

DESIGN CONSIDERATIONS FOR XML-BASED T&E STANDARDS

Tim Darr, John Hamilton, Ronald Fernandes
Knowledge Based Systems, Inc
1408 University Drive East
College Station, TX 77840

Charles H. Jones
812TSS/ENTI
Edwards AFB, CA

ABSTRACT

The next generation of telemetry systems will rely heavily on XML-based standards. Multiple standards are currently being developed and reviewed by the T&E community, including iNET's Metadata Description Language (MDL), the XML-version of IRIG 106, Chapter 9 (TMATS XML), the Instrumentation Hardware Abstraction Language (IHAL), and the Data Display Markup Language (DDML). In this paper, we share design considerations for developing XML-based T&E standards, gained from our experiences in designing IHAL and DDML.

KEYWORDS

XML, XML Schema, TMATS, DDML, MDL, IHAL

INTRODUCTION

Existing T & E standards (MDL [3], TMATS [5], IHAL [2], DDML [6]) cover a unique scope and define ways to describe several important T&E concepts. Ideally, future T&E systems and sub-systems will make use of concepts from several of these standards, combining only the portions of each standard that are relevant to the given sub-system. For example, a telemetry ground station may need to leverage concepts related to measurements (MDL), measurement packaging (MDL and TMATS XML), and Data Display (DDML). Similarly, an Instrumentation Engineer's system will need to make use of concepts related to instrumentation hardware (IHAL), measurements (MDL), and the relationships between instrumentation and measurements.

In the sections that follow, we provide a short introduction to XML for those that are not familiar with it, and then describe three simple schema design guidelines that will enable the sharing of XML standards in existing and new standards. These guidelines include

dividing a schema into sub-schemas and declaring global types and elements. We illustrate the application of these guidelines using the TMATS XML schema. This paper describes the guidelines at a very high level, intentionally avoiding the details of XML and XML schemas in order to make the presentation valuable to those that are not familiar with XML design concepts. The guidelines outlined in this paper are currently being applied to refactor the TMATS XML schema.

Our intent is not to criticize the design of any existing standard. Rather, our purpose is to highlight some common design practices that inhibit the integration and re-use of existing standards into new standards, and to show alternative design practices that alleviate these issues. The XML schema design practices that prevent integration and re-use of existing standards include the following:

- Duplication of identical or nearly identical structures. It is not uncommon to repeat the same structure over and over again in the XML schema. It is sometimes easier to “copy and paste” structures instead of taking the time to design a proper schema. This design practice should be replaced by the “declare global types” guideline.
- Duplication of elements. Similar to the previous practice, it is not uncommon to define local elements that refer to XML structures. This design practice should be replaced by the “declare global elements” guideline.
- Monolithic schemas. It is very common to design an XML schema as a single monolithic file. Decomposing a schema into logical components is a difficult task and requires careful design and thought. This design practice should be replaced by the “dividing a schema into sub-schemas” guideline.

The next section provides an overview of XML and XML schemas for those that are not familiar with these technologies.

XML OVERVIEW

XML is a specification produced by the W3C¹ whose original intent was to provide a standard, machine-readable format for describing documents. Because of its popularity, wide adoption and prevalence on the Internet, its use has expanded to describe arbitrary data structures such as web services, and in this paper, T & E metadata. The example below in Figure 1 shows a portion of an XML document (an XML “snippet”) from a prototype version of the TMATS XML format.

In XML, each piece of data, or element, is surrounded by a “tag” such as `<d:Measurement>` and `<Parity>`. The structure of an XML file is such that tags can be enclosed in other tags to an arbitrary depth (`<MeasurementLocation>` is a sub-element of `<d:Measurement>`, `<MeasurementFragments>` is a sub-element

¹ <http://www.w3.org/>

of <MeasurementLocation>, etc.). This is the basic idea behind the structure of an XML document.

```
<d:Measurement Name="ENGINETEMPERATURE" d:id="engine-temperature">
  <Parity>Default</Parity>
  <ParityTransferOrder>Default</ParityTransferOrder>
  <MeasurementTransferOrder>MSB First</MeasurementTransferOrder>
  <MeasurementLocation>
    <MeasurementFragments>
      <StartWord>3</StartWord>
      <WordInterval>0</WordInterval>
      <StartFrame>1</StartFrame>
      <FrameInterval>0</FrameInterval>
      <BitMask>Full Word</BitMask>
    </MeasurementFragments>
  </MeasurementLocation>
</d:Measurement>
```

Figure 1: Example TMATS XML Snippet

The remainder of this section will get into more detail about the specific parts of an element, and how a schema defines the rules for a specific XML document type.

Anatomy of an XML Element

This section provides a brief overview of the structure of an XML element. The component parts of an XML element are identified in Figure 2. Each of these components is defined below:

- **Element:** “Element” is the term used to define a complete unit of XML information. It begins with a start tag and ends with an end tag. The value of an element can be a simple value or one or more sub-elements (children).
- **Start Tag:** The start tag identifies the beginning of the element, and consists of the element’s name (and possibly a namespace and attributes) included between a “<” and a “>” symbol.
- **End Tag:** The end tag identifies the end of the element, and looks identical to the start tag, except it includes a “/” (forward-slash) after the “<” symbol. An end tag does not contain attributes.
- **Namespace:** The namespace is optional in XML, but can be used to define the scope within which the element is defined. In our TMATS example, we define a “d” namespace (for the TMATS D Group) and make all of the D Group elements members of it.
- **Element name:** The name of the element is what appears in the start and end tags, and is what actually identifies the piece of information being defined.
- **Attribute:** Attributes are another method of associating information with an XML element. An attribute consists of a name followed by a “=” (equals) sign, followed by a value enclosed in quotes. There is currently some controversy among users of XML as to when it is appropriate to use an attribute instead of simply adding a child element with the same name and value.
- **Element value:** The value of the element is everything that lies between the start tag and the end tag. The value can EITHER be a single value (e.g. 7, “John”, true, etc) OR a collection of one or more sub-elements (children).

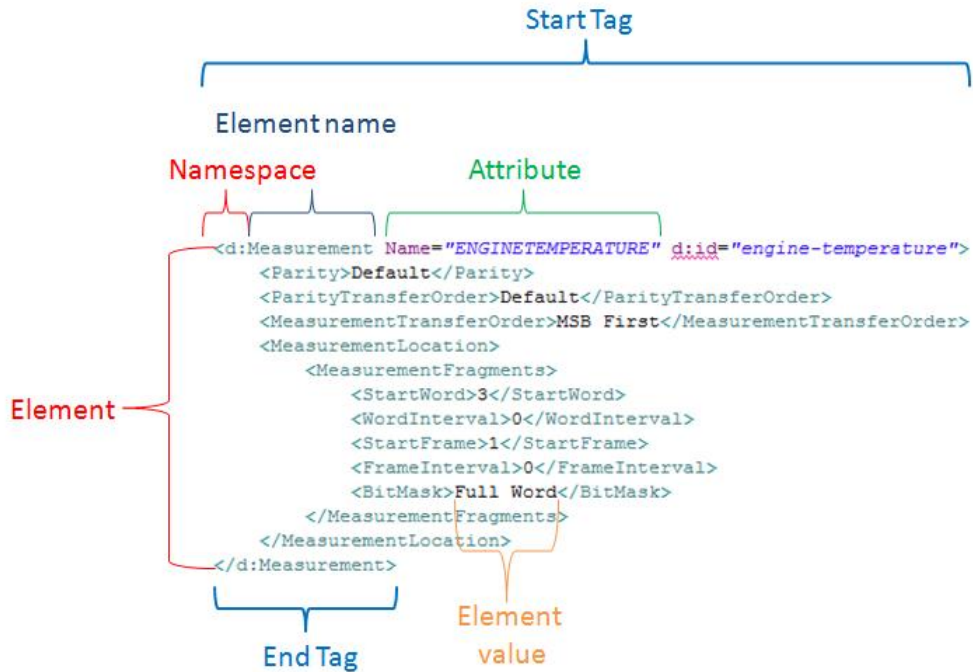


Figure 2: Components of an XML element

XML Schemas

An XML schema is a design document used to describe a specific language that is based on XML. The rules for formatting proper XML are very simple and unrestricted. A schema defines which element names are valid, which elements can have which children, and which values are valid for each element. Types organize XML documents by defining the allowed structure for specific groupings of elements.

Even though an XML schema is itself a document, it is usually more useful to view the schema as a diagram. In this document, we use diagrams generated by the XMLSpy® tool. In order to understand these diagrams, we'll put together a simple schema based on TMATS XML.

An example schema diagram for our TMATS example is shown in Figure 3. The TMATS example shown in Figure 2 is a valid XML instance document that conforms to this schema. In this diagram, XML elements appear as boxes with their names printed inside. Attributes appear in an aptly-named "attributes" box. For both attributes and elements, a solid border indicates that it is required, while a broken or "dotted" border indicates that the element is optional. You will notice, for example, that both the "common:id" attribute and the <Parity> sub-element are optional.

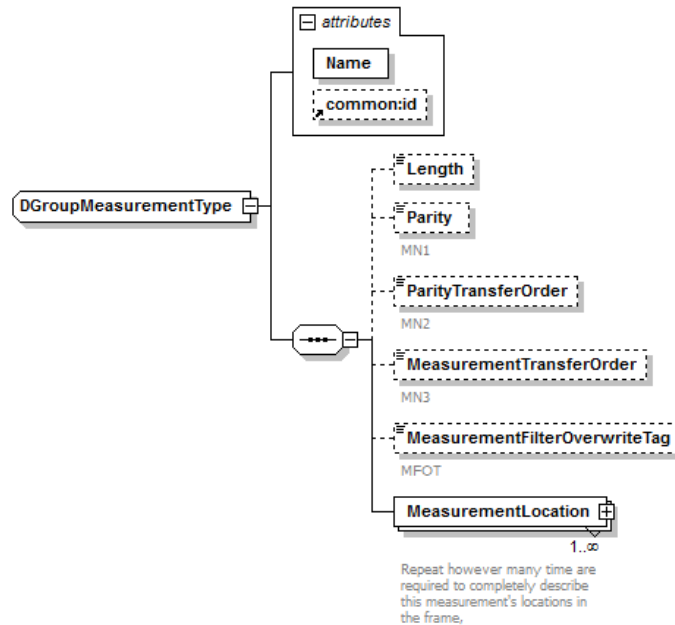




Figure 3: Example schema diagram

Sub-elements are connected to their parents with lines that pass through a special symbol. Each of these symbols specifies the rules for how the elements connected to the right side of it must appear in an instance document. A “sequence” symbol () indicates that each of the child elements must appear in an instance document in the same order in which they appear in the schema. A “choice” symbol () indicates that the instance document must contain exactly 1 of the child elements.

The XML and schema concepts presented in this section should be enough for the reader to understand all of the examples in the remainder of this paper.

Global vs. local definitions

Syntactically, global types and elements are defined as top-level elements in XML. Local types and elements are defined as sub-types or sub-elements of other XML components. Semantically, global types and elements can be reused throughout the rest of the schema, while local types and elements can only be used in their local context.

Figure 4 shows global and local distinction for the TMATS D Group schema. In Figure 4(a) The elements `<SelectedMeasurementName>` and `<Measurement>` and the type `<MeasurementListType>` are global components (they are sub-components of the `<xs:schema>` element). The global element `<Measurement>` has a reference to the global type `<DGroupMeasurementType>`. Similarly, the global type `<MeasurementListType>` has a reference to the global element `<Measurement>`. In Figure 4(b) the element `<Length>` is local to the definition of the type `<DGroupMeasurementType>`; it cannot be referenced or reused outside this type.

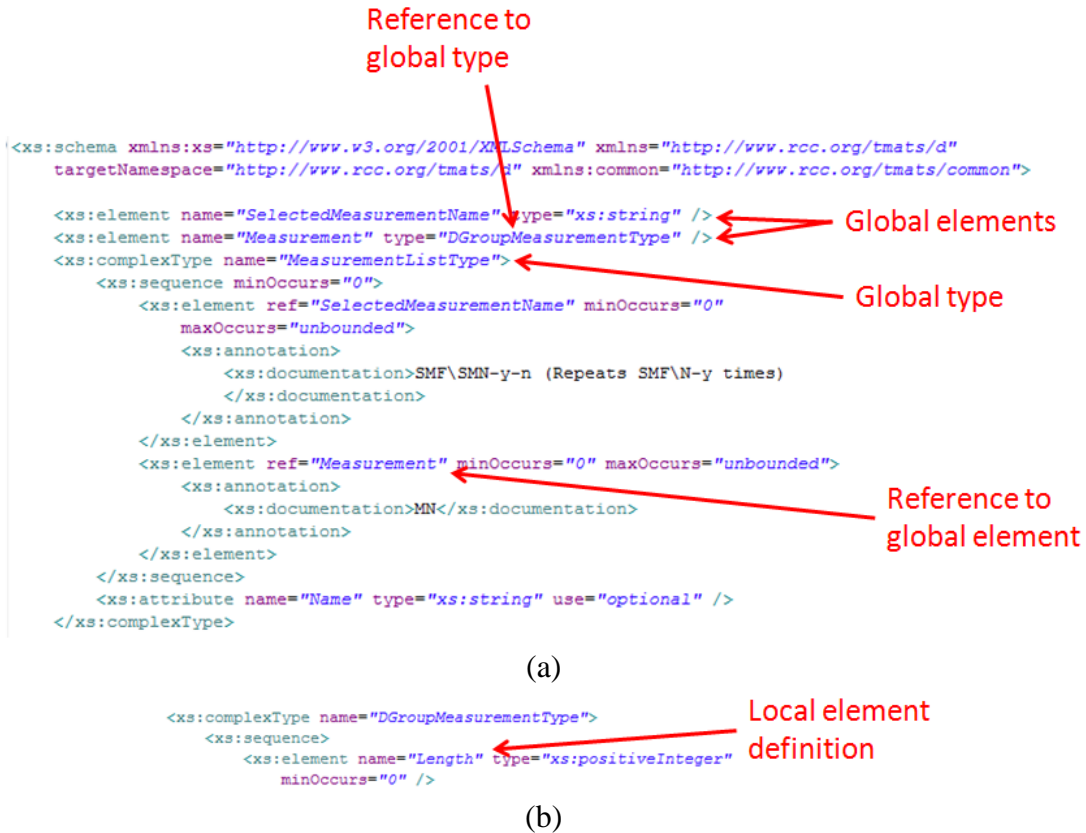


Figure 4: Global vs. Local Example

THE CASE FOR EMPLOYING THESE GUIDELINES

There are costs for not employing the guidelines described in this paper and benefits for employing the guidelines. The costs include the following:

- Difficulty in understanding large schemas
- Large schemas increase the size of other schemas that import them
- Changing redundant schema components in multiple places is time-consuming and can lead to errors

If a single, large, monolithic schema is used (misapplication of the “Modularization” guideline), it can be difficult for a human to understand the schema. This is especially true if the schema contains multiple naturally separable logical components. As an example, the current version of the TMATS XML schema contains XML structures for all of the TMATS groups. If a user is not interested in the recorder (R) group, the user must still navigate through these structures.

Large schemas also cause problems when a user is interested in using a subset of that schema in another schema (misapplication of the “Modularization” guideline). In addition to the problem identified above in understanding the schema, importing (or including) a large schema has performance costs. The entirety of the schema must be imported (or included) which can cause problems in loading the schema or downloading

the schema from the internet. For example, in an experimental version of IHAL, KBSI was interested in re-using the TMATS PCM measurements structures defined in the TMATS XML schema. The monolithic TMATS schema is an order of magnitude larger than a modular TMATS P group. It is true that the TMATS XML schema is relatively small at this point, but these costs can become significant for larger schemas.

When redundant schema components are not consolidated into shared components, it is time consuming to maintain the schema and the likelihood of introducing errors during maintenance is increased (misapplication of the “Global Types” and “Global Elements” guidelines). In the current TMATS XML schema, certain identical structures are defined multiple times (representation of parity is one example). If the user needs to change this structure in some way, the user must search for all instances of that structure and make the desired changes. This takes time, and if the user misses one of the structures, errors will be introduced into the schema. This is not so much of an issue for a structure that is not likely to change, such as parity, but for evolving structures, this could be a major issue.

The benefits of following the guidelines presented in this paper include the following:

- Promotes the use of a schema in other schemas
- Reduces the amount of time needed to understand a schema
- Reduces the time needed to make changes
- Reduces the errors when changing the schema
- Composability means you only use what you need

The primary benefit of these guidelines is the reuse of schema structures in other schemas. During the development of the IHAL standard, it was necessary to represent measurement units. There were several candidate options: develop a custom representation of units, reuse the UnitsML schema standard developed by NIST [4], or reuse the units representation that is part of MDL. We chose to reuse the MDL representation for two primary reasons: it promotes interoperability between MDL and IHAL, and it reduces the amount of work that would have been required to create a new units representation.

By modularizing a schema, it becomes easier for a user to understand. In the example given previously, by separating out the TMATS R group schema, it would be easier for a new user of the schema to understand not only the R group, but also the other groups. If a user is interested in using only a small portion of a schema, such as the TMATS PCM measurement structures, a modular schema provides a way to only use what you need.

By defining XML types and elements globally and reusing these structures by reference, maintenance becomes much easier. Changing types only needs to be done in one place, reducing the time to make changes. Having to make changes to only one structure significantly reduces the potential for errors. Extending a type becomes much easier as well as this only needs to be done in a single place.

GUIDELINES

This section describes the three guidelines and illustrates their use via examples taken from the TMATS XML schema and the IHAL XML schema.

Global Types

In this guideline, all types are defined globally and used as needed to build up other types by referencing the globally-defined type. This guideline promotes reuse, certain XML processing software does not work well if this guideline is not followed², and use of this guideline results in a more compact schema (no duplication of information).

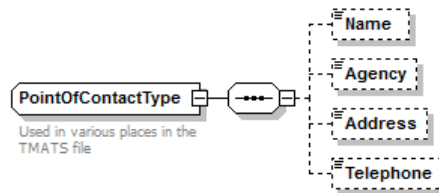


Figure 5: Global Types Positive Example

Figure 5 shows a positive example of this guideline in the TMATS XML schema for the “point of contact” type. This is a type that is used in many other types. Instead of creating a local type definition in each type that uses it, the developers of the TMATS XML schema created a global type. This promotes sharing of types within the TMATS XML schema and promotes reusability and composability in other schemas.

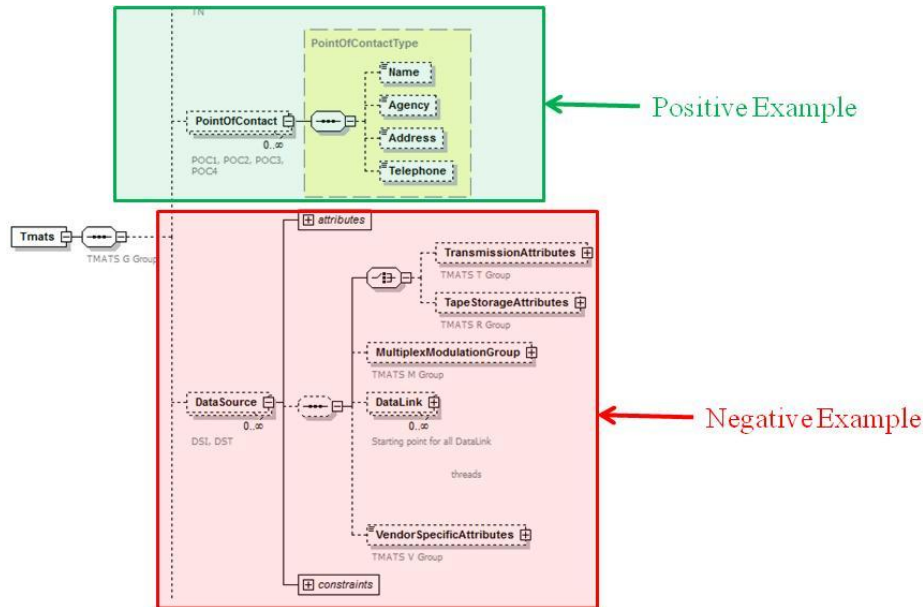


Figure 6: Global Elements Positive and Negative Example

² <http://xmlbeans.apache.org/>

Global elements

In this guideline, all elements are defined globally and used as needed to reference global types. This guideline promotes reuse, and reduces maintenance errors.

Figure 6 shows a positive and negative example of this guideline. The “POC” element is defined globally to reference the point of contact type. Each use of the “point of contact” concept should use this element. Conversely, the “data source” element is defined locally to the TMATS element. Defining the reference to the data source in this type is an inconsistent way to reference the same type.

Modularization

In this guideline, the schema is decomposed into logical sub-schemas, with each schema identified by a namespace. This guideline promotes a “building block” approach to schema design and composition. New schemas are built by identifying existing schemas that provide types and elements that can be used in the new schemas. Structures are reused or extended to satisfy the requirements of the new schema.

Figure 7 shows a positive example of this guideline as employed in an experimental version of IHAL. The TMATS XML schema was decomposed into sub-schemas for each TMATS group (P, D, R, G, etc.). The blue boundary shows the IHAL use element for a DAU; the green boundary shows an included TMATS P group element to describe the measurements processed by the DAU. This promotes interoperability (a TMATS and IHAL schema share the same description of measurements) and reduces work in designing a schema (the IHAL developers did not have to design a measurement schema).

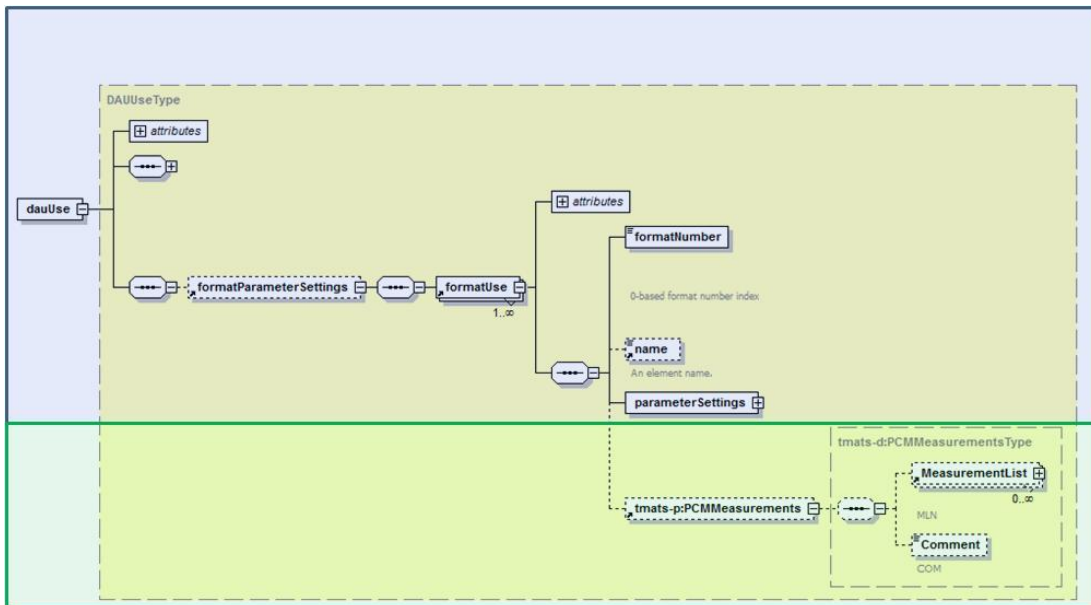


Figure 7: Modularization

CONCLUSION

This paper presented three guidelines for XML schema design that if implemented will promote the reuse of schemas, reduce the amount of time needed to understand a schema, reduce maintenance and development effort, and facilitate schema composability. The guidelines “Modularization”, “Global Types”, and “Global elements” are simple to understand and implement with a small amount of discipline. Positive and negative examples of these guidelines were presented for illustration purposes. These examples were drawn from our experience with IHAL, MDL and the TMATS XML schema.

REFERENCES

- [1] XML Tutorial, <http://www.w3schools.com/xml/default.asp>, accessed June 7, 2011.
- [2] Hamilton, Fernandes, Koola, and Jones, *An Instrumentation Hardware Abstraction Language*, Proc. International Telemetry Conf., Vol. XLII, (2006) Paper 06-10-02, San Diego, CA.
- [3] Moore, Price, Cormier, and Malatesta, *Metadata Description Language: The iNET Metadata Standard Language*, Proc. International Telemetry Conf., Vol XLV (2009), Paper 09-01-01, Las Vegas, NV
- [4] Units Markup Language home page, <http://unitsml.nist.gov/>, accessed June 9, 2011.
- [5] <https://wsmrc2vger.wsmr.army.mil/rcc/manuals/106-09/Tmats.xml>, accessed June 20, 2011.
- [6] http://www.wsmr.army.mil/RCCsite/Documents/106-09_Telemetry%20Standards/ddml31.xsd, accessed June 20, 2011.