CAMP LLC

Vehicle-to-Infrastructure 2 (V2I-2) Consortium

Ford GM HYUNDAI MOTOR GROUP TOYOTA

Vehicle-to-Infrastructure Program

V2I Safety Applications

Event Driven Configurable Messaging (EDCM) XML Schema

Submitted to the United States Department of Transportation
Federal Highway Administration (FHWA)

August 28, 2020

In Response to Cooperative Agreement Number
DTFH6114H00002

Executive Summary

This document describes messaging schema developed for the Event Driven Configurable Messaging (EDCM) Project. The EDCM enables a Transportation Management Center (TMC) to request information from Connected Vehicles (CV) equipped with EDCM capabilities in specified areas regarding current conditions at varying rates and time of day. EDCM-equipped CVs then provide vehicle dynamics and status data in response when, where, and as often as requested by TMC using a flexible messaging schema.

The Query Message (QM) sent by the TMC and the Response Message (RM) from the CV is based on a well-defined data dictionary, known to both the connected vehicle and the infrastructure system. The EDCM flexible messaging schema is based on a widely adapted industry standard eXtensible Markup Language (XML) which is used to formulate the QM and RM.

The EDCM Project was conducted by the Crash Avoidance Metrics Partners LLC (CAMP) Vehicle-to-Infrastructure 2 (V2I-2) Consortium, consisting of Ford Motor Company, General Motors LLC, Hyundai Motor Group and Toyota, in cooperation with the Virginia Tech Transportation Institute. The Project was sponsored by the Federal Highway Administration (FHWA) through Cooperative Agreement DTFH6114H00002.

Table of Contents

List of Tables

List of Acronyms

Acronym	Definition
CAMP	Crash Avoidance Metrics Partners LLC
CV	Connected Vehicle
EDCM	Event-Driven Configurable Messaging
FHWA	Federal Highway Administration
QM	Query Message
RM	Response Message
TMC	Transportation Management Center
V2I-2	Vehicle-to-Infrastructure 2 Consortium
XML	eXtensible Markup Language

1 Introduction

This document provides the data dictionary for exchanging messages between the Connected Vehicle (CV) and the Transportation Management Center (TMC) developed as part of the Event Driven Configurable Messaging (EDCM) Project. The EDCM Project was conducted by the Crash Avoidance Metrics Partners LLC (CAMP) Vehicle-to-Infrastructure 2 (V2I-2) Consortium, consisting of Ford Motor Company, General Motors LLC, Hyundai Motor Group and Toyota, in cooperation with the Virginia Tech Transportation Institute. The Project was sponsored by the Federal Highway Administration (FHWA) through Cooperative Agreement DTFH6114H00002.

The EDCM flexible messaging schema is based on a widely adapted industry standard eXtensible Markup Language (XML) which is used to formulate the Query Message (QM) from TMC and Response Message (RM) from CV.

An element in a schema is defined by its name and content model. The content model of an element is defined by its type. Appendix A lists the instance elements in QM and RM that can have only values that fit the types defined in the schema. A type can be simple or complex. A simple type cannot contain elements or attributes in its value. A complex type can associate attributes with an element. Appendix B lists the attributes for used in complex elements.

Appendix A – XML Schema Elements for QM / RM

Table 1: List Vehicle Parameters for QM and RM

XML Tag	Description	Data Type	Value Range
vehType	Vehicle type	Integer	[0..9]
speedMps	Vehicle speed	Decimal (Meter per sec)	[0.0 .. 99.0]
speedChangeMps	Vehicle speed change	Decimal (Meter per sec)	[-99.0 .. +99.0]
speedChangePct	Vehicle speed change percent	Decimal	[0.0 .. 100.0]
headingDeg	Vehicle heading angle degrees	Integer	[0 .. 359]
toleranceDeg	Heading tolerance degrees	Integer	[0 .. 359]
steeringWheelAngle	Steering wheel angle	Decimal – deg	[-30.0 .. +30.0]
longAccel	Longitudinal acceleration	Decimal – m/s^2	[-9.9 .. +9.9]
latAcccel	Lateral acceleration	Decimal – m/s^2	[-9.9 .. +9.9]
vertAccel	Vertical acceleration	Decimal – m/s^2	[-1.0 .. +9.9]
yawRate	Yaw rate	Integer	[-99 .. +99]
brakeApplied	Vehicle brake applied	Boolean	[true | false]
traction	Vehicle traction status	Boolean	[true | false]
abs	Vehicle ABS status	Boolean	[true | false]
scs	Stability control system status	Boolean	[true | false]
brakeBoost	Vehicle brake boost status	Boolean	[true | false]
auxBrake	Vehicle aux brake system	Boolean	[true | false]
panicBrake	Vehicle panic brake status	Boolean	[true | false]
wiperPos	Wiper position status	Integer	[1 | 2 | 3]
normalBeam	Exterior light – normal beam status	Boolean	[true | false]
highBeam	Exterior light – high beam status	Boolean	[true | false]
fogLight	Exterior light – fog light status	Boolean	[true | false]
hazardLight	Exterior light – hazard light status	Boolean	[true | false]
extAirTempC	Exterior air temperature	Decimal in deg C	-40.0 .. +100.0
dataCond	Conditional data	Logical operators	[LT | GT |LE | GE | EQ | NE | true | false | 1 | 0]
timeDur	Time interval	Built-in duration	[PnYnMnDTnHnMnS]*
schemaVer	XML Schema version	Decimal	[1.0 .. 20.0]

* - The time interval is specified in the following form "PnYnMnDTnHnMnS" where:

- P indicates the period (required)

- nY indicates the number of years
- nM indicates the number of months
- nD indicates the number of days
- T indicates the start of a time section (required if you are going to specify hours, minutes, or seconds)
- nH indicates the number of hours
- nM indicates the number of minutes
- nS indicates the number of seconds

Following section describes the XML schema elements implemented in this project. Associated attributes used with the schema elements are in Appendix B.

```
<!-- ============================ I am Here... ================================

   <iamHere> - To be used by CV when first connection is established with TMC.

   When TMC needs periodic update from CV, it can request such using query message

Example:
   <iamHere>
      <myVitals msgType="iamhere" vehType="1" vehID="EDCM-1" speedMps="32.5"
      latDeg="42.232" longDeg="-83.234" elevMet="245" headingDeg="145"/>
   </iamHere>

-->

<xs:element name="iamHere">
   <xs:complexType>
      <xs:sequence>
         <xs:element name="myVitals" maxOccurs="1" minOccurs="1">
            <xs:complexType>
               <xs:simpleContent>
                  <xs:extension base="xs:string">
                     <xs:attribute ref="msgType"      use="required"/>
                     <xs:attribute ref="msgDateTime"  use="required"/> <!-- Timestamp -->
                     <xs:attribute ref="vehType"      use="required"/>
                     <xs:attribute ref="vehID"        use="optional"/>
                     <xs:attribute ref="speedMps"     use="required"/>
                     <xs:attribute ref="latDeg"       use="required"/>
                     <xs:attribute ref="longDeg"      use="required"/>
                     <xs:attribute ref="elevMet"      use="required"/>
                     <xs:attribute ref="headingDeg"   use="required"/>
                  </xs:extension>
               </xs:simpleContent>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
</xs:element>


<!-- ======================  Start of Query Message Section  ===========================

   <qmFrame> - Root for Query Message Frame

Example:
   <qmFrame>
      <eventMsg followed by attributes.../>      Required
      <dataRequest>    ...    </dataRequest>      Required
      <qmDur    followed by attributes.../>       Optional
      <qmAction followed by attributes.../>       Optional
      <gfRegion>       ...    </gfRegion>         optional
      <trigCond>       ...    </trigCond>         Optional
      <dataAvg>        ...    </dataAvg>          Optional
   </qmFrame>

-->

<xs:element name="qmFrame">
   <xs:complexType>
      <xs:sequence     minOccurs="0"    maxOccurs="1">
         <xs:element  ref="eventMsg"   minOccurs="1"    maxOccurs="1"/>
         <xs:element  ref="dataRequest" minOccurs="1"  maxOccurs="1"/>
         <xs:element  ref="gfRegionEntryExitStatus"     minOccurs="0" maxOccurs="1"/>
         <xs:element  ref="qmDur"      minOccurs="0"    maxOccurs="1"/>
         <xs:element  ref="qmAction"   minOccurs="0"    maxOccurs="1"/>
         <xs:element  ref="gfRegion"   minOccurs="0"    maxOccurs="1"/>
         <xs:element  ref="qmTrigger"  minOccurs="0"    maxOccurs="10"/>
      </xs:sequence>
   </xs:complexType>
</xs:element>


<!--
   <eventMsg> - Event for which this query is generated

Example:
   <eventMsg msgType="query" msgPriority="3" eventID="23" msgDateTime="2010-10-23T05:00:00"
      eventInfo="Construction Zone" cCode="30" scCode="2" msgCount="15"
      rmCommType="cellAndDSRC" schemaVer="1.0"/>
  -->

<xs:element name="eventMsg">
   <xs:complexType>
      <xs:simpleContent>
         <xs:extension    base="xs:string">
            <!-- <xs:attribute ref="msgID"    use="required"   ...Removed...  -->
            <xs:attribute ref="eventID"      use="required"/>
            <xs:attribute ref="msgDateTime"  use="required"/> <!-- Time of measurement -->
            <xs:attribute ref="vehType"      use="optional"/>
            <xs:attribute ref="eventInfo"    use="optional"/>
            <xs:attribute ref="rmCommType"   use="required"/>
            <xs:attribute ref="msgType"      use="required"/>
            <xs:attribute ref="msgPriority"  use="optional"/>
            <xs:attribute ref="vehResponsePct"   use="optional"/> <!-- Request for % of
                                             vehicle response the query -->
            <xs:attribute ref="vehID"        use="optional"/>
            <xs:attribute ref="cCode"        use="optional"/>
            <xs:attribute ref="scCode"       use="optional"/>
            <xs:attribute ref="msgCount"     use="optional"/>
            <xs:attribute ref="schemaVer"    use="required"/>
            </xs:extension>
         </xs:simpleContent>
      </xs:complexType>
   </xs:element>




<!--
```

```
    <gfRegionEntryExitStatus> - Geo Fence Region Entry Exit Status

    When CV enters/exits geo fenced area defined in the active query,
    the CV responds its status for each active QM in progress.

    See example above for <qmFrame> for TMC to initiate a query and for
    <rmFrame> for CV to respond.

    This query has a unique eventID = "999" and is active all the time
    by default, unless terminated by TMC using <qmAction> in a QM.
-->

<xs:element name="gfRegionEntryExitStatus">
    <xs:complexType>
        <xs:attribute ref="eventID"   use="required"/>
        <xs:attribute ref="gfStatus"  use="required"/>
    </xs:complexType>
</xs:element>


<!--
    Following modified on 3/18-23/2020 to address how to provide query data
    to TMC

    <dataRequest> - For requesting vehicle data

Example:   <provide> must come before <provideAvg> if present
    <dataRequest>
        <provide dataName="speedMps" intervalDistMet="160"/> Report every 1/10th mile
        <provide dataName="speedMps" intervalDistMet="160" timeDur="PT1H30M"/>   Report every
                                                      1/10th mile for 1.5 hrs
        <provide dataName="vehBrakeStatus"/>
        <provide dataName="vehAccelStatus"/>
        <provide dataName="vehPos"/>
        <provide dataName="wiperPos"  intervalTime="00:10:00"/>  After trigger, every10 min
        <provideAvg dataAvgName="speedChangeMps" preTrigSamples="10"
                            intervalTime="00:00:01"/> Avg of pre-trig samples
    </dataRequest>

Example: trigger when speed change is 60% or more in 20s or less

    <qmTrigger>
        <when speedChangePct="60.0" dataCond="GE"/>
        <when timeDur="PT20s" dataCond="LE"/>
    </qmTrigger>

In above example, CV to respond 12 times for the same "eventID" with different information

-->

<xs:element name="dataRequest">
    <xs:complexType>
        <xs:sequence minOccurs="0"       maxOccurs="1">

        <!--   Provide values of specific data in QM... -->

            <xs:element name="provide" minOccurs="0" maxOccurs="10">
                <xs:complexType>
                    <xs:simpleContent>
                        <xs:extension    base="xs:string">
                            <xs:attribute ref="dataName"        use="required"/>
                            <xs:attribute ref="timeDur"         use="required"/>
                            <xs:attribute ref="intervalTime"    use="optional"/>
                            <xs:attribute ref="intervalDistMet"  use="optional"/>
                        </xs:extension>
                    </xs:simpleContent>
                </xs:complexType>
            </xs:element>

            <!--   Provide average value of specific data in QM... -->

            <xs:element name="provideAvg" minOccurs="0" maxOccurs="4">
                <xs:complexType>
                    <xs:simpleContent>
                        <xs:extension    base="xs:string">
                            <xs:attribute ref="dataAvgName"      use="required"/>
                            <xs:attribute ref="preTrigSamples"   use="optional"/>
                            <xs:attribute ref="postTrigSamples"  use="optional"/>
                            <xs:attribute ref="intervalTime"     use="optional"/>
                            <xs:attribute ref="intervalDistMet"  use="optional"/>
                        </xs:extension>
                    </xs:simpleContent>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
</xs:element>

<!--
    <qmDur> - Query Message Duration.

Example:
    <qmDur startDate="2010-10-23" startTime="06:00:00"  endTime="10:00:00"></qmDur>

-->

<xs:element name="qmDur">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension    base="xs:string">
                <xs:attribute ref="startDate"/>
                <xs:attribute ref="endDate"  />
                <xs:attribute ref="startTime"/>
                <xs:attribute ref="endTime"  />
        <!--    <xs:attribute ref="timeDur"  />  Removed. Use start/end, date/time...  -->
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:element>

<!--
    <qmAction> - Query Message Action...
    start|stop...

Example:
    <qmAction action="stop" time="2019-10-23T10:00:00"></qmAction>
-->

<xs:element name="qmAction">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension       base="xs:string">
                <xs:attribute    name="time" type="xs:dateTime"/>
                <xs:attribute    name="action" default="start">
                    <xs:simpleType>
                        <xs:restriction     base="xs:string">
                        <xs:enumeration  value="start"/>
                        <xs:enumeration  value="stop" />
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
        </xs:extension>
    </xs:simpleContent>
   </xs:complexType>
</xs:element>

<!--
   <qmTrigger> - Query Message trigger condition

   Example:  when all of the following conditions are met (Logical AND)
             for vehicle type 3, speed is >= 32.5 mps, speed change is >= 60% in less than
             20s and exterior temp is <= 3.5 degrees C.

   <qmTrigger>
      <when vehType="3"/>
      <when speedMps="32.5"          dataCond="GE"/>
      <when speedChangePct="60.0"   timeDur="PT20s" dataCond="GE"/>
      <when timeDur="PT20s"          dataCond="LE"/>
      <when extAirTempC="3.5"        dataCond="LE"/>
   </qmTrigger>
-->

<xs:element name="qmTrigger">
   <xs:complexType>
      <xs:sequence>
         <xs:element name="when" minOccurs="1" maxOccurs="10">
            <xs:complexType>
               <xs:simpleContent>
                  <xs:extension   base="xs:string">
                     <xs:attribute ref="speedMps"         />
                     <xs:attribute ref="speedChangeMps"   />
                     <xs:attribute ref="speedChangePct"   />
                     <xs:attribute ref="headingDeg"       />
                     <xs:attribute ref="toleranceDeg"     />
                     <xs:attribute ref="steeringWheelAngle"/>
                     <xs:attribute ref="longAccel"        />
                     <xs:attribute ref="latAccel"         />
                     <xs:attribute ref="vertAccel"        />
                     <xs:attribute ref="yawRate"          />
                     <xs:attribute ref="brakeApplied"     />     <!-- Vehicle control -->
                     <xs:attribute ref="traction"         />
                     <xs:attribute ref="abs"              />
                     <xs:attribute ref="scs"              />
                     <xs:attribute ref="brakeBoost"       />
                     <xs:attribute ref="auxBrake"         />
                     <xs:attribute ref="panicBrake"       />
                     <xs:attribute ref="wiperPos"         />
                     <xs:attribute ref="normalBeam"       />     <!-- Individual light -->
                     <xs:attribute ref="highBeam"         />
                     <xs:attribute ref="fogLight"         />
                     <xs:attribute ref="hazardLight"      />
                     <xs:attribute ref="extAirTempC"      />
                     <xs:attribute ref="dataCond"         />
      <!-- 3/18/20     <xs:attribute ref="timeCond"         /> In <dataRequest>        -->
                     <xs:attribute ref="timeDur"          /> <!-- built-in data type-->
      <!-- 3/23/20     <xs:attribute ref="intervalTime"   /> REMOVED, in dataRequest  -->
      <!-- 3/23/20     <xs:attribute ref="intervalDistMet"/> REMOVED, in dataRequest  -->
      <!-- 3/18/20     <xs:attribute ref="samples"        />        REMOVED           -->
                  </xs:extension>
                </xs:simpleContent>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
  </xs:complexType>
</xs:element>


<!--  <gfRegion> - For defining geo fenced region...

    1. Polygon
    2. Circle
    3. Drive distance from a node point
    4. From → To location


    Added: All GF regions include optional min/max elevation that applies to
           the entire defined region
           March 11, 2020

Examples:
    <gfRegion>   - Polygon
        <gfRegionElev>
            <elev elevMinMet="23" elevMaxMet="123"/>
        <gfRegionElev>
        <poly>
            <node latDeg="xxx.xx" longDeg="xx.xx"/>
            <node latDeg="xxx.xx" longDeg="xx.xx"/>
            <node latDeg="xxx.xx" longDeg="xx.xx"/>
            <node latDeg="xxx.xx" longDeg="xx.xx"/>
        </poly>
  </gfRegion>

  <gfRegion>   - Circle
        <gfRegionElev>
            <elev elevMinMet="23" elevMaxMet="123"/>
        </gfRegionElev>
        <circle>
            <center latDeg="xxx.xx" longDeg="xx.xx" radiusMet="nnn"/>
        </circle>
  </gfRegion>

    <gfRegion>   - from2toLocation
        <gfRegionElev>
            <elev elevMinMet="23" elevMaxMet="123"/>
        <gfRegionElev>
        <from2toLocatoin>
            <fromLocation latDeg="xx.xx" longDeg="xx.xx"
                headingDeg="nnn" toleranceDeg="nnn"/>
            <toLocation latDeg="xx.xx" longDeg="xx.xx"
                headingDeg="nnn" toleranceDeg="nnn"/>
        </from2toLocation>
    </gfRegion>

    <gfRegion>   - driveDistKm
        <gfRegionElev>
            <elev elevMinMet="23" elevMaxMet="123"/>
        <gfRegionElev>

        <driveDistKm>
            <from  latDeg="xxx.xx" longDeg="xx.xx" distKm="50" headingDeg="nnn"
                   toleranceDeg="nnn"/>
        </driveDistKm>
```

```
   </gfRegion>
-->

<xs:element name="gfRegion">
   <xs:complexType>
      <xs:sequence>

      <!-- Optional "Min/Max Elevation" that applies to all regions defined below... -->

       <xs:element name="gfRegionElev" minOccurs="0" maxOccurs="1">
          <xs:complexType>
             <xs:sequence>
                <xs:element name="elev" minOccurs="1" maxOccurs="1">
                   <xs:complexType>
                      <xs:simpleContent>
                         <xs:extension   base="xs:string">
                            <xs:attribute ref="elevMinMet" use="optional"/>
                            <xs:attribute ref="elevMaxMet" use="optional"/>
                         </xs:extension>
                      </xs:simpleContent>
                   </xs:complexType>
                </xs:element>
             </xs:sequence>
          </xs:complexType>
      </xs:element>

      <!-- Polygon -->

      <xs:element name="poly"              minOccurs="0" maxOccurs="1">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="node"    minOccurs="3" maxOccurs="50">
                  <xs:complexType>
                     <xs:simpleContent>
                        <xs:extension       base="xs:string">
                           <xs:attribute    ref="latDeg"     use="required"/>
                           <xs:attribute    ref="longDeg"    use="required"/>
                        </xs:extension>
                     </xs:simpleContent>
                  </xs:complexType>
               </xs:element>
            </xs:sequence>
         </xs:complexType>
      </xs:element>

      <!-- Circle -->

      <xs:element name="circle"  minOccurs="0" maxOccurs="1">
         <xs:complexType>
            <xs:sequence>
               <xs:element name="center">
                  <xs:complexType>
                     <xs:simpleContent>
                        <xs:extension       base="xs:string">
                           <xs:attribute    ref="latDeg"     use="required"/>
                           <xs:attribute    ref="longDeg"    use="required"/>
                           <xs:attribute    ref="radiusMet"  use="required"/>
                        </xs:extension>
                     </xs:simpleContent>
                  </xs:complexType>
               </xs:element>
            </xs:sequence>
         </xs:complexType>
```

```
        </xs:element>

        <!-- From to location -->

        <xs:element name="from2toLocation" minOccurs="0" maxOccurs="1">
           <xs:complexType>
              <xs:sequence>
                 <xs:element name="fromLocation">
                    <xs:complexType>
                       <xs:simpleContent>
                          <xs:extension       base="xs:string">
                             <xs:attribute    ref="latDeg"        use="required"/>
                             <xs:attribute    ref="longDeg"       use="required"/>
                             <xs:attribute    ref="headingDeg"    use="optional"/>
                             <xs:attribute    ref="toleranceDeg"  use="optional"/>
                             <xs:attribute    ref="radiusMet"     use="optional"/>
                          </xs:extension>
                       </xs:simpleContent>
                    </xs:complexType>
                 </xs:element>

                 <xs:element name="toLocation">
                    <xs:complexType>
                       <xs:simpleContent>
                          <xs:extension       base="xs:string">
                             <xs:attribute    ref="latDeg"        use="required"/>
                             <xs:attribute    ref="longDeg"       use="required"/>
                             <xs:attribute    ref="headingDeg"    use="optional"/>
                             <xs:attribute    ref="toleranceDeg"  use="optional"/>
                             <xs:attribute    ref="radiusMet"     use="optional"/>
                          </xs:extension>
                       </xs:simpleContent>
                    </xs:complexType>
                 </xs:element>
              </xs:sequence>
           </xs:complexType>
        </xs:element>

        <!-- Drive Distance -->

        <xs:element name="driveDistKm" minOccurs="0" maxOccurs="1">
           <xs:complexType>
              <xs:sequence>
                 <xs:element name="from">
                    <xs:complexType>
                       <xs:simpleContent>
                          <xs:extension       base="xs:string">
                             <xs:attribute    ref="latDeg"        use="required"/>
                             <xs:attribute    ref="longDeg"       use="required"/>
                             <xs:attribute    ref="distKm"        use="required"/>
                             <xs:attribute    ref="headingDeg"    use="optional"/>
                             <xs:attribute    ref="toleranceDeg"  use="optional"/>
                             <xs:attribute    ref="radiusMet"     use="optional"/>
                          </xs:extension>
                       </xs:simpleContent>
                    </xs:complexType>
                 </xs:element>
              </xs:sequence>
           </xs:complexType>
        </xs:element>

        </xs:sequence>
     </xs:complexType>
```

```
</xs:element>

<!--
   <dataAvg> - Data averaging...

      Following is eliminated and incorporated in <dataRequest> in <provide>
      along with number of pre and post samples, intervalTime and intervalDistMet.
      3/23/2020
   -->

<!--
<xs:element name="dataAvg">
   <xs:complexType>
      <xs:sequence>
         <xs:element name="avg" minOccurs="1" maxOccurs="5" >
            <xs:complexType>
               <xs:simpleContent>
                  <xs:extension       base="xs:string">
                     <xs:attribute    ref="dataName"      />
                     <xs:attribute    ref="intervalTime"  />
                     <xs:attribute    ref="samples"       />
                     <xs:attribute    ref="preTrigSamples"/>
                  </xs:extension>
               </xs:simpleContent>
            </xs:complexType>
         </xs:element>
      </xs:sequence>
   </xs:complexType>
</xs:element>

   -->

<!--   ========================== END of Query Message Section ==========================
       ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
       ========================== START of Query Response Section ======================

   <rmFrame> - Root for Response Message Frame

Example:
   <qmFrame>
      <eventMsg followed by attributes... />               Required
      <vehVars> followed by attributes... />               Required
      <vehPos   followed by attributes.../>                Required
      <vehAccelStatus followed by attributes.../>          Optional
      <vehBrakeStatus followed by attributes.../>          Optional
      <wxtLightStatus followed by attributes.../>          Optional
      <gfRegionEntryExitStatus followed by attributes.../> Optional
   </rmFrame>

-->

<xs:element name="rmFrame">
   <xs:complexType>
      <xs:sequence minOccurs="1" maxOccurs="1">
         <xs:element ref="eventMsg"       minOccurs="1"    maxOccurs="1"/>
         <xs:element ref="vehVars"        minOccurs="1"    maxOccurs="1"/>
         <xs:element ref="vehPos"         minOccurs="1"    maxOccurs="1"/>
         <xs:element ref="vehAccelStatus" minOccurs="0"    maxOccurs="1"/>
         <xs:element ref="vehBrakeStatus" minOccurs="0"    maxOccurs="1"/>
         <xs:element ref="extLightStatus" minOccurs="0"    maxOccurs="1"/>
         <xs:element ref="gfRegionEntryExitStatus" minOccurs="0"  maxOccurs="10"/>
      </xs:sequence>
   </xs:complexType>
```

```
</xs:element>

<!--
  <vehVars> - List of vehicle parameters for providing vehicle data/status.
              Multiple vehicle variables and their values are defined using
              attributes that allow for validity checking.

              Note: Many vehicle data are now aggregated in different groups and
              removed from <vehVars>

Example:
   <vehVars>
      <vehData vehType="3" longAccel="1.5" brakeApplied="yes"/>
   </vehVars>

   Following removed on Mar. 5, 2020. They are captured in <vehPos>
      letDeg
      longDeg
      elevMet
      headingDeg
      toleranceDeg
  -->

<!--  Following revised on Thursday, Mar. 12, 2020

   The elements taken out are now incorporated in <rmFrame> to alleviate ambiguity
   and duplicate use in RM.
   1. vehAccelStatus
   2. vehBrakeStatus
   3. extLightStatus

   Following are removed from the list since the TMC (I assume) is not
   interested to know in RM about them...

   1. speedChangeMps
   2. speedChangePct
   -->

<xs:element name="vehVars">
   <xs:complexType>
      <xs:sequence>
         <xs:element name="vehData" minOccurs="1" maxOccurs="25">
            <xs:complexType>
            <xs:simpleContent>
               <xs:extension       base="xs:string">
                  <xs:attribute    ref="vehType"       />
                  <xs:attribute    ref="speedMps"      />
                  <xs:attribute    ref="yawRate"       />
                  <xs:attribute    ref="steeringWheelAngle"/>
                  <xs:attribute    ref="wiperPos"      />
                  <xs:attribute    ref="extAirTempC"   />
               </xs:extension>
            </xs:simpleContent>
         </xs:complexType>
        </xs:element>
      </xs:sequence>
   </xs:complexType>
</xs:element>

<!-- =========================  Vehicle Status Section  ============================== -->

<!--
```

```
   <extLightList> - Exterior light status list for all lights.

      1. Normal Beam
      2. High Beam
      3. Fog Light
      4. Hazard Light

      All 4 values must be provided, separated by comma.

Example: Normal beam and fog lights are on...
   <extLightList>1,0,1,0</extLightList>

-->

<xs:element name="extLightList">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="[0|1],[0|1],[0|1],[0|1]"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>

<!--
   <extLightStatus> - Vehicle exterior light status for individual light.

Example:
   <extLightStatus normalBeam="false" highBeam="true"/>

   true or false can be replaced by 1 or 0
-->

<xs:element name="extLightStatus">
   <xs:complexType>
      <xs:attribute name="normalBeam"   type="xs:boolean"/>
      <xs:attribute name="highBeam"      type="xs:boolean"/>
      <xs:attribute name="fogLight"      type="xs:boolean"/>
      <xs:attribute name="hazardLight"  type="xs:boolean"/>
   </xs:complexType>
</xs:element>

<!--
   <vehBrakeList> - List of Vehicle brake system status for all brakes.
                   0 = off, 1 = on, -1 = unavailable

Example: All seven values must be present, separated by comma

   <vehBrakeList>1,0,1,0,-1,-1,0</vehBrakeList>

      1. brakeApplied = 1
      2. traction = 0
      3. abs = 1
      4. scs = 0  (stability control system)
      5. brakeBoost = -1
      6. auxBrake = -1
      7. panicBrake = 0
-->

<xs:element name="vehBrakeList">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="-?[0|1],-?[0|1],-?[0|1],-?[0|1],-?[0|1],-?[0|1],-?[0|1]"/>
      </xs:restriction>
   </xs:simpleType>
```

```
</xs:element>

<!--

   <vehBrakeStatus> - Alternative for individual brake status.
                      Possible values: yes|no|unavailable
Example:
   <vehBrakeStatus brakeApplied="yes" abs="yes" brakeBoost="unavailable"/>
-->

<xs:element name="vehBrakeStatus">
   <xs:complexType>
      <xs:attribute    ref="brakeApplied"/>
      <xs:attribute    ref="traction"/>
      <xs:attribute    ref="abs"/>
      <xs:attribute    ref="scs"/>
      <xs:attribute    ref="brakeBoost"/>
      <xs:attribute    ref="auxBrake"/>
      <xs:attribute    ref="panicBrake"/>
   </xs:complexType>
</xs:element>

<!--
  <vehAccelList> - List of all (long, lat & vert) vehicle accelerations.

  vehAccelList:
      long Acceleration - Along the vehicle longitudinal axis
      lat  Acceleration - Along the vehicle lateral axis
      vert Acceleration - Along the vehicle vertical axis

Example:
   <vehAccelList>-1.346,2.3456,1.3473287</vehAccelList>

-->

<xs:element name="vehAccelList">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern   value="-?[0-9].[0-9]+,-?[0-9].[0-9]+,-?[0-1].[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>


<!--
   <vehAccelStatus> - Alternate method for individual acceleration values...

Example:
   <vehAccelStatus longAccel="-1.4325" latAccel="0.345"/>

-->

<xs:element name="vehAccelStatus">
   <xs:complexType>
      <xs:attribute    ref="longAccel"/>
      <xs:attribute    ref="latAccel"/>
      <xs:attribute    ref="vertAccel"/>
   </xs:complexType>
</xs:element>

<!--
   <pos3MD> - Vehicle position in micro degrees, elev in meters..
```

```
Example:
   <pos3MD latMD="423456" longMD="-833476348" elev="23"/>

-->

<xs:element name="pos3MD">
   <xs:complexType>
      <xs:attribute    ref="latMD"      use="required"/>
      <xs:attribute    ref="longMD"     use="required"/>
      <xs:attribute    ref="elevMet"    use="optional"/>
   </xs:complexType>
</xs:element>


<!--
   <pos3D> - Vehicle position in degrees.

   latDeg  - In degrees
   longDeg - In degrees
   elevMet - In meters

Example:  position in degrees, elev in m.
   <pos3D latDeg="42.3456" longDeg="-83.3476348 elevMet="23"/>

   Note:
   pos3D is replaced by vehPos which includes heading and tolerance.

-->

<xs:element name="pos3D">
   <xs:complexType>
      <xs:attribute    ref="latDeg"     use="required"/>
      <xs:attribute    ref="longDeg"    use="required"/>
      <xs:attribute    ref="elevMet"    use="optional"/>
   </xs:complexType>
</xs:element>

<!--
   <vehPos> - Vehicle position including heading and heading tolerance.

Example:  Position in degrees, elev in m.

   <vehPos latDeg="42.3456" longDeg="-83.3476348 elevMet="23" headingDeg="180"/>

-->

<xs:element name="vehPos">
   <xs:complexType>
      <xs:attribute    ref="latDeg"        use="required"/>
      <xs:attribute    ref="longDeg"       use="required"/>
      <xs:attribute    ref="elevMet"       use="optional"/>
      <xs:attribute    ref="headingDeg"    use="optional"/>
      <xs:attribute    ref="toleranceDeg"  use="optional"/>
   </xs:complexType>
</xs:element>

<!--
   <pos3MDList> - List of lat, long in micro degrees, elev in meters...

Example:
   <pos3MDList>423456,-833476348,23</pos3MDList>

-->
```

```
<xs:element name="pos3MDList">
   <xs:simpleType>
      <xs:restriction base="xs:string">
         <xs:pattern  value="-?[0-9]+,-?[0-9]+,-?[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>

<!--
   <pos3DList> - list of lat, long in degrees, elev in meters.

Example:
   <pos3DList>42.3456,-83.3476348,23</pos3DList>

-->

<xs:element name="pos3DList">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="-?[0-9]{2}.[0-9]+,-?[0-9]{2}.[0-9]+,-?[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

Appendix B – XML Schema Attributes for QM / RM

Following section describes implemented XML schema attributes for the elements.

```
<!-- ======================  SCHEMA ATTRIBUTE Definitions...  ====================== -->

<xs:attribute name="msgDateTime" type="xs:dateTime"/>   <!-- Message timestamp -->
<xs:attribute name="rseID"       type="xs:string"  />   <!-- RSE ID - For future use -->
<xs:attribute name="eventInfo"   type="xs:string"  />   <!-- event information -->
<xs:attribute name="radiusMet"   type="xs:integer" />   <!-- for geo fence -->

<!--
   Following added to support averaging of pre- and post-trigger samples in
   <dataRequest>
   Date: 3/19/2020

-->

<xs:attribute name="preTrigSamples"     type="xs:integer"/>
<xs:attribute name="postTrigSamples"    type="xs:integer"/>
<xs:attribute name="intervalTime"       type="xs:time"/>
<xs:attribute name="intervalDistMet"    type="xs:integer"/>


<!--
   Following removed. Data average is used in <provideAvg> in <dataRequest>

   Mar. 23, 2020

<xs:attribute name="dataAverage">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="yes|no"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

-->

<!--
   <eventID> - Unique event ID generated by TMC for which the query is issued.

   -->

<xs:attribute name="eventID">
   <xs:simpleType>
      <xs:restriction        base="xs:integer">
         <xs:minInclusive    value="0"/>
         <xs:maxInclusive    value="999"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   <gfStatus> - Vehicle's Geo-Fence status for gfRegionEntryExitStatus data request.

   Added on Mar 2, 2020
   0 = outside/not applicable for GF
   1 = Entered the GF
   2 = Inside the GF
   3 = Exited the GF

-->

<xs:attribute name="gfStatus">
   <xs:simpleType>
      <xs:restriction  base="xs:integer">
         <xs:pattern   value="0|1|2|3"/>
       </xs:restriction>
   </xs:simpleType>
</xs:attribute>


<!--
   <dataName> - List of vehicle data names

   Following defines data names used for requesting data in QM using <dataRequest> within
   qmFrame

-->

<xs:attribute name="dataName">
   <xs:simpleType>
      <xs:restriction     base="xs:string">
         <xs:enumeration  value="vehType"            />
         <xs:enumeration  value="vehPos"             />
         <xs:enumeration  value="pos3D"              />
         <xs:enumeration  value="pos3DList"          />
         <xs:enumeration  value="pos3MD"             />
         <xs:enumeration  value="pos3MDList"         />
         <xs:enumeration  value="headingDeg"         />
         <xs:enumeration  value="speedMps"           />
         <xs:enumeration  value="speedChangeMps"     />
         <xs:enumeration  value="vehAccelStatus"     />
         <xs:enumeration  value="vehAccelLst"        />
         <xs:enumeration  value="longAccel"          />
         <xs:enumeration  value="latAccel"           />
         <xs:enumeration  value="vertAccel"          />
         <xs:enumeration  value="yawRate"            />
         <xs:enumeration  value="steeringWheelAngle" />
         <xs:enumeration  value="vehBrakeStatus"     />
         <xs:enumeration  value="brakeApplied"       />
         <xs:enumeration  value="traction"           />
         <xs:enumeration  value="abs"                />
         <xs:enumeration  value="scs"                />
         <xs:enumeration  value="vehBrakeList"       />
         <xs:enumeration  value="brakeBoost"         />
         <xs:enumeration  value="auxBrake"           />
         <xs:enumeration  value="panicBrake"         />
         <xs:enumeration  value="wiperPos"           />
         <xs:enumeration  value="extLightStatus"     />
         <xs:enumeration  value="extLightList"       />
         <xs:enumeration  value="normalBeam"         />
         <xs:enumeration  value="highBeam"           />
         <xs:enumeration  value="fogLight"           />
         <xs:enumeration  value="hazardLight"        />
         <xs:enumeration  value="extAirTempC"        />
         <xs:enumeration  value="gfRegionEntryExitStatus"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>
```

```
<!--
   <dataAvgName> - List of vehicle data names for averaging pre- and post-triggered
   samples. These are limited number of variables to support data averaging in place of
   <dataAverage>
   Following defines data averaging names that can be used in:
   <dataRequest> from qmFrame

   Mar. 23, 2020

-->

<xs:attribute name="dataAvgName">
   <xs:simpleType>
      <xs:restriction     base="xs:string">
         <xs:enumeration  value="speedMps"        />
         <xs:enumeration  value="speedChangeMps"  />
         <xs:enumeration  value="longAccel"       />
         <xs:enumeration  value="headingDeg"      />
         <xs:enumeration  value="extAirTempC"     />
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>


<!--
   <rmCommType> - Communication Type DSRC, Cellular, Cellular and DSRC.

-->

<xs:attribute name="rmCommType">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern   value="cell|DSRC|cellAndDSRC"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   <msgCount> - Message Count is incremented to indicate a new message for the
   same event

-->

<xs:attribute name="msgCount">
   <xs:simpleType>
      <xs:restriction        base="xs:integer">
         <xs:minInclusive    value="0"/>
         <xs:maxInclusive    value="127"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   <msgType> - Message type: iamhere, query, response or inform.

  -->

<xs:attribute name="msgType">
   <xs:simpleType>
      <xs:restriction base="xs:string">
         <xs:pattern value="iamhere|query|response|inform"/>
      </xs:restriction>
   </xs:simpleType>
```

```
</xs:attribute>

<!--
   <msgPriority> - Message priority level (0..9), 9 being the highest.

-->

<xs:attribute name="msgPriority">
   <xs:simpleType>
      <xs:restriction        base="xs:integer">
         <xs:minInclusive    value="0"/>
         <xs:maxInclusive    value="9"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   <vehResponsePct> - % of vehicles to respond to the query message

   Note: Responding vehicle to generate and use a random number. If the number is within
   requested %, respond to query.

-->

<xs:attribute name="vehResponsePct">
   <xs:simpleType>
      <xs:restriction        base="xs:integer">
         <xs:minInclusive    value="0"/>
         <xs:maxInclusive    value="100"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
  <schemaVer> - Schema Version indicates the version of schema used for query and response

-->

<xs:attribute name="schemaVer">
   <xs:simpleType>
      <xs:restriction        base="xs:decimal">
         <xs:minInclusive    value="1.0"/>
         <xs:maxInclusive    value="20.0"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>



<!--
  <cCode> - Cause Code
```

causeCode and subCauseCode define the type of event. The value of subCauseCode is application dependent. The values for causeCode are defined in ETSI EN 302 637-3.

Table 2: List of ETSI Cause Codes and Sub Cause Codes

Value / Units	**Cause Codes -** ETSI EN 302 637-3
0	reserved
1	trafficCondition
2	accident
3	roadworks
6	adverseWeatherCondition-Adhesion
9	hazardousLocation-SurfaceCondition
10	hazardousLocation-ObstacleOnTheRoad
11	hazardousLocation-AnimalOnTheRoad
12	humanPresenceOnTheRoad
14	wrongWayDriving
15	rescueAndRecoveryWorkInProgress
17	adverseWeatherCondition-ExtremeWeatherCondition
18	adverseWeatherCondition-Visibility
19	adverseWeatherCondition-Precipitation
26	slowVehicle
27	dangerousEndOfQueue
91	vehicleBreakdown
92	postCrash
93	humanProblem
94	stationaryVehicle
95	emergencyVehicleApproaching
96	hazardousLocation-DangerousCurve
97	collisionRisk
98	signalViolation
99	dangerousSituation
(0..255)	SubCausecode

```
-->

<xs:attribute name="cCode">
   <xs:simpleType>
      <xs:restriction base="xs:integer">
         <xs:minInclusive value="0"/>
         <xs:maxInclusive value="99"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
  <scCode> - Sub Cause Code - See ETSI sub cause code list

  -->

<xs:attribute name="scCode">
   <xs:simpleType>
      <xs:restriction base="xs:integer">
         <xs:minInclusive value="0"/>
         <xs:maxInclusive value="4"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>


<!--
```

```
    <vehType> - Vehicle Types.

    0 = All vehicle types(default)
    1 = Passenger vehicle
    2 = Heavy duty truck
    3 = Specialty equipped vehicles
    4 = Snowplow
    5 = Road repair truck
    ...
-->

<xs:attribute name="vehType">
    <xs:simpleType>
        <xs:restriction  base="xs:integer">
            <xs:pattern  value="[0-9]"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!--
    <vehID> - Vehicle ID#.
    vehID - Randomly generated value

-->

<xs:attribute name="vehID">
    <xs:simpleType>
        <xs:restriction  base="xs:string">
            <xs:pattern  value="EDCM\-[0-9]*"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!--
    <speedMps> - Vehicle Speed in m/s

-->

<xs:attribute name="speedMps">
    <xs:simpleType>
        <xs:restriction        base="xs:decimal">
            <xs:minInclusive    value="0.0"/>
            <xs:maxInclusive    value="99.0"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!--  <speedChangeMps> -   Veh speed change in m/s.
      Speed Change to accommodate both +ve and -ve values.

-->

<xs:attribute name="speedChangeMps">
    <xs:simpleType>
        <xs:restriction        base="xs:decimal">
            <xs:minInclusive    value="-99.0"/>
            <xs:maxInclusive    value="99.0"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!--  speedChangePct in %
```

```
-->

<xs:attribute name="speedChangePct">
   <xs:simpleType>
      <xs:restriction        base="xs:decimal">
         <xs:minInclusive    value="0.0"/>
         <xs:maxInclusive value="100.0"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   Vehicle acceleration...
    1. Longitudinal
    2. Lateral
    3. Vertical

-->

<xs:attribute name="longAccel">
   <xs:simpleType>
      <xs:restriction  base="xs:decimal">
         <xs:pattern   value="-?[0-9].[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<xs:attribute name="latAccel">
   <xs:simpleType>
      <xs:restriction  base="xs:decimal">
         <xs:pattern   value="-?[0-9].[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<xs:attribute name="vertAccel">
   <xs:simpleType>
      <xs:restriction  base="xs:decimal">
         <xs:pattern   value="-?[0-1].[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   steering wheel angle and Yaw

   -->

<xs:attribute name="steeringWheelAngle">
   <xs:simpleType>
      <xs:restriction     base="xs:decimal">
         <xs:minInclusive value="-30.0"/>
         <xs:maxInclusive value="30.0"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<xs:attribute name="yawRate">
   <xs:simpleType>
      <xs:restriction     base="xs:integer">
         <xs:minInclusive value="-99"/>
         <xs:maxInclusive value="99"/>
      </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!--
    Vehicle brake status... (yes|no|unavailable)

    1. brake
    2. traction
    3. abs
    4. scs
    5. brakeBoost
    6. auxBrake
    7. panicBrake

-->

<xs:attribute name="brakeApplied">
    <xs:simpleType>
        <xs:restriction  base="xs:string">
            <xs:pattern  value="yes|no|unavailable"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!-- traction -->

<xs:attribute name="traction">
    <xs:simpleType>
        <xs:restriction  base="xs:string">
            <xs:pattern  value="yes|no|unavailable"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!-- abs -->

<xs:attribute name="abs">
    <xs:simpleType>
        <xs:restriction  base="xs:string">
            <xs:pattern  value="yes|no|unavailable"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!-- scs - Stability Control System -->

<xs:attribute name="scs">
    <xs:simpleType>
        <xs:restriction  base="xs:string">
            <xs:pattern  value="yes|no|unavailable"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>

<!-- brake boost -->

<xs:attribute name="brakeBoost">
    <xs:simpleType>
        <xs:restriction  base="xs:string">
            <xs:pattern  value="yes|no|unavailable"/>
        </xs:restriction>
    </xs:simpleType>
</xs:attribute>
```

```
<!-- aux brake -->

<xs:attribute name="auxBrake">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="yes|no|unavailable"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   The deceleration value associated for the panic brake is set by the vehicle.

-->

<xs:attribute name="panicBrake">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="yes|no|unavailable"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>


<!--
   Exterior light status... (true|false) OR (1|0)
      1. Normal Beam
      2. High Beam
      3. Fog Light
      4. Hazard Light
  -->

<xs:attribute name="normalBeam"  type="xs:boolean"/>
<xs:attribute name="highBeam"    type="xs:boolean"/>
<xs:attribute name="fogLight"    type="xs:boolean"/>
<xs:attribute name="hazardLight" type="xs:boolean"/>


<!--
   Wiper Position
   0=off, 1=normal, 2=high, 3=intermittent

-->

<xs:attribute name="wiperPos">
   <xs:simpleType>
      <xs:restriction  base="xs:integer">
         <xs:minInclusive    value="0"/>
         <xs:maxInclusive    value="3"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   Exterior Air Temp in Celsius

-->

<xs:attribute name="extAirTempC">
   <xs:simpleType>
      <xs:restriction  base="xs:decimal">
         <xs:minInclusive    value="-40.0"/>
```

```
          <xs:maxInclusive    value="100.0"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>


<!--
  Distance, heading and heading tolerance.

  -->

<!--  Distance in km -->

<xs:attribute name="distKm">
   <xs:simpleType>
      <xs:restriction base="xs:integer">
          <xs:minInclusive    value="1"/>
          <xs:maxInclusive    value="50"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--  Heading in Degrees -->

<xs:attribute name="headingDeg">
   <xs:simpleType>
      <xs:restriction base="xs:integer">
          <xs:minInclusive    value="0"/>
          <xs:maxInclusive    value="359"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--  Tolerance in Degrees... -->

<xs:attribute name="toleranceDeg">
   <xs:simpleType>
      <xs:restriction  base="xs:integer">
          <xs:minInclusive    value="0"/>
          <xs:maxInclusive    value="359"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--
   Vehicle position
   1. in degrees
   2. in micro degrees

   latDeg in degrees
   longDeg in degrees
   elevMet in meters

-->

<!--  Latitude in Degrees... -->

<xs:attribute name="latDeg">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
          <xs:pattern  value="-?[0-9]{2}.[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
```

```
</xs:attribute>

<!--  Longitude in Degrees... -->

<xs:attribute name="longDeg">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="-?[0-9]{2}.[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--  Elevation in meters... -->

<xs:attribute name="elevMet">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="-?[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--  Minimum Elevation in meters... -->

<xs:attribute name="elevMinMet">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="-?[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<!--  Maximum Elevation in meters... -->

<xs:attribute name="elevMaxMet">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="-?[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>


<!--
  Vehicle position in micro degrees

   latMD     - in micro degrees
   longMD    - in micro degrees
   elevMet   - in meters

-->

<xs:attribute name="latMD">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="-?[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>

<xs:attribute name="longMD">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
```

```
          <xs:pattern  value="-?[0-9]+"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>


<!--
   Message trigger condition based on data
   1. Comparison
   2. Logical operation
   3. Boolean

-->

<xs:attribute name="dataCond">
   <xs:simpleType>
      <xs:restriction  base="xs:string">
         <xs:pattern  value="LT|GT|LE|GE|EQ|NE|true|false|1|0"/>
      </xs:restriction>
   </xs:simpleType>
</xs:attribute>


<!--

   1. Message duration
   2. start and end dates
   3. start and end times
   4. time duration - PnYnMnDTnHnMnS
      P followed by Year, Month, Days, Time in H, M and S
   5. time intervalTime - hh:mm:ss

-->

<xs:attribute name="startDate"       type="xs:date"/>
<xs:attribute name="endDate"         type="xs:date"/>
<xs:attribute name="startTime"       type="xs:time"/>
<xs:attribute name="endTime"         type="xs:time"/>
<xs:attribute name="startDateTime"   type="xs:dateTime"/>
<xs:attribute name="endDateTime"     type="xs:dateTime"/>
<xs:attribute name="timeDur"        type="xs:duration"/>
```