CountryData
Technologies for Data Exchange

SDMX Markup Language
(SDMX-ML)

SDMX-ML

- Implementation of SDMX Information Model
- Uses eXtensible Markup Language (XML)
    - Another implementation is based on GESMES/TS but SDMX-ML is far the most commonly used SDMX format

SDMX Versions

- SDMX-ML 2.1 is substantially different from SDMX-ML 2.0.
- This presentation focuses on SDMX 2.0, which at the moment is more widely used.
  - Many available tools, especially Eurostat’s, only support SDMX 2.0.
- CountryData uses SDMX 2.0.

SDMX Messages

- Any SDMX-related data are exchanged in the form of documents called *messages*.
- There are several types of SDMX messages, each serving a particular purpose.

SDMX-ML Namespace Modules

- SDMX-ML defines several namespaces. Each namespace defines constructs for a certain area.
  - E.g. there are namespaces for Structure, Generic, Query, and other message types.
- In some cases, user may define a namespace for their data, but it is still wrapped as a standard SDMX message.
  - E.g. Compact data message.

Message Envelope

- A single type of "envelope" is used for all SDMX messages.
- Has its own namespace, commonly aliased as *message* or *mes*

  http://www.SDMX.org/resources/SDMXML/schemas/v2_0/message
- Defines root element, which reflects message type (e.g. *message:Structure*, *message:Query*), a header, and some other elements.

Message Header

- Shared by all SDMX-ML message types
- Must be provided in every message
- Includes information on sender and receiver, and other relevant information

Namespaces in an SDMX Message

```xml
<?xml version="1.0" encoding="utf-8"?>
<Structure xmlns="http://www.SDMX.org/resources/SDMXML/schemas/v2_0/message"
xmlns:common="http://www.SDMX.org/resources/SDMXML/schemas/v2_0/common"
xmlns:structure="http://www.SDMX.org/resources/SDMXML/schemas/v2_0/structure"
xmlns:utility="http://www.SDMX.org/resources/SDMXML/schemas/v2_0/utility">
```

message

```xml
<Header>
  <ID>CountryData_DSD_v_0_8</ID>
  <Test>true</Test>
  <Name xml:lang="en">Data Structure Definition for CountryData</Name>
  <Prepared>2012-01-30T15:38:19.917Z</Prepared>
  <Sender id="UNSD">
    <Name xml:lang="en">UNSD</Name>
    <Contact>
      <Name xml:lang="en">Contact person in SENDER agency</Name>
      <Department xml:lang="en">Department of SENDER agency</Department>
      <Role xml:lang="en">Role of contact person in SENDER agency</Role>
    </Contact>
  </Sender>
  <Receiver id="receiver">
    <Name xml:lang="en"></Name>
    <Contact>
      <Name xml:lang="en"></Name>
      <Department xml:lang="en"></Department>
      <Role xml:lang="en"></Role>
    </Contact>
  </Receiver>
</Header>

<CodeLists>
```

structure

```xml
  <structure:CodeList id="CL_AGE_GROUP_COUNTRY_DATA" agencyID="UNSD" version="0.8">
    <structure:Name xml:lang="en">Age group code list</structure:Name>
    <structure:Code value = "NA">
      <structure:Description xml:lang="en">Not applicable</structure:Description>
    </structure:Code>
    <structure:Code value = "000_000_Y">
```

Structure Message

- Defines the concepts, dimensions, codes, dataflows, and/or other structural information, but carries no data itself
- Similar in purpose to a database's data dictionary
- Uses Structure namespace, commonly aliased as *structure* or *str*

http://www.SDMX.org/resources/SDMXML/schemas/v2_0/structure

Generic Data Message

- Conveys data in a form independent of a data structure definition
- Can be easily validated against the generic schema, but not against a specific DSD
- Can be used in situations where the recipient is not expected to know the details of underlying key family
- Namespace aliased as *generic*

http://www.SDMX.org/resources/SDMXML/schemas/v2_0/generic

Compact Data Message

- Designed to exchange large data sets in a DSD-specific form
  - Can be validated against a DSD
- Less verbose than generic message; broader use of XML attributes
- Can be used for incremental updates

Compact Message Namespaces

- Compact message namespace defines elements of compact message.
  - DataSet, Group, Series, Obs
- The actual namespace used on these elements is defined by the user.

```
<cd:DataSet
    xmlns:cd="urn:sdmx:org.sdmx.infomodel.keyfamily.KeyFamily=UNSD:
    CountryData:compact">
   <cd:Series FREQ="A" SERIES="AG_LND_FRST" UNIT="PERCENT"
    LOCATION="T" AGE_GROUP="NA" SEX="NA" REF_AREA="GHA"
    SOURCE_TYPE="NA">
```

Cross-sectional Data Message

- Designed primarily to exchange many observations at a single point in time, in a DSD-specific form
- Can be validated against a DSD

Utility Message

- Special-purpose message, used mostly in schema-based validation functions
- DSD-specific format

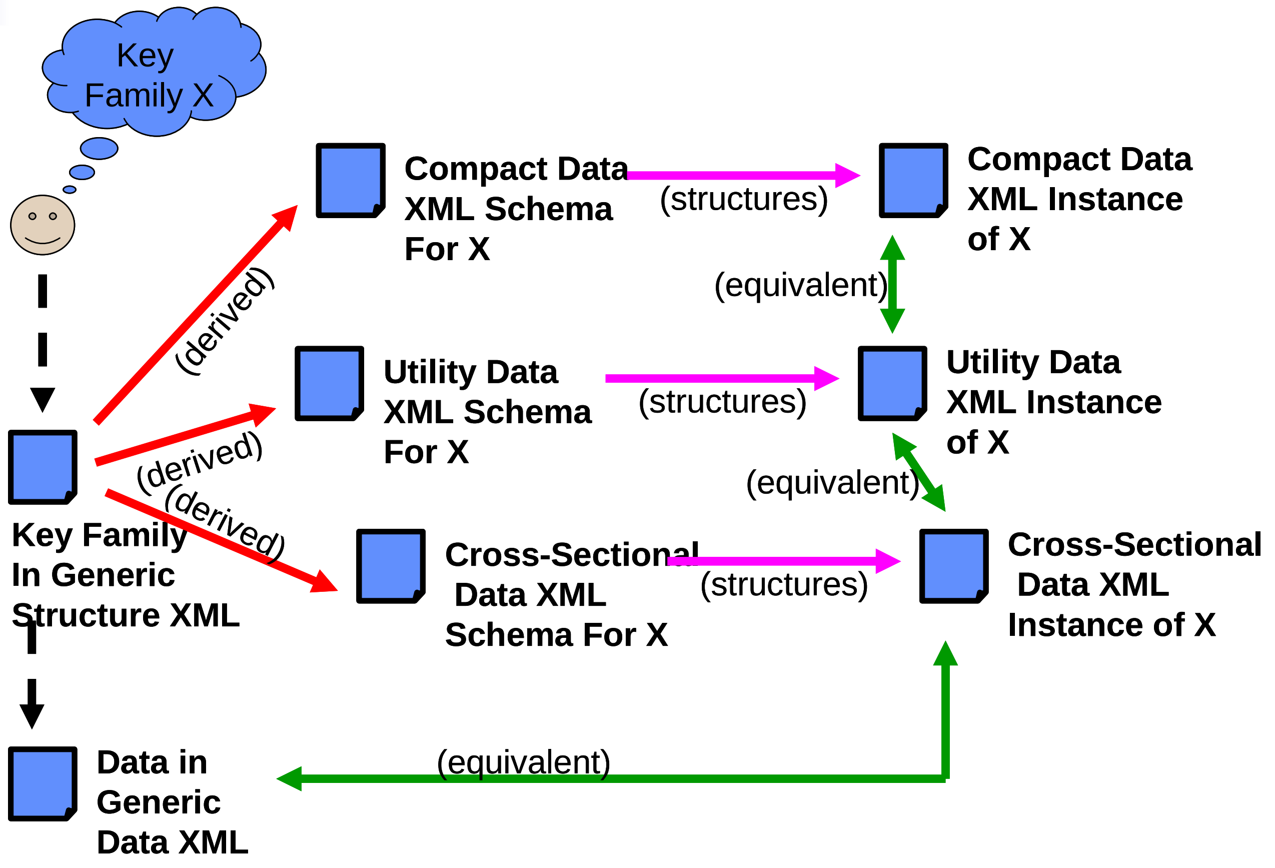
Query Message

- Designed to query SDMX registries, web services, and other applications
- An SDMX-ML message is returned in response

Deriving SDMX-ML messages

- Because all types of SDMX-ML messages rely on the single underlying information model, many messages types can be derived from one another.
  - Schemas for compact, cross-sectional, and utility messages can be derived from DSD.
  - Generic, compact, cross-sectional, and utility messages based on the same DSD can be derived from one another.

Model-based equivalence of SDMX-ML messages


Key Family X
Compact Data XML Schema For X
(structures)
Compact Data XML Instance of X
(derived)
(equivalent)
Utility Data XML Schema For X
(structures)
Utility Data XML Instance of X
(derived)
(derived)
(equivalent)
Key Family In Generic Structure XML
Cross-Sectional Data XML Schema For X
(structures)
Cross-Sectional Data XML Instance of X
Data in Generic Data XML
(equivalent)

Source: SDMX User Guide, v. 1.3

Versioning of SDMX Artefacts

- Identification of an SDMX artefact can be thought to consist of 3 parts:
  - Maintenance agency
  - Id
  - Version
- The isFinal attribute specifies whether the artefact has been finalized.
  - When isFinal is set to True, no further updates are allowed without versioning

Agency (organisation)

- “A unique framework of authority within which a person or persons act, or are designated to act, towards some purpose”
- Agency has a very important role in SDMX.
  - Maintenance Agency, Data Provider, Data Consumer all reference agency.

DSD and schemas

- A Structure Message in itself is not an XSD schema: it is an XML (SDMX-ML) document.
- XSD schemas can be derived from the DSD for Utility, Compact, and Cross-Sectional messages.
- SDMX tools support automatic schema generation from DSD.

Names and Descriptions

- Many of SDMX constructs, especially in the Structure namespace, can (or must) be given a short human-readable Name and/or a longer Description.
  - E.g. for every code there should be at least one Description.
- Names and Descriptions can be expressed in any language, as specified by the *xml:lang* attribute.

Annotations

- Annotations supply additional explanatory information and can be embedded in most SDMX constructs.
- Can be very useful for reporting simple metadata such as footnotes, or even mapping information.

Structure Namespace: ConceptScheme

- “Descriptive information for an arrangement or division of concepts into groups based on characteristics, which the objects have in common ”
- Optional in SDMX 1.0 and 2.0, mandatory in SDMX 2.1
- MDG data concepts are currently standalone. MDG metadata concepts are placed in a concept scheme.

Structure Namespace: Concept

- "A unit of knowledge created by a unique combination of characteristics"
- Concept is declared separately from dimensional structure.
- When a dimension, attribute, or measure is declared in a DSD, a reference to an existing concept is provided.

Structure Namespace: CodeList

- "A predefined list from which some statistical coded concepts take their values"
- Acts as a wrapper for codes to be used in a concept or concepts
- CodeLists are often the most changeable component in a DSD
  - Must be versioned carefully

Structure Namespace: Code

- “A language-independent set of letters, numbers or symbols that represent a concept whose meaning is described in a natural language”
- Valid characters: A-Z, a-z, @, 0-9, _, -, $
- Names and Descriptions can be supplied in multiple languages.

Structure Namespace: KeyFamily (DSD)

- “Set of structural metadata associated to a data set, which includes information about how concepts are associated with the measures, dimensions, and attributes of a data cube, along with information about the representation of data and related descriptive metadata”

Structural Metadata: KeyFamily

- The element KeyFamily declares dimensional structure.
- References are made to concepts and codelists, declared outside the element.

Structure Namespace: MetadataStructureDefinition

- Defines "targets" against which reference metadata can be reported, and metadata attributes, i.e. the actual content
- Metadata can be reported against structural metadata such as Concept, and against keys or partial keys from a DSD.

Structure Namespace: MetadataStructureDefinition

- Footnotes are often reported as observation-level attributes, as in MDG DSD.
- Another way to report footnotes is use Annotations.
  - In many cases preferable since it uses standard SDMX constructs and therefore supports multiple languages

Reference Metadata Support

- Few existing SDMX tools support SDMX 2.0 reference metadata.
  - Reflects insufficient attention given to SDMX metadata exchange until recently
  - Documentation on MSD in SDMX 2.0 is often ambiguous
- Much stronger support in SDMX 2.1
  - Many tools available
  - Better documentation

Independence of SDMX Artefacts

- Many types of SDMX artefacts are maintained independently and used by reference.
  - E.g. concepts from a concept scheme can be used in many DSDs.