EAD-Technical Considerations

Recommendations:

1. Maintain the EAD schema in RelaxNG. Distribute derived W3C XML Schema and DTD versions as well.
2. If elements and attributes from other namespaces are to be included at all, use only those from namespaces maintained by SAA (i.e., currently, EAC and EAD) or, perhaps, those which are highly generalized and widely adopted (e.g., xml:id and xml:lang). EAD elements could also be closely aligned with elements/attributes from other external namespaces (e.g., MODS) but remain within EAD namespace.
3. Consider EAD Tag Library as a schema language neutral definition of EAD XML elements, attributes, and structures. Maintain in the same manner as EAC (i.e., TEI).
4. Develop a formal vocabulary/ontology for EAD, and the entities/concepts of interest to EAD.
5. Develop a subset of EAD, "EAD-strict", to satisfy desire for simpler, "database-friendly", interchange-oriented schema. Continue to maintain "EAD-full" for "document-centric", retrospective encoding, or other purposes.
6. Provide supplementary tools and resources to educate and facilitate application of EAD. For example, Tag Library in multiple languages and multiple formats; XSLT stylesheets for translation from EAD 2002 to new version, "EAD-Full" to "EAD-Strict", "tidying" of instances, EAD to RDF, etc...; Schematron schema for checking rules not expressible in XSD, RelaxNG, and DTD, etc...

Discussion:

Schema Language Options

When EAD was first developed the only schema language available to define the elements, attributes and structures of conformant documents was Document Type Definition (DTD). Since then other schema languages have emerged, the two most significant being the World Wide Web Consortium XML Schema (W3C XML Schema, or XSD) and RelaxNG. In 2004, RelaxNG and XSD versions derived from the EAD 2002 DTD were made available, chiefly to enable the validation of EAD instances wrapped in documents valid against the Metadata Encoding and Transmission Schema (METS) XSD schema.

Detailing the differences among DTD, XSD, and RelaxNG is beyond the scope of the current document. Further, it should be understood that the differences do not manifest themselves or have the same degree of importance at every point of EAD document lifecycle. Many of the differences relate to the writing, maintenance, or extension of the schema itself, fewer to the creation, editing, and validation of instances by users. With regard to maintenance, RelaxNG is probably the preferable language for the "maintenance" version of the schema, with XSD and DTD versions derived from it. Local customizations of EAD would likely have to be performed on the maintenance version. Whichever schema language is used for the maintenance version, it is recommended that EAC-CPF should also be maintained in that language.

To most users, it likely will not make much of a difference in which language the schema used to validate an instance is written. Documents valid against the RelaxNG version of EAD ought to be valid against the XSD version and vice versa, as is the case with instances valid against the EAD 2002 schemas in either language. While it is not the case with EAD 2002, it is certainly possible for instances valid against either the RelaxNG and XSD schemas to be valid against a DTD version (and vice versa) as well. It will depend upon the use of datatyping and namespace features (both relatively weakly supported in DTD) in the upcoming revision. Nevertheless, a conversion stylesheet could be made available to convert between RelaxNG/XSD-valid instances and DTD-valid instances in any case. It is therefore recommended to distribute versions of EAD in all three schema languages.

RelaxNG and W3C XML Schema offer the possibility of inclusion of elements and attributes from other namespaces in EAD. Caution is warranted. Processing and validation of XML instances which contain multiple namespaces can be complicated and difficult, especially for those new to XML and XSLT. Namespaces can also greatly increase the maintenance burden, both for the EAD schema and for EAD instances, as the versions of the schemas of included namespaces change or become unavailable. For these reasons it is recommended that if external namespaces are to be used at all, they should be restricted to widely used and general purpose namespaces (e.g., xml:id and xml:lang) or those maintained by SAA (i.e. EAC). It is possible, however, to closely model EAD elements and attributes on those from external namespaces. For example, the content models of many of the EAD <did> elements can be designed to be similar to those of analogous MODS elements in order to facilitate translation between the two.

EAD as a Vocabulary/Ontology

EAD models a type of document (most often finding aids). It describes the elements, attributes and structure of *descriptions* of archival collections and their components. It does not, however, model or define archival collections and their components (as well as other related entities) themselves. The fullest and most authoritative expression of EAD is the Tag Library, not the EAD DTD/Schema, which ought to be seen as a particular formal expression of much, but not the entirety, of the model defined by the Tag Library. Indeed, there are “rules” set down in the Tag Library which are not expressible in the DTD (or XSD) schema languages. For example, the definition of the OTHERLEVEL attribute states “Set LEVEL to "otherlevel" and then supply the preferred term in the OTHERLEVEL attribute”, yet this constraint (if using OTHERLEVEL, then LEVEL must have the value “otherlevel”) is something which cannot be expressed in the DTD schema or XSD (version 1.0) schema language.

It is thus already the case that the EAD standard already exists at multiple levels of expression and formality. At the highest is the Tag Library, fully but not formally expressed for machine use in prose, and this is formally but not fully expressed in XML schema languages. It is recommended that this system be retained, but it would also be advantageous to have a more formal expression of the entities defined by the Tag Library, represented in a formal vocabulary or ontology. At the very least and at its most simple, such an ontology could be a (Simple Knowledge Organization (SKOS) vocabulary in which each of the elements and attributes defined by the Tag Library are given a persistent URI, definitions in multiple languages, and assertions of equivalence or other relationships to related concepts in other vocabularies. Such an ontology could facilitate integration with semantic web applications and resources, and at least provide a URI for the significant concepts in EAD which could be used in RDF triples, in linked data contexts, and for dereferencing to retrieve a definition and assertions of equivalence to terms/concepts in other vocabularies. Beyond this, development of an ontology/vocabulary defining

the “things”/entities/concepts of interest and unique to archives (e.g., series, subseries), to be used alongside vocabularies being established for other entities/concepts to facilitate incorporation of data about archives in the Semantic Web/Linked Data environments.

Development of an EAD vocabulary is not an essential part of the Revision, but given the great interest in Linked Open Data and Semantic Web technologies in archives and libraries more generally (e.g., the LOCAH, SNAC, etc...), the time may be right to assemble efforts around EAD and the EAD revision.

EAD Subset

The comments on the upcoming revision submitted by the EAD community and the Design Goals document show a desire for a more constrained (“simple”, “database friendly”, “interoperable”) EAD. It should be noted that constrained subsets of EAD may be easily created by users or groups of users using Schematron or less easily by directly customizing the EAD DTD. TS-EAD could also develop and distribute XSLT stylesheets which could perform a “down-translation” of instances so they would validate against a constrained version of EAD. Indeed, it would be very beneficial to the community to have such a Schematron schema and conversion stylesheets, if only as a means to educate on technical solutions already available and easily implemented (see Tools and Supplementary Resources, below)

To the extent that a constrained subset of EAD should be maintained and distributed by TS-EAD, it is not clear that this is best accomplished by revising or providing a constrained version of the “base” EAD itself. TS-EAD could very well provide one (or multiple) supplemental Schematron schemas while leaving the base schema as is. Another (and not exclusive) option is providing a constrained version (“EAD-Strict”) which is a subset of the base schema (“EAD-Full”). This option is attractive in that it will provide a single schema file (for each schema language) to be used in ways already familiar to users, but will allow other users who have need for the features left out of the “strict” version. It’s likely that an “EAD-Strict” will be most appropriate for newly created finding aids/archival description records and for use in “data-centric” contexts. However, there will certainly be many users who will use EAD to encode *existing* instances and for whom “document-centric” purposes are of concern. There is no reason why both cannot be accommodated.

Tools and Supplementary Resources

The EAD standard does not consist merely of a single schema file. EAD has always benefited from making available tools and other supplementary resources which facilitate its application. Perhaps the most important resource is the Tag Library, which should continue to be distributed in multiple languages and in multiple formats. It would be advantageous to the SDT if the Tag Libraries of EAC and EAD be maintained in the same way. SDT member Florence Clavaud has proposed a TEI-based model for EAC which would suit the purposes of the EAD as well.

The revision Design Goals document mentions that a mechanism for conversion of EAD 2002 instances to the next version of EAD be made available. It is likely that an XSLT stylesheet could be developed for this purpose, although it remains to be seen whether the conversion can be entirely automatic. Other useful tools could be XSLT stylesheets to facilitate conversion from “EAD-full” to “EAD-Strict”

instances and to perform simple “tidying” (e.g., removal of elements empty of content), and EAD to RDF. Such stylesheets would perhaps be most beneficial by demonstrating techniques for manipulating and managing EAD data. Similarly, a simple but extensible Schematron schema, expressing, for example, the constraints on “EAD-full” which define “EAD-strict”, could educate users and enable them to easily subset EAD for their own purposes.