Report of the

Tactical Situation (TACSIT) Controller Interface (TCI) for Combat Management Services (CMS) Systems

Joint Revised Submission

Finalization Task Force 2.0

to the

Object Management Group (OMG)

Technical Committee

~~19~~ 18 ~~April~~ June 2012

Document Number:	**dtc/2012-~~04-12~~06-22**
Task Force Chair(s):	**Matt Wilson**

Specification

Revised specification (clean):	**dtc/2012-~~04-07~~06-20**
Revised specification (change-bar):	**dtc/2012-06-21**
~~Answering OMG RFP:~~	**~~c4i/07-06-16~~**

Accompanying documents

Inventory:	**dtc/2012-~~04~~06-~~04~~19**	**Non-normative**
~~Updated Specification w/ Change Bars~~	**~~dtc/2012-04-05~~**	
Updated UML PIM for TCI	**dtc/2012-04-10**	**Normative**
Updated UML Java PSM for TCI	**dtc/2012-04-09**	**Normative**
Updated Java Code PSM for TCI	**dtc/2012-04-15**	**Normative**

Formatted: Font: 11 pt

Formatted: Font: 11 pt

Formatted: Font: 11 pt

Formatted: Font: 11 pt

Formatted: Font: 11 pt

Template: omg/~~09~~12-~~06~~03-01

Table of Contents

Summary of TCI for CMS Systems FTF 2.0 Activities

Formation

- Chartered By: C4I Domain Task Force / OMG Domain Technical Committee
- On: September 23, 2011 – Salt Lake City, UT
- Comments Due Date: February 02, 2012
- Report Due Date: March 26, 2012
- Report Deadline: April 30, 2012

Revision / Finalization Task Force Membership

Member	Organization	Status
Matt Wilson	SimVentions	Charter (Chair)
Ron Townsen	GDAIS	Charter
Simon Mettrick	BAE Systems	Charter
Joel Russ	Raytheon	Charter
Ken Rode	Gallium	Charter
Gunther Sablon	Luciad	Charter
Charlie Fudge	NSWCDD	Charter
Dario Laccarino	Selex SI	Charter

Issue Disposition:

Disposition	Number of Occurrences	Meaning of Disposition
Resolved	~~16~~15	The RTF/FTF agreed that there is a problem that needs fixing, and has proposed a resolution (which may or may not agree with any resolution the issue submitter proposed)
Deferred	~~3~~4	The RTF/FTF agrees that there is a problem that needs fixing, but did not agree on a resolution and deferred its resolution to a future RTF/FTF.
Transferred	0	The RTF/FTF decided that the issue report relates to another specification, and recommends that it be transferred to the relevant RTF.
Closed, no change		The RTF/FTF decided that the issue report does not, in fact, identify a problem with this (or any other) OMG specification.
Closed, Out of Scope		The RTF/FTF decided that the issue report is an enhancement request, and therefore out of scope for this or any future FTF or RTF working on this major version of the specification. The RTF/FTF has closed the issue without making any specification changes, but RFP or RFC submission teams may like to consider these enhancement requests when proposing future new major versions of the specification.
Duplicate or merged	13	This issue is either an exact duplicate of another issue, or very closely related to another issue: see that issue for disposition. For this document, the issues are listed together in the same resolution description however they are counted separately as occurances.

Voting Record:

Poll No.	Closing date	Issues included
1	December 13, 2012	Issue numbers had not been generated at the time of voting, however all issues in this report were discussed and voted upon.
2	February 16, 2012	OMG Issue No: 15603 OMG Issue No: 15793 OMG Issue No: 15799 OMG Issue No: 17050, 17000 (duplicate issues) OMG Issue No: 17051, 17001 (duplicate issues) OMG Issue No: 17052, 17002 (duplicate issues) OMG Issue No: 17003 OMG Issue No: 17004 OMG Issue No: 17053, 17005 (duplicate issues) OMG Issue No: 17006 OMG Issue No: 17054, 17008 (duplicate issues) OMG Issue No: 17055, 17007 (duplicate issues) OMG Issue No: 17056, 17009 (duplicate issues) OMG Issue No: 17057, 17010 (duplicate issues) OMG Issue No: 17058, 17011 (duplicate issues) OMG Issue No: 17059, 17012 (duplicate issues) OMG Issue No: 17060, 17013 (duplicate issues) OMG Issue No: 17061, 17014 (duplicate issues) OMG Issue No: 17062, 17015 (duplicate issues)
3	March 19, 2012	OMG Issue No: 15603 OMG Issue No: 15799 OMG Issue No: 17050, 17000 (duplicate issues) OMG Issue No: 17051, 17001 (duplicate issues) OMG Issue No: 17052, 17002 (duplicate issues) OMG Issue No: 17003 OMG Issue No: 17004 OMG Issue No: 17053, 17005 (duplicate issues) OMG Issue No: 17054, 17008 (duplicate issues) OMG Issue No: 17055, 17007 (duplicate issues) OMG Issue No: 17056, 17009 (duplicate issues)
4	April 25, 2012	OMG Issue No: 15603 OMG Issue No: 15793 OMG Issue No: 15799 OMG Issue No: 17050 OMG Issue No: 17051 OMG Issue No: 17052 OMG Issue No: 17003 OMG Issue No: 17004

		OMG Issue No: 17053 OMG Issue No: 17054 OMG Issue No: 17055 OMG Issue No: 17056 OMG Issue No: 17061

Poll 1 was conducted at the OMG Technical Meeting in Santa Clara, CA. All issues were discussed and addressed during that meeting. The votes at the meeting reflect the final resolutions in this report.
Poll 2 was conducted via email prior to completion of this effort to provide submitters one final opportunity to weigh in with any final changes.
Poll 3 was conducted via email and in person at the OMG technical meeting in Reston, VA. This poll resolves problems identified in our initial report submission.
Poll 4 was conducted via email. This poll resolves the remaining problems identified by the Architecture Board during the OMG meeting in Reston, VA.

Voter	Vote in poll 1	Vote in poll 2	Vote in poll 3	Vote in poll 4
Matt Wilson	Yes	Yes	Yes	Yes
Ron Townsen	Yes	Yes	Yes	Yes
Simon Mettrick	Yes	Yes	Yes	Did not vote
Joel Russ	Did not vote	Did not vote	Did not vote	Did not vote
Ken Rode	Did not vote	Did not vote	Did not vote	Did not vote
Gunther Sablon	Did not vote	Did not vote	Did not vote	Did not vote
Charlie Fudge	Yes	Yes	Yes	Yes
Dario Laccarino	Yes	Did not vote	Did not vote	Did not vote

Summary of Changes Made

The TCI for CMS Systems made changes that:

1. Address carry-over issues from FTF 1.
2. Added features that improved the usability of the profiles and improved their suitability to original purpose;
3. Corrected features that impeded implementation and/or did not properly match the language being modeled;
4. Increased readability, clarity, correctness, and precision of the text.

Here is the FTF's categorization of the resolutions applied to the specification according to their impact on the clarity and precision of the specification:

Extent of Change	Number of Issues	OMG Issue Numbers
Critical/Urgent - Fixed problems with normative parts of the specification which prevented implementation work	**0**	
Significant - Fixed problems with normative parts of the specification that raised concern about implementability	**8**	**15793,** **15799,** **17004,** **17053/17005,** **17006,** **17054/17008**
Minor - Fixed minor problems with normative parts of the specification	**12**	**15603** **17050/17000,** **17051/17001,** **17052/17002,** **17003,** **17056/17009,** **17060/17013**
Support Text -Changes to descriptive, explanatory, or supporting material.	**12**	**17055/17007,** **17057/17010,** **17058/17011,** **17059/17012,** **17061/17014,** **17062/17015**

Disposition: Resolved

OMG Issue No: 15603

Title: Normative PIM and MagicDraw file

Source:

Steve Cook - Steve.Cook@microsoft.com

Severity: Minor

Summary:

In the AB today, when the TACSIT submission was presented, I noted that the normative PIM is expressed as a MagicDraw file, not a pure OMG-compliant XMI file. The normative PIM should be an OMG-compliant file and the MagicDraw file should be a convenience artifact.

Resolution:

Generation of UML PIM in pure OMG XMI.

Discussion:

The generated XMI is compliant with the OMG standard profile, however it still contains references to the MagicDraw profile.

There is no text to revise. The change is made in the UML PIM.

Disposition: Resolved

Disposition: Deferred

OMG Issue No: 15793

Title: Dealing with error at PIM level

Source:

Hugues Vincent <Hugues Vincent@thalesgroup.com>

Severity: Significant

Summary:

Error should be dealt with at the PIM level for each method.

Resolution:

Chapter/Section: all
Page 29. convertScreenPos / convertGeoPos…
Pg 34: getViewport
Need to resolve how to handle Exception case for GeodeticPosition.
Options: PIM level Exception handling vs. return a "special" value for invalid.

If we are to "throw" an exception, we need to find out how to represent this at the PIM level.

We were unable to resolve this issue during FTF period.

Discussion:

Throwing an exception seems like the correct course of action here, provided it can be represented appropriately in the PIM. Unfortunately, the FTF was unable to come to a solution during this period.

There has been no change recommended related to this issue since the first FTF.

Disposition: Deferred

Disposition: Deferred

OMG Issue No: 15799

Title: Selection by track number

Source:

Hugues Vincent <Hugues Vincent@thalesgroup.com>

Severity: Significant

Summary:

Chapter/Section: 7.1
Nature: Enhancement

The ways to select by track number as well as to select within geographic geometry seem to me inefficient. Indeed, in that way there are 2 accesses to the TACSIT server(s): one to query on one to effectively select when an interface proposing to select from a an interface with a query as parameter would be more efficient since it implied just one access call to the TACSIT server.

Resolution:

No Change.

Discussion:

The issue is partly due to the complexity of TN. There are multiple possible types of "TN" for any entity. We could add a convenience method of "addToSelection(Query, SelectionType)" which would prevent the need to go to the QueryManager in the middle of every one of these types of transactions. However this would create an unnecessary association between the SelectionManager and the QueryManager, which dilutes the purpose of the SelectionManager. The TacsitController already provides a single point of access for the SelectionManager and QueryManager, so associating both objects to a component which performs a query and adds it to the selection would already be simple during implementation. The addition of 1 line of code to retrieve the query results seems like a small price to pay to keep the cohesion of the SelectionManager concept.

Revised Text:

No change recommended at this time. This may be revisited in a future FTF/RTF.

Disposition: Deferred

Disposition: Resolved

OMG Issue No: 17050, 17000

Title: SelectionMethodology not implemented.

Source:

Matt Wilson - mwilson@simventions.com

Severity: Minor

Summary:

This may be a translation error from the platform specific model, but this must be implemented. It would best be implemented as an enumeration.

Resolution:

Added the following files in the Java PSM:
org.omg.tacsit.controller.SelectionMethodology

Updated the following files in the Java PSM:
org.omg.tacsit.controller.TacsitController

- Changed getSelectionMethodology() to return the enumeration value type.

Added the following files in the C++ PSM:
SelectionMethodology.h

Updated the following files in the C++ PSM:
TacsitController.h

- Changed getSelectionMethodology() to return the enumeration value type.

Discussion:

This is already addressed in paragraph 7.1.1.16 with the Enumeration Class called SelectionMethodology. Its two valid values are “ViewportDependent” and “ViewportIndependent”

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17051, 17001

Title: SelectionEvent takes a single Entity parameter

Source:

Matt Wilson - mwilson@simventions.com

Severity: Minor

Summary:

A translation error occurred from the java PSM to the specification source files. This should be a Collection or List of Entity objects, rather than a single Entity. The affected methods are:

- Constructor
- getEntities

To ensure the integrity of the event object, the internally stored Collection or List should be unmodifiable. This can be easily achieved by copying the List and invoking Collections.unmodifiableList(source).

Resolution:

Update the PSM class methods to take/return a List<Entity> respectively.

Updated the following files in the Java PSM:
org.omg.tacsit.controller.SelectionEvent

- Changed the constructor to take a List<Entity>.

Updated the following files in the C++ PSM:
SelectionEvent.h

- Changed the constructor to take a vector of Entity objects.

Revised Text:

There is no modification necessary to the text of the specification. Java PSM/C++ files only.

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17052, 17002

Title: SelectionManager selection methods take a single Entity parameter

Source:

Matt Wilson - mwilson@simventions.com

Severity: Minor

Summary:

A translation error occurred from the java PSM to the specification source files. This should be a Collection or List of Entity objects, rather than a single entity. The affected methods are:

- setSelection
- addToSelection
- removeFromSelection

Resolution:

Update the Java PSM class methods to take a List<Entity>.

Updated the following files in the Java PSM:
org.omg.tacsit.controller.SelectionManager

- Changed *setSelection* to accept a List<Entity>.
- Changed *addToSelection* to accept a List<Entity>.
- Changed removeFromSelection to accept a List<Entity>

Updated the following files in the C++ PSM:
SelectionManager.h

- Changed *setSelection* to accept a vector<Entity>.
- Changed *addToSelection* to accept a vector<Entity>.
- Changed removeFromSelection to accept a vector<Entity>

Revised Text:

There is no modification necessary to the text of the specification. Java PSM files only.

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17003

Title: Viewport scaleToPoints takes a single GeodeticPosition parameter

Source:

Matt Wilson - mwilson@simventions.com

Severity: Minor

Summary:

A translation error occurred from the java PSM to the specification source files. scaleToPoints takes only a single GeodeticPosition, whereas it should take a List of GeodeticPositions. In addition, since GeodeticPosition is an interface, it would be appropriate if wildcards were used. The suggested fix is as follows:

```
void scaleToPoints(List<? Extends GeodeticPosition> points,
double margin);
```

Resolution:

Update the Java PSM class method for scaleToPoints to take a List<GeodeticPosition>.

Updated the following files in the Java PSM:
org.omg.tacsit.controller.Viewport

- Changed *scaleToPoints* to accept a List<GeodeticPosition>.

Updated the following files in the C++ PSM:
Viewport.h

- Changed *scaleToPoints* to accept a vector<GeodeticPosition>.

The suggestion to use wildcards is deferred for future resolution; New issue created for this portion of the request - 17240.

Revised Text:

There is no modification necessary to the text of the specification. Java PSM files only.

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17004

Title: Ref. to 6.3.2, Add an *Angle Class to the Standard*

Source:

Matt Wilson - mwilson@simventions.com

Severity: Significant / Enhancement

Summary:

Having an Angle class would be very useful. This would free you from potential errors about worrying whether a double represents degrees or radians. This worry is particularly troublesome, since interface components present angles as degrees, whereas processing is most frequently done in radians. When dealing with unstructured primitive values, it's an easy mistake to make. An Angle class would allow both to operate in whatever unit was most convenient by providing getters for each unit type.

In addition to unit safety checking, it also makes property editors (for example, for tables and forms) far simpler to construct. The simplest way to specify an editor or renderer is to set the default cell editor or renderer. This facilitates minimal coupling between UI implementations and the data being presented. If the data is stored and presented as doubles, this approach is not possible because an editor that works for a unit of distance is probably not appropriate to also work for a unit of angle. Since GeodeticPosition contains both, this problem is likely to come up very frequently. The problem can be worked around by explicitly setting a particular column's editor or renderer, but this strongly couples the display capability to the ordering of the displayed TableModel.

The following properties should be changed to an Angle:

- Rectangle
  - *Orientation* property
- Geodetic Position
  - *Latitude* property
  - *Longitude* property
- ViewEyeProperties
  - *Orientation* property

Resolution:

An Angle class has been added to the specification and referenced in the places suggested.

Added the following interface in the PIM/PSM:
org.omg.tacsit.geometry.Angle

Updated the following classes in the PIM/PSM:
org.omg.tacsit.geometry.Rectangle

- Changed *getOrientation* to return an Angle

org.omg.tacsit.geometry.GeodeticPosition

- Changed *getLongitude* to return an Angle
- Changed *getLatitude* to return an Angle

org.omg.tacsit.geometry.ViewEyeProperties

- Changed *getOrientation* to return an Angle
- Changed *setOrientation* to accept an Angle parameter

Revised Text:

Table of Contents modified to include section 7.3.1.7 on the Angle class.

Paragraph 7.1.1.9.2. ViewEyeProperties.orientation
modified table to reflect use of Angle

Paragraph 7.3.1.2.1 GeodeticPosition.getLatitude()
modified table to reflect change from double to Angle

Paragraph 7.3.1.2.2 GeodeticPosition.getLongitude()
modified table to reflect change from double to Angle

Paragraph 7.3.1.6.0 Rectangle.getOrientation()
modified table to reflect change from double to Angle

Figure 7.34 modified to reflect the changes seen
In the query interfaces

Paragraph 7.3.1.7 added to describe the Angle interface.
Added Figure 7.41 depicting the Angle interface.
Added Paragraph 7.3.1.7.1 Angle.getDegrees()
Added Paragraph 7.3.1.7.2 Angle.getRadians()
Added Paragraph 7.3.1.7.3 Angle.add()
Added Paragraph 7.3.1.7.4 Angle.subtract()

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17053, 17005

Title: Add a Distance Class to the Standard

Source:

Matt Wilson - mwilson@simventions.com

Severity: Significant / Enhancement

Summary:

Using this as an object rather than a plain double value is a cheap way to avoid costly errors. NASA lost a $125 million Mars orbiter because one team was using imperial units instead of metric. Representing distance as an object makes it glaringly obvious in the code which calculations are using which units.

Consider the following method of *Rectangle*:

```
double getHeight();
```

Embedded in the comments of the interface contract is that "getHeight()" returns a double in meters. But that's not at all obvious in client code that uses getHeight(), especially if it's 2 or 3 levels removed from the place it is called.
Instead, if it were changed to:

```
Distance getHeight();
```

Distance could have methods "double getMeters()", and "double getFeet()", that make it immediately obvious what the returned value represents.

If adopted, the following changes should be made:

- GeodeticPosition
  - Change *altitude* property from double to Distance
- Rectangle
  - Change getHeight() to return Distance
  - Change getWidth() to return Distance
- Circle
  - Change getRadius() to return Distance
- ViewEyeProperties
  - Change get/setRangeScale to use Distance
- Viewport
  - Change scaleToPoints(points, margin) margin parameter to Distance

Resolution:

A Distance class has been added to the specification and referenced in the places suggested.

Added the following interface in the PIM/PSM:
org.omg.tacsit.geometry.Distance

Updated the following classes in the PIM/PSM:
org.omg.tacsit.geometry.GeodeticPosition

- Changed *getAltitude* to return a Distance

org.omg.tacsit.geometry.Rectangle

- Changed *getHeight* to return a Distance
- Changed *getWidth* to return a Distance

org.omg.tacsit.geometry.Circle

- Changed *getRadius* to return a Distance

org.omg.tacsit.geometry.ViewEyeProperties

- Changed *getRangeScale* to return a Distance
- Changed *setRangeScale* to accept a Distance parameter

org.omg.tacsit.controller.Viewport

- Changed *scaleToPoints* to accept a Distance Parameter to return a Distance

Revised Text:

Paragraph 7.1.1.9.4 ViewEyeProperties.rangeScale
modified table and graphic to reflect change to use Distance

Paragraph 7.1.1.10.7 Viewport.scaleToPoints()
modified table and graphic to reflect change of the “margin” parameter from double to Distance

Paragraph 7.3.1.2.3 GeodeticPosition.getAltitude()
modified table and graphic to reflect change from double to Distance

Paragraph 7.3.1.5 Circle.getRadius()
modified table and graphic to reflect change from double to Distance

Paragraph 7.3.1.6.1Rectangle. getHeight()
modified table and graphic to reflect change from double to Distance
Paragraph 7.3.1.6.3Rectangle. getWidth()
modified table and graphic to reflect change from double to Distance

Paragraph 7.3.1.8 added to describe the Distance interface.
Added Figure 7.42 depicting the Distance interface.
Added Paragraph 7.3.1.8.1 Distance.getFeet()
Added Paragraph 7.3.1.8.2 Distance.getYards()
Added Paragraph 7.3.1.8.3 Distance.getMiles()
Added Paragraph 7.3.1.8.4 Distance.getMeters()
Added Paragraph 7.3.1.8.5 Distance.getKilometers()
Added Paragraph 7.3.1.8.6 Distance.getNauticalMiles()
Added Paragraph 7.3.1.8.7 Distance.add()
Added Paragraph 7.3.1.8.8 Distance.subtract()
Added Paragraph 7.3.1.8.9 Distance.multipliedBy()
Added Paragraph 7.3.1.8.10 Distance.dividedBy(scalarValue)
Added Paragraph 7.3.1.8.11 Distance.dividedBy(distance)

Disposition: Resolved

Disposition: Deferred

OMG Issue No: 17006

Title: Add an *EntityRepository Concept / Class to the Standard*

Source:

Matt Wilson - mwilson@simventions.com

Severity: Significant / Enhancement

Summary:

The definition of how Entity objects are related to and move around through a TacsitController, ViewportManager, and Viewport seems to be vague by design. Presumably, vendors have different implementations that make creating a cohesive abstraction difficult. It is a worthy goal, however – if it can be achieved. Separating the subsystems into more clearly defined subcomponents (Track providing, and track viewing) promotes encapsulation and division of responsibility. It also allows simplified compliance to the Tacsit specification for implementations which really provide only a single subsystem.

Consider NASA's Worldwind. It is a general geographic map package. It provides little facility for getting track information from remote sources. The entire entity store concept is foreign to Worldwind; the reference implementation is purely incidental. Worldwind could work with *any* entity store, provided it had a well defined interface.

The reference implementation contains a sample of what an EntityRepository would look like. The definition is given here (sans comments):

```
public interface EntityRepository<E extends Entity> extends
QueryManager
{
  public Iterator<E> getEntities();
  public void addRepositoryListener(RepositoryListener listener);
  public void removeRepositoryListener(RepositoryListener
listener);
}
```

Repository Listener would be defined as follows:

```
public interface RepositoryListener
{
  public void entitiesAdded(RepositoryChangeEvent event);
  public void entitiesRemoved(RepositoryChangeEvent event);
```

```
  public void entitiesUpdated(RepositoryChangeEvent event);
  public void entitiesCleared(RepositoryChangeEvent event);
}
```

This interface description should be general enough to allow for abstraction of existing implementations.

An area of common concern for this type of implementation is update speed. If data is being imported at a very fast speed, the user interface may not be able to handle the volume of updates. For instance, consider the common scenario that each update received is an array of structs containing track data that contains all the values for the track in the system. If a repository were to fire an "entitiesUpdated" event every time a property was copied into the entity, that would potentially be many new objects created:

O = (number of objects created to fire the event) *(number of properties set) * (number of tracks)

Each track will probably have 10 properties, 2 objects will be needed, and there may be roughly 2000 tracks in the system. That's 40,000 objects *every* time a new track block comes in – excluding anything that happens once it gets to the UI.

However, this problem can be alleviated by deviating slightly from the standard Listener implementation. Instead of broadcasting to a RepositoryListener any time a track is changed, the EntityRepository can poll its Entity objects for changes based on an update rate. It would then fire a summary of those changes to its associated listeners, effectively coalescing all the redundant events into a single one. This drastically cuts down on the number of objects created to:

O = (number of objects created to fire the event)

Each entity will need to be "polled" every update, but the polling consists simply of checking a time value against a known value. In the example above, 2000 lastModified values would have to be compared.

This allows the update interval to be easily scaled to the fastest possible rate that still gives the desired responsiveness in the user interface. The key attribute is this: from the perspective of the client of the entity repository interface, it behaves exactly as per a "normal" Listening pattern. The complexity is masked by the interface and doesn't creep into client code.

An example of how such an implementation would work is included in the reference implementation, *PolledEntityRepository*.

.

Discussion:

This concept has merit but is too involved to introduce at this phase of the specification life cycle. This issue is deferred to a future revision of the specification.

Revised Text:

No Change.

Disposition: Deferred

Disposition: Resolved

OMG Issue No: 17054, 17008

Title: Add method to Entity interface to "getReferencePoint"

Source:

Matt Wilson - mwilson@simventions.com

Severity: Significant/ Enhancement

Summary:

There is no way to get even a general understanding about where an Entity exists at a general interface level.

Resolution:

Define a new method in the entity interface:

```
GeodeticPosition getReferencePoint();
```

This is logically consistent across any entity type. An entity could still consist of multiple points (e.g., isPointEntity() would return false), and still return a center point. This keeps the conceptual integrity of the entity interface, but expands its utility.

One such concrete example of the utility: Creating a user interface component for scaling a Viewport to a set of points. The most common use case is for those points to be entity positions. The QueryManager can be used to retrieve a list of entities, but from that point on you have to dive into implementation specific details to convert the Entities to a position to choose from.

~~Updated the following classes in the PIM/PSM:~~
~~*org.omg.tacsit.controller.Entity*~~

- ~~Added method *getReferencePoint()*, which returns a Distance~~

Revised Text:

Part of this resolution is the editorial fix to clean up the "extra" tables in 7.1.1.1.3.

Update the model to reflect the new method in the Entity interface to add getReferencePoint() with a return type of GeodeticPosition.

Add a new section to the specification and update the graphic in paragraph 7.1.1.1 on figure 7.11to reflect the new method.

7.1.1.1.5 getReferencePoint

Returns the GeodeticPosition to be used as the main point of reference for this entity.

Type	GeodeticPosition
Visibility	public
Is Abstract	false
Parameter	

Updated the following classes in the PIM/PSM:
org.omg.tacsit.controller.Entity

- Added method *getReferencePoint*(), which returns a Distance

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17055, 17007

Title: Modify the comments in ContainQuery

Source:

Matt Wilson - mwilson@simventions.com

Severity: Support Text

Summary: The comments of this class reference state this:

```
Containment is determined by using the Entity's Geometry as
an argument to the contains() method on the Geometry
specified by this Query, i.e., this.satisfies( entity ) =
this.getGeometry().contains( entity.getGeometry() )
```

This is not possible, however. The entity class has no "Geometry" in its interface. If it were present, the Geometry interface would not provide enough information to evaluate such a query.

Resolution:

The Geometry class "contains" method accepts a GeodeticPosition, not a Geometry. By adding the "getReferencePoint" method to entity (as proposed by issue 17054) and modifying the comment on this method

Change the comment to a more accurate description of how to use it. For example: *Containment is determined by checking whether or not an entity's reference point (GeodeticPosition) is contained within a specific Geometry, i.e. this.satisfies(entity) = this.getGeometry().contains(entity.getReferencePoint()).*

Revised Text:

Section 7.2.1.1 changed the description / comment for the method to "Containment is determined by checking whether or not an entity's reference point (GeodeticPosition) is contained within a specific Geometry, i.e. this.satisfies(entity) = this.getGeometry().contains(entity.getReferencePoint())."

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17056, 17009

Title: Remove "plural" designation of method call in EntityTypeQuery

Source:

Matt Wilson - mwilson@simventions.com

Severity: Minor

Summary:

Interface specification is pluralized and the figure on 7.2.1.3 shows a return of multiple EntityType objects. This implies that multiple EntityTypes are returned, however, the return value specified in the table and the Java PSM is singular.

```
EntityType getEntityTypes( );
```

Resolution / Discussion:

Change the return type in the table to match the model and update the Java PSM.

The Java PSM will return a List<EntityType> instead of a single EntityType object.

Updated the following classes in the Java PSM:
org.omg.tacsit.controller.TacsitController

- Changed method *getEntityTypes*(), to return a List<EntityType>

Updated the following classes in the C++ PSM:
TacsitController.h

- Changed method *getEntityTypes*(), to return a vector<EntityType>

Revised Text:In Paragraph 7.2.1.3.1, modify the table to:

7.2.1.3.1 getEntityTypes

Returns the list of EntityType objects to compare with.

Type	EntityType [1..n]
Visibility	public
IsAbstract	false
Parameter	

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17057, 17010

Title: Fix Comment in IntersectionQuery

Source:

Matt Wilson - mwilson@simventions.com

Severity: Support Text

Summary:

The intersection query has a comment that states:

```
Intersection is determined by using the Entity's Geometry
as an argument to the intersects() method on the Geometry
specified by this Query, i.e., this.satisfies( entity ) =
this.getGeometry().intersects( entity.getGeometry() )
```

This is not possible, however. The entity class has no “Geometry” in its interface. If it were present, the Geometry interface does not provide enough information to evaluate such a query.

Resolution / Discussion:

Change the comment to a more general description of the problem space. For example: *Intersection is determined by checking whether or not one entity partially overlaps another entity.*

Revised Text:

Changed text in 7.2.1.5 to read:
The IntersectionQuery is used to determine if an Entity intersects geometrically with a Geometry. Intersection is determined by checking whether or not one entity partially overlaps another entity

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17058, 17011

Title: Add comments to QueryManager

Source:

Matt Wilson - mwilson@simventions.com

Severity: Minor

Summary:

It is unclear whether or not a null query is permissible.

Resolution / Discussion:

Explicitly state in the interface documentation that a null query should return all Entities in the QueryManager.

Revised Text:

Add sentence to 7.2.1.6 "In the case where a null query is passed to the QueryManager, the QueryManager is to return all Entities in the QueryManager."

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17059, 17012

Title: Revise comments in code/model for clarity in TacsitController.getEntityTypes

Source:

Matt Wilson - mwilson@simventions.com

Severity: Support Text

Summary:

The comment for this method reads:

```
/**
  * Return the Entity Types which are supported by the TACSIT. This will
  * return a list of all Entity Types currently available for Selection and
  * Query by this TACSITController
  *
  * @return
  */
```

The first sentence gives the impression that it will return anything that can possibly be in the TacsitController. The second sentence implies only entities that currently exist in the TacsitController.

Resolution:

It seems likely that the first sentence is correct; the wording of the second sentence should be changed to "This will return a list of all Entity Types available for Selection and Query by this TACSITController".

Revised Text:

Change the comments in paragraph 7.1.1.8.2 and the model to read:

Returns the Entity Types that are supported by the TACSIT. This will return a list of all Entity Types available for Selection and Query by this TACSITController.

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17060, 17013

Title: Revise comments in code/model for clarity in TacsitController

Source:

Matt Wilson - mwilson@simventions.com

Severity: Support Text

Summary:

It is not clear from the comments that the collections returned by getProjections() and getEntityTypes() cannot be changed.

Resolution / Discussion:

The comments should say the collections returned by getProjections and getEntityTypes() are unmodifiable.

Revised Text:

Add text to paragraphs 7.1.1.8.1 and 7.1.1.8.8.
Attempted changes to the collection returned by this method will have no effect on the behavior and functionality of the TacsitController.

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17061, 17014

Title: Add comments to ViewEye

Source:

Matt Wilson - mwilson@simventions.com

Severity: Support Text

Summary:

The comments of the member variables are very descriptive. However, these comments should appear in the Javadoc for the get and set methods for each property. This provides much better integration with IDEs – most especially important for remembering whether the values are in degrees or radians.

Resolution / Discussion:

Copy the comments for member variables ViewEyeProperties to their respective get/set methods.

Revised Text:

Update the PIM model and Java PSM to reflect comments in section 7.1.1.9.
There is no change needed to the specification document.

Updated the following classes in the PIM/PSM:
org.omg.tacsit.controller.ViewEyeProperties

- Updated method documentation for *getProjection*()
- Updated method documentation for *setProjection*()
- Updated method documentation for *getOrientation*()
- Updated method documentation for *setOrientation*()
- Updated method documentation for *getGeoCenter*()
- Updated method documentation for *setGeoCenter*()
- Updated method documentation for *getRangeScale*()
- Updated method documentation for *setRangeScale*()

Disposition: Resolved

Disposition: Resolved

OMG Issue No: 17062, 17015

Title: Add clarification / comment to ViewportManager.getViewports method

Source:

Matt Wilson - mwilson@simventions.com

Severity: Support Text

Summary:

The method getViewports() returns a Collection of Viewports. A user of this class may think modifying the contents of the Collection would modify the state of the ViewportManager.

Resolution / Discussion:

Explicitly state (and enforce) that the returned Collection is unmodifiable.

Revised Text:

Add sentence to section 7.1.1.13.4 stating:
The collection returned by this method is unmodifiable

Disposition: Resolved