14. Information Model

14.6 Export XML Schema

14.6.1 Schema design

The MoReq2010® export XML schema is used when exporting entities from an MCRS as described in **11. Export**. The schema describes how entities are exported, in what order, and the structure of the resulting XML data.

The schema inherently reflects the design considerations behind the export function in MoReq2010®. An export is envisaged as a continuous stream of entity data rather than as sets of individual datafiles. This steam is structured to ensure that, on import, an MCRS need only process it once with a single sequential pass through the data.

The XML export format is therefore optimised for output by walking through an MCRS solution's internal data structures, and for import using stream based parsing technologies (for example, with a SAX based XML parser, or equivalent) rather than document based parsing (using a DOM based XML parser, or equivalent). Sequential import is necessary because it cannot be assumed by the importing MCRS that the whole of the export data will be small enough to allow it to be loaded into system memory simultaneously.

14.6.2 Exporting large amounts of data

The approach taken to export derives, in part, from the need to efficiently export large and sometimes even huge amounts of data from an MCRS. Depending on the operating environment into which this data is exported, it may become necessary to split the exported data across more than one datafile, so as to ensure manageable datafile sizes. This is explained in **11.2.3 Use of XML** and is also shown in **Figure 14a**.

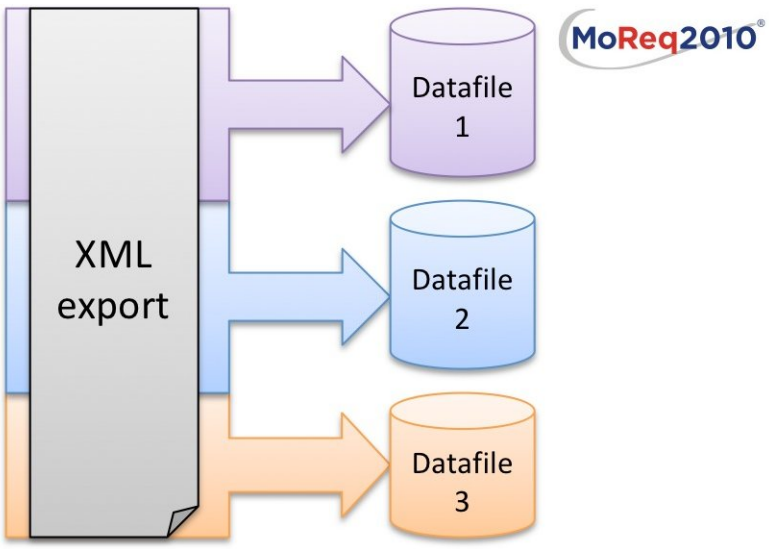

***Figure 14a** - A large XML export may be split across multiple datafiles*

MoReq2010® does not specify how many or what sized datafiles should be created when exports are split, and export data may be broken at any arbitrary point. However, an export

Copyright © 2010 & 2011 DLM Forum Foundation, all rights reserved

stream should always be split in such a way that the content of each datafile follows on from that of the previous datafile. In other words, it must be possible to recreate the original export in its entirety by simply joining the content of the datafiles back together, or by transmitting the datafiles sequentially as a single unified stream of data.

Exported datafiles may be encrypted and/or compressed. Again, MoReq2010® does not specify any particular approach or technology.

14.6.3 Anatomy of an export

Together, **Figures 14b** and **14c** give a high level representation of the structured content of an XML export. **Figure 14b** gives the first part of this representation.

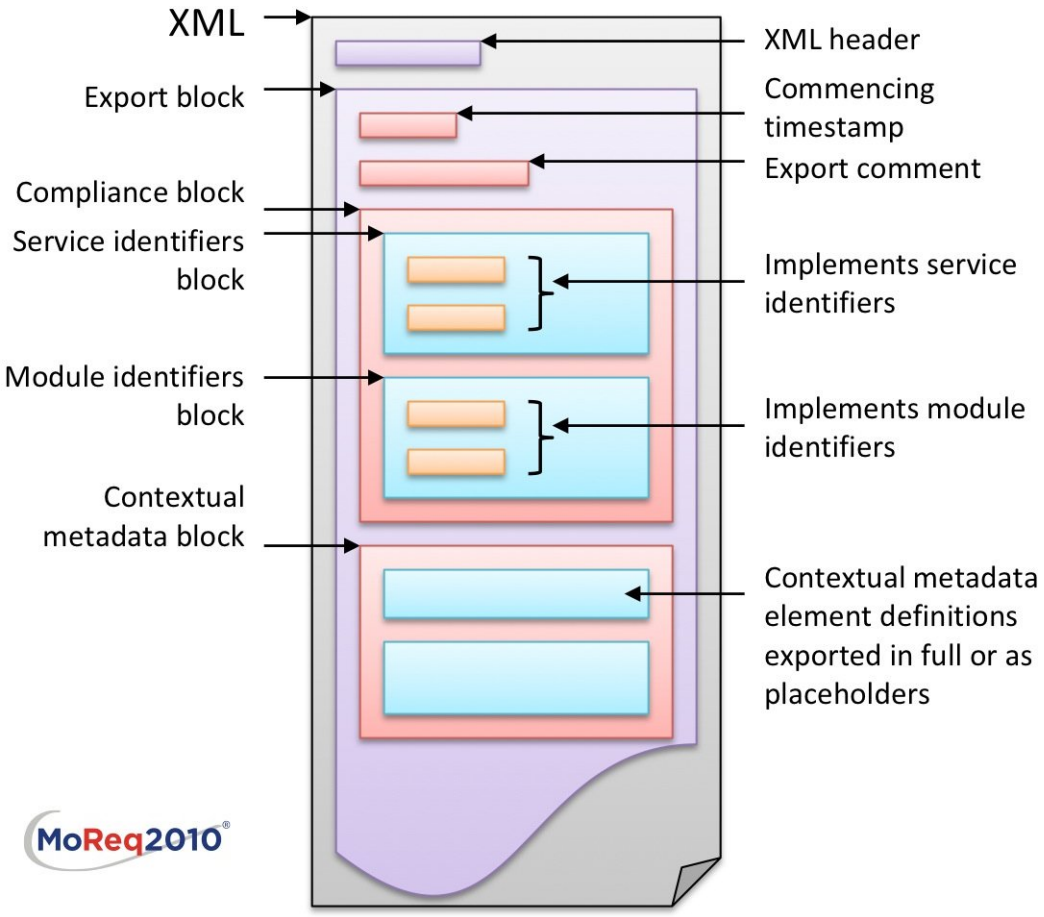

***Figure 14b** - The first part of the structured content of an XML export datafile*

As **Figure 14b** shows, each export is effectively a long fully formed XML datafile that can be validated against the MoReq2010® XML schema. The datafile is comprised of an XML header followed by an export block.

The XML header contains the usual XML declarations, and should also include a Unicode declaration, as all MCRS solutions must provide support for Unicode under **R2.4.8**. The export block contains the MCRS data that is being exported and is explained in greater detail below.

Figure 14c follows on from **Figure 14b** and shows the second part of a representation of the XML structure inside an export.

Copyright © 2010 & 2011 DLM Forum Foundation, all rights reserved

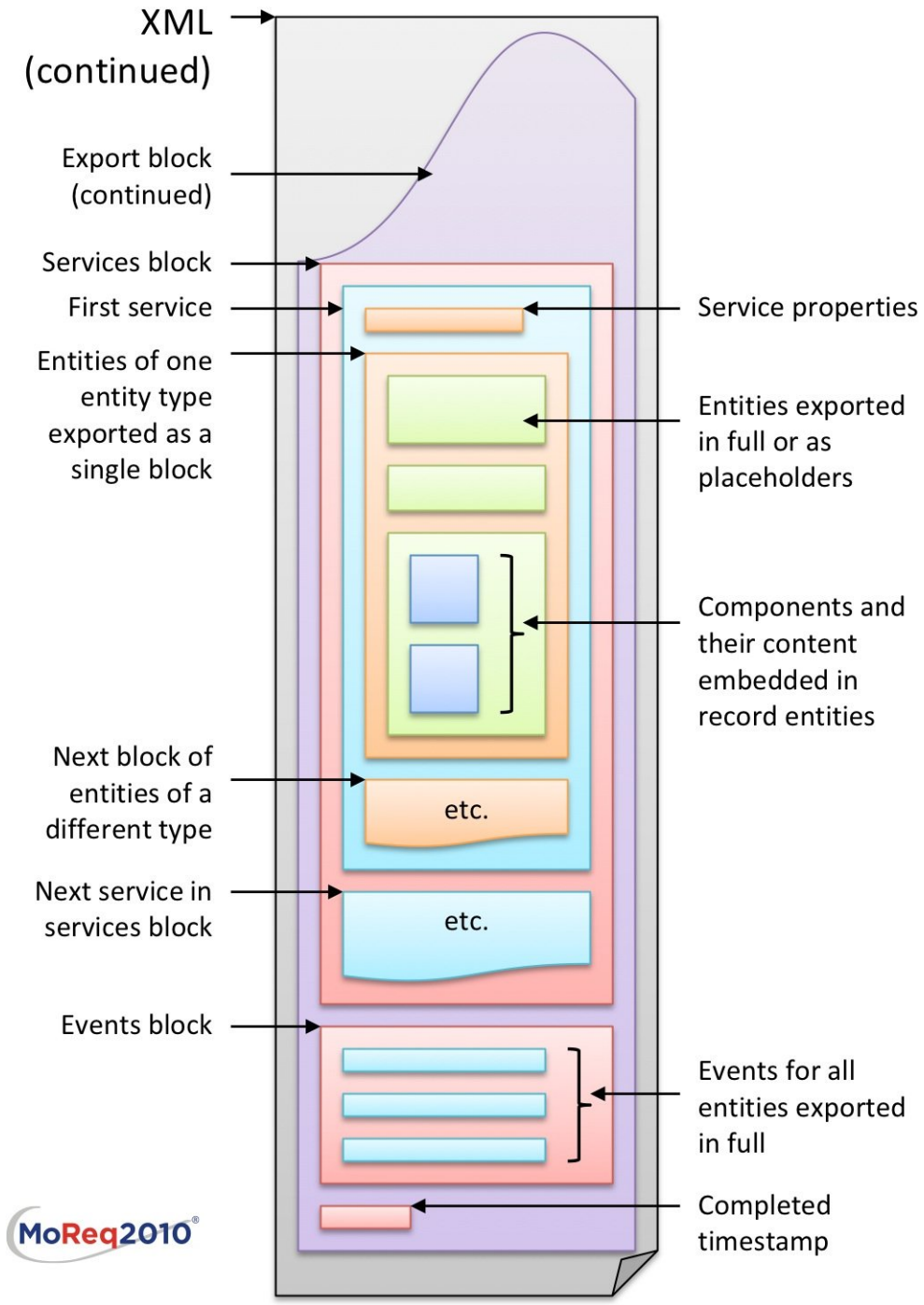

***Figure 14c** - The second part of the structured content of an XML export datafile*

Any XML export datafile should be capable of validation against the MoReq2010® XML schema. In practice, this is not a single schema but rather multiple schemas. The biggest XML schema covers the core services of MoReq2010®, while additional supplementary XML schemas provide coverage for the additional plug-in modules, or extension modules, supported by the MCRS.

A single XML schema is available that includes all other schemas, and is updated as additional modules are released.

Copyright © 2010 & 2011 DLM Forum Foundation, all rights reserved

14.6.4 The export block

The export block contains the body of the export. The "Export" XML element that wraps the export block includes the Export Identifier as an XML attribute, see **R11.4.4**. Each export from an MCRS is uniquely identified. Throughout the XML schema, some metadata such as the Export Identifier are expressed as XML attributes. This is mostly confined to System Identifiers and Language Identifiers.

The export block contains the following sub-sections:

- The export comment provided by the user performing the export, see **R11.4.5**;
- The export commencing timestamp, see **R11.4.5**;
- A block of compliance information;
- A block of contextual metadata element definitions;
- A block of services and their entities;
- A block of events; and
- The export completing timestamp, see **R11.4.5**.

The compliance block comes first and contains a list of the services and modules that are implemented by the MCRS exporting the data. These are also included later under the metadata for each service. This compliance information is specified at the start of the export block so that the MCRS importing the data can immediately determine whether or not to proceed to load the remainder of the XML data. One or more of the services or modules listed in the compliance block may be incompatible with those of the MCRS performing the import. The importing MCRS checks this by reading the compliance block and can then stop processing at that point.

The contextual metadata block contains the definitions of contextual metadata elements that are used by the entities being exported. These will be included as placeholders, unless they have been selected for export in full.

Contextual metadata element definitions are defined in this separate block, ahead of the services block, so that the importing MCRS can correctly anticipate and interpret the datatypes and values of contextual metadata elements when they are encountered in the remainder of the data.

Because they are exported outside their respective service block, contextual metadata element definitions have an additional XML attribute which is used to identify the service to which they belong.

The contextual metadata block is followed by a block of services each containing the entity data for the entities that are being exported. The composition of each of these service blocks is explained under **14.6.5 Service blocks**.

The event block is the final block within the export block and is followed only by the Export Completed Timestamp, see **R11.4.6**.

The event block collectively contains all of the events for every other entity that is being exported in full. Events are listed separately from services as several entities may participate in the same event, including entities from different services. This ensures that each event is only exported once, regardless of how many participating entities it has.

Copyright © 2010 & 2011 DLM Forum Foundation, all rights reserved

14.6.5 Service blocks

The services block is made up of a series of individual service blocks. Each service block represents the data from a single service or bundle of services, as defined under **R2.4.1**.

The make up of each service block is the same. At the start of the block are the service properties. These are the system metadata and access control list for the service itself. If the whole service is being exported in full then its contextual metadata and events are also exported.

The service properties are followed by blocks of entities. There is a single block of entities for each of the entity types managed by the service (see the rationale to **R2.4.9**). The entities for each entity type, placed together inside the same block, may be exported in full or as placeholders.

Where entities are exported in full their contextual metadata and events are also exported. Components are encapsulated inside record entities, along with their content.

14.6.6 Necessary data

It is important to ensure that only necessary data is included in any export. If an entity is exported then only those related entities that provide context should be exported with it (see the summary under **11.2.13 Export summary table**).

Unnecessary entities should not be included in the export. Nor should entities be exported in full if MoReq2010® only requires that they be exported as placeholders. Similarly, no entity should be included twice in any export.

For example, a contextual metadata element definition should only be included in the contextual metadata block if has been selected by the user to be exported in full, or as a placeholder if one or more of the other entities being exported in full have a contextual metadata element with that definition. The contextual metadata element definition should not be included if none of the other data in the export refer to it.

Having been included in the contextual metadata block, contextual metadata element definitions are not also included under their respective services (note that the XML schema prevents this).

14.6.7 Validation

Although the MoReq2010® XML schema can be used to validate the structure and format of data exported from an MCRS, being well formed and validated alone does not ensure that the data has been correctly exported. There are many instances where the XML schema alone is unable to interpret whether appropriate processing rules have been applied. For example, the XML schema does not check whether there are events in the events block for all entities exported in full.

Similarly, a record or child aggregation will have a "ClassId" element if a class has been directly associated with it overriding its default classification, and will not have a "ClassId" element if it inherits its class from its parent aggregation (see **5.2.2 Inheriting classification**). Validation alone cannot determine whether inheritance has been correctly applied in the export XML. Validation by XML schema should therefore be seen as only part of any proof of correctness of XML data exported from an MCRS.

Copyright © 2010 & 2011 DLM Forum Foundation, all rights reserved

13. Glossary of Terms

These are additional glossary terms included from **14.6 Export XML Schema**.

Term	Explanation and relationship to general concepts
DOM	(*acronym*) Document Object Model for **XML**. A DOM based XML parser allows random access to XML data by loading it as a single "document" containing a tree like structure of **XML elements**. While **DOM** based XML parsers are more flexible than **SAX** based parsers they are not considered suitable for processing large amounts of sequential data, such as **export data** from an **MCRS**, as the whole of the XML document must first be loaded into memory or stored locally.
Element	(*noun*) See **metadata schema** and **XML element**.
SAX	(*acronym*) Simple API for **XML**. SAX based parsers sequentially process XML data. They are considered faster and lighter than **DOM** based parsers (*disambiguation*) The term "API" when used as part of "Simple API for XML" (i.e. SAX), refers to an application programming interface used to process XML and not to an **API** for a **records system**.
Schema	(*noun*) See **XML schema**.
XML attribute	(*noun*) A property of an **XML element** included as a descriptor within the element start tag.
XML block	(*noun*) Where an **XML element** contains a list of XML elements, all of the same type, this is described as a "block".
XML element	(*noun*) An XML element is a discrete section of **XML data** delineated by a start tag and an end tag. (*disambiguation*) An XML element used in **XML data** should not be confused with a **metadata element**, used to describe an **entity** in an **MCRS**.

Copyright © 2010 & 2011 DLM Forum Foundation, all rights reserved

Term	Explanation and relationship to general concepts
XML schema	(*noun*) A technique for describing and constraining the structure and grammar of an **XML** document. An XML schema is used to validate **XML data**. Validating XML can prove that it is structurally correct, but cannot necessarily test whether it is logically correct according to various processing rules. For example, it cannot be used to check **export data** from an **MCRS** and prove that every **system identifier** it contains is unique.

Copyright © 2010 & 2011 DLM Forum Foundation, all rights reserved