Introduction

The following topics are covered:

- XML Toolkit Features
- XML Toolkit Description
- Outlook
- Considerations and Limitations

XML Toolkit Features

- Natural-based XML parser using dynamic variables.
- Functions for
  - conversion of Natural data structures into DTD definitions;
  - generation of COMPRESS statements to save a Natural data structure as an XML document;
  - generation of callback for the Natural-based parser.

XML Toolkit Description

Objective

The objective of the Natural XML Toolkit is to provide additional XML functionality with Natural and improve the integration of Natural applications with XML.

General Architecture

The Natural XML Toolkit consists of a collection of Natural programs. The Toolkit programs may be integrated into customer applications to provide access to XML data or to deliver data from Natural in XML format.

The Natural XML Toolkit calls the functions listed below:

XML Toolkit Functions

1. Mapping of Natural Data Definition to DTD and vice versa.
2. **XML Token => NAT**
   Data After the Natural data structure has been created, the XML document has to be parsed and saved into the data structure. A Natural implementation is generated that is capable of saving the given data into the Natural data structure.

3. **NAT Data => XML Document ("Serialize")**
   Serialization is the process of taking the data stored in the Natural data structures and creating an XML document according to the description provided.

A Natural dialog implements the user interface to the XML Toolkit functions. The DTD will be accessed as a work file and the generated Natural objects will be saved directly to the Natural system file.

Map Natural Data Definitions to DTD

This mapping is the first step to bind Natural data structures to XML tags and is required to implement a representation of Natural data as XML tags. The example below shows the mapping as well as some obvious differences between Natural and a DTD.

Natural PDA

```
                       Press ESC to enter command mode
 Mem: EMPL     Lib: SYSEXXT  Type: PARAMETER  Bytes:  1072  Line:    0 of:  26
C T   Comment
  *   *** Top of Data Area ***
    1 EMPLOYEE
    2 ATTRIBUTES_OF_EMPLOYEE
    3 PERSONNEL-ID                    A          8
  *
    2 FULL-NAME
    3 FIRST-NAME                      A         20
    3 NAME                            A         20
  *
    2 FULL-ADDRESS
    3 C@ADDRESS-LINE                  I          4
    3 ADDRESS-LINE                    A         20 (1:6)
    3 CITY                            A         20
    3 ZIP                             A         20
    3 COUNTRY                         A          3
  *
    2 TELEPHONE
    3 AREA-CODE                       A          6
    3 PHONE                           A         15
```

Generated DTD

```
<!ELEMENT EMPLOYEE (PERSONNEL-ID, FULL-NAME, FULL-ADDRESS, TELEPHONE, INCOME* )>

<!ELEMENT PERSONNEL-ID (#PCDATA ) >

<!ELEMENT FULL-NAME (FIRST-NAME, NAME )>
         <!ELEMENT FIRST-NAME  (#PCDATA )>
         <!ELEMENT NAME  (#PCDATA )>

<!ELEMENT FULL-ADDRESS (ADDRESS-LINE*, CITY, ZIP, COUNTRY )>
         <!ELEMENT ADDRESS-LINE  (#PCDATA )>
         <!ELEMENT CITY  (#PCDATA )>
         <!ELEMENT ZIP  (#PCDATA )>
         <!ELEMENT COUNTRY  (#PCDATA )>
...
```

The generated DTD will be used later on during serialization to a XML document (see below).

Serialize Data to XML

During execution of a Natural program, the content of the data defined in the DEFINE DATA statement will be filled with "real" content. This content will be written to a dynamic variable in XML format during serialization and will use the formerly generated DTD as input.

The XML Toolkit generates the program to serialize the data.

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<EMPLOYEE PERSONNEL-ID="30016509">
<FULL-NAME>
  <FIRST-NAME>ELSPETH</FIRST-NAME>
  <NAME>TROWBRIDGE</NAME>
</FULL-NAME>
<FULL-ADDRESS>
  <ADDRESS-LINE>91 BACK LANE</ADDRESS-LINE>
  <ADDRESS-LINE>BILSTON</ADDRESS-LINE>
  <ADDRESS-LINE>STAFFORDSHIRE</ADDRESS-LINE>
  <CITY>BILSTON</CITY>
  <ZIP>ST2 3KA</ZIP>
  <COUNTRY>UK</COUNTRY>
</FULL-ADDRESS>
<TELEPHONE>
  <PHONE>863322</PHONE>
  <AREA-CODE>0602</AREA-CODE>
</TELEPHONE>
...
```

Map DTD to Natural Data Definitions

The mapping of a DTD to Natural data structures again shows differences. The DTD does not specify how many person records will be included in the XML document, therefore the Toolkit assumes that a maximum number of "v" persons will be included. The application programmer might know the exact number and the data structure could be adapted accordingly. A similar limitation exists with the length of the data. The DTD does not include information about the length of the data in a person’s record. Therefore the Toolkit creates fields in the data structure with a length of 253, the current maximum.

```
* DTD E:\SAG\nat\4.2\fnat\SYSEXXT\RES\empl.dtd
COMPRESS &1& ’<EMPLOYEE’
  ’ PERSONNEL-ID="’EMPLOYEE.PERSONNEL-ID "’
  ’>’ INTO &1& LEAVING NO
/* now the children
COMPRESS &1& ’<FULL-NAME’
  ’>’ INTO &1& LEAVING NO
/* now the children
COMPRESS &1& ’<FIRST-NAME’
  ’>’
  EMPLOYEE.FIRST-NAME
  ’</FIRST-NAME>’ INTO &1& LEAVING NO
COMPRESS &1& ’<NAME’
  ’>’
  EMPLOYEE.NAME
  ’</NAME>’ INTO &1& LEAVING NO
/*
COMPRESS &1& ’</FULL-NAME>’ INTO &1& LEAVING NO
COMPRESS &1& ’<FULL-ADDRESS’
  ’>’ INTO &1& LEAVING NO
```

```
/* now the children
FOR &2& = 1 TO EMPLOYEE.C@ADDRESS-LINE
  COMPRESS &1& '<ADDRESS-LINE'
    '>'
    EMPLOYEE.ADDRESS-LINE(&2&)
    '</ADDRESS-LINE>' INTO &1& LEAVING NO
END-FOR
...
```

Parse XML File and Assign to Natural Data

```
* DTD E:\SAG\nat\4.2\fnat\SYSEXXT\RES\empl.dtd
DECIDE ON FIRST &1&
  VALUE 'EMPLOYEE'
    RESET INITIAL EMPLOYEE
  VALUE 'EMPLOYEE/@PERSONNEL-ID'
    /* #REQUIRED
    EMPLOYEE.PERSONNEL-ID := &3&
  VALUE 'EMPLOYEE/FULL-NAME'
    IGNORE
  VALUE 'EMPLOYEE/FULL-NAME/FIRST-NAME'
    IGNORE
  VALUE 'EMPLOYEE/FULL-NAME/FIRST-NAME/$'
    EMPLOYEE.FIRST-NAME := &3&
  VALUE 'EMPLOYEE/FULL-NAME/NAME'
    IGNORE
  VALUE 'EMPLOYEE/FULL-NAME/NAME/$'
    EMPLOYEE.NAME := &3&

...
```

Outlook

The XML Toolkit is another step forward to full XML support with Natural. The XML Toolkit might be extended after the first release. However, the main objective is to implement XML functionality in one of the forthcoming releases as part of Natural's powerful language.

Considerations and Limitations

When using the XML Toolkit, the following limitations should be considered.

- Very Large Data Structures

Very Large Data Structures

Data structures which will result in more then approximately 700 data fields and groups will end up with the message:

```
Input Structure too big
```

Solution

Split up the data structure into smaller sections.