Project no. TIP5-CT-2006-031415O

INNOTRACK

Integrated Project (IP)

Thematic Priority 6: Sustainable Development, Global Change and Ecosystems

D1.4.5 – Linking of Tools

Due date of deliverable: 28/2/09

Actual submission date: 17/3/09

Start date of project: 1 September 2006 Duration: 36 months

Organisation name of lead contractor for this deliverable: University of Birmingham

Revision Final

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Table of Contents

Glossary

CAD – Computer Aided Design
B2B – Business 2 Business
SOA – Service Oriented Architecture
XML eXtensible Markup Language
RPC- Remote Procedure Call
SOAP – Simple Object Access Protocol
HTTP – Hyper Text Transfer Protocol
API – Application Programming Interface

1. Executive Summary

The objective of this deliverable is to demonstrate that web based technologies can easily enable the linking of different tools supporting the interchange of data between them. The main aim is to illustrate through a basic prototype example, the technology, code and resource required. The deliverable will follow the following main themes:

- The architectural arrangement linking of tools
- The prerequisite models for integration of tools and exchange of data
- The resources required to support the integration and interpret the data exchanged
- The code features required to support the architecture

The linking of tools is an area of focus in a number of industries that have a requirement to exchange product or design data. The actual solutions proposed vary depending on the specific requirement. There have been a number of standards developed to support work undertaken in this area [2], [3], [4]. This deliverable will show how current available technology can be applied in a track related application to provide a service that can integrate data from a number of difference sources.

2. Introduction

The linking of tools within the railway context is equivalent to many exchange mechanisms such as business to business (B2B) transactions that are now commonplace in many industries. The objective of this deliverable is to build on the existing study of requirements and potential for linking of models provided in D1.4.3.

There are numerous tools used in the railway domain for modelling rolling stock and infrastructure. In general these tools are based on stand alone applications that were not intended to be integrated with each other. However, recent work in the area of CAD design integration has shown that the ability to exchange data about the design of a component is useful. The same approach can be applied to railway tools where there is an opportunity to link to our existing tools so that they can automatically exchange data. This solution provides a number of advantages since a number of tools can be chained together reducing the time required to manually pass data between them. There is also a reduced chance of errors being made during manual translation into each of the tools.

The starting point of this deliverable is the consideration of the types of systems that could exchange data and how the architecture to connect them would be arranged. The deliverable D1.4.3 introduces two systems that use and generate model data. One of these is a vehicle modelling package and another is track recording data. The interesting feature of these packages is that the vehicle modelling package can accept a track model in order to evaluate the effect of a track on a vehicles performance. This creates and ideal case study for the demonstration of the linking of tools. The deliverable D1.4.3 introduces the type of data that these systems produce and consume. This deliverable illustrates how these data sets can be exchanged in prototype demonstration.

The following sections describe the architecture for a prototype deployment followed by a demonstration of the prototype system. Some code samples are provided to give enough information about the proposed solution.

3. Architecture

3.1 Background

The Internet and associated standards has created opportunities for systems that were independently deployed to be integrated with other systems. This feature is important for many industries that regularly exchange transaction data. The advantages of this is that transactions are faster since there is less opportunity for error as the data is not manually transferred.

Recent activity in the area of CAD designers has seen developers creating interfaces between systems to support the sharing of designs between different systems. This approach leads to considerable time saving in not needing to manually enter data in the receiving system.

In the railway domain there are similar requirements for tools integration. Deliverable D1.4.3 introduces the requirement for exchange of data between tools and the technology used to support that exchange. Essentially a case study is devised that illustrates the integration of vehicle design data with data from a track measurement system. In this section the architecture for the proposed integration is described.

3.2 Web Services

Web services are based on a Service Oriented Architecture (SOA). An SOA is essentially a collection of loosely coupled functions across the IT infrastructure - internal and external - that communicate with each other to provide a particular service or business process. This solution is appropriate for linking of tools because there are many tools with differing requirements that may need to be linked.

The alternative is to stay with siloed, standalone applications that can't easily talk to each other. Non-SOA applications are designed to carry out fairly specific functions and the code in them is often not easily reusable. To modify and customise them so they can talk to other applications becomes costly and complex.

One real-world example is the London Stock Exchange, which has used an SOA to enable its systems to communicate directly with those of the various financial firms it must work with. Other hypothetical examples might be the bank that has grown by one acquisition after another - for example allowing the systems of its retail banking, insurance and mortgage arms to speak with each other and a common system - that uses the SOA middleware layer to pull information from the various legacy systems. Or the travel agent that uses SOA to allow its customers to book cars from a separate car rental company.

The SOA approach has actually been around for more than a decade but it's only started to come into more widespread use in businesses now because of the standard technologies, protocols and languages associated with web services, especially XML. To put it simply, web services is the connectivity that enables various applications or functions in an enterprise to plug in and talk to each other and provide the 'service' in SOA. By reusing the functionality in applications instead of developing lots of standalone apps it cuts development costs as well as simplifying an enterprise's IT infrastructure while making it much more flexible and agile. In an SOA, services can be quickly adapted to new or changing business processes where previously it might have meant completely redesigning existing applications or buying in expensive new packages.

3.3 Proposed Architecture

The proposed architecture is based on a potentially realistic scenario. A train designer wants to test the dynamic properties of new train design using a third party tool. To achieve this, the designer will also need to use data from a measured section of track geometry. The train design characteristics and track geometry data are held in a format that is proprietary to each system.

To achieve the linking of the tools the user will want to transfer the data from the train and track systems to the dynamics modelling tool. This is achieved through the implementation of web services as described previously.

Figure 1 illustrates the arrangement of component hardware to support the demonstration. Each physical server runs a web server such as Tomcat [1]. The main aim of the architecture is to provide a simple demonstration of how tools can be linked together.

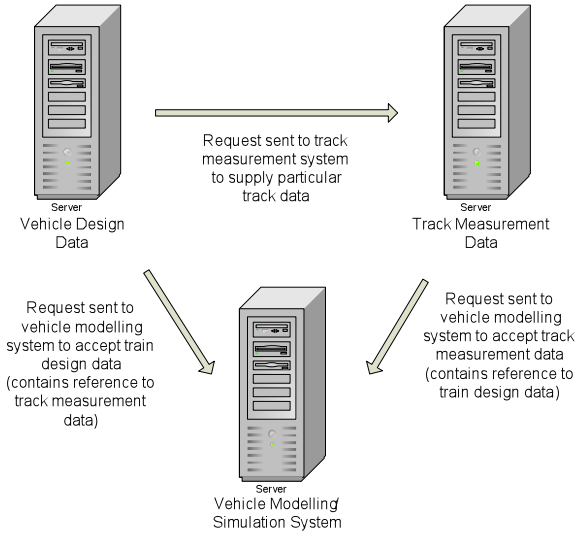

Figure 1 Simplified prototype architecture

Each Tomcat sever can host a number of web services where a service requestor can initiate the request of a function available from a web service as shown in figure 2. The functions are requested using a common method referred to as Remote Procedure Call (RPC), where SOAP is an abbreviation for Simple Object Access Protocol – an alternative to HTTP for XML data transfer.

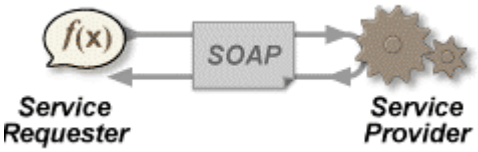

Figure 2 – Architectural Elements involved in XML-RPC

In the prototype deployment, there are three main functions required. The first is a request from the vehicle design package to the vehicle modelling package, the second is from the vehicle design package to the track measurement system and the third is from the track measurement system to the vehicle modelling package. A sequence diagram outlining this arrangement is shown in figure 3.

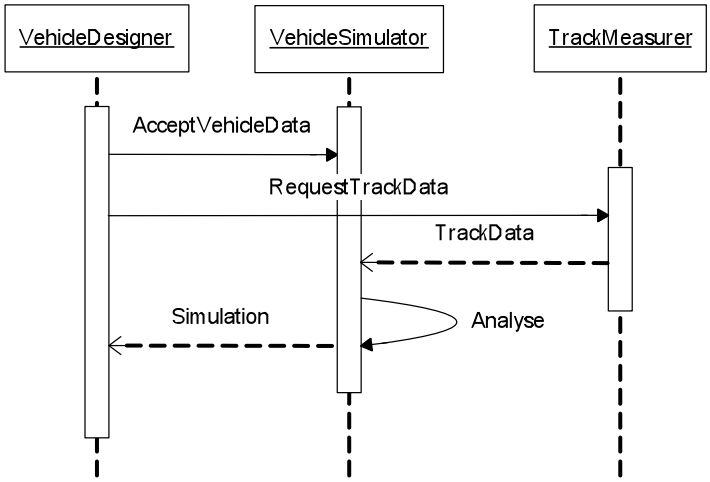

Figure 3 Prototype sequence diagram

3.4 Deployment

Each service call requires the transfer of data from the service requestor to the service provider. This is achieved using XML where an XML schema is used to validate the data that has been sent. Figure 4 shows a more detailed representation of the system architecture supporting the prototype deployment. The XML file in each case is passed as a parameter of the function call.

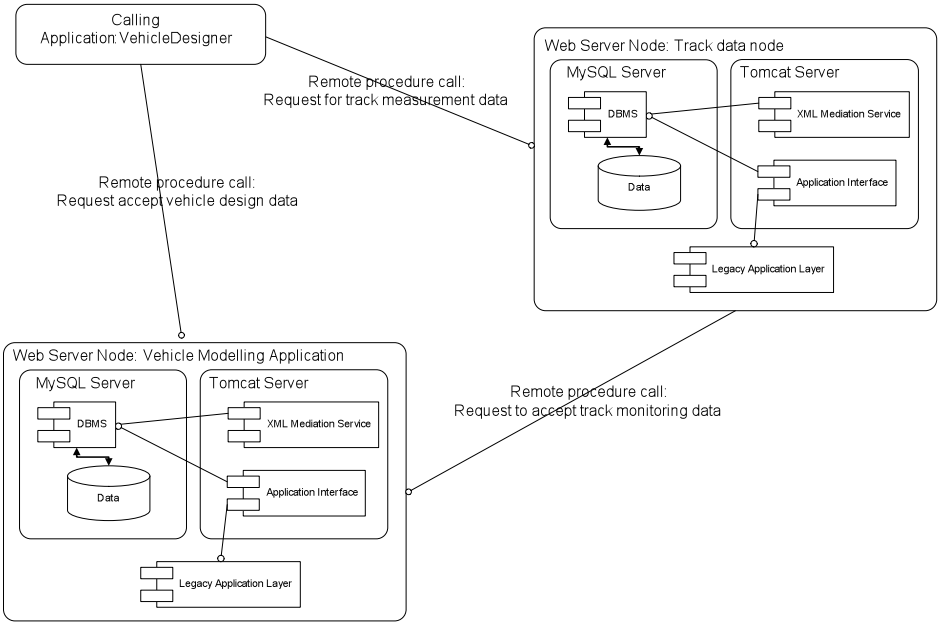

Figure 4 Detailed prototype deployment.

Each node contains a function that can interpret XML data called the XML mediation service. The node also contains a database server. The reason for this is for cases where there are multiple models to be sent such as numerous vehicles forming a train set or a number of tracks forming a track section. The database is also used to translate the data into a format that is appropriate for the receiving application.

To demonstrate the requirements to link tools using XML, we consider a number of different applications that enables the user to build a dynamic model of a rail vehicle. These dynamic models are used to study the response to real measured track geometry or user specified inputs in the form of track displacements and external force inputs. These systems are Vampire (http://www.vampire-dynamics.com/index.html), Gensys (http://www.gensys.se/index.html) and NuCars (http://www.aar.com/nucars/about.asp). In general they perform identical functions using similar data represented in different formats.

In this case study, we consider the requirement to compare the results from all applications of this type. Here, it would be desirable to create one dynamic model of the vehicle and apply it to all three of the tools. To demonstrate this we consider the Vampire vehicle specification in Section 7.

The equivalent of this model in XML is an XML schema that describes the specification and an XML document that provides an instance of a vehicle model.

3.4.1 XML Schema and Data

Vehicle Schema

The example XML model is built on a number of assumptions. First, that each file represents an individual vehicle and second, that each vehicle consists of at least two and not more than two bogies. It can be seen that XML is ideal for representing this type of information for two reasons: first, the restriction can be captured in the schema so that is can be observed by the user and second the restriction can be used to check if the XML data is valid to the schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 3 U (http://www.xmlspy.com) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="VEHICLEMODEL">
        <xs:annotation>
            <xs:documentation>Piecewise linear vehicle model</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="BOGIE" minOccurs="2" maxOccurs="2">
                    <xs:complexType>
                        <xs:choice>
                            <xs:element name="SUSPENSION_CHARACTERISTICS">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="TRAILING_ARM_BUSH" maxOccurs="4">
                                            <xs:complexType>
                                                <xs:sequence>
                                                    <xs:element name="LONGITUDINAL_STIFFNESS" type="xs:decimal">
                                                        <xs:annotation>
                                                            <xs:documentation>Units: kN/m</xs:documentation>
                                                        </xs:annotation>
                                                    </xs:element>
```

The complete XML schema for this model can be found in the Appendix.

Using this schema a vehicle model can be constructed that all of the vehicle modelling tools could potentially accept as input. The vehicle data is created as an XML file that matches the schema. An example of this data file is shown below – note that the data has been generated by an editor so the values are not true data values for the model.

Track Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 3 U (http://www.xmlspy.com) by Richard Lewis (University of Birmingham) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="TRACK_DATA">
        <xs:annotation>
            <xs:documentation>Comment describing your root element</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="OBSERVATION">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="ELR" type="xs:string"/>
                            <xs:element name="TRACK" maxOccurs="unbounded">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="RAILWAY_LOCATION" type="xs:string"/>
                                        <xs:element name="WORLD_LOCATION" type="xs:string"/>
                                        <xs:element name="FAULT">
                                            <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                    <xs:enumeration value="T35R"/>
                                                    <xs:enumeration value="TW3M"/>
                                                    <xs:enumeration value="AL35"/>
                                                    <xs:enumeration value="T35L"/>
                                                    <xs:enumeration value="T35R"/>
                                                    <xs:enumeration value="A35R"/>
                                                    <xs:enumeration value="A35L"/>
                                                    <xs:enumeration value="MT70"/>
                                                    <xs:enumeration value="AL70"/>
                                                    <xs:enumeration value="MT120"/>
                                                    <xs:enumeration value="AL120"/>
                                                </xs:restriction>
                                            </xs:simpleType>
                                        </xs:element>
                                        <xs:element name="MAGNETUDE" type="xs:double"/>
                                        <xs:element name="TIME" type="xs:time"/>
                                    </xs:sequence>
                                    <xs:attribute name="trackCode" type="xs:int" use="required"/>
                                </xs:complexType>
                            </xs:element>
                        </xs:sequence>
                        <xs:attribute name="date" type="xs:date" use="required"/>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>

</xs:schema>
```

3.4.2 Sample Data files

Vehicle XML File

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSPY v5 rel. 3 U (http://www.xmlspy.com)-->
<VEHICLEMODEL xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\MAINDRIVE\Innotrack Files\VehicleModel_1.xsd" model_ID="String">
    <BOGIE bogie_num="1">
        <SUSPENSION_CHARACTERISTICS>
            <TRAILING_ARM_BUSH>
                <LONGITUDINAL_STIFFNESS>30.0772</LONGITUDINAL_STIFFNESS>
                <LATERAL_STIFFNESS>3.267</LATERAL_STIFFNESS>
                <VERTICAL_STIFFNESS>50.0</VERTICAL_STIFFNESS>
                <ROLL_STIFFNESS>0.020</ROLL_STIFFNESS>
                <PITCH_STIFFNESS>0.010</PITCH_STIFFNESS>
                <YAW_STIFFNESS>20.0</YAW_STIFFNESS>
            </TRAILING_ARM_BUSH>
            <PRIMARY_SPRINGS>
                <LONGITUDINAL_SHEAR_STIFFNESS>0.617</LONGITUDINAL_SHEAR_STIFFNESS>
                <LATERAL_SHEAR_STIFFNESS>0.617</LATERAL_SHEAR_STIFFNESS>
                <VERTICAL_STIFFNESS>0.732</VERTICAL_STIFFNESS>
                <BENDING_STIFFNESS>0.00728</BENDING_STIFFNESS>
            </PRIMARY_SPRINGS>
            <SECONDARY_SPRINGS>
                <LONGITUDINAL_SHEAR_STIFFNESS>0.165</LONGITUDINAL_SHEAR_STIFFNESS>
                <LATERAL_SHEAR_STIFFNESS>0.16</LATERAL_SHEAR_STIFFNESS>
                <VERTICAL_STIFFNESS>0.43</VERTICAL_STIFFNESS>
                <BENDING_STIFFNESS>0.01</BENDING_STIFFNESS>
            </SECONDARY_SPRINGS>

    <SECONDARY_ROLL_BAR_STIFFNESS>3.14159265358979323846264338327 95</SECONDARY_ROLL_BAR_STIFFN
ESS>
            <BUMPSTOP_SYMMETRIC_CHARACTERISTIC>
                <MAG>-65</MAG>
                <FORCE>230</FORCE>
            </BUMPSTOP_SYMMETRIC_CHARACTERISTIC>
        </SUSPENSION_CHARACTERISTICS>
    </BOGIE>
    <BOGIE bogie_num="2">
        <SUSPENSION_CHARACTERISTICS>
            <TRAILING_ARM_BUSH>
                <LONGITUDINAL_STIFFNESS>30.0772</LONGITUDINAL_STIFFNESS>
                <LATERAL_STIFFNESS>3.267</LATERAL_STIFFNESS>
                <VERTICAL_STIFFNESS>50.0</VERTICAL_STIFFNESS>
                <ROLL_STIFFNESS>0.020</ROLL_STIFFNESS>
                <PITCH_STIFFNESS>0.010</PITCH_STIFFNESS>
                <YAW_STIFFNESS>20.0</YAW_STIFFNESS>
            </TRAILING_ARM_BUSH>
            <PRIMARY_SPRINGS>
                <LONGITUDINAL_SHEAR_STIFFNESS>0.617</LONGITUDINAL_SHEAR_STIFFNESS>
                <LATERAL_SHEAR_STIFFNESS>0.617</LATERAL_SHEAR_STIFFNESS>
                <VERTICAL_STIFFNESS>0.732</VERTICAL_STIFFNESS>
                <BENDING_STIFFNESS>0.00728</BENDING_STIFFNESS>
            </PRIMARY_SPRINGS>
            <SECONDARY_SPRINGS>
                <LONGITUDINAL_SHEAR_STIFFNESS>0.165</LONGITUDINAL_SHEAR_STIFFNESS>
                <LATERAL_SHEAR_STIFFNESS>0.16</LATERAL_SHEAR_STIFFNESS>
                <VERTICAL_STIFFNESS>0.43</VERTICAL_STIFFNESS>
                <BENDING_STIFFNESS>0.01</BENDING_STIFFNESS>
            </SECONDARY_SPRINGS>
            <SECONDARY_ROLL_BAR_STIFFNESS>0.05</SECONDARY_ROLL_BAR_STIFFNESS>
            <BUMPSTOP_SYMMETRIC_CHARACTERISTIC>
                <MAG>-65</MAG>
                <FORCE>230</FORCE>
```

```
            </BUMPSTOP_SYMMETRIC_CHARACTERISTIC>
        </SUSPENSION_CHARACTERISTICS>
    </BOGIE>
    <VEHICLE_DIMENSIONS>
        <BOGIE_SEMI_PIVOT_SPACING>0.05</BOGIE_SEMI_PIVOT_SPACING>
    </VEHICLE_DIMENSIONS>
    <SUSPENSION_GEOMETRY>
        <TRAILING_ARM_BUSHES>
            <HEIGHT_ABOVE_RAIL_LEVEL>1</HEIGHT_ABOVE_RAIL_LEVEL>
            <LONGITUDINAL_SEMI_SPACING>1</LONGITUDINAL_SEMI_SPACING>
            <LATERAL_SEMI_SPACING>1</LATERAL_SEMI_SPACING>
        </TRAILING_ARM_BUSHES>
    </SUSPENSION_GEOMETRY>
</VEHICLEMODEL>
```

Track XML File

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSPY v5 rel. 3 U (http://www.xmlspy.com)-->
<TRACK_DATA xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\MAINDRIVE\Innotrack Files\Docs\D1_4_5\TrackData_1.xsd">
    <OBSERVATION date="2006-02-11">
        <ELR>ECM1</ELR>
        <TRACK trackCode="3200">
            <RAILWAY_LOCATION>155m1402y</RAILWAY_LOCATION>
            <WORLD_LOCATION>53.51962:-1.14001</WORLD_LOCATION>
            <FAULT>T35R</FAULT>
            <MAGNETUDE>-21.15</MAGNETUDE>
            <TIME>13:34:16-00:00</TIME>
        </TRACK>
    </OBSERVATION>
</TRACK_DATA>
```

4. Applications

4.1 General Information

A **Web service** is defined by the W3C as "a software system designed to support interoperable machine-to-machine interaction over a network. Web services are frequently just Web APIs that can be accessed over a network, such as the Internet, and executed on a remote system hosting the requested services.

The W3C Web service definition encompasses many different systems, but in common usage the term refers to clients and servers that communicate over the HTTP protocol used on the Web. Such services tend to fall into one of two camps: Big Web Services and RESTful Web Services.

"Big Web Services" use XML messages that follow the SOAP standard and have been popular with traditional enterprise. In such systems, there is often a machine-readable description of the operations offered by the service written in the Web Services Description Language (WSDL). The latter is not a requirement of a SOAP *endpoint*, but it is a prerequisite for automated client-side code generation in many Java and .NET SOAP frameworks (frameworks such as Spring, Apache Axis2 and Apache CXF being notable exceptions). Some industry organizations, such as the WS-I, mandate both SOAP and WSDL in their definition of a Web service.

4.2 Validating XML in Java

In order to process an XML document, the receiving application must first validate that document against its schema. The following example demonstrates validating an XML document with the Validation API (for readability, some exception handling is not shown):

```
    // parse an XML document into a DOM tree
    DocumentBuilder parser =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
    Document document = parser.parse(new File("instance.xml"));

    // create a SchemaFactory capable of understanding WXS schemas
    SchemaFactory factory =
SchemaFactory.newInstance(XMLConstants.W3C_XML_SCHEMA_NS_URI);

    // load a WXS schema, represented by a Schema instance
    Source schemaFile = new StreamSource(new File("mySchema.xsd"));
    Schema schema = factory.newSchema(schemaFile);

    // create a Validator instance, which can be used to validate an instance
document
    Validator validator = schema.newValidator();

    // validate the DOM tree
    try {
        validator.validate(new DOMSource(document));
    } catch (SAXException e) {
        // instance document is invalid!
    }
```

The JAXP parsing API has been integrated with the Validation API. Applications may create a Schema with the validation API and associate it with a DocumentBuilderFactory or a SAXParserFactory instance by using the DocumentBuilderFactory.setSchema(Schema) and SAXParserFactory.setSchema(Schema) methods.

4.3 Reading XML data concepts in Java

Using SAX

This program SAXParserExample.java parses a XML document and prints it on the console. Sax parsing is event based modelling. When a SAX parser parses an XML document every time it encounters a tag it calls the corresponding tag handler methods.

When it encounters a Start Tag it calls this method
public void startElement(String uri,..

When it encounters a End Tag it calls this method
public void endElement(String uri,...

This program also parses the xml file, creates a list of employees and prints it to the console. The steps involved are:

- Create a SAX parser and parse the XML
- In the event handler create the employee object
- Print out the data

Basically the class extends DefaultHandler to listen for call back events. And we register this handler with the SAX parser to notify us of call back events. We are only interested in *start event, end event and character event*.
In the example below, *start event*, if the element is employee we create a new instant of employee object and if the element is Name/Id/Age we initialize the character buffer to get the text value.
In *end event* if the node is employee then we know we are at the end of the employee node and we add the Employee object to the list.If it is any other node like Name/Id/Age we call the corresponding methods like setName/SetId/setAge on the Employee object.
In *character event* we store the data in a temp string variable.

a) Create a SAX Parser and parse the XML

```
        private void parseDocument() {

                //get a factory
                SAXParserFactory spf = SAXParserFactory.newInstance();
                try {

                        //get a new instance of parser
                        SAXParser sp = spf.newSAXParser();

                        //parse the file and also register this class for call
backs
                        sp.parse("employees.xml", this);

                }catch(SAXException se) {
                        se.printStackTrace();
                }catch(ParserConfigurationException pce) {
                        pce.printStackTrace();
                }catch (IOException ie) {
                        ie.printStackTrace();
                }
        }
```

b) In the event handlers create the Employee object and call the corresponding setter methods.

```
        //Event Handlers
        public void startElement(String uri, String localName, String qName,
                Attributes attributes) throws SAXException {
                //reset
                tempVal = "";
                if(qName.equalsIgnoreCase("Employee")) {
                        //create a new instance of employee
                        tempEmp = new Employee();
                        tempEmp.setType(attributes.getValue("type"));
                }
        }


        public void characters(char[] ch, int start, int length) throws
SAXException {
                tempVal = new String(ch,start,length);
        }

        public void endElement(String uri, String localName,
                String qName) throws SAXException {

                if(qName.equalsIgnoreCase("Employee")) {
                        //add it to the list
                        myEmpls.add(tempEmp);

                }else if (qName.equalsIgnoreCase("Name")) {
                        tempEmp.setName(tempVal);
                }else if (qName.equalsIgnoreCase("Id")) {
                        tempEmp.setId(Integer.parseInt(tempVal));
                }else if (qName.equalsIgnoreCase("Age")) {
                        tempEmp.setAge(Integer.parseInt(tempVal));
                }

        }
```

c) Iterating and printing.

```
        private void printData(){

                System.out.println("No of Employees '" + myEmpls.size() + "'.");

                Iterator it = myEmpls.iterator();
                while(it.hasNext()) {
                        System.out.println(it.next().toString());
                }
        }
```

In order to process the data transferred as shown in figure 4 it is necessary to deploy two services from the Vehicle modelling node. These are described in the next section.

4.4 Web Services

Creating a Web service using IDE such as NetBeans is relatively simple process. NetBeans creates the imports to the web service libraries and generates the simple web service class with the name that the user provides.

```
import javax.jws.WebService; // Program uses the annotation @WebService
import javax.jws.WebMethod; //Program uses the annotation @WebMethod
import javax.jws.WebParam; //Program uses the annotation @WebParam

@WebService( // annotates the class as a web service
name = "VehicleDesign", // sets class name
serviceName="VehicleDesignService") // sets the service name
public class VehicleDesign{
public double acceptVehicleDesignData(fileName inputXML){
…
}
public double acceptTrackMonitoringData(fileName inputXML){
…
}
```

4.4.1 Web Methods

```
public double acceptVehicleDesignData(fileName inputXML){

  public void loopAround(Node node, int level) throws SQLException
    {
          NodeList nl=node.getChildNodes();
          int cnt=nl.getLength();
          NamedNodeMap attributes;
          for (int i=0;i<cnt;i++){
                    if (nl.item(i).getNodeName()=="VEHICLEMODEL"){
                    attributes=nl.item(i).getAttributes();
                     for (int j=0; j<attributes.getLength();j++){
                     if (attributes.item(j).getNodeName()=="model_ID"){
                              items[0]=attributes.item(j).getNodeValue();
                     }
                    }
                    }else if (nl.item(i).getNodeName()=="BOGIE"){
                    attributes=nl.item(i).getAttributes();
                    for (int j=0; j<attributes.getLength();j++){
                              if (attributes.item(j).getNodeName()=="bogie_num"){
                              items[j]=attributes.item(j).getNodeValue();
                       }
                    }
```

```
                    } else if (nl.item(i).getNodeName()=="VEHICLE_DIMENSIONS"){
                     attributes=nl.item(i).getAttributes();
                    } else if (nl.item(i).getNodeName()=="SUSPENSION_GEOMETRY"){
                     attributes=nl.item(i).getAttributes();
                    }
                    } else if (nl.item(i).getNodeName()=="SUSPENSION_CHARACTERISTICS"){
                     attributes=nl.item(i).getAttributes();
                    }
                    …
                    …
                }
             }
            //Put second loop here
           loopAround(nl.item(i),level+1);
           if (level==2) {
              if (items[0].toString()=="nothing")
              {
                 // do nothing
              } else
                    // add the data to the database
              {
       }
}

public double acceptTrackMonitoringData(fileName inputXML){
  public void loopAround(Node node, int level) throws SQLException
    {
           NodeList nl=node.getChildNodes();
           int cnt=nl.getLength();
           NamedNodeMap attributes;
           for (int i=0;i<cnt;i++){
                    if (nl.item(i).getNodeName()=="TRACK_DATA"){
                    attributes=nl.item(i).getAttributes();

                    }else if (nl.item(i).getNodeName()=="OBSERVATION"){
                    attributes=nl.item(i).getAttributes();
                    for (int j=0; j<attributes.getLength();j++){
                              if (attributes.item(j).getNodeName()=="date"){
                             items[j]=attributes.item(j).getNodeValue();
                       }
                    }
                    } else if (nl.item(i).getNodeName()=="ELR"){
                     attributes=nl.item(i).getAttributes();
                    } else if (nl.item(i).getNodeName()=="TRACK"){
                     attributes=nl.item(i).getAttributes();
                    for (int j=0; j<attributes.getLength();j++){
```

```
                        if (attributes.item(j).getNodeName()=="trackCode"){
                        items[j]=attributes.item(j).getNodeValue();
                    }
                }
            }
            } else if (nl.item(i).getNodeName()=="RAILWAY_LOCATION"){
                attributes=nl.item(i).getAttributes();
            }
                …
                …
            }
        }
        //Put second loop here
        loopAround(nl.item(i),level+1);
        if (level==2) {
            if (items[0].toString()=="nothing")
            {
                // do nothing
            } else
                // add the data to the database
            {
    }
}
}
```

5. Conclusions

This deliverable has provided a background to data integration and the linking of tools. It represents a case study for the linking of tools associated with vehicle design, track measurement and vehicle dynamics profiling. A background to the available technology has been provided which includes a prototype architecture and code samples.

The deliverable provide a framework from which further work can be undertaken. There is potential to extend the work presented here in a number of ways:

- This document has discussed an example where an XML model of a vehicle has been generated in order to provide a common input to a number of modelling tools. However, there would also be advantages in bringing the output from the same tools into a similar, common standard. This could be achieved either as an export option from the tools themselves, or, as would certainly be a more appropriate solution in the short term, by a web service that would parse the software-specific output formats from the tools and translate it into XML.
- Deciding upon the level of abstraction appropriate to a particular model is always a difficult task, the extensible nature of XML data formats mean that additional tags can be defined in situations where additional information needs to be stored. This process does not interfere with the ability of other software to read the file as it can in a traditional file format, since the XML parser will simply pass over tags it does not recognise. The creation of extended sets of tags for particular modelling tasks, parameters required by individual software packages, or non-SI measurement units such as the `chains' used on the UK rail network, would be an important extension of the example discussed here.
- Software packages for railway vehicle dynamics also use postprocessors for the evaluation of: track-shift forces, risk for flange climbing, ride indexes, pantograph sway, vehicle overturning, RCF, wheel/rail-wear, etc. In order to allow the interchange of these postprocessors it would be beneficial to additionally use an XML schema at this level.

It is important to remember that the adoption of an XML standard for data exchange is not a zero-cost solution. Software currently in use would either need to be updated to support the new standard, or services to translate data between the formats need to be provided. A substantial effort is also required to make sure that the standard supports all the features required by the parties involved in the most economical way possible. Much of this work falls to the companies developing the software, and certainly in the early stages of adoption the costs may outweigh the benefits. Ultimately however, the greater flexibility XML data formats allow to customers mean that tools that support them are much more attractive than those which don't, and market forces tend to drive the adoption of the standard across a sector.

6. Bibliography

[1] Tomcat - http://tomcat.apache.org/

[2] ISO 10303 - **Industrial automation systems and integration -- Product data representation and exchange -- Part 218: Application protocol: Ship structures**

[3] ISO 8000 **Data quality -- Part 110: Master data: Exchange of characteristic data: Syntax, semantic encoding, and conformance to data specification**

[4] ISO 22745 **Industrial automation systems and integration -- Open technical dictionaries and their application to master data -- Part 1: Overview and fundamental principles**

7. Annexes

7.1 Appendix - Vehicle Specification XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v5 rel. 3 U (http://www.xmlspy.com) by Richard Lewis (University of Birmingham) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="VEHICLEMODEL">
        <xs:annotation>
            <xs:documentation>Piecewise linear vehicle model</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="BOGIE" minOccurs="2" maxOccurs="2">
                    <xs:complexType>
                        <xs:choice>
                            <xs:element name="SUSPENSION_CHARACTERISTICS">
                                <xs:complexType>
                                    <xs:sequence>
                                        <xs:element name="TRAILING_ARM_BUSH" maxOccurs="4">
                                            <xs:complexType>
                                                <xs:sequence>
                                                    <xs:element name="LONGITUDINAL_STIFFNESS" type="xs:decimal">
                                                        <xs:annotation>
                                                            <xs:documentation>Units: kN/m</xs:documentation>
                                                        </xs:annotation>
                                                    </xs:element>
                                                    <xs:element name="LATERAL_STIFFNESS" type="xs:decimal">
                                                        <xs:annotation>
                                                            <xs:documentation>Units: kN/m</xs:documentation>
                                                        </xs:annotation>
                                                    </xs:element>
                                                    <xs:element name="VERTICAL_STIFFNESS" type="xs:decimal">
                                                        <xs:annotation>
                                                            <xs:documentation>Units: kN/m</xs:documentation>
                                                        </xs:annotation>
                                                    </xs:element>
                                                    <xs:element name="ROLL_STIFFNESS" type="xs:decimal">
                                                        <xs:annotation>
                                                            <xs:documentation>Units: kNm/rad</xs:documentation>
                                                        </xs:annotation>
                                                    </xs:element>
                                                    <xs:element name="PITCH_STIFFNESS" type="xs:decimal">
                                                        <xs:annotation>
                                                            <xs:documentation>Units: kNm/rad</xs:documentation>
                                                        </xs:annotation>
                                                    </xs:element>
                                                    <xs:element name="YAW_STIFFNESS" type="xs:decimal">
                                                        <xs:annotation>
                                                            <xs:documentation>Units: kNm/rad</xs:documentation>
                                                        </xs:annotation>
                                                    </xs:element>
                                                </xs:sequence>
                                            </xs:complexType>
                                        </xs:element>
                                        <xs:element name="PRIMARY_SPRINGS" type="SPRING_SUSP_TYPE"
maxOccurs="4"/>
                                        <xs:element name="SECONDARY_SPRINGS" type="SPRING_SUSP_TYPE"
maxOccurs="2"/>
                                        <xs:element name="SECONDARY_ROLL_BAR_STIFFNESS" type="xs:decimal">
                                            <xs:annotation>
                                                <xs:documentation>Units: kNm/rad</xs:documentation>
                                            </xs:annotation>
                                        </xs:element>
```

```
                                <xs:element name="BUMPSTOP_SYMMETRIC_CHARACTERISTIC" minOccurs="10"
maxOccurs="10">
                                    <xs:complexType>
                                        <xs:sequence>
                                            <xs:element name="MAG" type="xs:int">
                                                <xs:annotation>
                                                    <xs:documentation>Units: mm</xs:documentation>
                                                </xs:annotation>
                                            </xs:element>
                                            <xs:element name="FORCE" type="xs:decimal">
                                                <xs:annotation>
                                                    <xs:documentation>Units: kN</xs:documentation>
                                                </xs:annotation>
                                            </xs:element>
                                        </xs:sequence>
                                    </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="MASS">
                        <xs:complexType>
                            <xs:choice>
                                <xs:element name="BOGIE_MASS">
                                    <xs:complexType>
                                        <xs:complexContent>
                                            <xs:extension base="MASS_TYPE"/>
                                        </xs:complexContent>
                                    </xs:complexType>
                                </xs:element>
                                <xs:element name="TRAILING_ARM" type="MASS_TYPE" maxOccurs="4"/>
                                <xs:element name="WHEELSET" type="MASS_TYPE"/>
                                <xs:element name="VEHICLE_BODY" type="MASS_TYPE"/>
                            </xs:choice>
                        </xs:complexType>
                    </xs:element>
                    <xs:element name="DAMPER_CHARACTERISTICS">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name="PRIMARY_VERTICAL_DAMPERS" maxOccurs="4"/>
                                <xs:element name="SECONDARY_LATERAL_DAMPERS" maxOccurs="2"/>
                                <xs:element name="SECONDARY_VERTICAL_DAMPERS" maxOccurs="2"/>
                                <xs:element name="SECONDARY_YAW_DAMPERS" maxOccurs="2"/>
                            </xs:sequence>
                        </xs:complexType>
                    </xs:element>
                </xs:choice>
                <xs:attribute name="bogie_num" type="xs:int" use="required"/>
            </xs:complexType>
        </xs:element>
        <xs:element name="VEHICLE_DIMENSIONS" type="SIZE_TYPE"/>
        <xs:element name="SUSPENSION_GEOMETRY" type="POSITION_TYPE"/>
    </xs:sequence>
    <xs:attribute name="model_ID" type="xs:string" use="required"/>
</xs:complexType>
</xs:element>
<xs:complexType name="MASS_TYPE">
    <xs:annotation>
        <xs:documentation>Mass and inertia of components</xs:documentation>
    </xs:annotation>
    <xs:choice>
        <xs:element name="INERTIA">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="MASS" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: kg</xs:documentation>
                        </xs:annotation>
                    </xs:element>
```

```
                    <xs:element name="ROLL_INERTIA" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: kgm^2</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="PITCH_INERTIA" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: kgm^2</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="YAW_INERTIA" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: kgm^2</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="SIZE" type="SIZE_TYPE" minOccurs="0"/>
        <xs:element name="POSITION" type="POSITION_TYPE" maxOccurs="unbounded"/>
    </xs:choice>
</xs:complexType>
<xs:complexType name="POSITION_TYPE">
    <xs:annotation>
        <xs:documentation>Postion of vehicle components</xs:documentation>
    </xs:annotation>
    <xs:choice>
        <xs:element name="TRAILING_ARM_BUSHES">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="LONGITUDINAL_SEMI_SPACING" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="LATERAL_SEMI_SPACING" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="PRIMARY_SPRINGS">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="LONGITUDINAL_SEMI_SPACING" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="LATERAL_SEMI_SPACING" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_TOP" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BOTTOM" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
```

```
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SECONDARY_SPRINGS">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="LATERAL_SEMI_SPACINGS" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_TOP" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BOTTOM" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="PRIMARY_VERTICAL_DAMPERS">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="LONGITUDINAL_SEMI_SPACING" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="LATERAL_SEMI_SPACING" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_TOP" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BOTTOM" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>
<xs:element name="SECONDARY_LATERAL_DAMPERS">
    <xs:complexType>
        <xs:sequence>
            <xs:element name="LATERAL_SEMI_SPACING_OF_BOLSTER_END" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="LATERAL_SEMI_SPACING_OF_BOGIE_END" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BOLSTER_END" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
```

```
                    </xs:element>
                    <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BOGIE" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="SECONDARY_VERTICAL_DAMPERS">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="LATERAL_SEMI_SPACING_OF_TOP" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="LATERAL_SEMI_SPACING_OF_BOTTOM" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_TOP" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BOTTOM" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="SECONDARY_YAW_DAMPERS">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="LONGITUDINAL_OFFSET_BODY_END_FROM_BOGIE_CTR" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="LONGITUDINAL_OFFSET_BOGIE_END_FROM_BOGIE_CTR"
type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="LATERAL_SEMI_SPACING" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BODY_END" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BOGIE_END" type="xs:decimal">
                        <xs:annotation>
                            <xs:documentation>Units: mm</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="SECONDARY_LATERAL_BUMP_STOP">
            <xs:complexType>
```

```
                    <xs:sequence>
                        <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL" type="xs:decimal">
                            <xs:annotation>
                                <xs:documentation>Units: mm</xs:documentation>
                            </xs:annotation>
                        </xs:element>
                    </xs:sequence>
                </xs:complexType>
            </xs:element>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="SIZE_TYPE">
        <xs:annotation>
            <xs:documentation>Vehicle dimensions</xs:documentation>
        </xs:annotation>
        <xs:choice>
            <xs:element name="BOGIE_SEMI_PIVOT_SPACING" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="BOGIE_SEMI_WHEELBASE" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="WHEEL_RADIUS" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units:mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_TRAINING_ARM" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="LONGITUDINAL_OFFSET__OF_TRAILING_ARM_COFG" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: mm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BOGIE_COFG"/>
            <xs:element name="HEIGHT_ABOVE_RAIL_LEVEL_OF_BODY_COFG"/>
            <xs:element name="LATERAL_OFFSET_OF_BODY_PTV_TO_RIGHT"/>
            <xs:element name="LOGITUDINAL_OFFSET_BODY_PTV_FORWARD"/>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="SPRING_SUSP_TYPE">
        <xs:annotation>
            <xs:documentation>Common spring suspension components</xs:documentation>
        </xs:annotation>
        <xs:sequence>
            <xs:element name="LONGITUDINAL_SHEAR_STIFFNESS" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: kNm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="LATERAL_SHEAR_STIFFNESS" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: kNm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="VERTICAL_STIFFNESS" type="xs:decimal">
                <xs:annotation>
                    <xs:documentation>Units: kNm</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="BENDING_STIFFNESS" type="xs:decimal">
                <xs:annotation>
```

```
                <xs:documentation>Units: kNm/rad</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```