



**MUVES-S2 Software: IrReader Application  
Programming Interface and the Ir2En Application for  
use with EnSight Visualization**

**by Robert Gonzalez**

**ARL-TN-0402**

**August 2010**

## **NOTICES**

### **Disclaimers**

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

# **Army Research Laboratory**

Aberdeen Proving Ground, MD 21005

---

**ARL-TN-0402****August 2010**

---

## **MUVES-S2 Software: IrReader Application Programming Interface and the Ir2En Application for use with EnSight Visualization**

**Robert Gonzalez**  
**Survivability/Lethality Analysis Directorate, ARL**

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
<p>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.</p> <p><b>PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</b></p>					
1. REPORT DATE (DD-MM-YYYY) August 2010		2. REPORT TYPE Final		3. DATES COVERED (From - To) July to August 2009	
4. TITLE AND SUBTITLE MUVES-S2 Software: IrReader Application Programming Interface and the Ir2En Application for use with EnSight Visualization				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S) Robert Gonzalez				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-SLB-S Aberdeen Proving Ground, MD 21005				8. PERFORMING ORGANIZATION REPORT NUMBER  ARL-TN-0402	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT This technical note introduces two new MUVES-S2 software products, IrReader and Ir2En for use with the EnSight Visualization tool. It provides access to the intermediate result data created by MUVES-S2 and makes the data available for EnSight to animate the interaction of a threat on a target. This document describes the initial requirements, software design, technical aspects and usage of the IrReader and Ir2En.					
15. SUBJECT TERMS Visualization, MUVES, MUVES-S2, intermediate results, EnSight, vulnerability/lethality					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT  UU	18. NUMBER OF PAGES  30	19a. NAME OF RESPONSIBLE PERSON Robert Gonzalez
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified			19b. TELEPHONE NUMBER (Include area code) (410) 278-6690

---

## Contents

---

<b>List of Figures</b>	<b>iv</b>
<b>Summary</b>	<b>1</b>
<b>1. Introduction</b>	<b>3</b>
<b>2. Methods and Procedures</b>	<b>3</b>
<b>3. Developing the Software</b>	<b>7</b>
<b>4. Using the Application - Ir2En</b>	<b>15</b>
<b>5. Results</b>	<b>16</b>
<b>6. Conclusions</b>	<b>16</b>
<b>7. Bibliography</b>	<b>18</b>
<b>Appendix A. IR file list</b>	<b>19</b>
<b>List of Symbols, Abbreviations, and Acronyms</b>	<b>21</b>
<b>Distribution List</b>	<b>23</b>

---

## List of Figures

---

Figure 1. Generating IR files in the MUVES-S2 GUI.....	4
Figure 2. Header of the MUVES-S2 IR file. ....	5
Figure 3. Tagged Structure of the IR file.....	5
Figure 4. IRFile data structure. ....	9
Figure 5. IRView data structure.....	10
Figure 6. IrShotline data structure. ....	11
Figure 7. IRShotlinethreat data structure.....	12
Figure 8. IRThreatPacket data structure. ....	13
Figure 9. IRTraceDamage data structure.....	13
Figure 10. Data Structures IRSummaryAssessment and IRORCAOperationalRequirement. ....	14
Figure 11. Ir2En interaction.....	15

---

## List of Tables

---

Table 1. Ir2En files. ....	16
----------------------------	----

---

## Summary

---

This report covers a new MUVES-S2<sup>1</sup> application, “Ir2En”, and an application programming interface (API), “IrReader”. These applications provide the capability to extract data from intermediate results (IR) files generated during a MUVES-S2 run. Ir2En generates the output that is used as an input to the EnSight<sup>2</sup> visualization tool. This allows EnSight to make use of the shotline<sup>3</sup> data to animate the simulation of a threat interacting with the target. Visualization of MUVES-S2 results are illustrated in 2-D and 3-D with the help of viewer applications developed by the U.S. Army Research Laboratory, Survivability Lethality Analysis Directorate (SLAD). However, these applications are not able to animate a threat on a target. The Ir2En software enhancement provides for that. The interaction between threat and target can now be visually enhanced through animation.

IR files contain a large amount of valuable information. This project has an untapped potential than just the visualization. It is dependent upon the imagination of the analysts. It was our task to gather this data and store it so that it can be accessible for future use. The IrReader stores all of the intermediate results from a MUVES-S2 run. It was then up to the development team to extract what data is required for this project and then to provide the necessary input files for EnSight. The use of IR files and EnSight was first developed by William Landis and needed to be put under the MUVES-S2 application suite. There is the potential for an additional tool to let a MUVES-S2 user see what the IrReader has stored and collect specific data as needed for whatever analyses are required.

---

<sup>1</sup>MUVES-S2 is not an acronym, but it used to stand for the Modular UNIX<sup>®</sup>-based Vulnerability Estimation Suite.

<sup>2</sup>EnSight<sup>®</sup> is a post-processing, visualization and animation product from Computational Engineering International (CEI).

<sup>3</sup>A shotline is the path of which a threat interacts with the target.

INTENTIONALLY LEFT BLANK.



---

## 1. Introduction

---

MUVES-S2 is a DoD software tool used to analyze weapon system vulnerability and munition lethality. An analysis provides two main output files. A final results (FR) file (e.g., 6.6.fr) and an Intermediate Results (IR) file. A FR file is always produced; it contains the end result of each threat munition shotline. Optionally generated IR files itemize each shot line's threat/target interaction in great detail. Intermediate results have been used extensively for display and analysis. The discussion will begin with the IR files and their creation. They are the basis for this enhancement. We will describe the data that EnSight requires for visualization and the data contained in the IR files. How to animate the MUVES results using EnSight will be left up to a separate document<sup>4</sup>. This report will focus on the software developed to facilitate that end.

---

## 2. Methods and Procedures

---

IR files are created by a MUVES-S2 analysis run as an option. The analyst must use the “intermediate” keyword in the session file to generate this type of output. Most commonly used is the MUVES-S2 graphical user interface (GUI) (see figure 1) application in which the analyst can “check” the intermediate results option. Both are shown below.

Session file excerpt

```
...  
vector Threatened  
eval inputs/des/sc-ktank  
interdata inputs/icurve/ktank.compart  
evaldata inputs/ecurve/ktank.compart  
viewfile inputs/view/orig.4views  
rayfile inputs/ray/no.rays  
intermediate results/1.0.ir ←  
analyze results/1.0.fr  
...
```

---

<sup>4</sup>Landis, William. *How to Animate MUVES Results Using EnSight*; ARL-MR-0733; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD, February 2010.

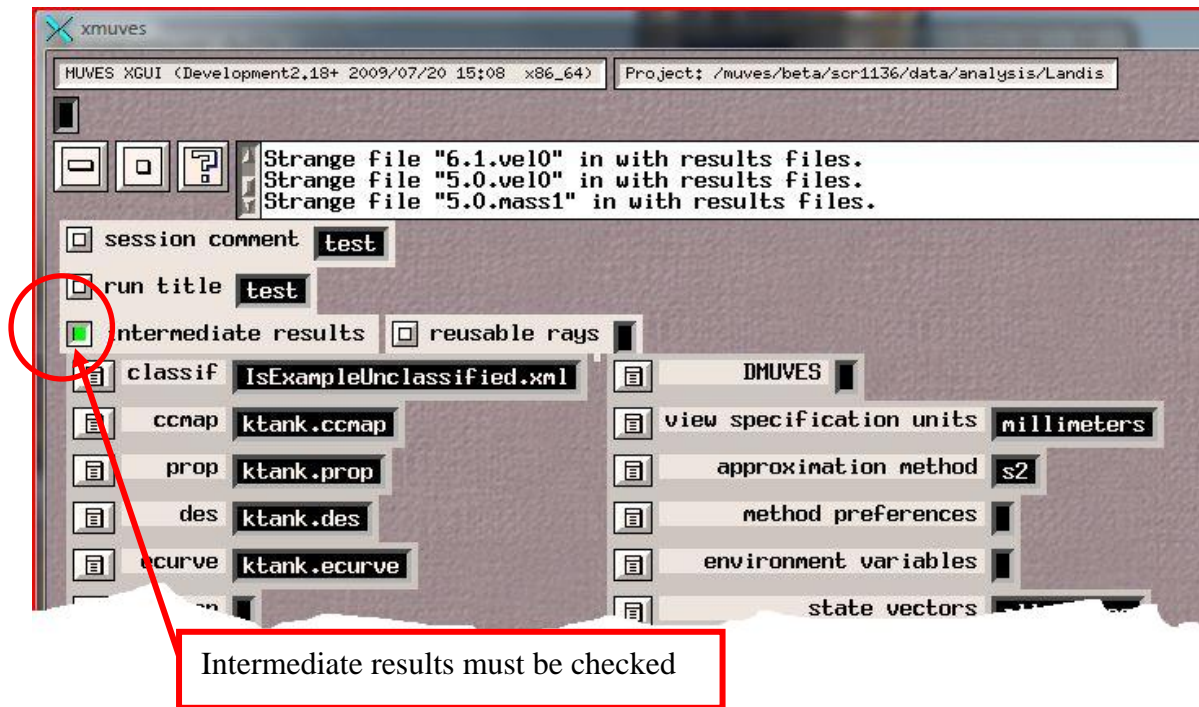


Figure 1. Generating IR files in the MUVES-S2 GUI.

Either of these will generate an IR file for this analysis run. We will now step through the IR file to show the types of data found there. Our goal is to extract information necessary to visualize what occurred during the analysis. For instance, if we wish to animate the direction and velocity of metal fragment threats greater than a certain mass, we need to extract at least the mass and initial velocity component for each fragment.

An IR file consists of a header as seen in figure 2, followed by the body of data organized by “tags” in the file as seen in figure 3. The header includes the version information of MUVES-S2, BRLCAD and ORCA used. It also contains a title, the approximation method, the name of the target and units of the shot coordinates.

```

#Is: header begin
#
#
#           #####
#           ##  UNCLASSIFIED  ##
#           #####
#
#
#
#
#
#Is: header end
MOVES Version:  Development2.19+ 2009/07/01 12:46   x86_64           BRLCAD:   7.14.8
View Units:      millimeters
Title:           "AJEM Sample Case 6.6, 240gr 6000fps Steel Cube Fragment vs Plat"
Approx:          s2
Target:          plates3.ajem
Modkey:          frag_ricochet = ON
Modkey:          frag_deflect = ON

```

Figure 2. Header of the MUVES-S2 IR file.

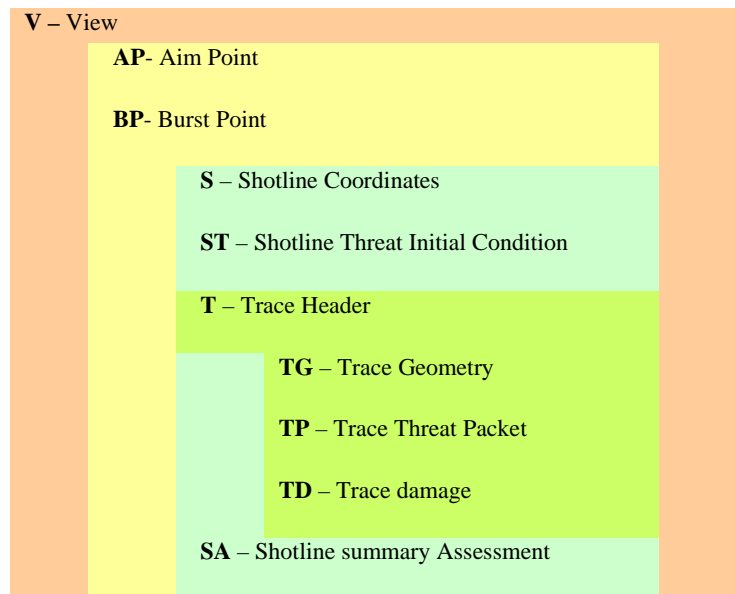


Figure 3. Tagged Structure of the IR file.

Following the header, each populated line of data in the file starts with a tag, a one or two letter key (e.g., V, AP, and BP) that identifies the nature of the line. The “tagged” structure of the IR file is outlined as follows in figure 3.

Below is a portion of a Shotline Threat Initial Condition (ST) line from a typical IR file.

ST: TestFragment MVF { mass=15.5517 vel=1828.8 smass=0 srmass=0 srvel=0 srshpf=0 matl=steel ...

The ST line echoes the initial threat information for the shot on a line. It holds the threat's initial condition such as it's mass and velocity, material and density, orientation, etc. The information provided depends upon the threat being assessed. We could have data for orientation and/or rotation. The type of threat dictates what data is being presented. Data from the entire line will be read and saved by the IrReader. However, from this line, we will make use of the initial mass and velocity of the threat.

Below is a portion of a Trace Geometry (TG) line from the same IR file.

```
TG: entry=< 53.975 -254 -254 > normal=< 1 -0 -0 > dir=< -1 0 0 > ...
```

The TG line echoes the shotline information for the component that is being hit by the threat. It holds the data for the x-y-z entry point, normal vector, direction vector, line of sight and more. From this line, we will make use of the entry data.

Below is an example of a Trace Threat Packet (TP) line.

```
TP: TestFragment MVF { mass=15.5517 vel=1828.8 dir=-1,0,0 Ap=158.008 sf=1 }
```

The threat packet echo provides the name and type of the threat impacting the component followed by a list of the current status of the threat. The list is dependent on the type of threat impacting the component.<sup>5</sup> This line contains the updated mass and velocity data that is of interest to us.

This is a sample of the data that is the primary focus which this new tool employs to enable EnSight to animate the threat event. As outlined above, the IR file has other lines of data, such as View (V), Aim Point (AP), Burst Point (BP), Shotline Coordinates (S), Trace Header (T), Trace Damage (TD) and Shotline Summary Assessment (SA). All of this data is being read and stored by the IrReader. While some of this data may be of use to analysts, it is not required by the EnSight tool for its visualization. We will, however, still examine each one and outline how the IrReader handles them.

Using the structure of the IR file, TAGS are stored as a linked data structure<sup>6</sup> within the computer's memory. Here is a simplified algorithm of how we sampled the data.

For each line in the IR file

- Read the line
- Isolate the tag at the beginning of the line
- Process the remainder of the line based on the tag
- Hold all contents in memory storing the data in linked lists<sup>7</sup> according to the tag

---

<sup>5</sup>Taken from "The AJEM User/Analyst Manual" version 2.18 June 5, 2009.

<sup>6</sup>A linked data structure is a data structure which consists of a set of data records or nodes linked together and organized by references or pointers.

<sup>7</sup>A linked list is a data structure consisting of a sequence of data records where in each record has a field that contains a reference or link to the next record in the list.

Name-value pairs will be parsed as a unit with numeric values converted. The text of the value will be preserved even if a numeric value was found.

The following data is often stored in name-value pairs (e.g., mass=15.5517). When a name-value pair is found, an attempt is made to extract the value as a number. If the attempt fails, the value is assumed to be a string of characters. In any case, the original string of characters is preserved because conversion from string to binary numeric value is so error prone.

When data is found that is not in the form of name-value pairs, each part is stored as a separate string of characters. The delimiter for these items is any whitespace character. No assumptions are made about the relationship between such free form data items. Characters that might imply a relationship (e.g., double quotes or parentheses) are themselves treated as data items and stored as strings of characters.<sup>8</sup>

A simplified algorithm of how we retrieved the data for EnSight:

Using the data structures stored by the IrReader which was organized by the IR Tags...

For each View (**V**) in the list

    For each Shotline threat list (**ST**)

        Read the members of the data structure looking for mass and/or velocity

        Get the initial mass and velocity and record it for EnSight

    For each Trace Geometry list (**TG**)

        Record the displacement vector

    For each Threat Packet list (**TP**)

        Get the last value of mass and velocity

Other tagged lines or data structures are not mentioned as they are not needed for the EnSight tool. All tags are still, however, available for arbitrary use. Therefore, there is a potential for future development for a tool to provide immediate access to this data library when the need arises. Its limit is only the limit of the imagination or ambition of the analyst.

---

### 3. Developing the Software

---

MUVES-S2 has an extensive library of development tools, some of which have been employed in this project. We will not go into each line of code in detail. However, this section focuses on the MUVES-S2 Ir library and in particular the Ir's IrReader and its data structures. The intent is to illustrate how this project uses the MUVES-S2 library and demonstrates how robust and

---

<sup>8</sup>This may appear inefficient, but as an explanation in historical context, the IR file has not been maintained as a database with strict formatting rules. Over the decades new information was added to, or formatting modified in, the IR file as needed. Therefore, while the basic field structures are known, a reading application should be robust enough to handle unexpected information.

versatile the MUVES-S2 product can be. The following highlights some examples of how this project was constructed in detail. Some of these highlights assume a background in “C” programming.

To begin, we need to read in the IR file. A simple low level `fopen()`<sup>9</sup> is all that is required. Within the MUVES-S2 library is a method, `IoOpenFile( )` that opens the file named by the concatenation of its variable character-string arguments for read/write/append according to the specified mode (which is passed to `fopen()`). It returns `NULL` if memory could not be allocated to build the pathname or if the file could not be opened, the error flag, `ErIndex` is set to identify the type of error that occurred). The final argument must be `(char *)NULL`. The method returns a `FILE` pointer upon success.

Reading the file from the top down, we start by processing the header. This portion of the file is at least consistent throughout the different IR files. The “MUVES Version”, “View Units”, “Title”, “Approx” (Approximation Method) and “Target” are in every IR file. We created a data member for each of the attributes in the header. They are stored in the top level data structure called, **IRFile** (figure 4). To help show how this entire data tree is built, we will begin filling out a data chart for each element of data. Next, the modkeys are processed and stored in a data structure member list called `mModkeyList` within the `IRFile` data structure. The following chart illustrates this and shows how the multiple modkey entries were handled. Note the modkey entry into the data structure. It splits into another data structure as required by the IR file. Modkeys are arbitrary and dependent on the analysis being run. This IR file had two modkeys set (see Figure 2. Header of the MUVES-S2 IR file), which caused the `IrReader` to add a branch to the initial data structure. It does this dynamically no matter how many modkeys are present.

---

<sup>9</sup>A file is opened using **fopen**, which returns an input/output stream attached to the specified file or other device from which reading and writing can be done.

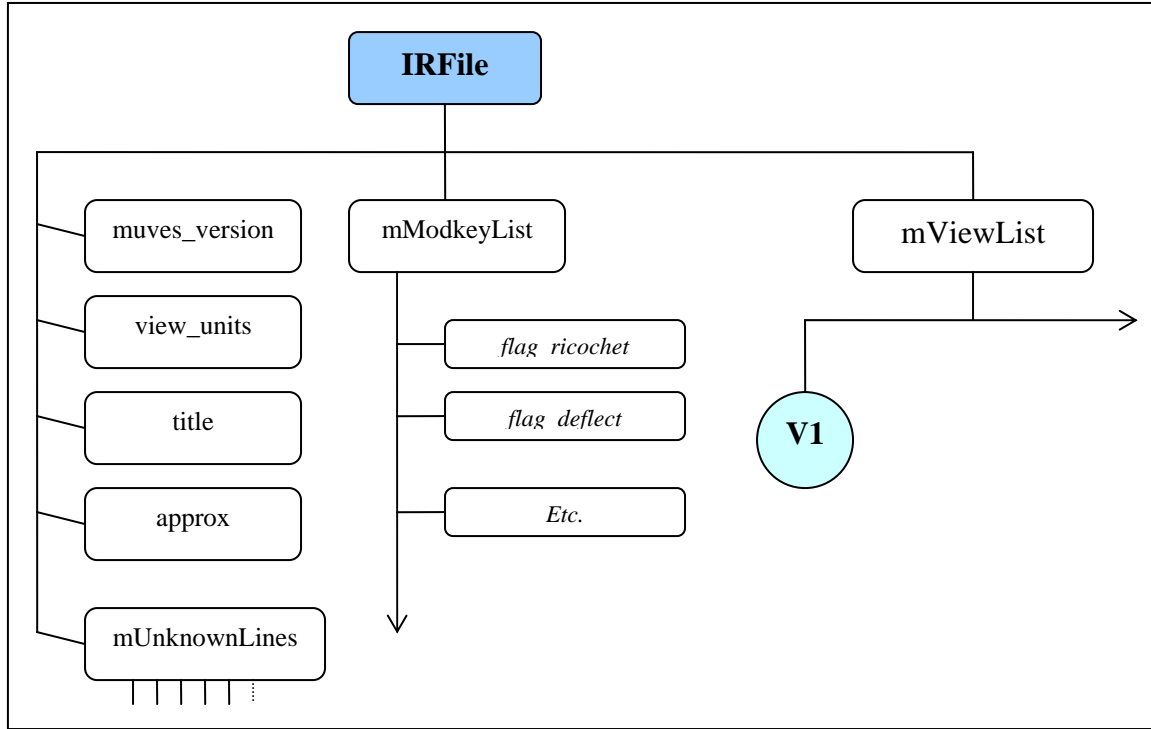


Figure 4. IRFile data structure.

As we continue down through the IR file we process each line of data from the shotline. Each line of data will be recorded. However, we will record data from some of the above mentioned lines of data for the EnSight visualization. If there happen to be any lines that do not have any particular structure or apparent data, we will store them as unknown, **mUnknownLines**.

The first line of data we should encounter is the VIEW (**V**). Here we setup a data structure, **IRView** (figure 5), with data from this line. Its members will be azimuth, elevation, direction vector and a group label. Subsequent data structures are linked using the MUVES development package, **Dq**<sup>10</sup>.

<sup>10</sup>The Dq package provides support for doubly linked lists. A doubly linked list, also known as a double-ended queue, is a data structure in which each node has two pointers, one forward ("to the right") and one backward ("to the left").

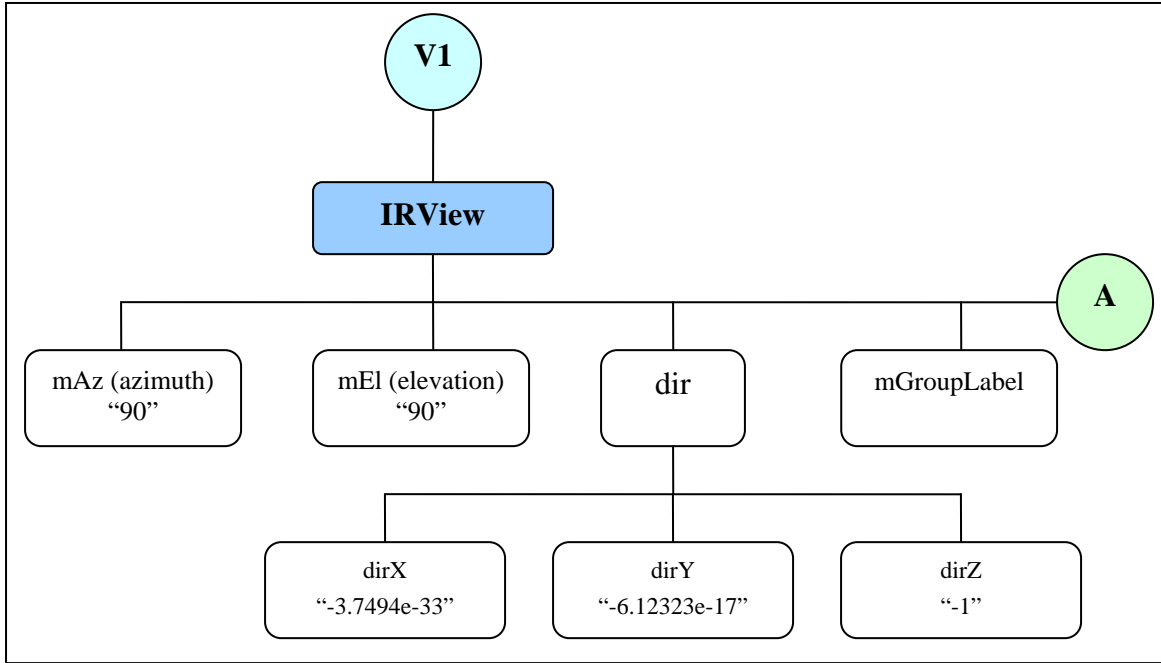


Figure 5. IRView data structure.

The next line of data is the Aim Point (**AP**). It contains the internal index number given to the shotline. Though it is a simple entry, it is the line of data that instantiates the shotline. We will start a shotline data structure from this “View” and build a data tree off of it. Since there can be multiple shotlines within a file, for every “View” we have a list of multiple shotlines. We create the **mShotlinelist** containing **IRShotline**(s) (figure 6). The Aim Point value is stored as **mAimPointIndex** under the **IRShotline** along with the **mFiringPointIndex** if present. A shotline starts with an initial threat, such as a shaped charge jet. Upon interaction with the target, additional shotlines are formed due to the spall<sup>11</sup>. Consequently, we introduce the **IRShotTrace** data structure to record the data of the fragments.

The Burst Point line (**BP**) contains the x-y-z coordinates of the burst point. BP is not always present, but when it is it is stored under the **mShotlineList** as **mBurstPoint** with a data structure **MPoint3D** to hold the x-y-z coordinate data (figure 6). The shotline data begins with the shotline coordinates (**S**) and cell coordinates (**C**). This data will be held at the top level of the **IRShotTrace** as **mShotlineCoord** and **mCellCoord**, each using the data structure **MPoint2D** to hold the x-y values. Now under the view we have added as follows (figure 6).

<sup>11</sup>**Spall** are flakes of a material that are broken off a larger solid body and can be produced by a variety of mechanisms, including as a result of projectile impact.



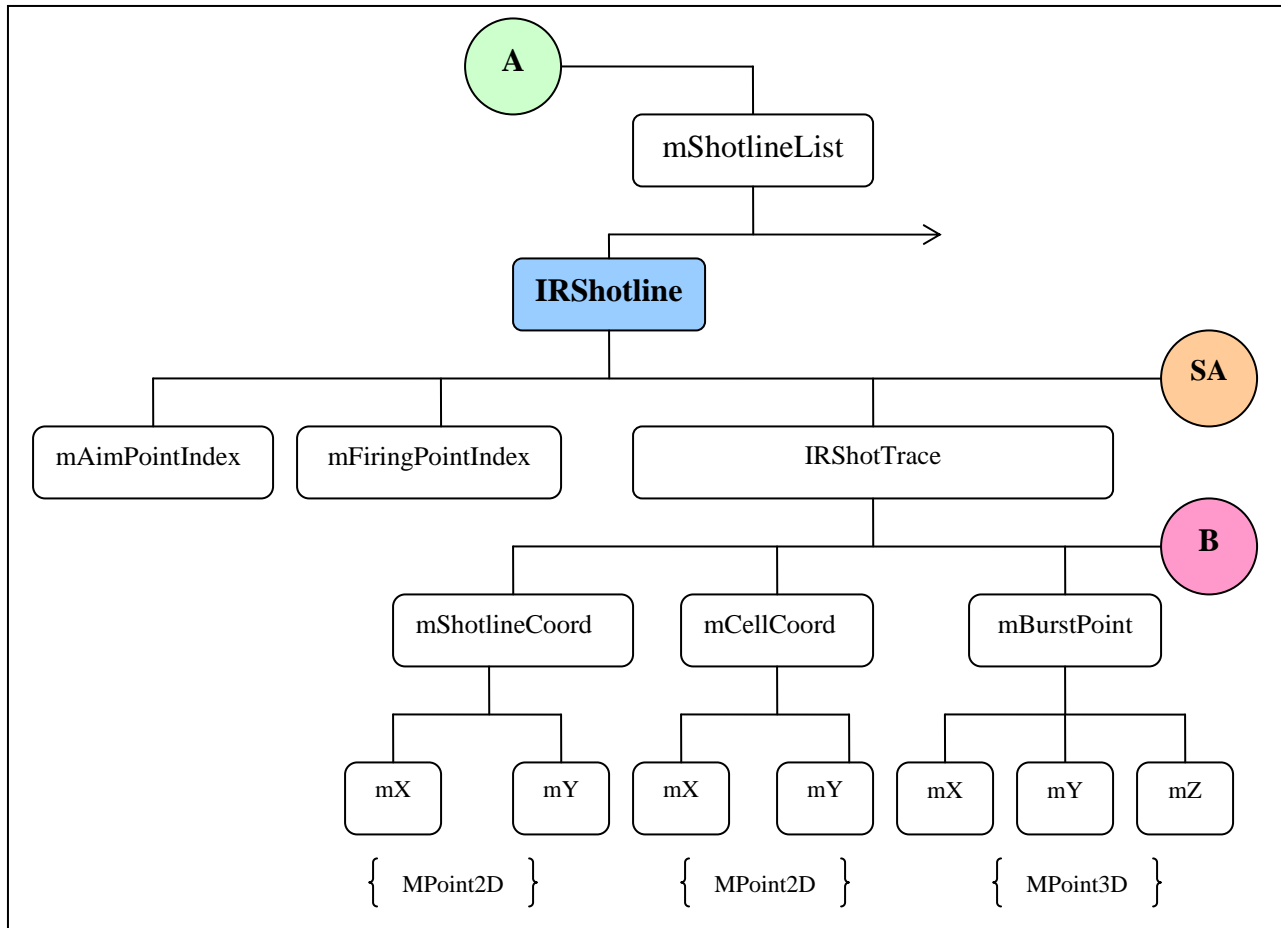


Figure 6. IRShotline data structure.

The Shotline Threat Initial (**ST**) conditions would be seen next. We create a new data structure, **IRShotlineThreat** (figure 7), to contain all of the data for the threat's initial conditions and the shot's trace (figure 7). Each trace represents the shot's travels through a component. Also, each component penetrated or perforated gets its own trace. We create **IRTrace** for this data. This is the line of data tagged, Trace Header (**T**). It will hold the threat type designator, component name and material. It will also be the parent level for the Trace Geometry (**TG**), Threat Packet (**TP**) and Trace Damage (**TD**). They are the three lines that will follow **T** in the IR file.

Following the trace header will be the trace geometry data. There will always be only one TG line followed by zero or more lines of TP followed by zero or more lines of TD. We will store this entry data in its own data structure, **IRTraceGeometry** under trace geometry. Next is the threat packet. This data will be stored as **IRThreatPacket** (figure 8). As this will hold the updated status of the threat, it will include data for the Subtrace Geometry (**STG**). This was not previously mentioned. However, some threats may include this line of data. This will be stored as **IRSubtraceGeometry** under the **mSubtraceGeomList** as there may be multiple entries. The trace damage, **IRTraceDamage**, line finishes the shot trace. We have completed the shot trace as follows (figures 8 and 9).

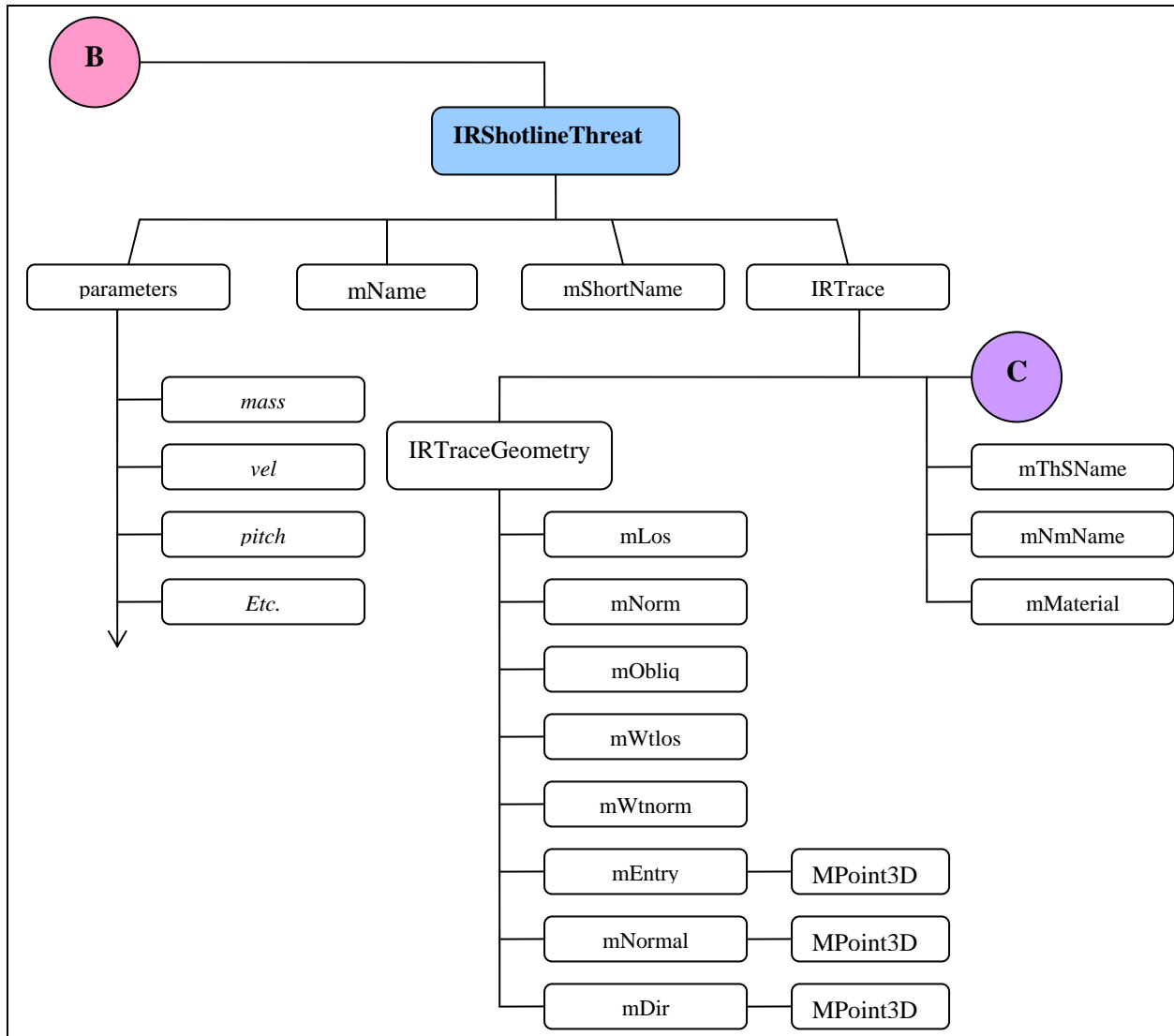


Figure 7. IRShotlinethreat data structure.

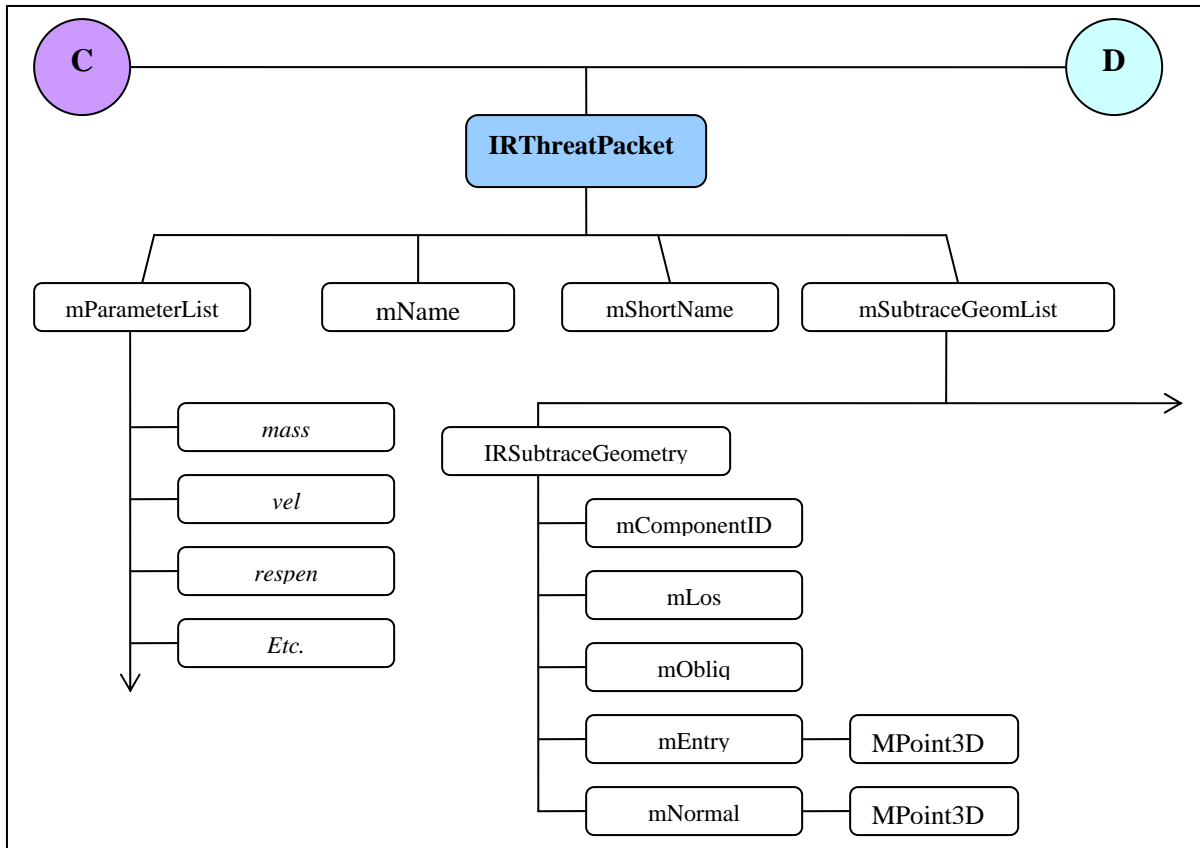


Figure 8. IRThreatPacket data structure.

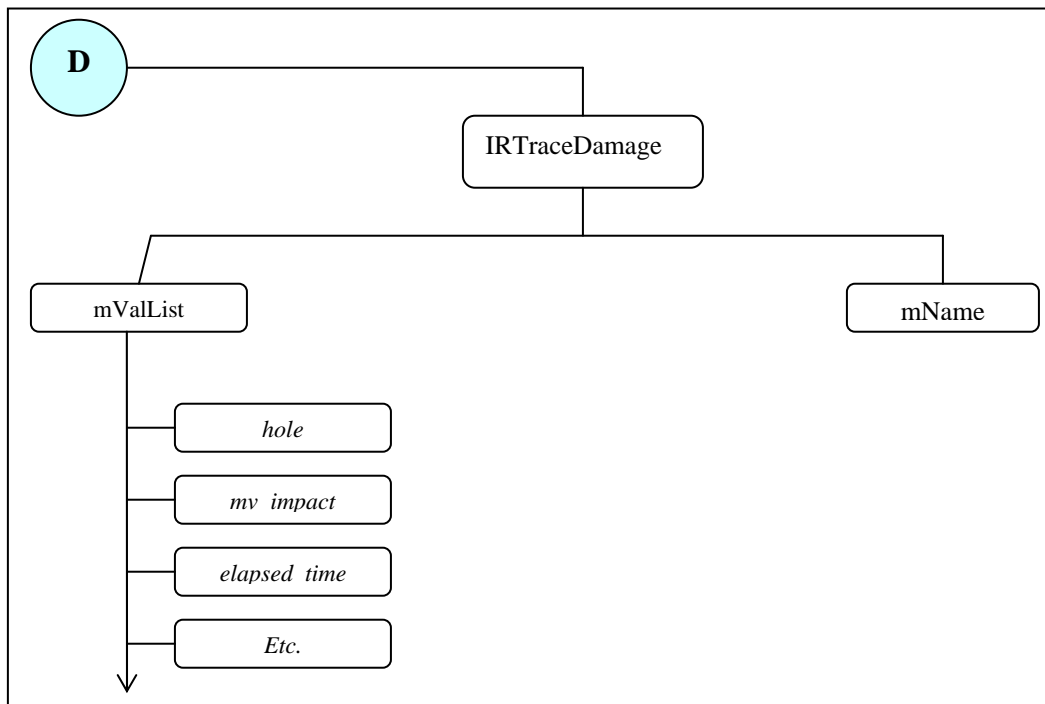


Figure 9. IRTraceDamage data structure.

Finally, the trace damage data is followed by the Shotline Summary Assessment (**SA**) (figure 10). This line of data will be held by data structure, **IRSummaryAssessment**. The types of values that may be shown are pk, ps, hit, killed, scalar, frf, or lof. There will be one of these for every shotline, so it belongs up a couple levels, under the IRShotline. Here is a small example of a SA line.

SA: "back"=0.205(pk) "back[hole]"=0.864(pk)

Also shown on figure 10 is the ORCA Operational Requirement (**OP**) to be held in memory by **IRORCAOperationalRequirement**. When needed, it will be within each shotline. We have shown it here for completion.

OP: Orca\_man job Assaulting\_Infantry\_Rifleman\_v2.1.odb 13 12 11 10 9 8 7 6 5 3 4 2

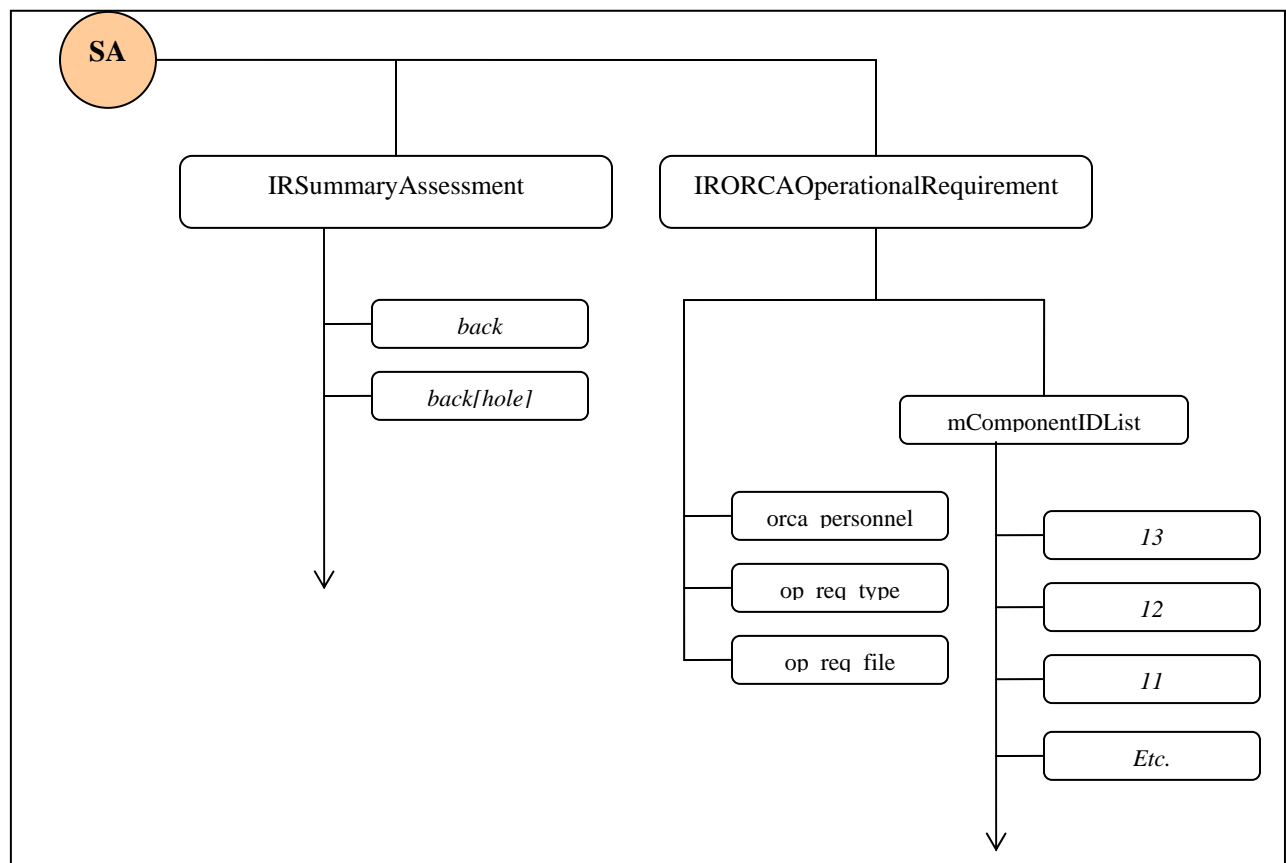


Figure 10. Data Structures IRSummaryAssessment and IRORCAOperationalRequirement.

---

## 4. Using the Application–Ir2En

---

Ir2En is a post-processing application (see figure 11). It is included with MUVES-S2 installations of software release 2.20 and above. The executable is located in the following path: \$MUVES/bin/Ir2En (where “\$MUVES” represents the directory where MUVES-S2 is installed) When the user sets their PATH in a MUVES environment, it can be run from any location, preferably where they have their IR files to be examined. After a MUVES run has produced IR files, typing **Ir2En** at the command prompt will start the application. It will interactively prompt the user for some additional information, IR filename, filename prefix of the EnSight input files and burst point coordinates ( x y z ). See the example interaction below.

```
$Ir2En
Enter MUVES-S2 .ir filename => 6.1.ir
Ir file, 6.1.ir, was entered
Enter burst point coordinates x y z => 3 3500 0
Burst point coords 3.000000e+00 3.500000e+03 0.000000e+00 were entered
Enter the EnSight filename prefix => test2
EnSight filename prefix test2 was entered
```

Figure 11. Ir2En interaction.

Alternatively, use options to enter all or some of the information at once, -f for the filename, -d for the displacement or burst point coordinates, and -p for the EnSight filename prefix.

```
$Ir2En -f 6.1.ir -d 3 3500 0 -p test2
```

More usage information is included in a man page<sup>12</sup> available from within the MUVES environment.

```
$man Ir2En
```

The application will create eight ASCII files. These are the files EnSight needs to animate the threat. Using the example above, here are the files Ir2En created (see table 1).

---

<sup>12</sup>Unix man pages are documentation or manuals of commands, macros, programs etc. retrieved by entering the command “man” followed by the program name.

Table 1. Ir2En files.

File Name	Description (where the data was acquired from in the IR file)
test2.vel0	Initial velocity from .ST. line
test2.vel1	Threat packet velocity from .TP. line
test2.mgeo0	Entry point user input
test2.mgeo1	Entry point from .TG. line
test2.mass0	Initial mass from .ST. line
test2.mass1	Threat packet mass from .TP. line
test2.geo	Fragments geometry file
test2.case	EnSight informational data

Noted to the right of the file names, we indicate where the data was acquired. This was spelled out in the algorithm explained in the introduction. Basically, we used our data tree and traversed down the IRShotline visiting all the branches and leaves and collected the data that we knew EnSight wanted to use. We then saved that data to the appropriate files.

EnSight can now provide the animation. This report was made to introduce the MUVES tools to get you there. The subsequent steps to run the visualization are explained in detail by William Landis in the “How to Animate MUVES Results Using EnSight” document of which Ir2En is a part of. A summary is available on the Ir2En man page as well.

---

## 5. Results

Accomplishing our objective, we have created a viable and useful tool from components of the MUVES development library in combination with a new software element to extract and house data from IR files for retrieval at will. In addition, we have shown that the data the “IrReader” collects can be acquired and made available for a specific analysis, in this case, the EnSight Visualization. Many such applications can be applied. It is up to the analyst to conjure the ideas for its use.

---

## 6. Conclusions

There is a wealth of information contained within IR files. It is a valuable and diverse source of threat analysis data that could be exploited. It just needed to be made more readily available for the analyst. The main issue with IR files is that they echo the shotline data in an arbitrary form e.g., (this=”that”, this=< 1 2 3 > or this=1,2,3). They were never designed to be a database file with predictable data structures that appear in a strict sequence. The IrReader attempted to read each IR file and parse out the data to store it into discrete and meaningful data structures. When a

file was encountered with data too far out of the norm, we produced an error message that the tool could not parse the file and return. For instance, of the 314 IR files generated in moves sample test cases, 20 cannot be processed by the IrReader. (See appendix A for a list of files)

We know what we gain from looking at a picture instead of raw data. With the help of this software and the use of the EnSight tool, the animation of threat against target adds an immeasurable value to understanding what happens to a vehicle during a munition impact. Though it will not be demonstrated here, one is encouraged to explore this type of analysis.

---

## 7. Bibliography

---

1. Butkiewicz, Mark; et. al. *Advanced Joint Effectiveness Model (AJEM), User/Analyst Manual Version 2.18*, The SURVICE Engineering Company, Aberdeen, Maryland, June 2009.
2. Rybczynski, Robert. *MUVES-S2 Change Design Document (CDD) for Software Change Request (SCR) #1133 Enhanced Visualization*, 14 September 2009, an internal AJEM document, [www.ajem.com](http://www.ajem.com)
3. Gonzalez, Roberto; Rybczynski, Robert; Landis, William. *MUVES-S2 Change Design Document for SCR #1246 Interface for EnSight Visualization*, 14 September 2009.
4. EnSight Gold – parallel processing and rendering visualization  
<http://www.ensight.com/ensight-gold.html>
5. Landis, William. *How to Animate MUVES Results Using EnSight*; ARL-MR-0733; U.S. Army Research Laboratory: Aberdeen Proving Ground, MD , February 2010.



---

## Appendix A. IR file list

---

Intermediate Results files that cannot be processed by the IrReader API.

1.1.ir  
4.2.ir  
8.2.ir  
8.6.ir  
8.7.ir  
8.8.ir  
18.0.ir  
18.12.ir  
18.1.ir  
18.5.ir  
18.6.ir  
18.7.ir  
20.10.ir  
20.6.ir  
20.7.ir  
20.9.ir  
21.0.ir  
21.1.ir  
26.1.ir  
26.3.ir

INTENTIONALLY LEFT BLANK.

---

## List of Symbols, Abbreviations, and Acronyms

---

**AJEM** – Advanced Joint Effectiveness Model is a lethality, vulnerability, and endgame computer simulation code

**AP:** - Aim Point index tag in MUVES output files

**B:** or **BP:** - Burst Point coordinate information tag in MUVES output files

**C:** - shotline Cell coordinate information tag in MUVES output files.

**BRL-CAD®** - Ballistic Research Laboratory Computer-Aided Design is a suite of tools developed by SLAD for creating and visualizing target descriptions.

**ErrIndex** – is a recorded error index (long) as MUVES defined errors in /muves/lib/errors

**frf** - Fractional Remaining Functionality.

**lof** - Loss of Function.

**LOS** - Line of Sight.

**modkey** – option(s) set by the analyst to select from multiple features in a MUVES session file

**ORCA** – Operational Requirement-based Casualty Assessment Model is a personnel vulnerability model, providing capabilities to perform ballistic survivability, vulnerability and lethality analyses of Soldier systems, weapons and weapons systems.

**Pk** – Probability of Kill (Note:  $P_k = 1 - P_s$ ).

**Ps** - Probability of Survival (Note:  $P_s = 1 - P_k$ ).

**S:** - Shotline coordinate information tag in MUVES output files

**SA:** - Summary Assessment index tag in MUVES output files. This indicates the line of data that lists critical components and their Pk values.

**ST:** - Shotline Threat information tag in MUVES output files. Information which follows this tag includes threat name and initial condition of threat parameters.

**T:** - Trace header tag in MUVES output files that contains threat identifier and component name.

**target** - the combat system which is being threatened

**TD:** - Trace Damage index tag in MUVES output files. Identifies damage packets produced from component/threat interaction.

**TG:** - Trace Geometry tag in MUVES output files. Line of sight geometry information through the component follows this tag.

**threat** - the entity which is aimed at, impacting on, detonating near, threatening the target.

**TP:** - Threat Packet tag in MUVES output files. Information which follows this tag includes threat name and status of threat parameters after impact.

**V:** - View index tag in MUVES output files.

No. of Copies	Organization
1 ELEC	ADMNSTR DEFNS TECHL INFO CTR ATTN DTIC OCP 8725 JOHN J KINGMAN RD STE 0944 FT BELVOIR VA 22060-6218
1 CD	OFC OF THE SECY OF DEFNS ATTN ODDRE (R&AT) THE PENTAGON WASHINGTON DC 20301-3080
1	US ARMY INFO SYS ENGRG CMND ATTN AMSEL IE TD A RIVERA FT HUACHUCA AZ 85613-5300
1	COMMANDER US ARMY RDECOM ATTN AMSRD AMR W C MCCORKLE 5400 FOWLER RD REDSTONE ARSENAL AL 35898-5000
3	US ARMY RSRCH LAB ATTN RDRL SLB S R GONZALEZ BLDG 247 ABERDEEN PROVING GROUND MD 21005
1	US ARMY RSRCH LAB ATTN RDRL CIM G T LANDFRIED BLDG 4600 ABERDEEN PROVING GROUND MD 21005-5066
2	US ARMY RSRCH LAB ATTN RDRL SLB A DATA MANAGEMENT TEAM ATTN RDRL SLB R BOWEN ABERDEEN PROVING GROUND MD 21005-5068
1 CD	US ARMY RSRCH LAB ATTN RDRL LOW MAILROOM (VAULT) J TRAMMEL, SR. (1 CD) WHITE SANDS MISSILE RANGE NM 88002-5513
1	US ARMY RSRCH LAB ATTN RDRL SLE R FLORES BLDG 1624 WHITE SANDS MISSILE RANGE NM 88002-5513

No. of Copies	Organization
1	U.S. EVALUATION CENTER ATTN TEAE SV (TECHNICAL DIRECTOR) 4120 SUSQUEHANNA AVE ABERDEEN PROVING GROUND MD 21005-3013
3	US ARMY RSRCH LAB ATTN IMNE ALC HRR MAIL & RECORDS MGMT ATTN RDRL CIM L TECHL LIB ATTN RDRL CIM P TECHL PUB ADELPHI MD 20783-1197
TOTAL: 16 (1 ELEC, 2 CDS, 13 HCS)	

INTENTIONALLY LEFT BLANK.