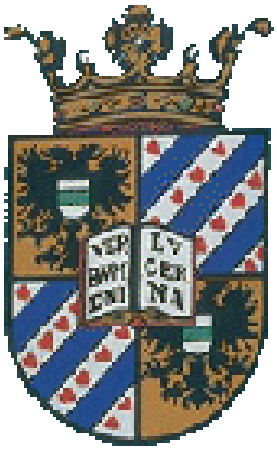

Rijksuniversiteit Groningen

Bachelor graduation project

An Implementation Of A Generic Web Services Library Architecture For Symbian OS

Author:
Alexander Jurjens

Supervisor:
Marco Aiello

January 13, 2010

Contents

1 Introduction

The evolution of more powerful processors for mobile devices (such as mobile phones, PDAs, etc.) has opened up new possibilities. These devices are small processing elements which can do a certain amount of work. They are becoming so powerful that they are able to run more complex software solutions such as web services. There exists a subdivision of these devices called cellular phones. Today's cellular phones are able to run web services. Web Services are Web APIs that can be accessed over a network and executed on a remote machine hosting the services. The frameworks that are now available (e.g. gSOAP [1], Nokia WS Framework [2] or Java ME[3]) for Web Service development come with a tool to generate C++ methods out of a WSDL [1] document at design time and have support for XML based communication at runtime. This approach of developing a Web Service means that the WSDL document must be known in advance to the programmer of the Web Service. This is actually a shortcoming for mobile devices, because it means that Web Services running on mobile devices can't be as portable and adaptable to change as they can be. By writing a library, which contains support for runtime WSDL and SOAP parsing, we try to make mobile phone applications more portable and adaptable to change.

This paper illustrates an implementation of a solution that provides good support for developing web services for Symbian OS [2], which is an operating system designed for mobile devices. The second chapter of this thesis provides an introduction to web services on mobile phones, introducing the features and highlighting the benefits. The third chapter provides an analysis of the web services library architecture (WESSYL). The fourth chapter illustrates an example of using library and the fifth chapter provides the conclusion.

[1] http://www.cs.fsu.edu/~engelen/soap.html

[2] http://wiki.forum.nokia.com/index.php?title=Special:PdfPrint&page=Web_Services_API

[3] http://wiki.forum.nokia.com/index.php?title=Special:PdfPrint&page=Web_Services_API

2 Web Services and Mobile Phones

2.1 Introduction to Web Services

A Web Service (web service) is defined as a piece of software designed to support interoperable machine-to-machine communication over a network. These web services may execute on geographically distributed machines. They provide an interface to operations, which are accessible through a network using eXtended Markup Language (XML) messaging. The messages are coded in XML, which means that web services run independently of transport protocols and platform on which the web service is running. This structure made the possibility for the emergence of Service Oriented Architecture (SOA). The key standards for web service-oriented architectures are:

1. A message interchanging standard that supports communication between services.

2. A standard that defines the way in which service providers should define the interface to web services.

3. A standard that defines the components of a service specification that may be used to discover the existence of a service.

4. A standard that is used as a workflow language to define process programs involving several different services.

The SOAP[4] (Simple Object Access Protocol) standard is used to describe the channel of communication between machines. The WSDL (Web Service Definition Language) standard is used to make representation of the functionality of a web service. The UDDI[5] (Universal Description, Discovery and Integration) standard defines the components of a service specification for service registration and service discovery. All the data is specified in XML. XML is a text-based standard, which means that it is readable for human beings. It can be used for storing documents and processing documents and the strict syntax of XML makes the parsing algorithms extremely simple. The XML messages are usually transmitted over transport protocols such as HTTP and HTTPS, which means that web services can be accessed using the Internet.

[4]http://www.w3.org/TR/soap/
[5]http://www.uddi.org

2.2 About Mobile Phones

Mobile phone technology is evolving so rapidly, that they are able to perform tasks that could only be performed by personal computers and high performance computers just a few years ago. The technology has evolved so much that it is now possible to implement web services on phones. Although it is possible to implement services, it is still not a very easy task to the actual implementation of the standards that are coming with web services. Therefore, it is decided that a library should be made for the most used mobile operating system for making the implementation of services easier. The most used mobile operating system is Symbian OS. Although there are SDK's available for processing XML and for making connections to other devices (Nokia Web Services Framework API for example), there is not one SDK available that supports WSDL document parsing and SOAP message parsing out of the box. This means that every (group of) developer(s) still has to implement new functionality using the existing tools in order to process WSDL documents, parsing SOAP messages and create SOAP messages. Although Symbian OS is an operating system for which you can write your own applications (with C++ or Java), it can be a bit of trouble to execute programs on the OS, because some versions of Symbian OS break binary compatibility of software that has been written for early versions of the OS. On the other hand, the API of the OS is quite stable so it is very easy to implement a piece of software for the target system.

3 Structure

3.1 WESSYL: The Web Services SYmbian Library

The architecture of WESSYL was presented by Franch, Aiello [3]. In this paper they proposes a generic web services library architecture, which is given the name WESSYL. They stated that one of the unique features of the library is runtime WSDL document parsing. The benefits of runtime parsing are the following:

- Greater modularity. One is able to actually add mobile phones to an existing network of web services and take away phones out of an existing network, without actually preventing the execution of the functionality of the web services.
- Adaptibility to change. If something changes dramatically in the environment of the phone, then that means that the phone keeps on running and trying to perform the tasks that it is supposed to do. If something changes in the service description, then it could still be possible to interact with the service.
- Portability in the creation of mobile phone applications. This means that the library can be used for applications that are running on different phones.

The flow of the application using the WESSYL library is the following:

1. After user input, a WSDL file is retrieved through a network connection.
2. a XML parser generates a hierarchical data structure representing the service description.
3. The application using the library then navigates the structure to list the available services and endpoints.
4. The application might decide to invoke some operation of the services.
5. The library prepares the SOAP message with the parameters.
6. The SOAP message can be send through the network (using gSOAP or Nokia WS Framework for example)
7. Possible return values will be parsed by the library and returned to the application.

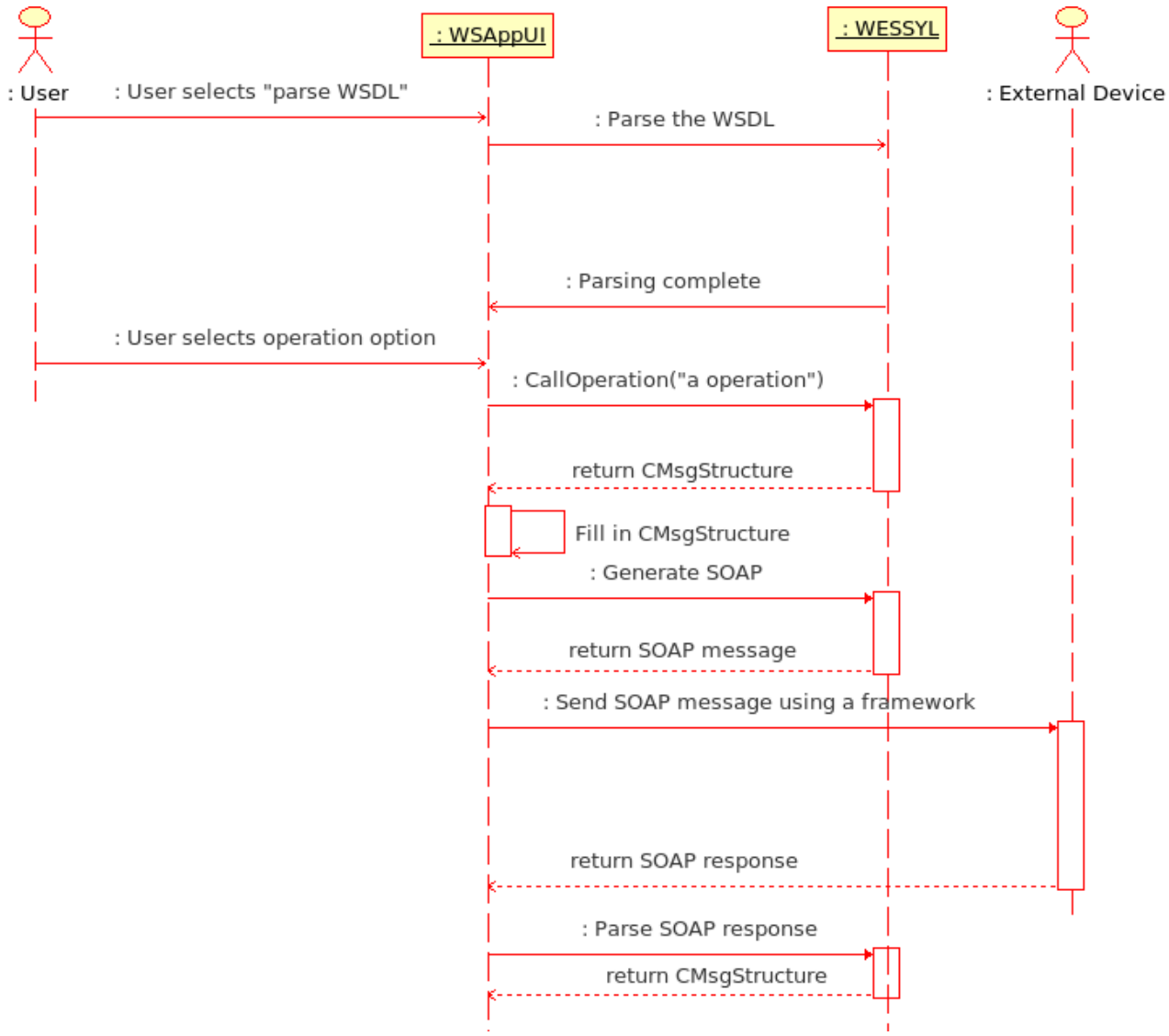

Figure 1: WESSYL sequence diagram

3.2 WSLib: the main class

The main entry point for the programmer of the application is the `WSLib` class. The WSLib class is a major class and it contains the following functions:

- `NewL()`: This function is called when a new WSLib instance needs to be initialized.
- `LoadWsdlL("a file")`: this function is used to let the library parse a given WSDL file. The library creates a CMsgStructure datastructure.
- `GetInputMessageStructureL("a servicename", "port", "a operationname")`: This function returns CMsgStructure object that contains the contains the entire structure of the message that we want to send to the web service.
- `CallOperationL("a servicename", "port", "a operationname", CMsgStructure message)`: This function converts an CMsgStructure into an array of 4 descriptor pointers:
  - The first entry is the SOAP data
  - The manner of transport (for example: http)
  - The target address of the SOAP message
  - The SOAPAction (i.e. the called function) which will be added to SOAP message when it is being send

  The following is done: for primitive datatypes, the function only checks the childs of the CMsgStructure object and whenever a child is a complexType (e.g. an array or a sequence of independent objects), the function `PreOrderTraversal` is called with the child. The subtree is then processed and every element of the complexType will be put in the soap message.

The navigation of the WSDL structure has been completed once all the above functions have been invoked.

3.3 About the classes CMsgStructure

The CMsgStructure class is also a major class and it is used for internal representations of SOAP messages. The CMsgStructure class contain the following data:

- Name
- Type
- Value
- Dynamic Array of pointers to other CMsgStructure objects

The name of the object is the name of the XML element that has been read in. The type of the object is the type of the element and the value is the content between the start XML element and the end XML element. This content may or may not be filled in by the application or the remote application, so the value datafield may not always be initialized. The CMsgStructure class also holds an array of pointers to other CMsgStructure objects. This means that one CMsgStructure instance can hold references to other objects and that the entire datastructure is actually a tree consisting of CMsgStructure objects; The application uses the CMsgStructure datastructure to retrieve all the data that it needs. In order to this, the application needs to apply a tree traversal algoritm on the structure in order to get to all the stored data.

3.4 About the CWSDLParser class

This class contains the function `LoadWsdlL`. This function needs to have a filename as the argument. After the file has been opened, it is parsed by a Simple API for XML (SAX) parser. The SAX parser comes with the third edition of the Nokia SDK. The parser contains an event-driven API which contains callback functions that the developer has to implement. As soon as the parser detects an XML element, it does a callback to the appropriate function. The most used function is the StartElement function. It contains if-then-else branches which are used to check which element the function is processing. The use of a SAX parser is highly beneficial, because it doesn't use a lot of memory and it is very fast compared to other types of XML parsers, such as the Document Object Model type parser. With the parser, a WSDL document can be transformed into a tree datastructure which can be traversed by other functions in the library.

3.5 About the CSoapMessageParser class

The class contains the function `parseSoapL` which gets a filename as argument and returns a pointer to a CMsgStructure object. The SoapMessageParser also contains a SAX parser which is used to parse the SOAP message. The contents between a starting XML element and an ending XML

element are parsed as well, but it can be possible that the contents are fully read in, which can cause inconsistencies. The solution to this inconsistency is to reallocate more memory to the contents buffer and parse the contents until an ending element is encountered. Then, the contents are assigned to the value datatype of the current object.

3.6 About The Other Classes

There are some additional classes that haven't been mentioned before in this paper, but are in the library nontheless. The classes are:

- CHashtable
- CBTDiscoverer
- CBluetoothInfo
- CDevicelistContainer

The CBTDiscoverer class is used to discover bluetooth-enabled devices in the vicinity. If a device is discovered, then the name and it's transformed bluetoothaddress will be inserted into a CBluetoothInfo object and added to te CDevicelistContainer object. The CDevicelistContainer object is an UI-element for Symbian OS that is used to list all the bluetooth-enabled devices in the vicinity. The data can then be send to another device using the OBEX protocol when a connection has been set up. The CHashtable class will be presented in the following subsection.

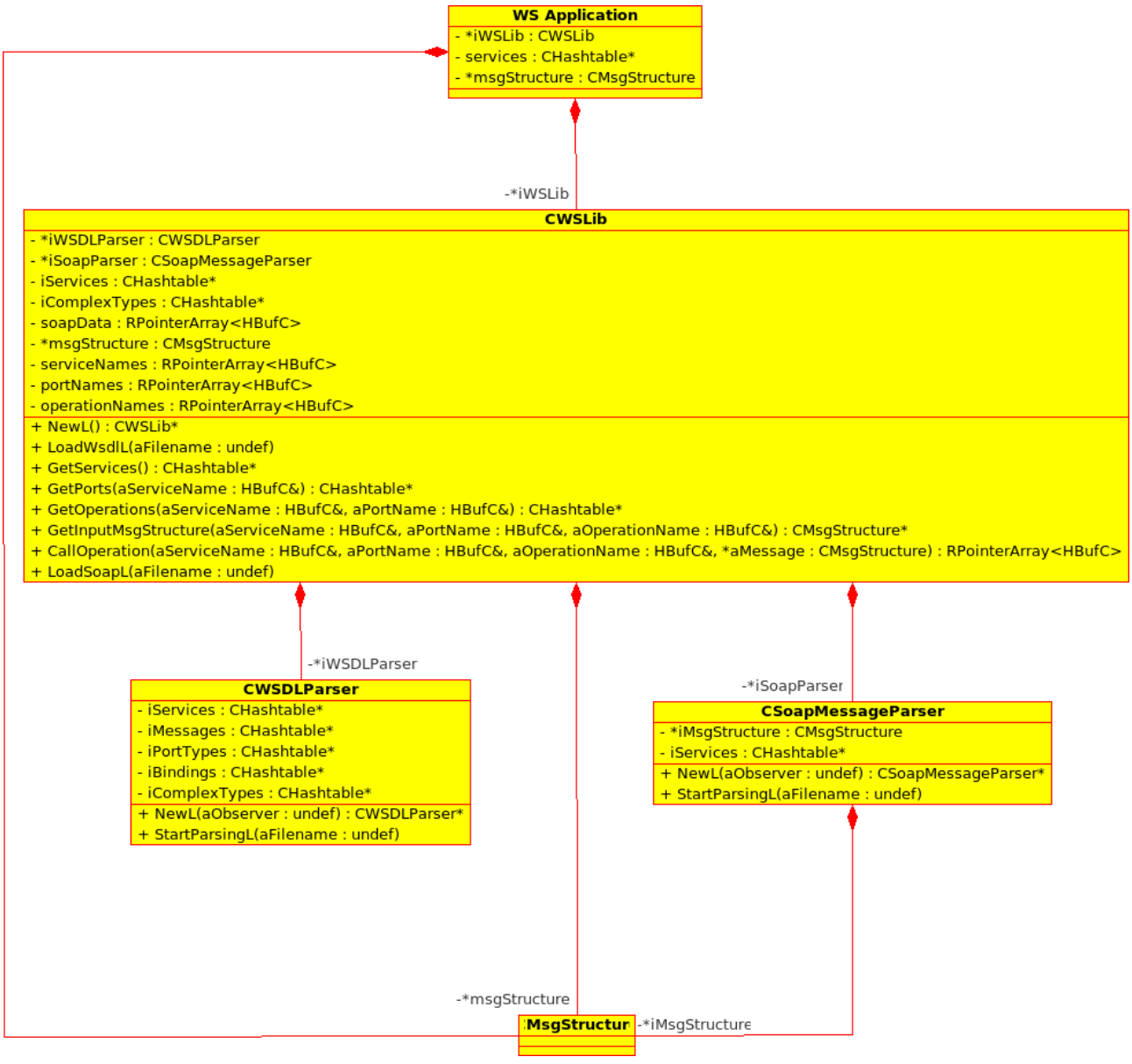

Figure 2: CMsgStructure message

3.7 About the CHashtable

The CHashtable class is a custom hashtable datastructure that is used to store pointers to objects. The class is fundamental and it is used everywhere where it can be used in WESSYL. The CHashtable class uses fixed length arrays in order to save memory. The pointers are put in the arrays using a hash code and a simple compression map. The nature of the hash code is polynomial. This means that the position of the characters in the key are taken into consideration while calculating the hash code. Words like `"stop"` and `"spot"` do not get the same code. The constant used in the hash function is 33. This method lowers the amount of collisions.

The compression map used is the division method. Open addressing is used for collision handling. There are no additional datastructures used in the hashtable. This means that each space in the array is occupied by at most one pointer. It also means that there are no additional datastructures required, thus there is no extra memory allocated.

If a certain place in the array is already occupied by a pointer, then a new place is searched in the array. This method is called Linear Probing. The problem with this method is that the function `removeElement(e)` can't delete the element on the place in the array immediately, because the place of the element, that has to be deleted, might have contained an element, that has been deleted before, and the element, that has to be deleted, can't be traced down. The solution to this problem is to use a secondary array that marks the spots where deleted elements have been placed. These deleted elements are called "deactivated elements". There is a boolean array used in the hashtable to keep track of deactivated elements. [4]

The CHashtable class also has caching abilities. It caches the pointer of the last object returned, the keys and the index of the pointer in the array. Multiple calls to the same operation will result in a $O(1)$ response of the hashtables used in the library.

4 An Example Of Using The Library

4.1 Introduction to the example

In this example a call to a service will be made by a device that is monitoring the heart rhithm of a heart patient. This example can be used as a reference for future mobile web service applications which want to use the functionality of the library. Only the functions that are used by the developer will be discussed. Reference to internal functions will only be made when they support the explanation.

4.2 Initializing the library

An application will be showed here which has no prior access to a WSDL file. The WSDL file will we received during the execution of the program and from that file the functions that the application can invoke will be derived. If the functionality is changed (e.g. a vital function disappears) then the application should not be able to access the service ever again until the function has been reimplemented at the remote service. The first step to initialize the library is to call the Symbian constructor function of the library which returns a pointer to the library object:

```
WSLib *library = WSLib::NewL();
```

The function `NewL()` is a function which is used to initialize an object. The character `L` is used when a function can Leave. A function will only leave when it is for example not able to allocate enough memory for the object. The WSLib class provides the functions LoadWsdlL() and parseSoapL(). The first function can be used to load a WSDL file into memory and the second function can be used to parse the contents of a SOAP file. The class also provides other functions, but these are used to navigate the structure of the parsed WSDL file. When the parsing is complete, a function will be called which is implemented in in an observer class. The function LoadWsdlL() has to be invoked the following way:

```
library->LoadWsdlL(_L("c:\\file.wsdl"));
```

The file `file.wsdl` is contained in the root directory (c:) of the file system of the phone. The function leaves when no file could be loaded and parsed by the function.

4.3 Navigating the structure

Once the WSDL file has been loaded and parsed, it is possible to query the services that are available on the remote service. In order to get all the services, the following function has to be called:

```
CMsgStructure message =
library->GetInputMessageStructureL("aService", "aPort",
"aOperation");
```

This function uses the hashtables of the services, ports and operations to trace down the called operation and it builds the CMsgStructure in place. The CMsgStructure is used to build the message.

4.4 Building the message and calling the operation

This section explains what should happen with the CMsgStructure `message` object once it has been returned from the GetInputMessageStructureL() function. The diagram of the message can be seen in figure 3.

The structure is going to be explained now. What can be seen first is the rootelement of the tree. The rootelement is always available and it is not always filled in, as can be seen here. The rest of the objects can be simple typed or complex typed. The simple type has three characteristics:

1. The type datafield always contains a built-in type of XMLSchema.
2. The value datafield always contains a value that is of the datatype filled in the type datafield.
3. A simple type node does not have any children.

The complex type has three characteristics:

1. The type can be a non-primitive type
2. the value is never valued, because a complextype consist of multiple simple type elements.
3. a complextype element has pointers to children.

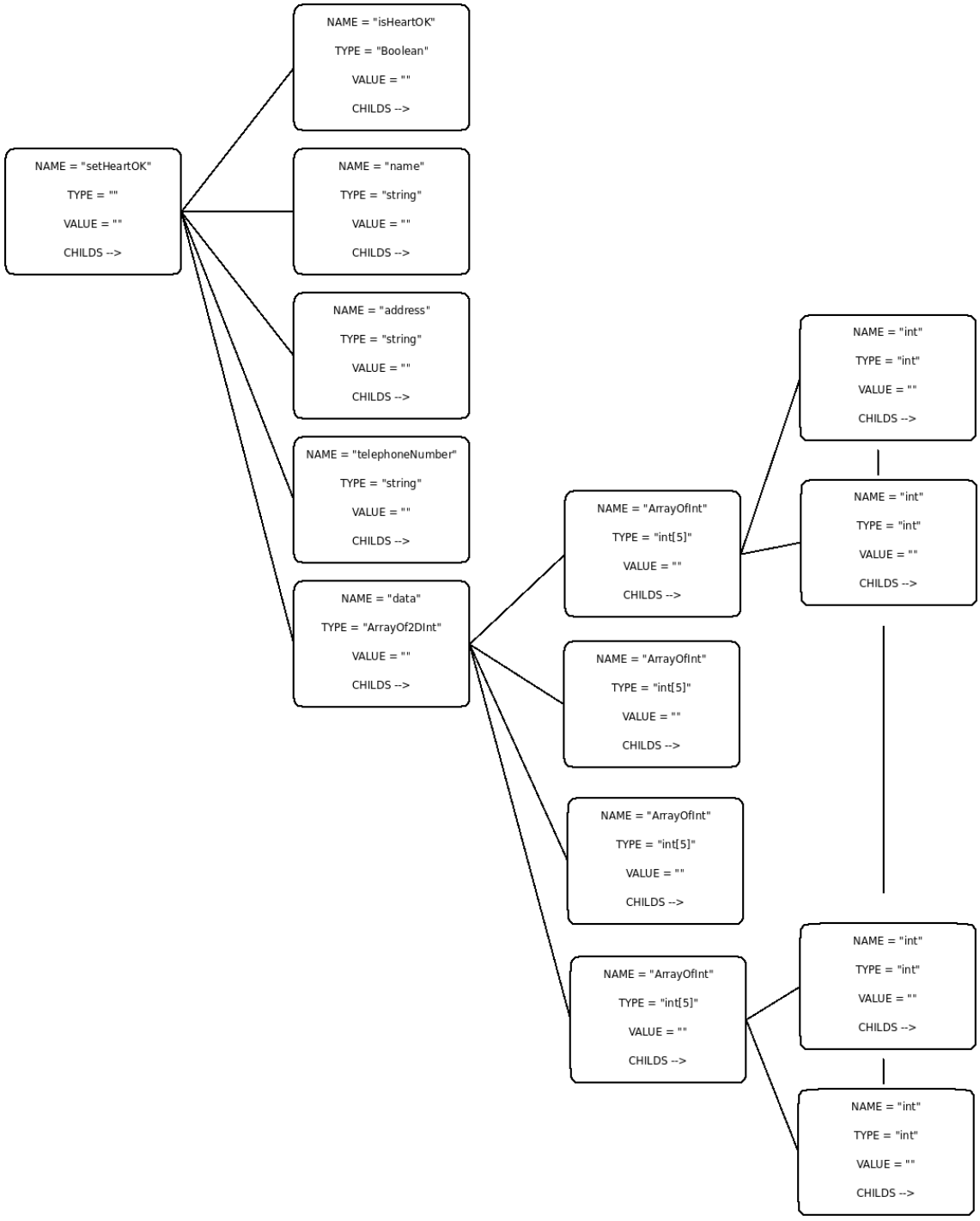

Figure 3: CMsgStructure message

All values of the structure has to be filled in before the message can be send. Suppose that the person is named Jan Klaasen and that he lives in Ergensstraat 1, that his telephonenumber is 012-34567890 and that there is something wrong with his heart. Additional data about for example his heart rhythm can be given in the children of the data node, which is an array of integers which can be transformed into an electrocardiogram. The following steps has to be done in order to fill in the structure:

First, the name, phonenumber, address and the heart condition have to be filled in. This is done as follows:

```
RPointerArray <CMsgStructure> children = message->GetChildren();
children[0]->SetValue(_L("false"));
children[1]->SetValue(_L("Jan Klaasen"));
children[2]->SetValue(_L("Ergensstraat 1"));
children[3]->SetValue(_L("012-34567890"));
```

Then, we want to send the local buffer of integers for an electrocardiogram as well. The integers have to be filled in as follows:

```
if (msgStructure) {

TInt buffer[5] = {1, 2, 3, 4, 5};

RPointerArray <CMsgStructure> children = msgStructure->GetChildren();

children[0]->SetValue(_L8("false"));

children[1]->SetValue(_L8("Jan Klaasen"));

children[2]->SetValue(_L8("Ergensstraat 1"));

children[3]->SetValue(_L8("012-34567890"));

        RPointerArray <CMsgStructure> array = children[4]->GetChildren();

        for (TInt i = 0; i < array.Count(); i++) {
```

```
                RPointerArray<CMsgStructure> array2 = array[i]->GetChildren();
                for (TInt j = 0; j < array2.Count(); j++) {
                        TBuf8<10> buf2;
                        buf2.Num(buffer[i]);
                        array2[j]->SetValue(buf2);
                }
        }
        iAppView->SetTextL(_L("The CMessageStructure contains values!"));
        break;
}
```

Now is everything filled in (see Figure 4) and the structure can be transformed into a SOAP message.

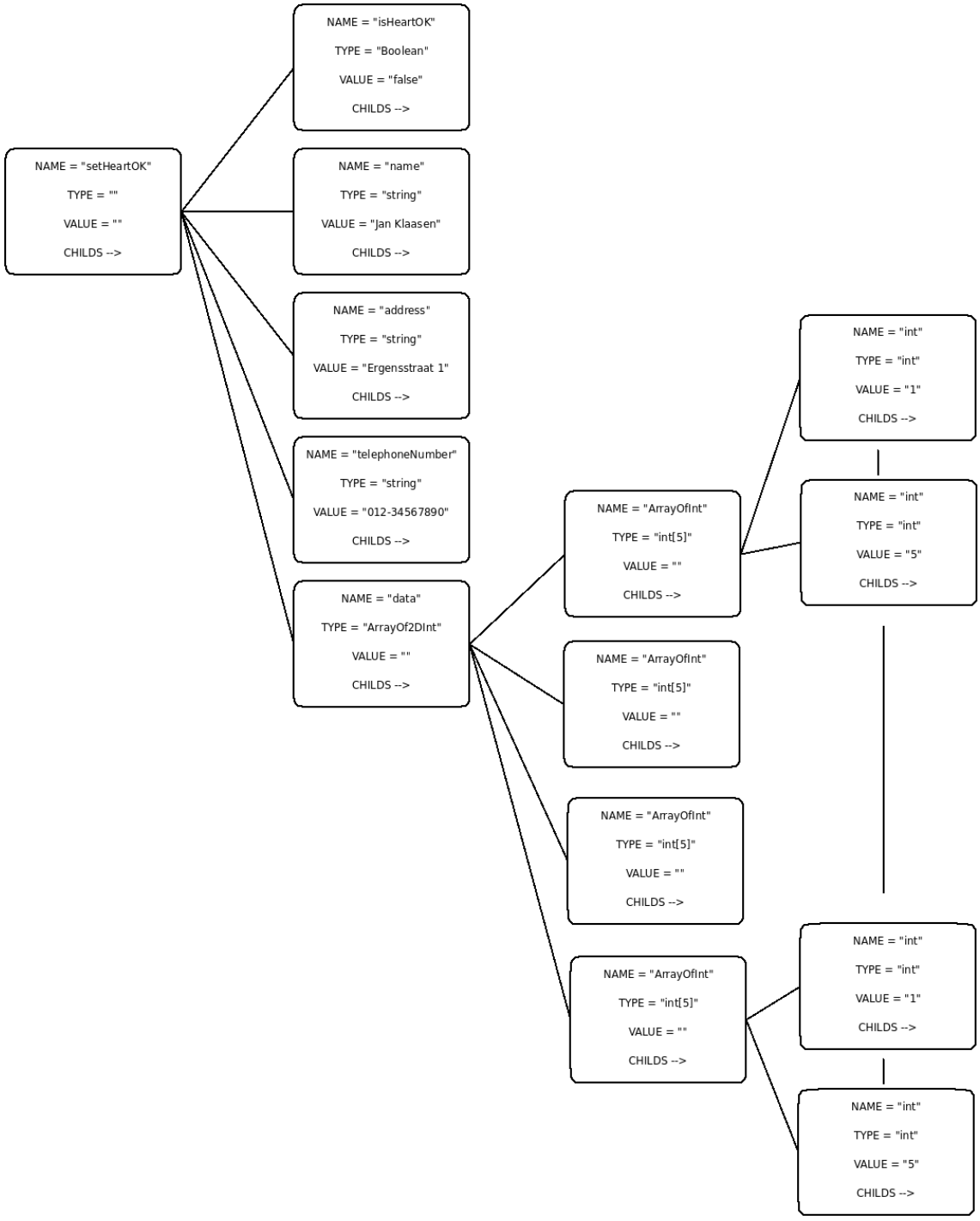

Figure 4: CMsgStructure message

The CMsgStructure structure is now fully filled in and it can now be converted to a SOAP message by invoking the following function:

```
RPointerArray<HbufC> soapData =
library->CallOperationL(services[0], ports[0], operations[0],
                        message);
```

The variable soapData contains all the information needed for sending the message to the remote service. The application can now send the SOAP message to a remote service using gSOAP or the Nokia Web Service Framework. The SOAP message of the CMsgStructure of Figure 2 is translated into the following:

```
<?xml version="1.0"?>
<soap:Envelope xmlns:soap="http://www.w3.org/2001/12/soap-envelope" soap:encodin
<soap:Body>
   <m:setHeartOK xmlns:m="http://example.com/file.wsdl">
      <m:isHeartOK type="boolean">false</m:isHeartOK>
<m:name type="string">Jan Klaasen</m:name>
<m:address type="string">Ergensstraat 1</m:address>
<m:telephonenumber type="string">012-3456789</m:telephonenumber>
<m:data type="array" arrayType="int[5]">
<m:item type="object">
<m:integer type="int">0</m:integer>
</m:item>
<m:item type="object">
<m:integer type="int">1</m:integer>
</m:item>
<m:item type="object">
<m:integer type="int">2</m:integer>
</m:item>
<m:item type="object">
<m:integer type="int">3</m:integer>
</m:item>
<m:item type="object">
<m:integer type="int">4</m:integer>
</m:item>
</m:data>
```

```
   </m:setHeartOK>
</soap:Body>

</soap:Envelope>
```

4.5 Handling a SOAP message

When a SOAP message is received from the endpoint, it needs to be transformed in such a way that it can be used by the application. For the transformation, the following function needs to be invoked:

```
CMsgStructure answer = library->parseSoapL(soapfile);
```

The function transforms the SOAP message above into a CMsgStructure that looks like the one in Figure 2.

5 Comparison of Web Service solutions

5.1 Qualitative Analysis

There are already a couple of Web Service frameworks in existence today. The most notable are gSOAP, Java ME and Nokia WSF.

- gSOAP: This toolkit is a portable open-source toolkit which is intended for C and C++. Its main focus is speed and a small footprint. It has support for WS-* protocols and it is the only SOAP/XML toolkit that uses automatic mappings for user-defined C and C++ datatypes. [6]
- Java ME Web Services API (WSA), JSR 172: This toolkit is geared towards phones with Java ME capabilities. This toolkit supports phones with CLDC 1.x and MIDP 1.x and 2.x. WSA may come with a optional packages, such as JAXP and JAX-RPC. The JAXP package contains the Java XML parser and the JAX-RPC package contains the stub generator. [7]
- Nokia Web Service Framework "Serene" (WSF): The Nokia WSF is available for both Java and Symbian C++ developers. This framework provides basic tools for parsing XML and SOAP messages. In addition to this framework, there is also a plugin tool available for creating Symbian C++ stub code by providing a WSDL file. [8]

	rpc/ encoded	rpc/ literal	document/[7] encoded	document/[8] literal
WESSYL	Yes	Yes	No	Yes
gSOAP	Yes	Yes	No	Yes
Java ME, JSR 172	Yes	Yes	Yes	Yes
Nokia S60 Web Services Framework "Serene"	Yes	Yes	Yes	Yes

[6] `http://www.cs.fsu.edu/~engelen/soap.html`

[7] `http://wiki.forum.nokia.com/index.php?title=Special:PdfPrint&page=Web_Services_API`

[8] `http://wiki.forum.nokia.com/index.php?title=Special:PdfPrint&page=Web_Services_API`

[7] "Nobody follows this style. It is not WS-I compliant. So let's move on." – `http://www.ibm.com/developerworks/webservices/library/ws-whichwsdl/`

[8] WESSYL supports the wrapped version

5.2 Quantitative Analysis

The performance of the WSDL parsers of gSOAP, WESSYL and the Java ME version of WESSYL are measured in this performance analysis.

5.2.1 Hardware

The hardware used for testing is the Nokia E52 mobile phone accessed through Nokia Remote Device Access [9]. This phone contains an ARM 11 600 MHz, 2.1 MIPS CPU , 60 megabyte internal RAM and can contain up to 16 gigabyte of microSD NAND flash memory.
The ARM 11 CPU features ARM Jazelle®technology for embedded Java execution. The operating system used is Symbian OS 9, S60 3rd Edition.

5.2.2 The Test Case

The WSDL parsers of gSOAP, WESSYL and the Java ME version of WESSYL are measured in this performance analysis. The testing is done using 2 WSDL files. The first WSDL file is a custom made WSDL file that can be used by WESSYL to generate a CMsgStructure for the web service application. The second WSDL file is an existing WSDL file used by Amazon. Although the file can't be used directly by WESSYL to generate a CMsgStructure for the application due to the limitations of the library, it can be used to test the WSDL parser.

The WSDL parser of WESSYL is a Simple API for XML parser, which is used to build up a representative structure of the WSDL file in memory. The performance of the XML parser and the buildup of the structure is measured for both WESSYL and Java ME WESSYL.

The WSDL parser of gSOAP converts a WSDL parser into a gSOAP C/C++ header file. However, the parser is not like WESSYL's parser and it is geared towards the creation of header files. The performance of the `read()` and `traverse()` (for validation and cycle checking) are tested. The actual compilation step is not included in the test.

The timings are measured using the Symbian C++ `MicroSecondsFrom("initialTime")`, C++ `gettimeofday()` and Java `Date.getTime()` functions. The timings are in milliseconds and they are the average of three runs.

[9] `http://www.forum.nokia.com/Technology_Topics/Application_Quality/Testing/Remote_Device_Access/`

5.2.3 Performance Test Results

	HeartMeter-Service WSDL	AmazonWeb-Services WSDL
WESSYL	120	332
gSOAP	10	65
Java ME WESSYL	168	567

5.2.4 Discussion

The gSOAP WSDL parser is faster than both versions of WESSYL. This is due to the fact that the gSOAP parser works different from WESSYL's parser. The gSOAP parser is a non-standard DOM parser that uses an internal structure for the representation of the WSDL file. The WESSYL parser builds a structure as well, but due to the fact that the parser is a SAX parser, it can easily be modified and ported to different platforms with different languages, because SAX is a standard XML parsing method used by multiple language development kits (such as Java and .NET) and the CMsgStructure is platform independent. The gSOAP parser on the other hand, is only used for the creation of C/C++ header files, which can only be parsed by gSOAP itself.
Although the processor is capable of executing Java bytecode in hardware, the Java version of WESSYL is slower than the Symbian C++ version of WESSYL. The virtual machine still creates some overhead for the application.

6 Future Work and Conclusion

There is still work to do in order to have a good functional library. There is still a optimization step that can be performed to get a more performance out of the functions. For instance, there are calls made to so-called setters and getters to set a value of a member and to retrieve the value of a member respectively. That can be changed to direct assignment to members. Also, due to the nature of the SAX parser, the validity of an XML file is not checked. The necessary data is only extracted from it. Although it is now a bit lightweight, the performance of the library could be bogged down due to the extra implementation of features.
Anyway, it is clear that mobile devices are powerful enough that web services can be run on them and that a web service can be made by using basic software tools, such as XML parsers. With such XML parsers WSDL and SOAP files can be parsed. WESSYL makes use of these parsers to parse WSDL and SOAP documents at runtime. There can be a seperate part (besides the parsers) in the software to access services and the operations belonging to these services. Based on the selection of the operation a SOAP message will be constructed by the library. This SOAP message will then be ready to be send to another web service. Incoming messages can be parsed and transformed into the internal datastructure. Nevertheless, the support for runtime WSDL parsing on all Symbian phones puts WESSYL in a unique position with respect to other libraries.

References

[1] E. Newcomer, *Understanding Web Services: XML, WSDL, SOAP and UDDI.*

[2] J. Stichbury, *Symbian OS Explained: Effective C++ Programming for Smartphones.*

[3] G. Franch and M. Aiello, "Web services on symbian os mobile devices: The web services symbian library (wessyl)," tech. rep.

[4] M. Goodrich and R. Tamassia, *Algorithm Design.*

7 Appendix A - HeartMeterService WSDL

```
<?xml version="1.0"?>

<definitions name="HeartMeter"

          targetNamespace="http://example.com/file.wsdl"

          xmlns:tns="http://example.com/file.wsdl"

          xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"

          xmlns:xsd1="http://example.com/file.xsd"

          xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"

          xmlns="http://schemas.xmlsoap.org/wsdl/">



  <types>

    <schema xmlns="http://www.w3.org/2001/XMLSchema"

         targetNamespace="http://www.example.com/schema"

         xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

         xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/">


      <complexType name="ArrayOf2DInt">

        <complexContent>

          <restriction base="soapenc:Array">
```

```
          <attribute ref="soapenc:arrayType"
              arrayType="ArrayOfInt[5]"/>
        </restriction>
      </complexContent>
    </complexType>

    <complexType name="ArrayOfInt">
      <complexContent>
        <restriction base="soapenc:Array">
          <attribute ref="soapenc:arrayType"
              arrayType="int[5]"/>
        </restriction>
      </complexContent>
    </complexType>
  </schema>

</types>
  <message name="setHeartOK">
    <part name="HeartOK" type="xsd:boolean"/>
  <part name="Name" type="xsd:string"/>
  <part name="Address" type="xsd:string"/>
```

```
  <part name="Telephonenumber" type="xsd:string"/>
  <part name="Data" type="tns:ArrayOf2DInt"/>
  </message>

<message name ="foo">
  <part name="bar" type="xsd:string"/>
</message>

  <portType name="HeartMeterPortType">
  <operation name="setHeartOK">
       <input message="tns:setHeartOK"/>
    </operation>
    <operation name="foo">
      <input message="tns:foo"/>
      <output message="tns:setHeartOK"/>
    </operation>
    <operation name="bar">
      <input message="tns:foo"/>
    </operation>
  </portType>
```

```
    <binding name="HeartMeterSoapBinding" type="tns:HeartMeterPortType">
        <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/htt
  <operation name="setHeartOK">
            <soap:operation soapAction="http://example.com/setHeartOK"/>
            <input>
                <soap:body use="encoded" namespace="http://example.com/heartmeter
                           encodingStyle="http://schemas.xmlsoap.org/soap/encodin
            </input>
        </operation>
    </binding>

    <service name="HeartMeterService">
        <documentation>My first service</documentation>
        <port name="HeartMeterPort" binding="tns:HeartMeterSoapBinding">
            <soap:address location="http://example.com/heartmeter"/>
        </port>
    </service>
</definitions>
```

8 Appendix B - AmazonWebServices WSDL

```
<wsdl:definitions xmlns:typens="http://soap.amazon.com" xmlns:xsd="http://www.w3

<wsdl:types>

<xsd:schema xmlns="" xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespac

<xsd:complexType name="ProductLineArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">

<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:ProductLine[]"/>

</xsd:restriction>

</xsd:complexContent>

</xsd:complexType>

<xsd:complexType name="ProductLine">

<xsd:all>

<xsd:element name="Mode" type="xsd:string" minOccurs="0"/>

<xsd:element name="ProductInfo" type="typens:ProductInfo" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<xsd:complexType name="ProductInfo">

<xsd:all>

<xsd:element name="TotalResults" type="xsd:string" minOccurs="0"/>
```

```
<!-- Total number of Search Results -->

<xsd:element name="TotalPages" type="xsd:string" minOccurs="0"/>

<!-- Total number of Pages of Search Results -->

<xsd:element name="ListName" type="xsd:string" minOccurs="0"/>

<!-- Listmania list name -->

<xsd:element name="Details" type="typens:DetailsArray" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<!-- Product Details

L - indicates that a piece of data is returned in a "lite" request

O - indicates that a piece of data will be returned only if it exists for the sp

<xsd:complexType name="DetailsArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">

<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:Details[]"/>

</xsd:restriction>

</xsd:complexContent>

</xsd:complexType>

<xsd:complexType name="Details">

<xsd:all>

<xsd:element name="Url" type="xsd:string" minOccurs="0"/>
```

```
<!-- L -->
<xsd:element name="Asin" type="xsd:string" minOccurs="0"/>
<!-- L -->
<xsd:element name="ProductName" type="xsd:string" minOccurs="0"/>
<!-- L -->
<xsd:element name="Catalog" type="xsd:string" minOccurs="0"/>
<!-- L -->
<xsd:element name="KeyPhrases" type="typens:KeyPhraseArray" minOccurs="0"/>
<xsd:element name="Artists" type="typens:ArtistArray" minOccurs="0"/>
<!-- LO -->
<xsd:element name="Authors" type="typens:AuthorArray" minOccurs="0"/>
<!-- LO -->
<xsd:element name="Mpn" type="xsd:string" minOccurs="0"/>
<xsd:element name="Starring" type="typens:StarringArray" minOccurs="0"/>
<xsd:element name="Directors" type="typens:DirectorArray" minOccurs="0"/>
<xsd:element name="TheatricalReleaseDate" type="xsd:string" minOccurs="0"/>
<xsd:element name="ReleaseDate" type="xsd:string" minOccurs="0"/>
<!-- LO -->
<xsd:element name="Manufacturer" type="xsd:string" minOccurs="0"/>
<!-- LO -->
```

```
<xsd:element name="Distributor" type="xsd:string" minOccurs="0"/>
<xsd:element name="ImageUrlSmall" type="xsd:string" minOccurs="0"/>
<!-- LO -->
<xsd:element name="ImageUrlMedium" type="xsd:string" minOccurs="0"/>
<!-- LO -->
<xsd:element name="ImageUrlLarge" type="xsd:string" minOccurs="0"/>
<!-- LO -->
<xsd:element name="ListPrice" type="xsd:string" minOccurs="0"/>
<!-- LO -->
<xsd:element name="OurPrice" type="xsd:string" minOccurs="0"/>
<!-- LO -->
<xsd:element name="UsedPrice" type="xsd:string" minOccurs="0"/>
<!-- LO -->
<xsd:element name="RefurbishedPrice" type="xsd:string" minOccurs="0"/>
<xsd:element name="CollectiblePrice" type="xsd:string" minOccurs="0"/>
<xsd:element name="ThirdPartyNewPrice" type="xsd:string" minOccurs="0"/>
<xsd:element name="NumberOfOfferings" type="xsd:string" minOccurs="0"/>
<xsd:element name="ThirdPartyNewCount" type="xsd:string" minOccurs="0"/>
<xsd:element name="UsedCount" type="xsd:string" minOccurs="0"/>
<xsd:element name="CollectibleCount" type="xsd:string" minOccurs="0"/>
<xsd:element name="RefurbishedCount" type="xsd:string" minOccurs="0"/>
```

```
<xsd:element name="ThirdPartyProductInfo" type="typens:ThirdPartyProductInfo" mi
<xsd:element name="SalesRank" type="xsd:string" minOccurs="0"/>
<xsd:element name="BrowseList" type="typens:BrowseNodeArray" minOccurs="0"/>
<xsd:element name="Media" type="xsd:string" minOccurs="0"/>
<xsd:element name="ReadingLevel" type="xsd:string" minOccurs="0"/>
                                        <xsd:element name="NumberOfPages" type="
                                        <xsd:element name="NumberOfIssues" type=
                                        <xsd:element name="IssuesPerYear" type="
                                        <xsd:element name="SubscriptionLength" t
                                        <xsd:element name="DeweyNumber" type="xs
                                        <xsd:element name="RunningTime" type="xs
<xsd:element name="Publisher" type="xsd:string" minOccurs="0"/>
<xsd:element name="NumMedia" type="xsd:string" minOccurs="0"/>
<xsd:element name="Isbn" type="xsd:string" minOccurs="0"/>
<xsd:element name="Features" type="typens:FeaturesArray" minOccurs="0"/>
<xsd:element name="MpaaRating" type="xsd:string" minOccurs="0"/>
<xsd:element name="EsrbRating" type="xsd:string" minOccurs="0"/>
<xsd:element name="AgeGroup" type="xsd:string" minOccurs="0"/>
<xsd:element name="Availability" type="xsd:string" minOccurs="0"/>
<xsd:element name="Upc" type="xsd:string" minOccurs="0"/>
```

```
<xsd:element name="Tracks" type="typens:TrackArray" minOccurs="0"/>
<xsd:element name="Accessories" type="typens:AccessoryArray" minOccurs="0"/>
<xsd:element name="Platforms" type="typens:PlatformArray" minOccurs="0"/>
<xsd:element name="Encoding" type="xsd:string" minOccurs="0"/>
<xsd:element name="Reviews" type="typens:Reviews" minOccurs="0"/>
<xsd:element name="SimilarProducts" type="typens:SimilarProductsArray" minOccurs
<xsd:element name="Lists" type="typens:ListArray" minOccurs="0"/>
<xsd:element name="Status" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="KeyPhraseArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:KeyPhrase[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="KeyPhrase">
<xsd:all>
<xsd:element name="KeyPhrase" type="xsd:string" minOccurs="0"/>
<xsd:element name="Type" type="xsd:string" minOccurs="0"/>
```

```
</xsd:all>
</xsd:complexType>
<xsd:complexType name="ArtistArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="AuthorArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="StarringArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
```

```
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="DirectorArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="BrowseNodeArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:BrowseNode[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="BrowseNode">
<xsd:all>
<xsd:element name="BrowseId" type="xsd:string" minOccurs="0"/>
<xsd:element name="BrowseName" type="xsd:string" minOccurs="0"/>
```

```
</xsd:all>

</xsd:complexType>

<xsd:complexType name="FeaturesArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">

<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>

</xsd:restriction>

</xsd:complexContent>

</xsd:complexType>

<xsd:complexType name="TrackArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">

<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:Track[]"/>

</xsd:restriction>

</xsd:complexContent>

</xsd:complexType>

<xsd:complexType name="Track">

<xsd:all>

<xsd:element name="TrackName" type="xsd:string" minOccurs="0"/>

<xsd:element name="ByArtist" type="xsd:string" minOccurs="0"/>
```

```
</xsd:all>
</xsd:complexType>
<xsd:complexType name="AccessoryArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="PlatformArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Reviews">
<xsd:all>
<xsd:element name="AvgCustomerRating" type="xsd:string" minOccurs="0"/>
<xsd:element name="TotalCustomerReviews" type="xsd:string" minOccurs="0"/>
<xsd:element name="CustomerReviews" type="typens:CustomerReviewArray" minOccurs=
```

```
</xsd:all>

</xsd:complexType>

<xsd:complexType name="CustomerReviewArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">

<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:CustomerReview[]"/

</xsd:restriction>

</xsd:complexContent>

</xsd:complexType>

<xsd:complexType name="CustomerReview">

<xsd:all>

<xsd:element name="Rating" type="xsd:string" minOccurs="0"/>

<xsd:element name="Summary" type="xsd:string" minOccurs="0"/>

<xsd:element name="Comment" type="xsd:string" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<xsd:complexType name="SimilarProductsArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">

<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>
```

```
</xsd:restriction>

</xsd:complexContent>

</xsd:complexType>

<xsd:complexType name="ListArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">

<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>

</xsd:restriction>

</xsd:complexContent>

</xsd:complexType>

<xsd:complexType name="MarketplaceSearch">

<xsd:all>

<xsd:element name="MarketplaceSearchDetails" type="typens:MarketplaceSearchDetai

</xsd:all>

</xsd:complexType>

<xsd:complexType name="SellerProfile">

<xsd:all>

<xsd:element name="SellerProfileDetails" type="typens:SellerProfileDetailsArray"

</xsd:all>

</xsd:complexType>

<xsd:complexType name="SellerSearch">
```

```
<xsd:all>
<xsd:element name="SellerSearchDetails" type="typens:SellerSearchDetailsArray"/>
</xsd:all>
</xsd:complexType>
<!--<xsd:complexType name="ExchangeSearch">
<xsd:all>
<xsd:element name="ListingProductDetails" type="typens:ListingProductDetailsArra
</xsd:all>
</xsd:complexType> -->
<xsd:complexType name="MarketplaceSearchDetails">
<xsd:all>
<xsd:element name="NumberOfOpenListings" type="xsd:string" minOccurs="0"/>
<xsd:element name="ListingProductInfo" type="typens:ListingProductInfo" minOccur
</xsd:all>
</xsd:complexType>
<xsd:complexType name="MarketplaceSearchDetailsArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:MarketplaceSearchD
</xsd:restriction>
```

```
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SellerProfileDetails">
<xsd:all>
<xsd:element name="SellerNickname" type="xsd:string" minOccurs="0"/>
<xsd:element name="OverallFeedbackRating" type="xsd:string" minOccurs="0"/>
<xsd:element name="NumberOfFeedback" type="xsd:string" minOccurs="0"/>
<xsd:element name="NumberOfCanceledBids" type="xsd:string" minOccurs="0"/>
<xsd:element name="NumberOfCanceledAuctions" type="xsd:string" minOccurs="0"/>
<xsd:element name="StoreId" type="xsd:string" minOccurs="0"/>
<xsd:element name="StoreName" type="xsd:string" minOccurs="0"/>
<xsd:element name="SellerFeedback" type="typens:SellerFeedback" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="SellerProfileDetailsArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:SellerProfileDetai
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
```

```
<xsd:complexType name="SellerSearchDetails">
<xsd:all>
<xsd:element name="SellerNickname" type="xsd:string" minOccurs="0"/>
<xsd:element name="StoreId" type="xsd:string" minOccurs="0"/>
<xsd:element name="StoreName" type="xsd:string" minOccurs="0"/>
<xsd:element name="NumberOfOpenListings" type="xsd:string" minOccurs="0"/>
<xsd:element name="ListingProductInfo" type="typens:ListingProductInfo" minOccur
</xsd:all>
</xsd:complexType>
<xsd:complexType name="SellerSearchDetailsArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:SellerSearchDetail
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ListingProductInfo">
<xsd:all>
<xsd:element name="ListingProductDetails" type="typens:ListingProductDetailsArra
</xsd:all>
```

```
</xsd:complexType>
<xsd:complexType name="ListingProductDetailsArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:ListingProductDeta
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ListingProductDetails">
<xsd:all>
<xsd:element name="ExchangeId" type="xsd:string" minOccurs="0"/>
<xsd:element name="ListingId" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeTitle" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangePrice" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeAsin" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeEndDate" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeTinyImage" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeSellerId" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeSellerNickname" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeStartDate" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeStatus" type="xsd:string" minOccurs="0"/>
```

```
<xsd:element name="ExchangeQuantity" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeQuantityAllocated" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeFeaturedCategory" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeCondition" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeConditionType" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeAvailability" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeOfferingType" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeSellerState" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeSellerCountry" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeSellerRating" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="SellerFeedback">
<xsd:all>
<xsd:element name="Feedback" type="typens:FeedbackArray"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="FeedbackArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
```

```
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:Feedback[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Feedback">
<xsd:all>
<xsd:element name="FeedbackRating" type="xsd:string" minOccurs="0"/>
<xsd:element name="FeedbackComments" type="xsd:string" minOccurs="0"/>
<xsd:element name="FeedbackDate" type="xsd:string" minOccurs="0"/>
<xsd:element name="FeedbackRater" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="ThirdPartyProductInfo">
<xsd:all>
<xsd:element name="ThirdPartyProductDetails" type="typens:ThirdPartyProductDetai
</xsd:all>
</xsd:complexType>
<xsd:complexType name="ThirdPartyProductDetailsArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:ThirdPartyProductD
```

```
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ThirdPartyProductDetails">
<xsd:all>
<xsd:element name="OfferingType" type="xsd:string" minOccurs="0"/>
<xsd:element name="SellerId" type="xsd:string" minOccurs="0"/>
<xsd:element name="SellerNickname" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeId" type="xsd:string" minOccurs="0"/>
<xsd:element name="OfferingPrice" type="xsd:string" minOccurs="0"/>
<xsd:element name="Condition" type="xsd:string" minOccurs="0"/>
<xsd:element name="ConditionType" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeAvailability" type="xsd:string" minOccurs="0"/>
<xsd:element name="SellerCountry" type="xsd:string" minOccurs="0"/>
<xsd:element name="SellerState" type="xsd:string" minOccurs="0"/>
<xsd:element name="ShipComments" type="xsd:string" minOccurs="0"/>
<xsd:element name="SellerRating" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="KeywordRequest">
```

```
<xsd:all>
<xsd:element name="keyword" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="mode" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
<xsd:element name="variations" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="PowerRequest">
<xsd:all>
<xsd:element name="power" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="mode" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
```

```
<xsd:element name="variations" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="BrowseNodeRequest">
<xsd:all>
<xsd:element name="browse_node" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="mode" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="AsinRequest">
<xsd:all>
<xsd:element name="asin" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
```

```
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="offer" type="xsd:string" minOccurs="0"/>
<xsd:element name="offerpage" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="BlendedRequest">
<xsd:all>
<xsd:element name="blended" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="UpcRequest">
<xsd:all>
<xsd:element name="upc" type="xsd:string"/>
<xsd:element name="mode" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
```

```
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
<xsd:element name="variations" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="ArtistRequest">
<xsd:all>
<xsd:element name="artist" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="mode" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
<xsd:element name="variations" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
```

```
<xsd:complexType name="AuthorRequest">
<xsd:all>
<xsd:element name="author" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="mode" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
<xsd:element name="variations" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="ActorRequest">
<xsd:all>
<xsd:element name="actor" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="mode" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
```

```
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
<xsd:element name="variations" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="DirectorRequest">
<xsd:all>
<xsd:element name="director" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="mode" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
<xsd:element name="variations" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="ExchangeRequest">
<xsd:all>
```

```
<xsd:element name="exchange_id" type="xsd:string"/>

<xsd:element name="tag" type="xsd:string"/>

<xsd:element name="type" type="xsd:string"/>

<xsd:element name="devtag" type="xsd:string"/>

<xsd:element name="locale" type="xsd:string" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<xsd:complexType name="ManufacturerRequest">

<xsd:all>

<xsd:element name="manufacturer" type="xsd:string"/>

<xsd:element name="page" type="xsd:string"/>

<xsd:element name="mode" type="xsd:string"/>

<xsd:element name="tag" type="xsd:string"/>

<xsd:element name="type" type="xsd:string"/>

<xsd:element name="devtag" type="xsd:string"/>

<xsd:element name="sort" type="xsd:string" minOccurs="0"/>

<xsd:element name="variations" type="xsd:string" minOccurs="0"/>

<xsd:element name="locale" type="xsd:string" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<xsd:complexType name="ListManiaRequest">
```

```
<xsd:all>
<xsd:element name="lm_id" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="WishlistRequest">
<xsd:all>
<xsd:element name="wishlist_id" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="MarketplaceRequest">
```

```
<xsd:all>
<xsd:element name="marketplace_search" type="xsd:string"/>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="type" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="page" type="xsd:string"/>
<xsd:element name="keyword" type="xsd:string" minOccurs="0"/>
<xsd:element name="keyword_search" type="xsd:string" minOccurs="0"/>
<xsd:element name="browse_id" type="xsd:string" minOccurs="0"/>
<xsd:element name="zipcode" type="xsd:string" minOccurs="0"/>
<xsd:element name="area_id" type="xsd:string" minOccurs="0"/>
<xsd:element name="geo" type="xsd:string" minOccurs="0"/>
<xsd:element name="sort" type="xsd:string" minOccurs="0"/>
<xsd:element name="listing_id" type="xsd:string" minOccurs="0"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
<xsd:element name="index" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="SellerProfileRequest">
<xsd:all>
<xsd:element name="seller_id" type="xsd:string"/>
```

```
<xsd:element name="tag" type="xsd:string"/>

<xsd:element name="type" type="xsd:string"/>

<xsd:element name="devtag" type="xsd:string"/>

<xsd:element name="page" type="xsd:string"/>

<xsd:element name="locale" type="xsd:string" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<xsd:complexType name="SellerRequest">

<xsd:all>

<xsd:element name="seller_id" type="xsd:string"/>

<xsd:element name="tag" type="xsd:string"/>

<xsd:element name="type" type="xsd:string"/>

<xsd:element name="devtag" type="xsd:string"/>

<xsd:element name="offerstatus" type="xsd:string"/>

<xsd:element name="page" type="xsd:string"/>

<xsd:element name="seller_browse_id" type="xsd:string" minOccurs="0"/>

<xsd:element name="keyword" type="xsd:string" minOccurs="0"/>

<xsd:element name="locale" type="xsd:string" minOccurs="0"/>

<xsd:element name="index" type="xsd:string" minOccurs="0"/>

</xsd:all>
```

```
</xsd:complexType>

<xsd:complexType name="SimilarityRequest">

<xsd:all>

<xsd:element name="asin" type="xsd:string"/>

<xsd:element name="tag" type="xsd:string"/>

<xsd:element name="type" type="xsd:string"/>

<xsd:element name="devtag" type="xsd:string"/>

<xsd:element name="locale" type="xsd:string" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<!-- Shopping Cart -->

<xsd:complexType name="ItemIdArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">

<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="xsd:string[]"/>

</xsd:restriction>

</xsd:complexContent>

</xsd:complexType>

<xsd:complexType name="ItemArray">

<xsd:complexContent>

<xsd:restriction base="soapenc:Array">
```

```
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:Item[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Item">
<xsd:all>
<xsd:element name="ItemId" type="xsd:string"/>
<xsd:element name="ProductName" type="xsd:string"/>
<xsd:element name="Catalog" type="xsd:string"/>
<xsd:element name="Asin" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeId" type="xsd:string" minOccurs="0"/>
<xsd:element name="Quantity" type="xsd:string"/>
<xsd:element name="ListPrice" type="xsd:string" minOccurs="0"/>
<xsd:element name="OurPrice" type="xsd:string"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="ItemQuantityArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:ItemQuantity[]"/>
```

```
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ItemQuantity">
<xsd:all>
<xsd:element name="ItemId" type="xsd:string"/>
<xsd:element name="Quantity" type="xsd:string"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="AddItemArray">
<xsd:complexContent>
<xsd:restriction base="soapenc:Array">
<xsd:attribute ref="soapenc:arrayType" wsdl:arrayType="typens:AddItem[]"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="AddItem">
<xsd:all>
<xsd:element name="Asin" type="xsd:string" minOccurs="0"/>
<xsd:element name="ExchangeId" type="xsd:string" minOccurs="0"/>
<xsd:element name="Quantity" type="xsd:string"/>
```

```
</xsd:all>

</xsd:complexType>

<xsd:complexType name="ShoppingCart">

<xsd:all>

<xsd:element name="CartId" type="xsd:string"/>

<xsd:element name="HMAC" type="xsd:string"/>

<xsd:element name="PurchaseUrl" type="xsd:string"/>

<xsd:element name="Items" type="typens:ItemArray" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<xsd:complexType name="GetShoppingCartRequest">

<xsd:all>

<xsd:element name="tag" type="xsd:string"/>

<xsd:element name="devtag" type="xsd:string"/>

<xsd:element name="CartId" type="xsd:string"/>

<xsd:element name="HMAC" type="xsd:string"/>

<xsd:element name="locale" type="xsd:string" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<xsd:complexType name="ClearShoppingCartRequest">
```

```
<xsd:all>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="CartId" type="xsd:string"/>
<xsd:element name="HMAC" type="xsd:string"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="AddShoppingCartItemsRequest">
<xsd:all>
<xsd:element name="tag" type="xsd:string"/>
<xsd:element name="devtag" type="xsd:string"/>
<xsd:element name="CartId" type="xsd:string" minOccurs="0"/>
<xsd:element name="HMAC" type="xsd:string" minOccurs="0"/>
<xsd:element name="Items" type="typens:AddItemArray"/>
<xsd:element name="locale" type="xsd:string" minOccurs="0"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="RemoveShoppingCartItemsRequest">
<xsd:all>
<xsd:element name="tag" type="xsd:string"/>
```

```
<xsd:element name="devtag" type="xsd:string"/>

<xsd:element name="CartId" type="xsd:string"/>

<xsd:element name="HMAC" type="xsd:string"/>

<xsd:element name="Items" type="typens:ItemIdArray"/>

<xsd:element name="locale" type="xsd:string" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

<xsd:complexType name="ModifyShoppingCartItemsRequest">

<xsd:all>

<xsd:element name="tag" type="xsd:string"/>

<xsd:element name="devtag" type="xsd:string"/>

<xsd:element name="CartId" type="xsd:string"/>

<xsd:element name="HMAC" type="xsd:string"/>

<xsd:element name="Items" type="typens:ItemQuantityArray"/>

<xsd:element name="locale" type="xsd:string" minOccurs="0"/>

</xsd:all>

</xsd:complexType>

</xsd:schema>

</wsdl:types>

<message name="KeywordSearchRequest">
```

```
<!-- Messages for Amazon Web APIs -->
<part name="KeywordSearchRequest" type="typens:KeywordRequest"/>
</message>
<message name="KeywordSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="PowerSearchRequest">
<part name="PowerSearchRequest" type="typens:PowerRequest"/>
</message>
<message name="PowerSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="BrowseNodeSearchRequest">
<part name="BrowseNodeSearchRequest" type="typens:BrowseNodeRequest"/>
</message>
<message name="BrowseNodeSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="AsinSearchRequest">
<part name="AsinSearchRequest" type="typens:AsinRequest"/>
</message>
```

```
<message name="AsinSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="BlendedSearchRequest">
<part name="BlendedSearchRequest" type="typens:BlendedRequest"/>
</message>
<message name="BlendedSearchResponse">
<part name="return" type="typens:ProductLineArray"/>
</message>
<message name="UpcSearchRequest">
<part name="UpcSearchRequest" type="typens:UpcRequest"/>
</message>
<message name="UpcSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="AuthorSearchRequest">
<part name="AuthorSearchRequest" type="typens:AuthorRequest"/>
</message>
<message name="AuthorSearchResponse">
<part name="return" type="typens:ProductInfo"/>
```

```
</message>
<message name="ArtistSearchRequest">
<part name="ArtistSearchRequest" type="typens:ArtistRequest"/>
</message>
<message name="ArtistSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="ActorSearchRequest">
<part name="ActorSearchRequest" type="typens:ActorRequest"/>
</message>
<message name="ActorSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="DirectorSearchRequest">
<part name="DirectorSearchRequest" type="typens:DirectorRequest"/>
</message>
<message name="DirectorSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="ExchangeSearchRequest">
<part name="ExchangeSearchRequest" type="typens:ExchangeRequest"/>
```

```
</message>
<message name="ExchangeSearchResponse">
<part name="return" type="typens:ListingProductDetails"/>
</message>
<message name="ManufacturerSearchRequest">
<part name="ManufacturerSearchRequest" type="typens:ManufacturerRequest"/>
</message>
<message name="ManufacturerSearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="MarketplaceSearchRequest">
<part name="MarketplaceSearchRequest" type="typens:MarketplaceRequest"/>
</message>
<message name="MarketplaceSearchResponse">
<part name="return" type="typens:MarketplaceSearch"/>
</message>
<message name="ListManiaSearchRequest">
<part name="ListManiaSearchRequest" type="typens:ListManiaRequest"/>
</message>
<message name="ListManiaSearchResponse">
```

```
<part name="return" type="typens:ProductInfo"/>

</message>

<message name="WishlistSearchRequest">

<part name="WishlistSearchRequest" type="typens:WishlistRequest"/>

</message>

<message name="WishlistSearchResponse">

<part name="return" type="typens:ProductInfo"/>

</message>

<message name="SellerProfileSearchRequest">

<part name="SellerProfileSearchRequest" type="typens:SellerProfileRequest"/>

</message>

<message name="SellerProfileSearchResponse">

<part name="return" type="typens:SellerProfile"/>

</message>

<message name="SellerSearchRequest">

<part name="SellerSearchRequest" type="typens:SellerRequest"/>

</message>

<message name="SellerSearchResponse">

<part name="return" type="typens:SellerSearch"/>

</message>

<message name="SimilaritySearchRequest">
```

```
<part name="SimilaritySearchRequest" type="typens:SimilarityRequest"/>
</message>
<message name="SimilaritySearchResponse">
<part name="return" type="typens:ProductInfo"/>
</message>
<message name="GetShoppingCartRequest">
<part name="GetShoppingCartRequest" type="typens:GetShoppingCartRequest"/>
</message>
<message name="ClearShoppingCartRequest">
<part name="ClearShoppingCartRequest" type="typens:ClearShoppingCartRequest"/>
</message>
<message name="AddShoppingCartItemsRequest">
<part name="AddShoppingCartItemsRequest" type="typens:AddShoppingCartItemsReques
</message>
<message name="RemoveShoppingCartItemsRequest">
<part name="RemoveShoppingCartItemsRequest" type="typens:RemoveShoppingCartItems
</message>
<message name="ModifyShoppingCartItemsRequest">
<part name="ModifyShoppingCartItemsRequest" type="typens:ModifyShoppingCartItems
</message>
```

```
<message name="ShoppingCartResponse">
<part name="ShoppingCart" type="typens:ShoppingCart"/>
</message>
<portType name="AmazonSearchPort">
<!-- Port for Amazon Web APIs -->
<operation name="KeywordSearchRequest">
<input message="typens:KeywordSearchRequest"/>
<output message="typens:KeywordSearchResponse"/>
</operation>
<operation name="PowerSearchRequest">
<input message="typens:PowerSearchRequest"/>
<output message="typens:PowerSearchResponse"/>
</operation>
<operation name="BrowseNodeSearchRequest">
<input message="typens:BrowseNodeSearchRequest"/>
<output message="typens:BrowseNodeSearchResponse"/>
</operation>
<operation name="AsinSearchRequest">
<input message="typens:AsinSearchRequest"/>
<output message="typens:AsinSearchResponse"/>
</operation>
```

```
<operation name="BlendedSearchRequest">
<input message="typens:BlendedSearchRequest"/>
<output message="typens:BlendedSearchResponse"/>
</operation>
<operation name="UpcSearchRequest">
<input message="typens:UpcSearchRequest"/>
<output message="typens:UpcSearchResponse"/>
</operation>
<operation name="AuthorSearchRequest">
<input message="typens:AuthorSearchRequest"/>
<output message="typens:AuthorSearchResponse"/>
</operation>
<operation name="ArtistSearchRequest">
<input message="typens:ArtistSearchRequest"/>
<output message="typens:ArtistSearchResponse"/>
</operation>
<operation name="ActorSearchRequest">
<input message="typens:ActorSearchRequest"/>
<output message="typens:ActorSearchResponse"/>
</operation>
```

```
<operation name="ManufacturerSearchRequest">
<input message="typens:ManufacturerSearchRequest"/>
<output message="typens:ManufacturerSearchResponse"/>
</operation>
<operation name="DirectorSearchRequest">
<input message="typens:DirectorSearchRequest"/>
<output message="typens:DirectorSearchResponse"/>
</operation>
<operation name="ListManiaSearchRequest">
<input message="typens:ListManiaSearchRequest"/>
<output message="typens:ListManiaSearchResponse"/>
</operation>
<operation name="WishlistSearchRequest">
<input message="typens:WishlistSearchRequest"/>
<output message="typens:WishlistSearchResponse"/>
</operation>
<operation name="ExchangeSearchRequest">
<input message="typens:ExchangeSearchRequest"/>
<output message="typens:ExchangeSearchResponse"/>
</operation>
<operation name="MarketplaceSearchRequest">
```

```
<input message="typens:MarketplaceSearchRequest"/>

<output message="typens:MarketplaceSearchResponse"/>

</operation>

<operation name="SellerProfileSearchRequest">

<input message="typens:SellerProfileSearchRequest"/>

<output message="typens:SellerProfileSearchResponse"/>

</operation>

<operation name="SellerSearchRequest">

<input message="typens:SellerSearchRequest"/>

<output message="typens:SellerSearchResponse"/>

</operation>

<operation name="SimilaritySearchRequest">

<input message="typens:SimilaritySearchRequest"/>

<output message="typens:SimilaritySearchResponse"/>

</operation>

<!-- Shopping Cart -->

<operation name="GetShoppingCartRequest">

<input message="typens:GetShoppingCartRequest"/>

<output message="typens:ShoppingCartResponse"/>

</operation>
```

```
<operation name="ClearShoppingCartRequest">
<input message="typens:ClearShoppingCartRequest"/>
<output message="typens:ShoppingCartResponse"/>
</operation>
<operation name="AddShoppingCartItemsRequest">
<input message="typens:AddShoppingCartItemsRequest"/>
<output message="typens:ShoppingCartResponse"/>
</operation>
<operation name="RemoveShoppingCartItemsRequest">
<input message="typens:RemoveShoppingCartItemsRequest"/>
<output message="typens:ShoppingCartResponse"/>
</operation>
<operation name="ModifyShoppingCartItemsRequest">
<input message="typens:ModifyShoppingCartItemsRequest"/>
<output message="typens:ShoppingCartResponse"/>
</operation>
</portType>
<binding name="AmazonSearchBinding" type="typens:AmazonSearchPort">
<soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http"/>
<!-- Binding for Amazon Web APIs - RPC, SOAP over HTTP -->
<operation name="KeywordSearchRequest">
```

```
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="PowerSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="BrowseNodeSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
```

```
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="AsinSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="BlendedSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
```

```
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="UpcSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="AuthorSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
```

```
</operation>
<operation name="ArtistSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="ActorSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="ManufacturerSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
```

```
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="DirectorSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="ExchangeSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
```

```
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="ListManiaSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="WishlistSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
```

```
</output>
</operation>
<operation name="SellerProfileSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="SellerSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
```

```
<operation name="MarketplaceSearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="SimilaritySearchRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<!-- Shopping Cart -->
<operation name="GetShoppingCartRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
```

```
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="ClearShoppingCartRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="AddShoppingCartItemsRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
```

```
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="RemoveShoppingCartItemsRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</output>
</operation>
<operation name="ModifyShoppingCartItemsRequest">
<soap:operation soapAction="http://soap.amazon.com"/>
<input>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
</input>
<output>
<soap:body use="encoded" encodingStyle="http://schemas.xmlsoap.org/soap/encoding
```

```
</output>

</operation>

</binding>

<service name="AmazonSearchService">

<!-- Endpoint for Amazon Web APIs -->

<port name="AmazonSearchPort" binding="typens:AmazonSearchBinding">

<soap:address location="http://soap.amazon.com/onca/soap2"/>

</port>

</service>

<!--Shopping Cart-->

</wsdl:definitions>
```