SISO-STD-014.2-2018

Standard for
Gateway Filtering Language

12 September 2018

Version 1.0

SAC Approved: 09/28/2018 **EXCOM Approved: 10/11/2018**

Prepared by:
Gateway Description and Configuration Languages (GDACL)
Product Development Group

P.O. Box 781238
Orlando, FL 32878-1238, USA

SISO Inc. Board of Directors
P.O. Box 781238
Orlando, FL 32878-1238, USA

Revision History

Version	Section	Date (MM/DD/YYYY)	Description
1.0	All	09/12/2018	Initial Version

Participants

At the time this product was submitted to the Standards Activity Committee (SAC) for approval, the Gateway Description and Configuration Languages Product Development Group (PDG) had the following membership and was assigned the following SAC Technical Area Director:

Product Development Group

Robert Lutz (Chair)
Dannie Cutts (Vice-Chair)
Kurt Lessmann (Vice-Chair)
David Drake (Editor)

Patrice Le Leydour (SAC Technical Area Director)

Jim Coolahan
Dannie Cutts
Lance Marrou
Angus (Thom) McLean
Michael O'Connor
Felix Rodriguez

The following individuals comprised the ballot group for this product.

Ballot Group

Jeffery Bergenthal
Curtis Blais
Dannie Cutts
David Drake
Michael Eady
Paul Gustavson
Scott Johnston
Michael Longworth
Robert Lutz
Lance Marrou
James McCall
Bjorn Moller
Roy Scrudder
Eugene Stoudenmire
Marcy Stutzman
Tom van den Berg

When the Standards Activity Committee approved this product on 28 September 2018, it had the following membership:

Standards Activity Committee

Katherine L. Morse (Chair)
Jean-Louis Igarza (Vice Chair)
Lance Marrou (Secretary)

Grant Bailey
Brad Dillman
Kevin Gupton
Paul Gustavson (CC Chair)
Michael Heaphy
Aerial Kreiner
Patrice Le Leydour
Chris McGroarty
Thom McLean
John Stevens

1

When the Executive Committee approved this product on 11 October 2018, it had the following membership:

Executive Committee

Michael O'Connor (Chair)
Robert Lutz (Vice Chair)
Jeff Abbott (Secretary)

Paul Gustavson	Lana McGlynn	Randy Saunders (CC Chair)
Kenneth Konwin	Katherine Morse (SAC Chair)	Roy Scrudder
Chris Metevier	Stefan Sandberg	Robert Siegfried

Introduction

This document defines a standard for describing data filters that reduce unneeded or unwanted data communication within a distributed simulation environment. The Gateway Filtering Language (GFL) defines a machine- readable Extensible Markup Language (XML) schema for defining data filters in gateways that are independent of any gateway implementation. GFL was originally developed in response to U.S. Department of Defense needs expressed in the Live- Virtual-Constructive Architecture Roadmap (LVCAR) effort. However, the GFL specification provided in this document provides a common language to address the broader, more general need in the international Modeling and Simulation (M&S) community to have common, standardized methods for building the gateway data filters needed in modern distributed applications.

The data files associated with this SISO Standard may be downloaded from this URL: http://www.sisostds.org/Schemas.aspx.

Table of Contents

Table of Figures

Standard for Gateway Filtering Language

1 Overview

Modeling and Simulation (M&S) is in wide use today within the international user community. While many M&S applications are standalone applications used in support of such areas as requirements development, concept exploration, and system design/development, the advent and increased maturity of technologies and standards for *distributed simulation* have led to much wider use of integrated live, virtual, and constructive (LVC) simulation environments in support of major Test and Evaluation (T&E) and training events. While most distributed simulation events apply a single underlying simulation architecture, such as Distributed Interactive Simulation (DIS), High Level Architecture (HLA), or Test and Training Enabling Architecture (TENA), sponsor requirements sometimes necessitate the selection of simulations whose external interfaces support different Simulation architectures. This is what is known as a *multi-architecture simulation environment*. When more than one simulation architecture must be used in the same environment, interoperability problems are compounded by the architectural differences. For instance, middleware incompatibilities, dissimilar metamodels for data exchange, and differences in the nature of the services that are provided by the architectures must all be reconciled for such environments to operate properly.

A *gateway* is an intelligent translator designed to link simulation enclaves that use dissimilar architectures. A *bridge* is very similar to a gateway but links enclaves using the same underlying architecture such as two HLA federations. Gateways and bridges are ubiquitous in the LVC community today and continue to represent one of the most widely used means of addressing interoperability concerns in multi-architecture simulation environments.

A gateway uses a Simulation Data Exchange Model (SDEM) to provide a representation of persistent and transient objects shared during a distributed simulation execution. When a user selects a gateway for their application, the mappings among the different simulation architectures and SDEM representations in the Exercise must be defined. *Filtering* is the elimination of simulation data from one Architecture/SDEM to another Architecture/SDEM, based on a criterion or criteria. The purpose of data filters is to reduce unneeded or unwanted data communication between Architecture/SDEM pairs. A gateway using GFL may have the same Architecture/SDEM on each side or different Architecture/SDEMs on each side.

Gateway Filtering Language (GFL) provides a common language for defining filters that is independent of the implementation. This permits exercise engineers who are not gateway experts to understand the filtering being performed by the gateway. GFL provides a common format and syntax for gateway developers to describe each filter to other distributed simulation professionals, and further lays the groundwork for the configuration of the gateways themselves. Since GFL is machine-readable, tools may be integrated into the gateway configuration process that enables users to search the GFL contents to determine the filtering aspects of the inter-architecture data communication.

This initial version of GFL supports filtering based on a class, class attribute, the sender of the message, and the geospatial location of a class instance. As additional experience accumulates using GFL in real applications, it is expected that modifications and extensions to the formal language will be necessary. Future revisions of this document will be produced as needed to support the evolving needs of the gateway user community.

1.1 Scope

This specification defines the requirements for the GFL schema along with a detailed description of the formal language that addresses these requirements. The description includes graphical representations of GFL schema elements to illustrate the types of filtering capabilities supported. A full XML schema for the GFL and an example are provided as annexes. This GFL applies to both gateways and bridges.

The standard for defining SDEM mapping between architectures is outside the scope of this document.

1.2 Purpose

The purpose of this Gateway Filtering Language (GFL) specification is to specify a formal Extensible Markup Language (XML) schema for defining data filters in gateways that is independent of any gateway implementation.

1.3 Objectives

The objective of GFL is to describe in detail the data that needs to be filtered by a gateway.

1.4 Intended Audience

The intended audience for this document includes:

- Exercise engineers who are not gateway experts can use this document to understand the kinds of information that can be filtered by a gateway and to characterize and select gateways based on filters.
- Gateway developers can use this document to record the filtering settings for a gateway in a machine- and human-readable form and to describe each filter to other distributed simulation professionals
- Resource developers can use this document to create tools to facilitate gateway selection, integration, and usage in support of Exercise planning, preparation, and simulation execution.

2 References

This document contains no normative references. Informative supporting documents may be found in the Bibliography.

3 Definitions, Acronyms, and Abbreviations

3.1 Definitions

bridge: A type of gateway where only a single simulation architecture is used. A bridge may translate between different SDEMs. Compare to *gateway*.

filtering: This is the elimination of distributed simulation traffic from one Architecture/SDEM to another Architecture/SDEM, based on a criterion or criteria. Filtering is intended to reduce unneeded or unwanted data communication between architectures.

gateway: An application for translating objects from one simulation architecture/SDEM pair to one or more simulation architecture/SDEM pair(s). Compare to *bridge*.

multi-architecture simulation environment. This is an integrated set of LVC resources that are employing the services of more than one architecture during a single simulation execution

Simulation Data Exchange Model (SDEM): This is a type of object model that provides a representation of persistent and transient objects shared during a distributed simulation execution.

3.2 Acronyms and Abbreviations

CRS	Coordinate Reference System
DIS	Distributed Interactive Simulation
DoD	Department of Defense
GFL	Gateway Filtering Language
HLA	High Level Architecture
ID	Identification
LVC	Live-Virtual-Constructive
M&S	Modeling and Simulation
SDEM	Simulation Data Exchange Model
T&E	Test and Evaluation
TENA	Test and Training Enabling Architecture
XML	Extensible Markup Language

4 GFL Definition

The requirements for GFL are as follows:

- A formal mechanism shall exist for defining filters in an implementation-independent format.
- Filters shall be specified in only one direction with a single input side and one or more output sides.
- Mechanisms shall be provided to give the user options for specifying the filter criteria. These include filtering based on class, class attributes, sender, and spatial constraints.

The current version of the GFL provides mechanisms permitting the most common types of filtering to be defined: class-based filtering, class attribute-based filtering, sender- and receiver-based filtering, and spatial-based filtering. Each filtering type requires its own treatment within the GFL.

GFL is implemented as an XML schema that allows users or developers to document the filtering settings for a gateway in a machine- and human-readable form. The following descriptions of GFL capabilities and design contain graphical depictions of the GFL schema elements to illustrate key points of usage. The legend for these depictions is presented in Figure 1. The actual GFL schema is provided in Annex A. In addition, an example GFL document is in Annex B, which contains three filter definitions that illustrate the application of the GFL schema.

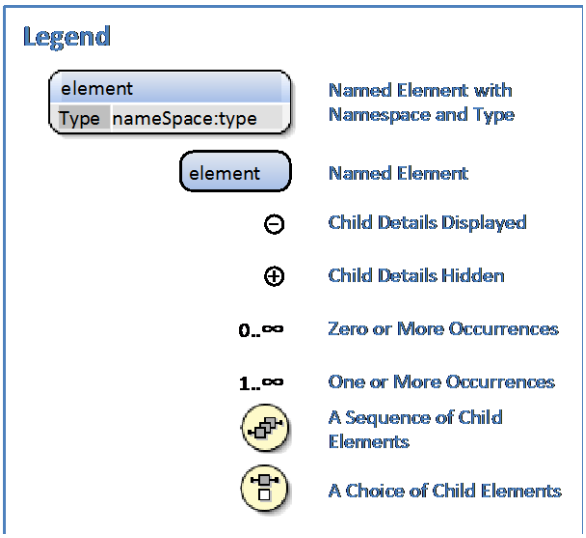

Figure 1: Legend for the XML diagrams

4.1 Class-Based Filtering

The primary structure in GFL supporting entity-based filtering is the *classFilter* element (see Figure 2). Within this element are:

- A mandatory *classIdentifier* element, which is a string representation from which the targeted entity shall be uniquely identified, and
- Zero or more optional *attribute* elements, which identify named attributes of the targeted entity and the value(s) on which to filter.

An example *classFilter* element is included in the example GFL document in Annex B.

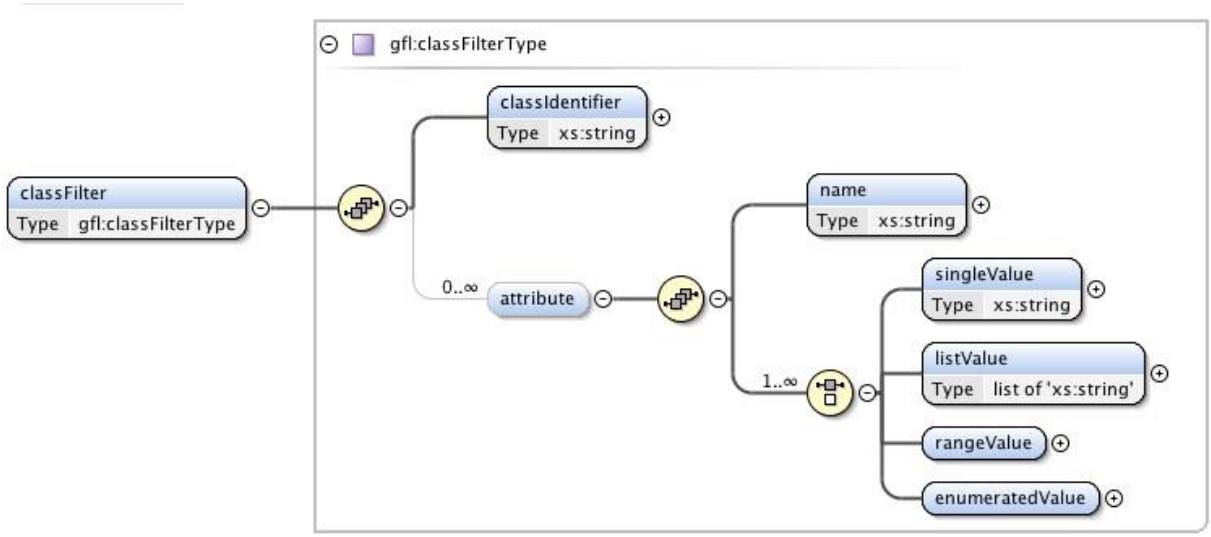

Figure 2: Graphical representation of *classFilter* element

4.2 Class Attribute-Based Filtering

The *classFilter* element shall also be used to define class attribute-based filters by delineating on which *attributes* of a class the gateway shall filter. Attribute filtering may consist of the following:

- Filtering by a single value by using the singleValue element,
- Filtering by a list of values by using the *listValue* element,
- Filtering by a numeric range of values, represented by high and low values of type *double,* or
- Filtering by possible enumerated values, represented by zero or more *enumerationValue* labels.

4.3 Sender- and Receiver-Based Filtering

Every filter statement in GFL contains an *appliesTo* element that consists of one or more *fromSide* elements and one or more *toSide* elements (see Figure 3). These have simple string values that represent a label usable by the gateway to uniquely identify the sender(s) and intended receiver(s) for simulation data being targeted for filtering.

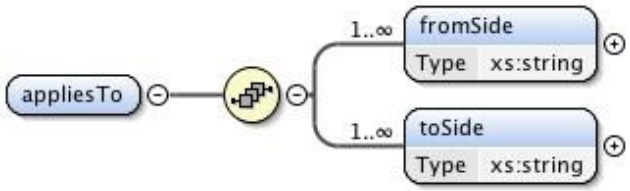

Figure 3: Graphical representation of *appliesTo* element

4.4 Spatial-Based Filtering

Spatial-based filtering is accomplished through the use of the *geoFilter* element (see Figure 4). This element allows a spatial filter to be defined as a *boundingBox*, *boundingCylinder*, or *boundingSphere*, where the volume defined by the chosen geometry shall either filter by inclusion (the default) or exclusion using the *inOrOut* attribute with a value of either “in” or “out.”

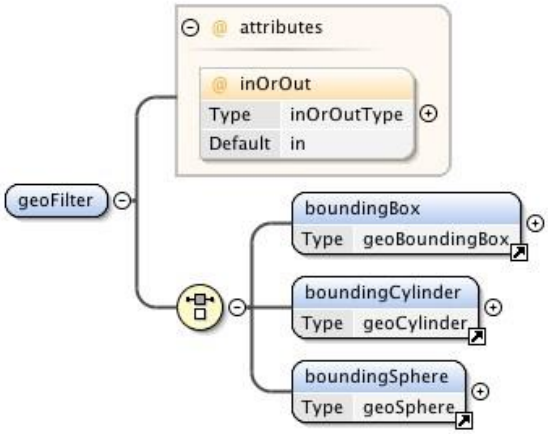

Figure 4: Graphical representation of *geoFilter* element

The *boundingBox* element is defined in a lower and an upper corner (see Figure 5). Each corner has associated *X*, *Y*, and *Z* coordinates. There is also an optional Coordinate Reference System (CRS) label for use in the case where the gateway supports more than one coordinate system.

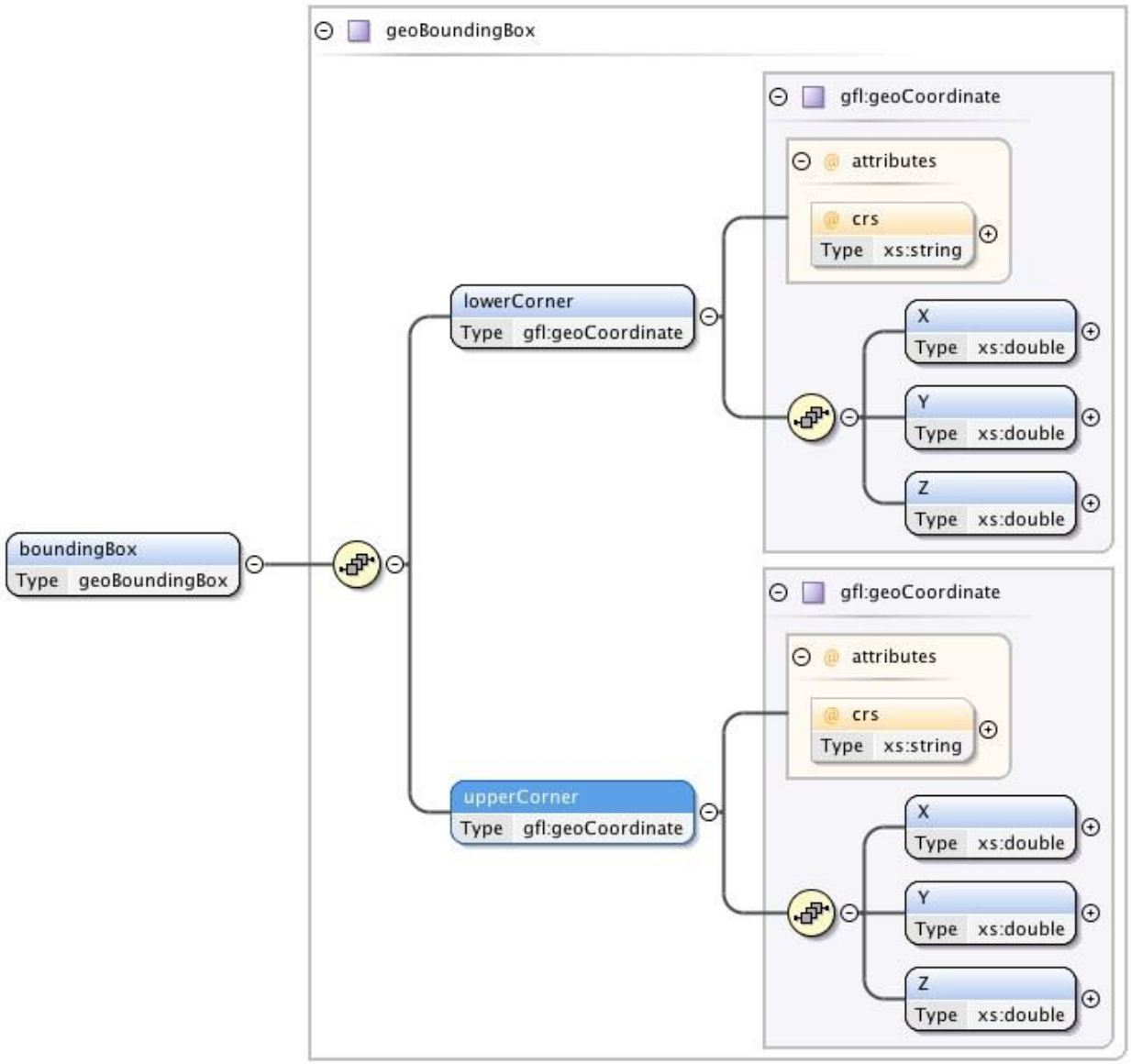

Figure 5: Graphical representation of *boundingBox* element

Also available is a cylindrical filtering volume as represented by the *boundingCylinder* element (see 5 Figure 6). This element consists of a center element that is represented as a *geoCoordinate*; a *radius*, 6 which is represented with a floating point double; and a *height*, also represented by a double, but also 7 including an *upOrDown* attribute, which determines if the cylinder extends “up” (the default) or 8 “down” the Z-axis for the length of the height value. Further, the sphere element itself has a mandatory 9 *lengthUnits* attribute, represented as a string, which indicates the units for height and radius.

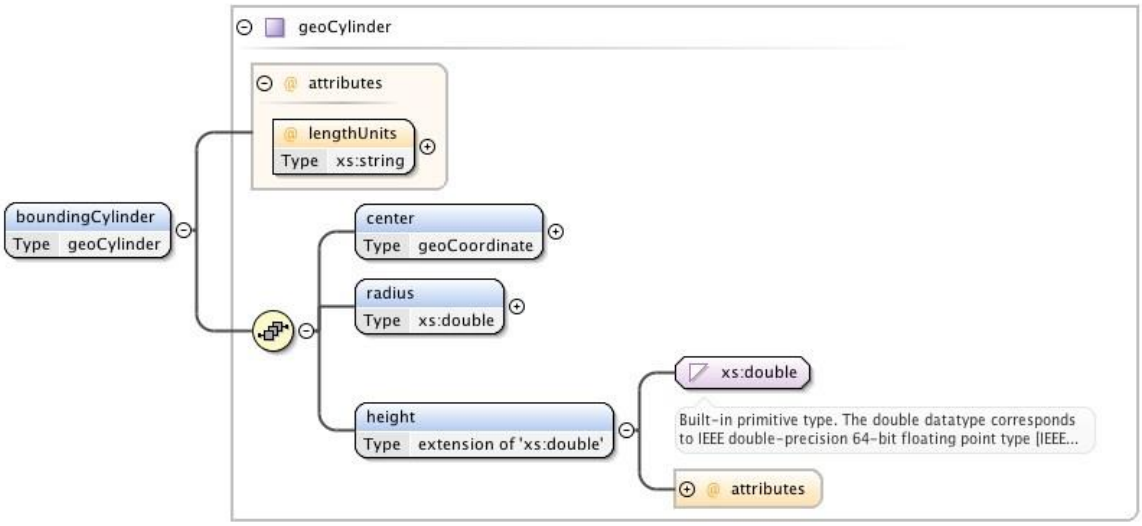

Figure 6: Graphical representation of *boundingCylinder* element

Finally, there is a spherical filtering volume as represented by the *boundingSphere* element (see Figure 7). This element consists of a center element, which is represented as a *geoCoordinate*, and a *radius*, which is represented with a floating point double. Further, the cylinder element itself has a mandatory *lengthUnits* attribute, represented as a string, which indicates the units for *radius*. An example *geoFilter* filter that uses *boundingSphere* element is included in the example GFL document in Annex B.

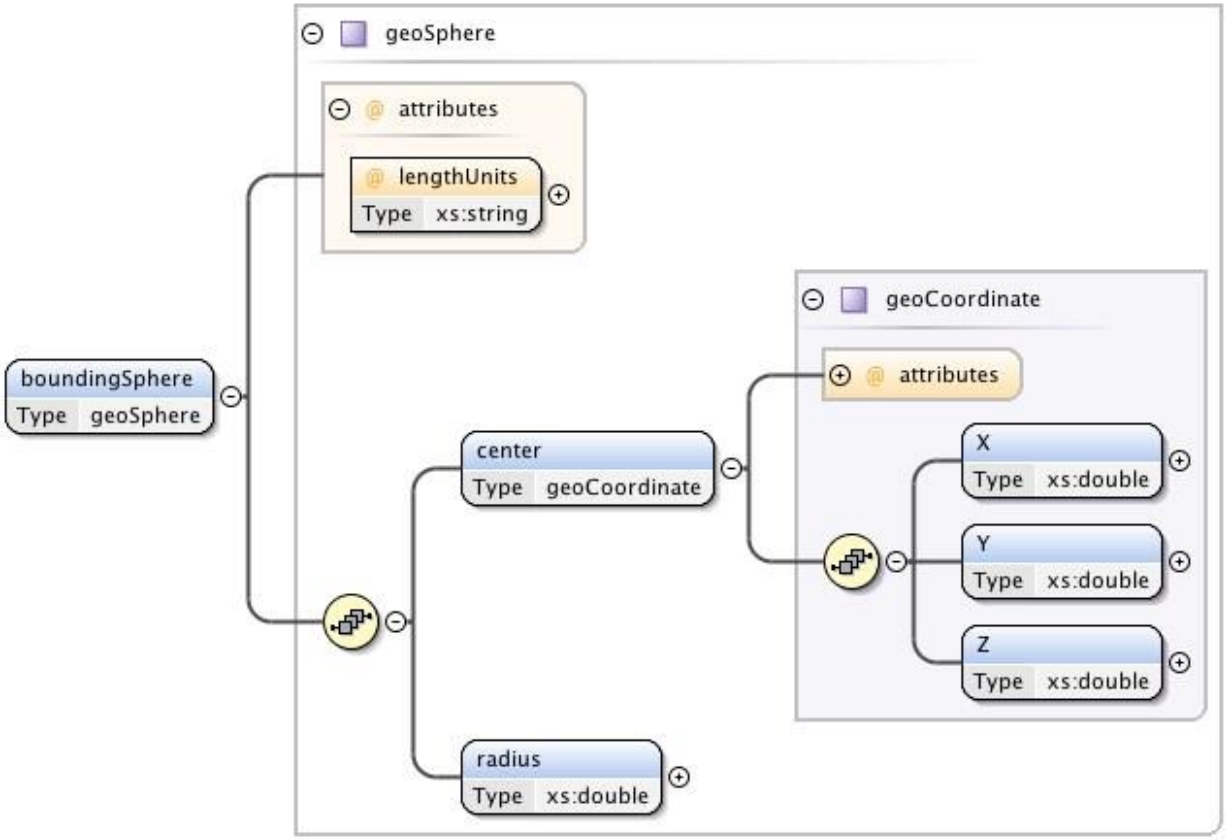

Figure 7: Graphical representation of *boundingSphere* element

In addition to spatial filtering on volumes set at fixed geospatial coordinates, GFL also supports filtering on spatial volumes centered on simulation entities. This is accomplished using the *relativeGeoFilter* element (see Figure 8). This element consists of an *instanceIdentifier*, used to uniquely identify the targeted entity, and an *enclosingVolume*, which defines the spatial volume centered on the entity. As with the previously covered fixed spatial filter types, cylinder, sphere, and box type volumes are supported. For box filters, distance along each axis is provided, with the understanding that the entity position bisects each line segment described. The cylinder filter requires a *radius*, centered on the entity, and a *height*, half above, half below the entity position. The sphere filter is the simplest, only requiring a *radius*, assuming a center at the entity's position.

An example *relativeGeoFilter* filter that uses *cylinder* as the enclosing volume is included in the example GFL document in Annex B.

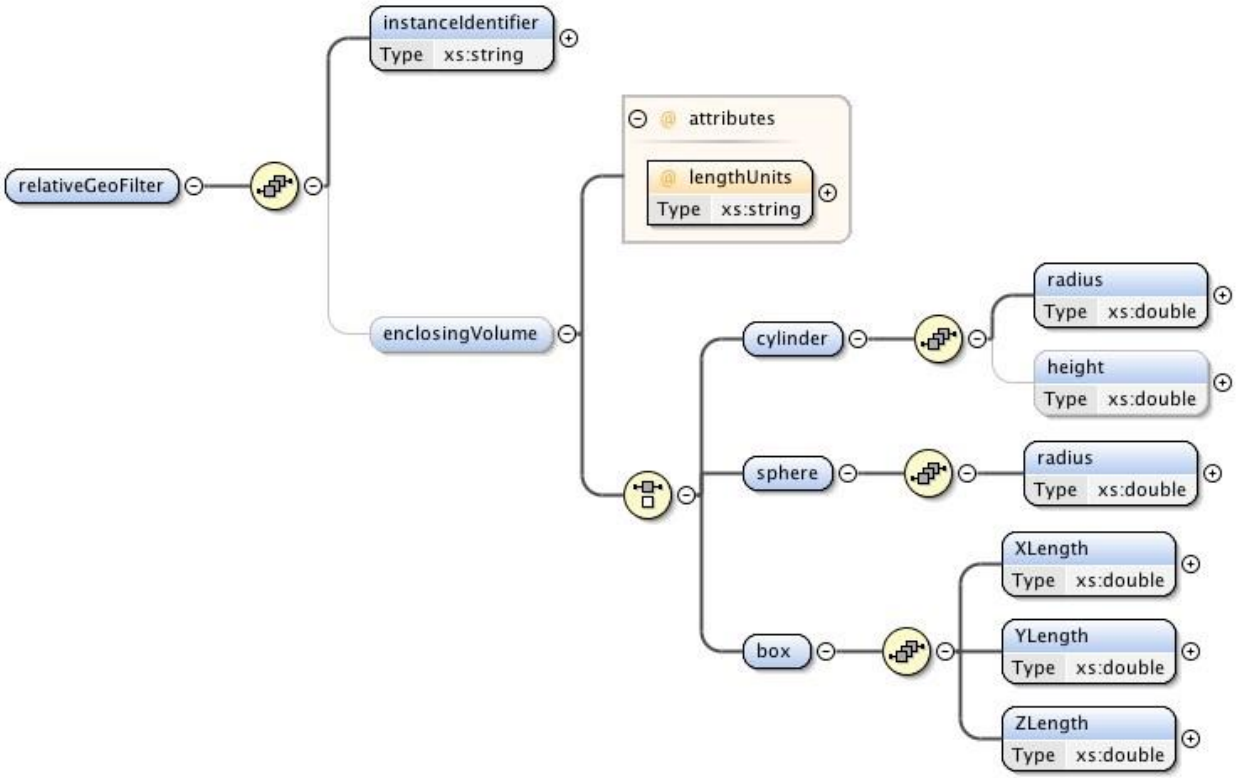

Figure 8: Graphical representation of *relativeGeoFilter* element

4.5 Statements

The core of the GFL specification consists of filter statements. The previously described sender, receiver, entity, and spatial filtering components are combined with actions and optional logic operators to form these filter statements. The basic structure of a GFL filter is shown in Figure 9. The rest of this section describes the actions and logic operations available for use with filters.

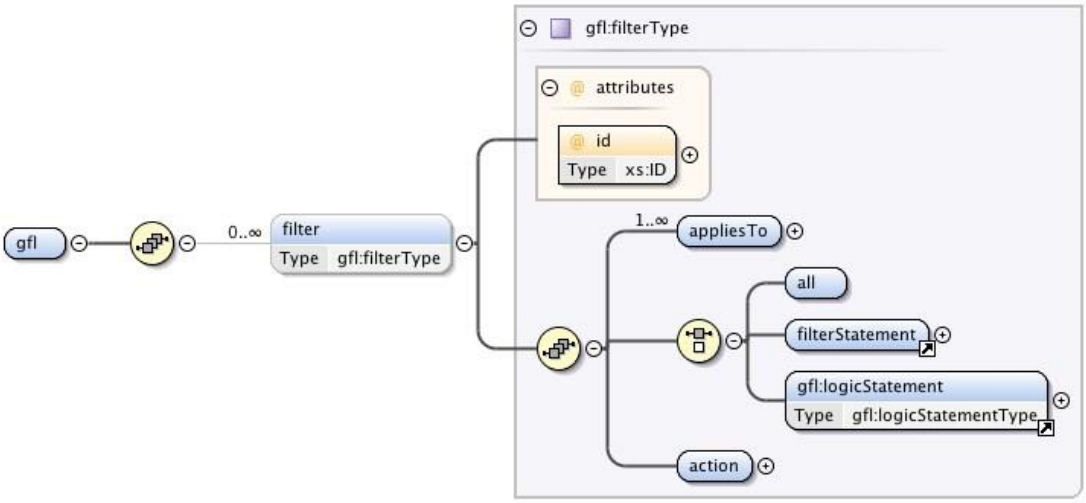

Figure 9: Graphical representation of *gfl* element

Each GFL filter element contains a unique *id* element, the aforementioned *appliesTo* element, an *action* (described below), and one of three filter types: *all*, *filterStatement*, or *logicStatement*. The *all* option is an empty element that instructs the gateway to apply the filter to everything from the *fromSide*(s) in the *appliesTo* element. The *filterStatement* element simply wraps one of the three filter elements seen previously: *geoFilter*, *relativeGeoFilter, and classFilter* (see Figure 10).

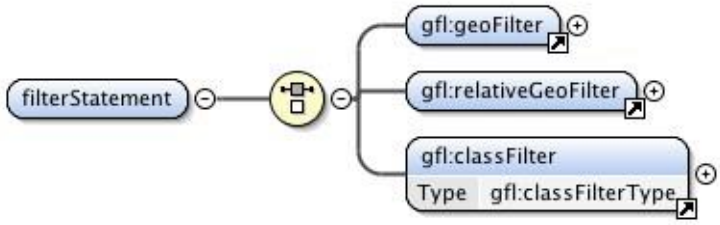

Figure 10: Graphical representation of *filterStatement* element

Each filter element contains an *action* element, which determines what action the gateway shall take in response to the filter conditions being met (see Figure 11). Currently, GFL supports dropping the message or forwarding it to the designated *toSide* target(s) as specified in the aforementioned *appliesTo* element. Further, if the specified action is to forward the message, an optional rate may be specified. This rate is represented using the *value* element and an associated *timeUnits* element.

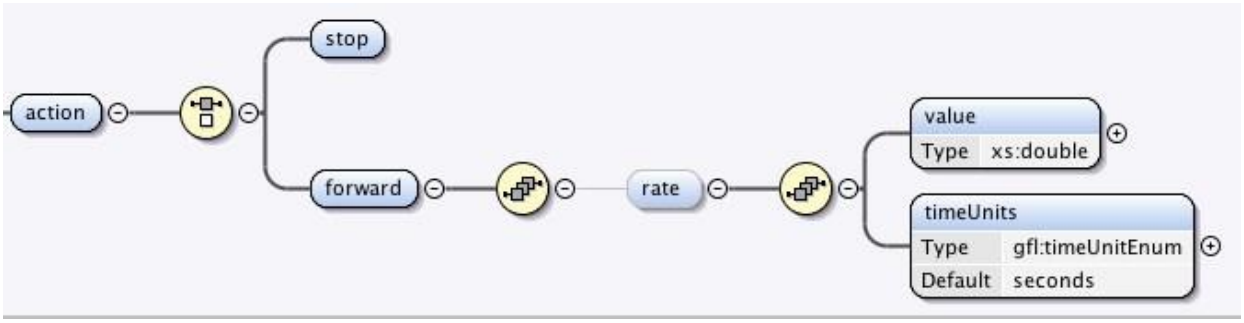

Figure 11: Graphical representation of *action* element

While simple filter statements may consist of either a single spatial or entity filter, GFL also supports more complex logic statements (see Figure 12). These complex logic statements may utilize the *and*, *or*, and *xor* (exclusive or) logic operators, and their negations, as well as arbitrary nesting of parentheses.

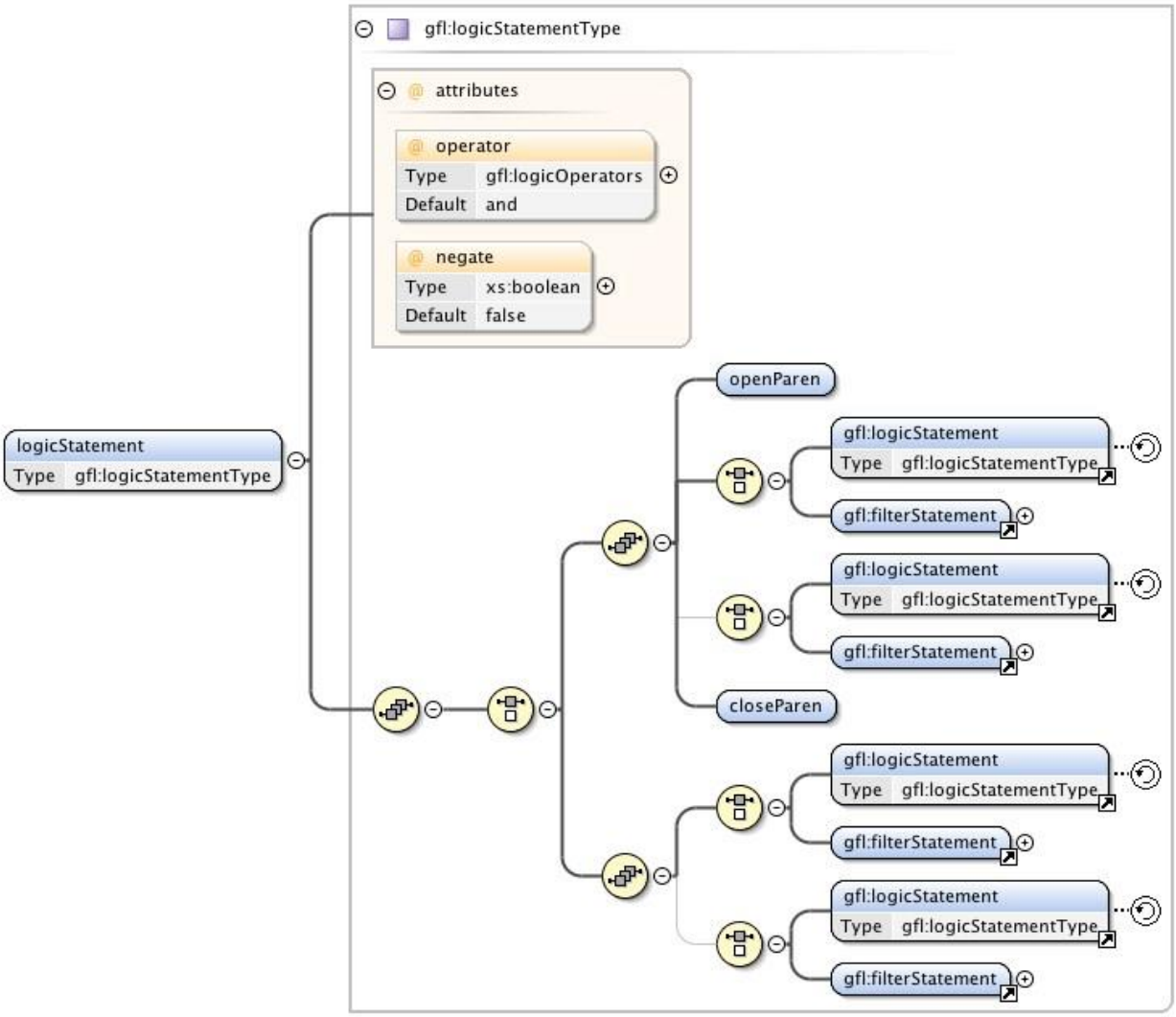

Figure 12: Graphical representation of *logicStatement* element

Annex A GFL Schema (Normative)

The *GFL.xsd* file defines the XML schema for Gateway Filtering Language (GFL). Readers should consult Section 4 (GFL Definition) for descriptions of all GFL components.

This file is a normative part of the specification. The files associated with Annexes A and B are stored in a Zip archive that can be downloaded from this URL: http://www.sisostds.org/Schemas.aspx.

Annex B Example of GFL Document (Informative)

The *GFL_example.xml* file is a valid Gateway Filtering Language (GFL) XML document demonstrating the use of filtering on fixed spatial volume (*geoFilter*), class-attribute value (*classFilter*), and relative spatial volume around an entity (*relativeGeoFilter*).

It is an informative part of the specification. The files associated with Annexes A and B are stored in a Zip archive that can be downloaded from this URL: http://www.sisostds.org/Schemas.aspx.

Annex C Bibliography (Informative)

The following documents were used for generating this standard.

[C1] Live-Virtual-Constructive Architecture Roadmap (LVCAR) Final Report, Institute for Defense Analyses, September 2008.

[C2] Live-Virtual-Constructive Architecture Roadmap Implementation, Common Gateways and Bridges – Gateways Configuration Model Report, JHU/APL NSAD-R-2011-034, April 2011.

[C3] Live-Virtual-Constructive Architecture Roadmap Implementation, Common Gateways and Bridges Characterization Report, JHU/APL NSAD-R-2010-031, May 2010.

[C4] Live-Virtual-Constructive Architecture Roadmap Implementation, SDEM Mapping Language (SML) Specification, JHU/APL NSAD-R-2011-220, August 2011.

[C5] The Open Geographic Information System Filter Encoding Standard (http://www.opengeospatial.org/standards/filter).