Native XML Parser

# Overview:

The video encoder needs to read the story data in xml format. The xml data is stored in blob. We have several choices:

1. Parse the xml data in managed code, and pass it to native code. This approach is easier to implement, but may potentially lead to marshal too much data between managed/native boundary.
2. Do everything in native code, including downloading the xml data from blob, and parse it. This may lead to a lot of work, as working with blob in native code is not easy.
3. Download the blob to a local storage file in managed code, and use native code to parse it. This approach is easier to implement compared to #2, and the performance is OK. Only 2 extra I/O are required for each story.

In this release, we choose the last option.

In addition, to parse xml in native code, we can either use an existing parser such as MSXML, or write a custom simple xml parser. Since currently our data structure is relatively simple, we choose the latter approach. Thus we don’t need to install MSXML on the cloud machine.

# Goals:

* Create a lightweight native xml parser.
* Support reading elements and attributes synchronously.

# None Goals:

* Support advanced features such as namespace and schema validation.
* Support asynchronous operations.
* Detect invalid xml files (we assume the input is valid, otherwise we throw a general error).

# Design:

We’ll create a class named XmlParser. Methods of this class should look similar to .NET XmlReader. In particular, the following methods should be supported:

* ReadStartElement
* ReadEndElement

In addition, we should allow the caller to obtain attributes of the current element. We will use a map<wstring, wstring> to store attributes.