Use Case: Processing based on XML configuration data

XML is commonly used for defining configuration data in a wide range of systems and applications. There are cases when the configuration parameters defining the behaviour of the system could only be received during runtime operations, and therefore the lightweight scripting-based XML processing could prove to be the most efficient solution.

1. Case Study: Provisioning of generic content for mobile device

Over the air (OTA) dynamic provisioning of mobile application and resources is a complicated multi-phased process. All major wireless companies (Nokia, Ericsson, Motorola, etc.) offer content provisioning solutions aimed at different categories of mobile devices. There is an ongoing effort in Open Mobile Alliance (OMA) to standardize the protocol and architecture for secure OTA content provisioning.
There are two actors in the OTA provisioning process: the mobile device provisioning agent and the server side content provisioning framework. In the current use case we deal with the operations of the mobile device provisioning agent, as platform-independent scripting-based XML processing is more important on the device side. The E4X could be successfully applied for the device agent's operations.
There are two different modes of content provisioning: user-initiated and server-initiated. In the user-initiated mode a mobile device initiates a provisioning sequence and drives the discovery and installation process. In the server-initiated mode (OTA directed push) the server side provisioning framework executes the provisioning steps on behalf of the mobile user. This installation mode is typically more appropriate for secure corporate solutions where IT departments are responsible for support and control of mobile devices deployed across the corporation. In the current use case we consider the user-initiated mode as more common and comprehensive.

2. Provisioning phases

Depending on the device platform, carrier, and application domain (public vs. corporate) the scenario for content provisioning could contain the following phases:

2.1 Content Discovery

Provisioning process starts from the content discovery phase. The result of this phase is the Universal Resource Locator (URL) of the content descriptor XML.

2.2 Content Descriptor Download

At this phase the content descriptor XML is downloaded to the mobile device. The content descriptor contains the number of attributes that define miscellaneous content parameters (size, origin, certificates, transport key, etc.) as well as additional attributes that define the behaviour of the device agent in the next provisioning phases (e.g. content download URL, registration/billing URL, initialization parameters, etc.).

2.3 Compatibility Analysis

The mobile device provisioning agent loads the content descriptor and processes XML attributes describing the content parameters. The agent's task is to ensure that the content can be downloaded onto and executed by the mobile device:

- The device should have sufficient free space to accommodate the application code and associated resources
- The required type and version of operational environment should match the device environment
- The required secure downloading protocol should be supported by the device and domain framework (e.g. HTTPS)
- The certification authority should be recognized by the domain operator and/or device owner (corporate)

2.4 Authorization request

This is an optional step of requesting download/installation permission from the domain authority. The device agent sends device information as well as content descriptor XML attributes to the predefined authorization URL. The domain authority can approve or reject content download. The reason for rejection could be an improper wireless plan of the device user, an invalid certificate, etc.

2.5 Content download and installation

The provisioning agent downloads the content from the specified URL and installs it on the mobile device.

2.6 Content registration and delivery acknowledgement

The registration URL of solution/service provider is supplied in the content descriptor XML and used to register installed content for billing, statistical, and digital rights management (DRM) purposes. At this point the installation is considered to be complete and the charging data could be sent from the download server to the carrier charging and billing systems.

2.7 Content (application) initialization

In case of application provisioning the device agent initializes the application and downloads/installs additional resources (if any) such as images, sound files, etc. After the initialization is completed the application icon is visible to the user.

3. Content descriptor XML

The following DTD could be used to define a content descriptor XML format:

```
<!ELEMENT content (desc?, screen?, env, user_prompt?, cert?,  auth, download, reg?)>
<!ATTLIST content name CDATA #REQUIRED>
<!ATTLIST content vendor CDATA #REQUIRED>
<!ATTLIST content type (app | res) “app”>
<!ATTLIST content size CDATA #REQUIRED>
```

```
<!ATTLIST content ext_size CDATA #REQUIRED>
<!ATTLIST content certified (yes | no) “yes”>
<!ATTLIST content transport_key CDATA #REQUIRED>
<!ELEMENT desc (#CDATA)>
<!ELEMENT screen (#PCDATA)>
<!ATTLIST screen height CDATA #REQUIRED>
<!ATTLIST screen width CDATA #REQUIRED>
<!ATTLIST screen type (mono | color) “mono”>
<!ELEMENT env (#PCDATA)>
<!ATTLIST env type CDATA #REQUIRED>
<!ATTLIST env version CDATA #REQUIRED>
<!ELEMENT user_prompt (param+)>
<!ELEMENT param (#PCDATA)>
<!ATTLIST param name CDATA #REQUIRED>
<!ELEMENT cert (#PCDATA)>
<!ATTLIST cert id CDATA #REQUIRED>
<!ATTLIST cert authority CDATA #REQUIRED>
<!ATTLIST cert type CDATA #REQUIRED>
<!ELEMENT auth (auth_param+)>
<!ELEMENT auth_param (#PCDATA)>
<!ATTLIST auth_param name CDATA #REQUIRED>
<!ATTLIST auth_param value CDATA #IMPLIED>
<!ELEMENT download (#PCDATA)>
<!ATTLIST download url CDATA #REQUIRED>
<!ATTLIST download url_res CDATA # IMPLIED>
<!ATTLIST download protocol (HTTP | HTTPS | any) “HTTP”>
<!ELEMENT reg (reg_param+)>
<!ATTLIST reg url CDATA #REQUIRED>
<!ELEMENT reg_param (#PCDATA)>
<!ATTLIST reg_param name CDATA #REQUIRED>
```

The following XML document represents a hypothetical content descriptor based on the DTD above.

```
<content
        name = “map viewer”
        vendor = ”Mapquest”
        type = ”app”
        size = “35,042”
        ext_size = “62,056”
        certified = “no”
        transport_key = “12345AWQ”/>
        <desc Generic map viewer for wireless device/>
        <screen height = “180” width = “180” type = “color”/>
        <env type = “Java” version = 1.4/>
        <user_prompt>
```

```
                <param name = “Subscription Type”/>
                <param name = “Subscription ID”/>
                <param name = “Password”/>
        </user_prompt >
        <auth>
                <auth_param name = “name”/>
                <auth_param name = “vendor”/>
                <auth_param name = “service” value = “priority”/>
                <auth_param name = “publisher” value = “100.101.102.103”/>
        </auth>
        <download
                url = “http://100.101.102.103/apps/mapv”
                url_res = “http://100.101.102.103/apps/mapv_res”
                protocol = “any”/>
        <reg url = “http://100.101.102.103/reg/apps/mapv”/>
                <reg_param name = “device ID”>
                <reg_param name = “device type”>
                <reg_param name = “transport_key”>
        </reg>
</content>
```

4. Sample E4X script

TBD…