-21)

Each node in the investigation design graph must be represented by an appropriate “Name” column, with the graph edges given as “Protocol REF” columns. Several other column types may be used to convey sample annotation. Data files are referenced using “Array Data File” and “Derived Array Data File” columns, shown here but omitted from subsequent examples for clarity (Figure 38). Note that this example has Genotype as experimental factor.

Source Name	Characteristics [Organism]	Term Source REF	Characteristics [Genotype]	Protocol REF	Extract Name	Protocol REF	Labeled Extract Name	Label	Hybridization Name	Array Data File	Derived Array Data File	Factor Value [Genotype]
CHIP_322_A	Arabidopsis thaliana	NCBI_tax	wild_type	P-TABM-41	CHIP_322_A	P-TABM-43	CHIP_322_A	biotin	CHIP_322_A	CHIP_322_A.CEL	CHIP_322_A.CHP	wild_type
CHIP_322_B	Arabidopsis thaliana	NCBI_tax	wild_type	P-TABM-41	CHIP_322_B	P-TABM-43	CHIP_322_B	biotin	CHIP_322_B	CHIP_322_B.CEL	CHIP_322_B.CHP	wild_type
CHIP_322_C	Arabidopsis thaliana	NCBI_tax	wild_type	P-TABM-41	CHIP_322_C	P-TABM-43	CHIP_322_C	biotin	CHIP_322_C	CHIP_322_C.CEL	CHIP_322_C.CHP	wild_type
CHIP_323_A	Arabidopsis thaliana	NCBI_tax	co	P-TABM-41	CHIP_323_A	P-TABM-43	CHIP_323_A	biotin	CHIP_323_A	CHIP_323_A.CEL	CHIP_323_A.CHP	co
CHIP_323_B	Arabidopsis thaliana	NCBI_tax	co	P-TABM-41	CHIP_323_B	P-TABM-43	CHIP_323_B	biotin	CHIP_323_B	CHIP_323_B.CEL	CHIP_323_B.CHP	co
CHIP_323_C	Arabidopsis thaliana	NCBI_tax	co	P-TABM-41	CHIP_323_C	P-TABM-43	CHIP_323_C	biotin	CHIP_323_C	CHIP_323_C.CEL	CHIP_323_C.CHP	co
CHIP_324_A	Arabidopsis thaliana	NCBI_tax	ft	P-TABM-41	CHIP_324_A	P-TABM-43	CHIP_324_A	biotin	CHIP_324_A	CHIP_324_A.CEL	CHIP_324_A.CHP	ft
CHIP_324_B	Arabidopsis thaliana	NCBI_tax	ft	P-TABM-41	CHIP_324_B	P-TABM-43	CHIP_324_B	biotin	CHIP_324_B	CHIP_324_B.CEL	CHIP_324_B.CHP	ft
CHIP_324_C	Arabidopsis thaliana	NCBI_tax	ft	P-TABM-41	CHIP_324_C	P-TABM-43	CHIP_324_C	biotin	CHIP_324_C	CHIP_324_C.CEL	CHIP_324_C.CHP	ft
CHIP_326_A	Arabidopsis thaliana	NCBI_tax	wild_type	P-TABM-41	CHIP_326_A	P-TABM-43	CHIP_326_A	biotin	CHIP_326_A	CHIP_326_A.CEL	CHIP_326_A.CHP	wild_type
CHIP_326_B	Arabidopsis thaliana	NCBI_tax	wild_type	P-TABM-41	CHIP_326_B	P-TABM-43	CHIP_326_B	biotin	CHIP_326_B	CHIP_326_B.CEL	CHIP_326_B.CHP	wild_type
CHIP_326_C	Arabidopsis thaliana	NCBI_tax	wild_type	P-TABM-41	CHIP_326_C	P-TABM-43	CHIP_326_C	biotin	CHIP_326_C	CHIP_326_C.CEL	CHIP_326_C.CHP	wild_type
CHIP_327_A	Arabidopsis thaliana	NCBI_tax	co	P-TABM-41	CHIP_327_A	P-TABM-43	CHIP_327_A	biotin	CHIP_327_A	CHIP_327_A.CEL	CHIP_327_A.CHP	co
CHIP_327_B	Arabidopsis thaliana	NCBI_tax	co	P-TABM-41	CHIP_327_B	P-TABM-43	CHIP_327_B	biotin	CHIP_327_B	CHIP_327_B.CEL	CHIP_327_B.CHP	co
CHIP_327_C	Arabidopsis thaliana	NCBI_tax	co	P-TABM-41	CHIP_327_C	P-TABM-43	CHIP_327_C	biotin	CHIP_327_C	CHIP_327_C.CEL	CHIP_327_C.CHP	co
CHIP_328_A	Arabidopsis thaliana	NCBI_tax	ft	P-TABM-41	CHIP_328_A	P-TABM-43	CHIP_328_A	biotin	CHIP_328_A	CHIP_328_A.CEL	CHIP_328_A.CHP	ft
CHIP_328_B	Arabidopsis thaliana	NCBI_tax	ft	P-TABM-41	CHIP_328_B	P-TABM-43	CHIP_328_B	biotin	CHIP_328_B	CHIP_328_B.CEL	CHIP_328_B.CHP	ft
CHIP_328_C	Arabidopsis thaliana	NCBI_tax	ft	P-TABM-41	CHIP_328_C	P-TABM-43	CHIP_328_C	biotin	CHIP_328_C	CHIP_328_C.CEL	CHIP_328_C.CHP	ft

Figure 38: ArrayExpress experiment E-TABM-21 — a simple iterated single-channel design

3.3.2 Example: Real-world example (ArrayExpress experiment E-MEXP-252)

Figure 39 shows how experimental factor values may be associated with a given assay. In this case, the “Behavior” Characteristic would be listed in the IDF as an experimental factor (Section 2.2). Other material characteristics may be included in a similar fashion, using as many columns as necessary to encode the annotation. In this way, variations in the materials used to generate the data set may be captured.

Source Name	Characteristics [OrganismPart]	Term Source REF	Characteristics [Sex]	Term Source REF	Characteristics [Organism]	Term Source REF	Characteristics [Behavior]	Protocol REF	Extract Name	Protocol REF	Labeled Extract Name	Label	Hybridization Name	Factor Value [Behavior]
Comb builder1	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	comb builder	P-MEXP-123	CB1	P-MEXP-125	CB1 Cy3	Cy3	CB1 vs G1	comb builder
Guard1	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	guard	P-MEXP-123	G1	P-MEXP-125	G1 Cy5	Cy5	CB1 vs G1	guard
Comb builder7	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	comb builder	P-MEXP-123	CB7	P-MEXP-125	CB7 Cy3	Cy3	CB7 vs N7	comb builder
Nurse7	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	nurse	P-MEXP-123	N7	P-MEXP-125	N7 Cy5	Cy5	CB7 vs N7	nurse
Comb builder1	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	comb builder	P-MEXP-123	CB1	P-MEXP-125	CB1 Cy5	Cy5	F1 vs CB1	comb builder
Forager1	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	forager	P-MEXP-123	F1	P-MEXP-125	F1 Cy3	Cy3	F1 vs CB1	forager
Forager1	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	forager	P-MEXP-123	F1	P-MEXP-125	F1 Cy5	Cy5	G1 vs F1	forager
Guard1	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	guard	P-MEXP-123	G1	P-MEXP-125	G1 Cy3	Cy3	G1 vs F1	guard
Guard7	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	guard	P-MEXP-123	G7	P-MEXP-125	G7 Cy3	Cy3	G7 vs U7	guard
Undertaker7	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	undertaker	P-MEXP-123	U7	P-MEXP-125	U7 Cy5	Cy5	G7 vs U7	undertaker
Guard7	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	guard	P-MEXP-123	G7	P-MEXP-125	G7 Cy5	Cy5	N7 vs G7	guard
Nurse7	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	nurse	P-MEXP-123	N7	P-MEXP-125	N7 Cy3	Cy3	N7 vs G7	nurse
Comb builder7	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	comb builder	P-MEXP-123	CB7	P-MEXP-125	CB7 Cy5	Cy5	U7 vs CB7	comb builder
Undertaker7	brain	SAEL	female	MGED Ontology	Apis mellifera	NCBI Taxonomy	undertaker	P-MEXP-123	U7	P-MEXP-125	U7 Cy3	Cy3	U7 vs CB7	undertaker

Figure 39: ArrayExpress experiment E-MEXP-252 (excerpt). A series of loop design investigations comparing the brains of worker bees which have different behaviors. Two sets of “loops” are shown here.

3.3.3 Example: Real-world example (ArrayExpress experiment E-MEXP-549)

Biological replicates should be represented by distinct biological sources, grouped together by common experimental factor values. An example of this is given in Figure 40, where biological replicates (*e.g.*, ARP1-0h, ARP2-0h and ARP3-0h) are represented as distinct Sources sharing the same factor value ("Time", in this example). In comparison, technical replicates are represented by branching of the investigation design graph at intermediate steps of the experimental processing, as shown in Figure 18.

Source Name	Characteristics [CellLine]	Characteristics [CellType]	Term Source REF	Characteristics [DiseaseState]	Term Source REF	Characteristics [Organism]	Term Source REF	Characteristics [StrainOrLine]	Hybridization Name	Array Design REF	Factor Value [Time]	Unit [TimeUnit]	Term Source REF
ARP1-0h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP1-0h	A-AFFY-33	0	hours	MO
ARP2-0h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP2-0h	A-AFFY-33	0	hours	MO
ARP3-0h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP3-0h	A-AFFY-33	0	hours	MO
ARP1-2h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP1-2h	A-AFFY-33	2	hours	MO
ARP2-2h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP2-2h	A-AFFY-33	2	hours	MO
ARP3-2h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP3-2h	A-AFFY-33	2	hours	MO
ARP1-4h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP1-4h	A-AFFY-33	4	hours	MO
ARP2-4h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP2-4h	A-AFFY-33	4	hours	MO
ARP3-4h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP3-4h	A-AFFY-33	4	hours	MO
ARP1-6h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP1-6h	A-AFFY-33	6	hours	MO
ARP2-6h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP2-6h	A-AFFY-33	6	hours	MO
ARP3-6h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP3-6h	A-AFFY-33	6	hours	MO
ARP1-8h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP1-8h	A-AFFY-33	8	hours	MO
ARP2-8h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP2-8h	A-AFFY-33	8	hours	MO
ARP3-8h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP3-8h	A-AFFY-33	8	hours	MO
ARP1-10h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP1-10h	A-AFFY-33	10	hours	MO
ARP2-10h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP2-10h	A-AFFY-33	10	hours	MO
ARP3-10h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP3-10h	A-AFFY-33	10	hours	MO
ARP1-12h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP1-12h	A-AFFY-33	12	hours	MO
ARP2-12h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP2-12h	A-AFFY-33	12	hours	MO
ARP3-12h	MOLT4	T cell	CTO	acute lymphoblastic leukemia	NCI_meta	Homo sapiens	NCBI_tax	CFARP011	H_ARP3-12h	A-AFFY-33	12	hours	MO

Figure 40: ArrayExpress experiment E-MEXP-54 9. Biological replicates are indicated by shared experimental factor values ("Time"). Protocols have been omitted for clarity.

3.3.4 Example: Treatment variation

Variations in the treatments used can also be indicated in the SDRF. These can be represented as distinct protocols, as shown in Figure 41.

Sample Name	Protocol REF	Extract Name	Labeled Extract Name	Label	Hybridization Name
ethanol-treated	P-XMPL-2	Eth. Ext	Eth. LE Cy3	Cy3	Hyb 1
butanol-treated	P-XMPL-1	But. Ext	But. LE Cy3	Cy5	Hyb 1
ethanol-treated	P-XMPL-2	Eth. Ext	Eth. LE Cy5	Cy3	Hyb 2
butanol-treated	P-XMPL-1	But. Ext	But. LE Cy5	Cy5	Hyb 2

Figure 41: Simple compound treatment using different protocols for different compounds. Two samples used in a dye swap

Sample Name	Protocol REF	Parameter Value [Compound]	Extract Name	Labeled Extract Name	Label	Hybridization Name
ethanol-treated	P-XMPL-3	ethanol	Eth. Ext	Eth. LE Cy3	Cy3	Hyb 1
butanol-treated	P-XMPL-3	butanol	But. Ext	But. LE Cy3	Cy5	Hyb 1
ethanol-treated	P-XMPL-3	ethanol	Eth. Ext	Eth. LE Cy5	Cy3	Hyb 2
butanol-treated	P-XMPL-3	butanol	But. Ext	But. LE Cy5	Cy5	Hyb 2

Figure 42: Same investigation as depicted in Figure 41, using protocol parameters rather than separate protocols.

Source Name	Protocol REF	Parameter Value [Temperature]	Unit [TemperatureUnit]	Term Source REF	Sample Name	Extract Name	Labeled Extract Name	Label	Hybridization Name
Source 1	P-XMPL-2	22	degree_C	MO	22 deg	22 deg. Ext	22 deg. Cy3	Cy3	Hyb 1
Source 1	P-XMPL-2	37	degree_C	MO	37 deg	37 deg. Ext	37 deg. Cy3	Cy5	Hyb 1
Source 1	P-XMPL-2	22	degree_C	MO	22 deg	22 deg. Ext	22 deg. Cy5	Cy3	Hyb 2
Source 1	P-XMPL-2	37	degree_C	MO	37 deg	37 deg. Ext	37 deg. Cy5	Cy5	Hyb 2

Figure 43: Example of Parameters with units.

Alternatively, a single protocol could be used with different parameter values. In this case the parameter would have to be linked to its protocol via the IDF header file (Figure 42). Parameter values may be specified with units (Figure 43), and included in the Factor Values for an investigation by creating a separate “Factor Value” column containing duplicated values.

3.3.5 Example: Variation in treatment application (ChIP-chip)

For investigations where some treatments are not applied to all the samples, gaps, separated by tabs, may be left in the table to indicate this. For example, ChIP-chip investigations typically compare a chromatin immunoprecipitate to the whole genomic DNA extract from which it was derived:

Source Name	Protocol REF	Extract Name	Protocol REF	Extract Name	Protocol REF	Labeled Extract Name	Label	Hybridization Name
yeast 1	P-XMPL-1	extract 1	P-XMPL-2	ip 1	P-XMPL-3	ip 1	Cy3	Hyb 1
yeast 1	P-XMPL-1	extract 1	->	->	P-XMPL-3	extract 1	Cy5	Hyb 1
yeast 1	P-XMPL-1	extract 2	P-XMPL-2	ip 2	P-XMPL-3	ip 2	Cy3	Hyb 2
yeast 1	P-XMPL-1	extract 2	->	->	P-XMPL-3	extract 2	Cy5	Hyb 2

Figure 44: Variation in treatment represented by empty fields.

In this example, P-XMPL-1, P-XMPL-2 and P-XMPL-3 reference a genomic DNA extraction protocol, an immunoprecipitation protocol, and a labeling protocol, respectively. These protocol types are specified in the IDF.

3.3.6 Example: Multi-layered SDRF example

The investigation shown in Figure 45 can also be represented in a two-layer structure, in which the immunoprecipitate and whole cell extract sample treatments are explicitly separated:

Source Name	Protocol REF	Extract Name	Labeled Extract Name	Label	Hybridization Name
yeast 1	P-XMPL-1	extract 1	extract 1	Cy5	Hyb 1
yeast 1	P-XMPL-1	extract 2	extract 2	Cy5	Hyb 2

Figure 45: Two-layered example ChIP-chip investigation, sample to assay

Extract Name	Protocol REF	Extract Name	Labeled Extract Name	Label	Hybridization Name
extract 1	P-XMPL-2	ip 1	ip 1	Cy3	Hyb 1
extract 2	P-XMPL-2	ip 2	ip 2	Cy3	Hyb 2

Figure 46: Two-layered example ChIP-chip investigation, extract to assay

This splitting into multiple spreadsheets can take place on any Name column.

3.3.7 Example: Association of data files with assays and samples

Measured and derived data files can be associated with a specific assay. In each case, either an “Array Design File” or “Array Design REF” column is needed, referencing either an included ADF file or an identifier in a public repository such as ArrayExpress, respectively. Note that the repository can optionally be indicated using a “Term Source REF” column:

Source Name	Hybridization Name	Array Design REF	Term Source REF	Array Data File	Derived Array Data File
Sample 1	Hybridization 1	A-AFFY-33	ArrayExpress	Data1.CEL	Data1.CHP
Sample 2	Hybridization 2	A-AFFY-33	ArrayExpress	Data2.CEL	Data2.CHP
Sample 3	Hybridization 3	A-AFFY-33	ArrayExpress	Data3.CEL	Data3.CHP
Sample 4	Hybridization 4	A-AFFY-33	ArrayExpress	Data4.CEL	Data4.CHP

Figure 47: Data files linked to assays on a per-assay basis (Affymetrix)

Source Name	Hybridization Name	Array Design File	Image File	Array Data File	Derived Array Data File
Sample 1	Hybridization 1	a-mexp-128_adf.txt	Data1.TIFF	Data1.gpr	Data1.txt
Sample 2	Hybridization 2	a-mexp-128_adf.txt	Data2.TIFF	Data2.gpr	Data2.txt
Sample 3	Hybridization 3	a-mexp-128_adf.txt	Data3.TIFF	Data3.gpr	Data3.txt
Sample 4	Hybridization 4	a-mexp-128_adf.txt	Data4.TIFF	Data4.gpr	Data4.txt

Figure 48: Data files linked to assays on a per-assay basis (GenePix)

In addition, it is common to include normalized data from multiple assays in a single common file. Please see Figure 9, above, for an example.

3.3.8 Example: Technology type - high-throughput sequencing

MAGE-TAB v1.1 adds support for technology types other than gene expression, e.g., high throughput sequencing. For experiments using such technology types, the SDRF should have an “Assay Name” column instead of a “Hybridization Name” column. The “Assay Name” column can be followed by a “Technology Type” column that describes the specific technology used. The following table shows a hypothetical experiment using high-throughput sequencing:

Source Name	Assay Name	Technology Type	Term Source REF	Protocol REF	Array Data File	Derived Array Data File
finch 1	marchesa 1	sequencing	EFO	Solexa Data Acquisition	run1.fastq	run1_norm_log2score.txt.gz
finch 2	pinta 2	sequencing	EFO	Solexa Data Acquisition	run2.fastq	run2_norm_log2score.txt.gz
finch 3	marchesa 1	sequencing	EFO	Solexa Data Acquisition	run3.fastq	run3_norm_log2score.txt.gz
finch 4	marchesa 2	sequencing	EFO	Solexa Data Acquisition	run4.fastq	run4_norm_log2score.txt.gz
finch 5	santiago 1	sequencing	EFO	Solexa Data Acquisition	run5.fastq	run5_norm_log2score.txt.gz

Figure 49: Example of Assays with associated technology type for sequencing experiments

3.4 Examples of array design descriptions — Array Description Format — ADF

The aim of the ADF component is to describe a microarray design in a spreadsheet or, for complex cases, a set of spreadsheets. Conceptually, microarray designs are devised to measure presence and/or abundance of genomic sequence entities in biological samples. Genomic sequences of interest are represented by one or more synthetic sequences which are in turn arranged in one or more physical locations in the two-dimensional space of a microarray surface. Therefore, to fully describe a microarray layout, information about genomic sequences, synthetic sequences, physical position on array and relationships (mappings) between those must be captured.

In this section we only give two examples of ADF. For a more formal specification see Section 2.4. Figure 50 shows a simple case, where there is a one-to-many (or one-to-one) mapping between Composite Elements and Reporters.

Block Column	Block Row	Column	Row	Reporter Name	Reporter Sequence	Reporter Group [role]	Control Type	Control Type Term Source REF	Composite Element Name
1	1	1	1	R1	ATGGTTGGTTACGTGT	experimental			PTEN
1	1	1	2	R2	CCGCGTTGCCCCGCC	experimental			PAX2
1	1	1	3	R3	CGTAGCTGATCGATGA	experimental			WWOX
1	1	1	4	R4	GGTTGGCTGAGATCGT	experimental			MAPK8
1	1	2	1	R1	ATGGTTGGTTACGTGT	experimental			PTEN
1	1	2	2	R2	CCGCGTTGCCCCGCC	experimental			PAX2
1	1	2	3	R3	CGTAGCTGATCGATGA	experimental			WWOX
1	1	2	4	R4	GGTTGGCTGAGATCGT	experimental			MAPK8
...	...	...	...	...	...	...	...	...	...
4	6	20	20	462020	TCCCTTCCGTTGTCCT	control	control_spike_calibration	MGED Ontology	

Figure 50: Simple design: one Reporter / one Composite Element relationship — use of a single spreadsheet

Note how the information about Reporter and Composite Element is duplicated, to indicate the fact the every synthetic sequence is spotted more than one time on the array. Figure 51, in contrast, illustrates the use of two spreadsheets to capture complex many-to-many mappings between Reporters and Composite Elements. This is provided as a future-proofing measure for cases which cannot be described concisely using the simple layout illustrated in Figure 50. Note in Figure 52 how the relationship between synthetic sequences (Reporter) and the genomic sequence of interest (Composite Element) is provided: as a semi-colon (;) separated list of Reporters contributing to the signal, indicated using the “Map2Reporters” column.

In complex situations such as this, the solution presented here has two advantages:

- the redundant information for describing Composite Element is kept to a minimum;
- the number of columns necessary in the Reporter – Composite Element spreadsheet (Figure 52) is kept constant.
- creation of Final Gene Expression Matrix would be made easier since mapping to Composite Element is already done (see Section 2.6.3).

Block Column	Block Row	Column	Row	Reporter Name	Reporter Sequence	Reporter Group [role]	Control Type	Control Type Term Source REF
1	1	1	1	R1	ATGGTTGGTTACGTGT	experimental		
1	1	1	2	R2	CCGCGTTGCCCCGCC	experimental		
1	1	1	3	R3	TCCCTTCCGTTGTCCT	experimental		
1	1	1	4	R4	CGTAGCTGATCGATGA	experimental		
1	1	1	5	R5	GGTTGGCTGAGATCGT	experimental		
1	1	1	6	R6	AATTAGCTAGAATGTTT	experimental		
1	1	1	7	R1	ATGGTTGGTTACGTGT	experimental		
1	1	1	8	R2	CCGCGTTGCCCCGCC	experimental		
1	1	1	9	R3	TCCCTTCCGTTGTCCT	experimental		
1	1	1	10	R4	CGTAGCTGATCGATGA	experimental		
1	1	1	11	R5	GGTTGGCTGAGATCGT	experimental		
1	1	1	12	R6	AATTAGCTAGAATGTTT	experimental		
...	...	...	...	...	...	...	...	...
8	4	12	12	R15550	TTGGTGGTGGTGGCCA	control	control_spike_calibration	MGED Ontology

Figure 51: Complex design: Reporters can be combined in more than one way to create Composite Elements. The Feature/Reporter spreadsheet is shown here

Composite Element Name	Map2Reporters	Composite Element Database Entry [refseq]	Composite Element Comment
WWOX transcript 1	R1;R3;R4	NM_016373	WW domain containing oxidoreductase
WWOX transcript 2	R2;R5	NM_130791	WW domain containing oxidoreductase
PTEN	R6;R4424	NM_000314	mutated in multiple advanced cancers 1

Figure 52: Complex design, showing the Reporter/Composite Element spreadsheet linked to the design in Figure 51.

3.5 Real-world example of a complete MAGE-TAB document

See the following files in the attachment:

- IDF: `e-mexp-428 v1.0.idf`,
- SDRF: `e-mexp-428sdrf v1.0.txt`,
- normalized data matrix: `e-mexp-428data v1.0.txt`,

Note that the ADF is not needed in this example because the SDRF references an array design in ArrayExpress. However, a separate ADF example document is included: `a-mexp-586adf_excerpt v1.0.txt`.

4 Correspondence to Other Models

4.1 Mapping from MAGE-TAB to MAGEv1.1

See the included file, MAGE-TAB_to_MAGEv1.1.txt, for a mapping from MAGE-TAB version 1.1 to MAGE version 1.1.

4.2 Changes from MAGE-TAB v1.0 to MAGE-TAB v1.1

1. Clarify how file packaging should work and how file references are interpreted.
2. Clarification that double quotes as escaping characters must be used to enclose entire cells.
3. Clarification on MAGE-TAB file character encodings.
4. Relax restrictions on object identifier format.
5. Clarify potential uses of Comment[] for extensibility in all MAGE-TAB files.
6. Add a MAGE-TAB Version field to IDF.
7. Added 'Investigation Accession' and 'Investigation Accession Term Source REF' to the IDF.
8. Specify ordering and cardinality of headers in SDRF and IDF files.
9. Replace “Hybridization Name” column with “Assay Name” colum in the SDRF.
10. Change other SDRF column types to support multi-technology investigations.
11. Allow multiple Characteristics of the same category in the SDRF.
12. Allow term source ref columns in ADF.
13. Allow explicit demarcation of ADF sections.
14. Clarify that factor values cannot be explicitly linked to corresponding characteristics or protocol parameters.
15. Clarify that differentially dimensioned arrays are allowed in the same investigation.
16. Explicitly allow blank lines.
17. Clarify quoting of column headers.
18. Disallow '->' to stand for an empty field.

Note that the MAGE-TAB 1.1 format is not backwards compatible with the 1.0 version, due primarily to changes 9. and 18. These changes are simple enough, however, that parsers should be able to preserve backwards compatibility by supporting both variations and allowing validation components to warn of violations to the rules.