Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

# MISR Visualization Experimental Environment

*-System Architecture Document (SAD)*
*-Visualisation Tool Preview and Data Set Integration*

*Rémi Blanchette*
*Thales Canada Inc.*
*1 Chrysalis Way*
*Ottawa, Ontario*
*K2G 6P9*

*Document Control No: 1773C.007-SAD-VT Rev. 01*
*Contract No.: W7701-071670/A*
*Scientific Authority: Alain Bouchard (418) 844-4000 x4744*

The scientific or technical validity of this Contract Report is entirely the responsibility of the contractor and the contents do not necessarily have the approval or endorsement of Defence R&D Canada.

**Defence R&D Canada – Valcartier**
Contract Report
DRDC Valcartier CR 2008-129
March 2008

Canada

THALES

# MISR Visualization Experimental Environment
# System Architecture Document (SAD)
# Visualisation Tool Preview and Data Set Integration

**Contract No: W7701-071670/A**

**Document Control No: 1773C.007-SAD-VT Rev. 01**

**RDDC-DRDC Document No: CR 2008-129**

**Date: 18 March 2008**

**– RESTRICTIONS ON DISCLOSURE –**

The information contained in this document has been prepared by Thales Systems Canada, a Division of Thales Canada Inc. for the Government of Canada in accordance with Contract W7701-071670/A. The restriction and rights on disclosure shall be in accordance with the contract terms and conditions.

THALES

# MISR Visualization Experimental Environment
# System Architecture Document (SAD)
# Visualisation Tool Preview and Data Set Integration

**Contract No: W7701-071670/A**

**Prepared by:**

Rémi Blanchette
Engineer

**Approved by:**

Richard Grenier
Project Manager

# REVISION HISTORY

| Date | Rev. | Internal Version | Description | Author |
|---|---|---|---|---|
| 18 March 2008 | 01 | 1 | Revision based on comment by Maxime Tardif. | Rémi Blanchette |
| 1 May 2008 | | 1.1 | Add DRDC number | Maxime Tardif |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

# TABLE OF CONTENTS

**Page**

**1 INTRODUCTION ..... 6**

1.1 Purpose ..... 6
1.2 Scope and Limitations ..... 6
1.3 Document Overview ..... 6

**2 REFERENCED DOCUMENTS ..... 6**

**3 FUNCTIONAL ANALYSIS ..... 6**

3.1 Data Source Selection ..... 7
3.1.1 Overview of the Functionality ..... 7
3.1.2 Actors ..... 7
3.1.3 High-Level Use Cases ..... 8
3.1.4 USE CASE: Select Data Source ..... 8
3.1.5 Implementation Overview ..... 9
3.1.6 USE CASE: Convert to Data Set ..... 9
3.1.7 Implementation Overview ..... 9
3.2 Data Requirements Specification and Validation ..... 10
3.2.1 Overview of the Functionality ..... 10
3.2.2 Actors ..... 10
3.2.3 High-Level Use Cases ..... 11
3.2.4 USE CASE: Interpret Data Source ..... 11
3.2.5 Implementation Overview ..... 11
3.2.6 USE CASE: Data Set Requirement Validation ..... 11
3.2.7 Implementation Overview ..... 12
3.3 Visualization Preview ..... 12
3.3.1 Overview of the Functionality ..... 12
3.3.2 Actors ..... 12
3.3.3 High-Level Use Cases ..... 13
3.3.4 USE CASE: Visualize ..... 13
3.3.5 Implementation Overview ..... 13
3.3.6 USE CASE: Create Loader Instance ..... 13
3.3.7 Implementation Overview ..... 14

**4 TECHNOLOGIES ASSESSMENT ..... 14**

4.1 Existing Infrastructure ..... 14
4.1.1 Imago ..... **Erreur ! Signet non défini.**
4.1.2 LiveSpaces ..... 15
4.2 Other Technologies ..... 15
4.3 User Interface ..... 15

**5 RECOMMENDATION ..... 15**

5.1 Other Consideration ..... 15

## LIST OF FIGURES

**Page**

Figure 3-1: Imago Solution Components ..... 7
Figure 3-3: Data Source Selection ..... 8
Figure 3-4: Data Source Type Class Diagram ..... 9
Figure 3-5: Data Set Class Diagram ..... 10
Figure 3-6: Data Set Requirements ..... 12
Figure 3-7: Loader Creation Preview ..... 14

## LIST OF TABLES

**Page**

Table 2-1: Referenced Documents ..... 6

# 1 INTRODUCTION

## 1.1 Purpose

The purpose of this document is to clearly define the technical problems posed by the dynamic preview of visualisation tool in the context of the Imago application and the architectural solution that address these problems.

As stated by the client: “we want to evaluate the possibility of giving the Imago user the capacity to preview and play with the different visualisation tools part of the knowledge base and the capacity to upload new data set into the visualisation tools”. This document will split this statement into underlying software functions necessary to get as close to that as we can. Thus explaining the problem and the reasoning behind the solution proposed.

## 1.2 Scope and Limitations

It is not in the scope of this document to go into the specifics of each visualisation tools preview and integration. Rather, this would be a global recipe with some common parts, and some “tool-specific” parts. Also, the initial scope of the proposed design will be very wide to encompass most possibilities but keep in mind that an implementation of this design could work with fewer components by reducing the scope accordingly.

Also, the deployment of the visualisation tool on the user station which is a very big issue in itself is not addressed by this document and considered out of the scope of this study.

## 1.3 Document Overview

The document will describe function by function the overall solution to the problem expressed by the use case.

# 2 REFERENCED DOCUMENTS

**Table 2-1: Referenced Documents**

| N° | Reference | Source | Title |
| --- | --- | --- | --- |
| [1] | http://services.alphaworks.ibm.com/manyeyes/home | Web | IBM many eyes website |
| [2] | | | |
| | | | |

# 3 FUNCTIONAL ANALYSIS

The following diagram (Figure 3-1: Imago Solution Components) provide an overview of the basic infrastructure of the Imago solution. Simply put it is a client server application that uses the “LiveSpace” component as the host of the knowledge objects and notification mechanism.

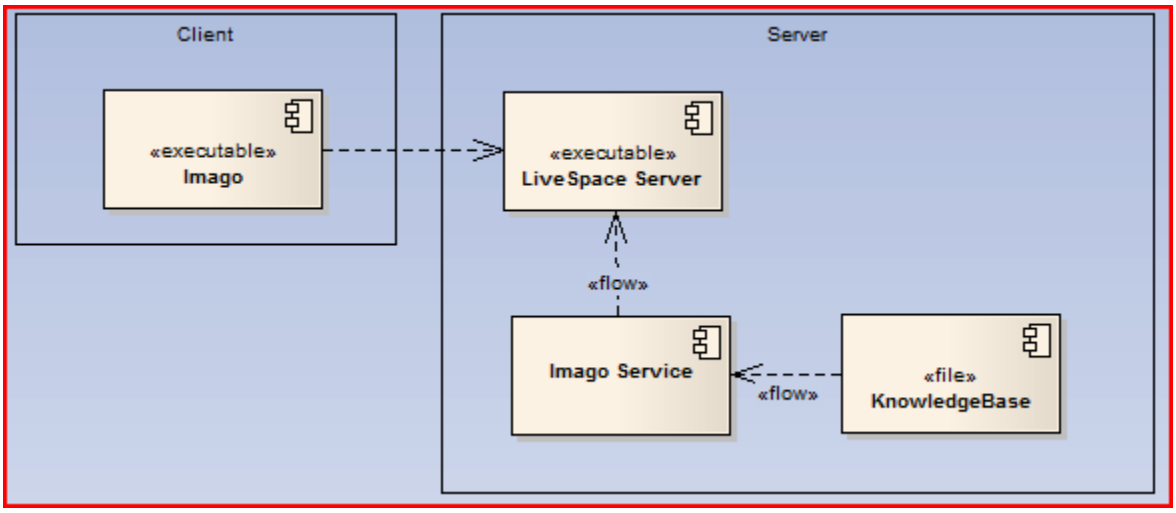



**Figure 3-1: Imago Solution Components**

## 3.1 Data Source Selection

### 3.1.1 Overview of the Functionality

In the process of previewing a visualisation tool, the user wants to select a data source to be used by the tool. He can select any data source, from a single text file, to database instance. The format of the data source does not matter to the user although the compliance of the format will be verified by the system.

### 3.1.2 Actors

**User**: the Imago user who in the process of previewing a visualisation tool wants to test it. The user has already selected the tool he wants to preview by browsing in Imago. He now needs to select which data to use.

### 3.1.3 High-Level Use Cases

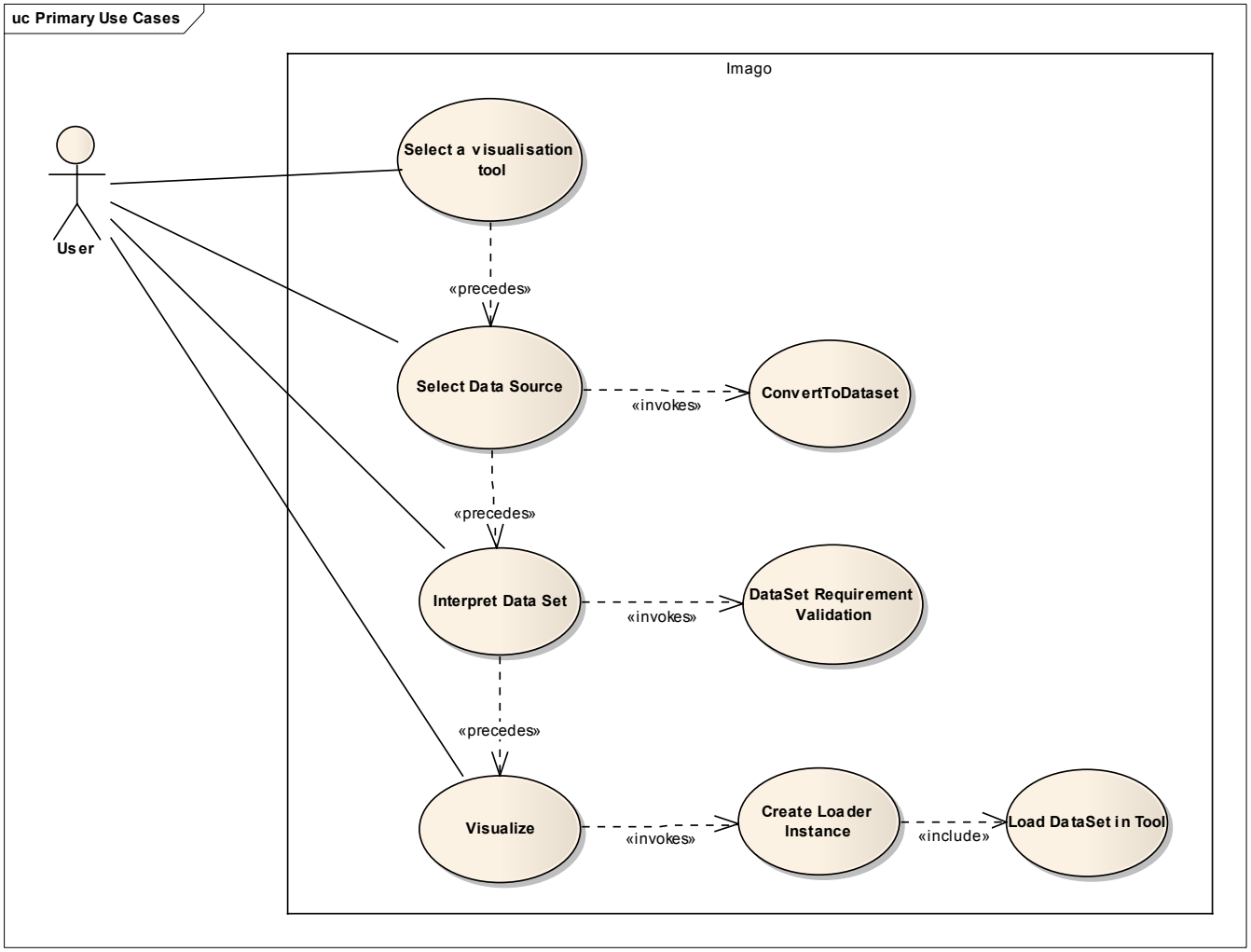



**Figure 3-2: Data Source Selection**

### 3.1.4 USE CASE: Select Data Source

The user is presented with a data source selection GUI that allows him to specify which data source he wants. He will first select the type of data source; i.e.:

- File
- Directory
- Database
- Web service

Then, the user will be presented a second panel dedicated to the selection of information regarding the data source; i.e.:

- File path;
- Directory path;
- Service url;
- Database connection string;

The wizard will then perform a validation for the data source existence and connectivity then finish appropriately.

### 3.1.5 Implementation Overview

This operation should be implemented by a step by step wizard to keep the flow of the task simple and focused of the type of data source selected in previous steps. Also, there is not much complexity here except in the validation part which will be very different for each type of data source. Checking for a file existence is not the same as verifying connectivity with an oracle 9i account. The complexity of this wizard will be proportional to the number of type of data source supported by the process.

**Option**: the preview process could work without the data source selection by "hard-coding" this step for each tool. The solution would be much less flexible but that much more simple.

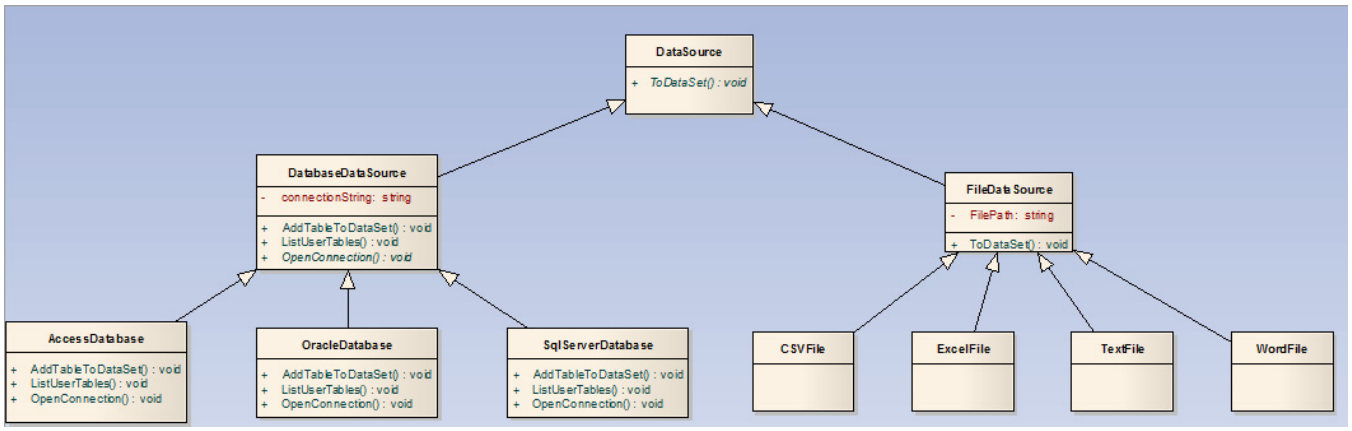



**Figure 3-3: Data Source Type Class Diagram**

### 3.1.6 USE CASE: Convert to Data Set

This use case is an internal software function triggered by the data source selection. The user does not specifically need to do anything. From a system point of view however, to achieve any sort of automatic and generic approach we need to convert the data from its original format to a common representation. This common representation should be a generic in-memory database that could be exported and stored in an XML format for optimization purpose.

### 3.1.7 Implementation Overview

In very general terms, the nature of data that we could integrate in a generic way will need to be relational in nature and consist of mostly simple attributes. We need to emphasize that only the most simple of attributes can fit into a very generic model.

For example, a list of person name, their age and address would fit that description while a picture of that person, a voice print and a satellite picture of that address would not.

Military related data tends to be very complex in nature because of the strategic value that needs to come from that data in order to get any sort of attention from military personnel.

The implementation of this use case is a virtual method in the data source hierarchy that produces a data structure in memory that is very generic. Tables, columns and rows are the common denominator or all relational data and they are the output of this function.

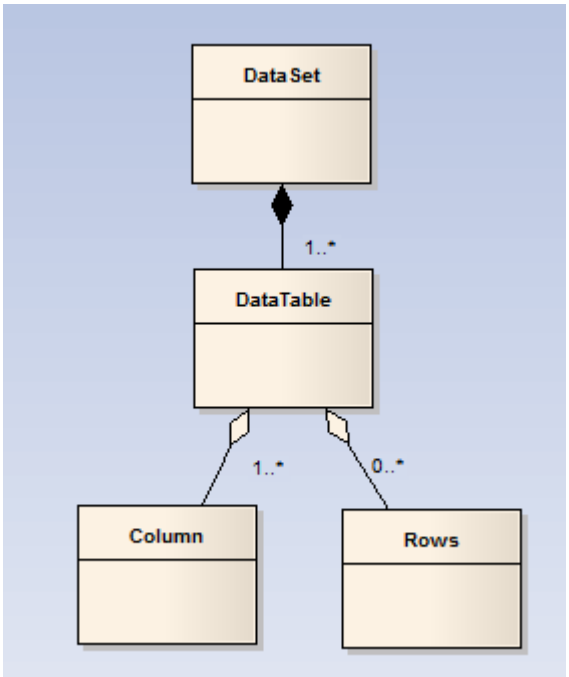



**Figure 3-4: Data Set Class Diagram**
This diagram need to be reworked with Java details and adjusted as necessary.

## 3.2 Data Requirements Specification and Validation

### 3.2.1 Overview of the Functionality

In the process of previewing a visualisation tool, the system need to validate that the data source selected by the user respected the minimum criteria needed by the visualisation tool. Needless to say that these data constraint are not something that we can discover automatically and thus they need to be added to the knowledge base or at least linked to the tools in one way or another.

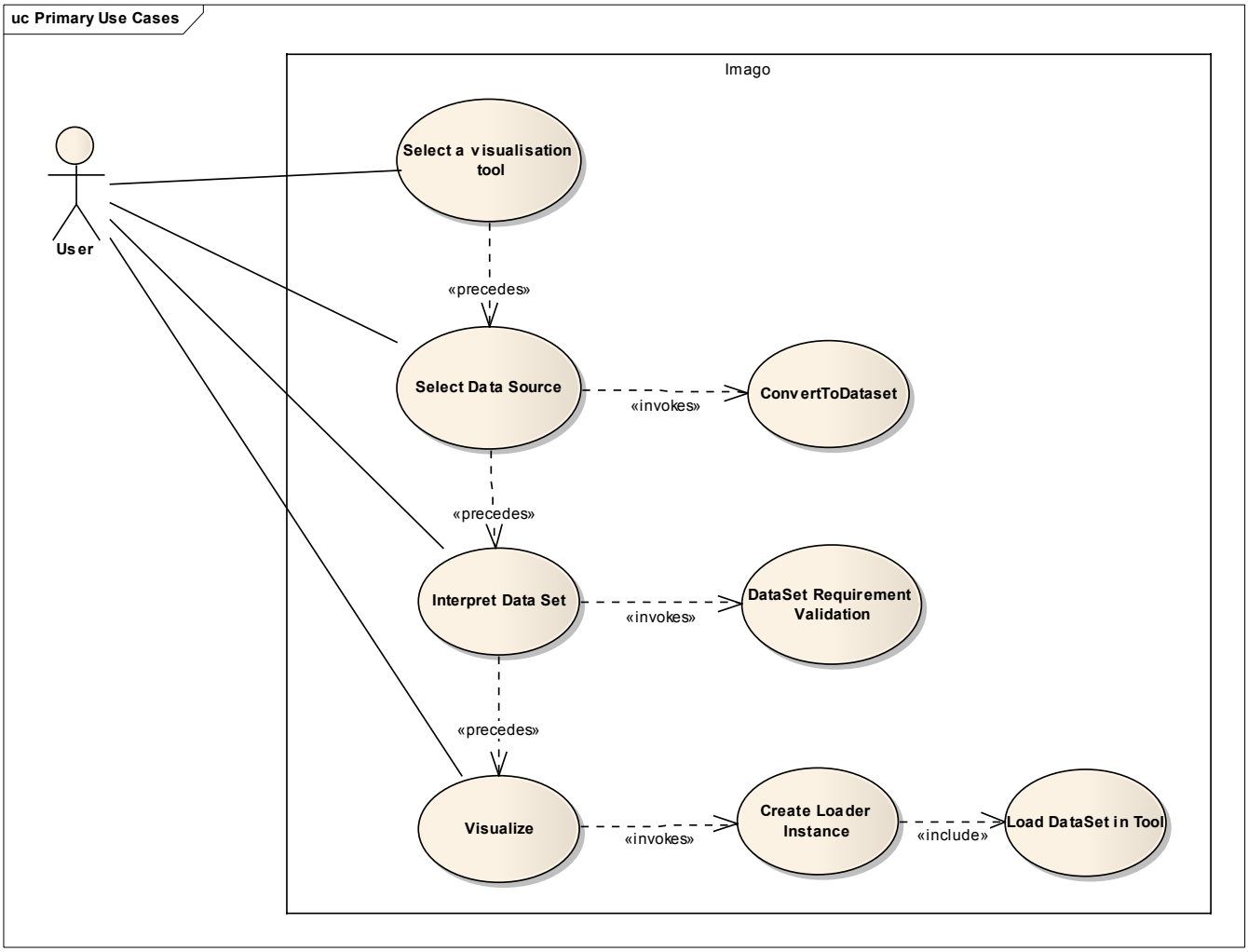
Also, the preview could very well work without this automatic validation and data interpretation. This software function adds the following quality to the solution;

- data set mismatch resiliency;
- human based data matching;
- error resiliency;

### 3.2.2 Actors

**User**: the Imago user who in the process of previewing a visualisation tool wants to test it, he will now fill in some blanks for the data interpretation of the data source.

### 3.2.3 High-Level Use Cases




### 3.2.4 USE CASE: Interpret Data Source

The user is presented with a GUI that present some questions based on the Data Set requirements identified for this tool. The questions are mostly presented as dropdown list input since what we are looking for is a matching between required table and columns and their actual name in the data source.

### 3.2.5 Implementation Overview

A simple GUI that generate question based on the requirement. The dropdown are filled with the possible values derived from the data source. These answers are not persisted.

### 3.2.6 USE CASE: Data Set Requirement Validation

This use case is an internal software function triggered by the interpretation of the data by the user. The user does not specifically need to do anything in this step except perhaps pressing a “validate” button or a “next” in a wizard.

### 3.2.7 Implementation Overview

A simple GUI that generate question based on the requirement. The dropdown are filled with the possible values derived from the data source. These answers are not persisted but are rather stored in a memory structure that will be sent to the loader down the line.

Also, the information stored in the data set requirement to the right should be added either to the visualisation tool in the knowledge base or added as an external configuration file.

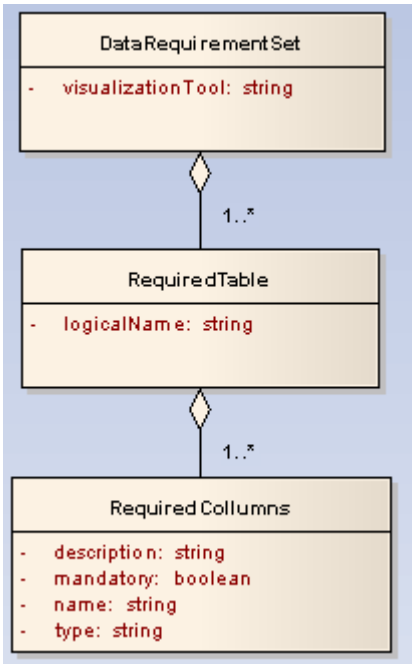



**Figure 3-5: Data Set Requirements**

## 3.3 Visualization Preview

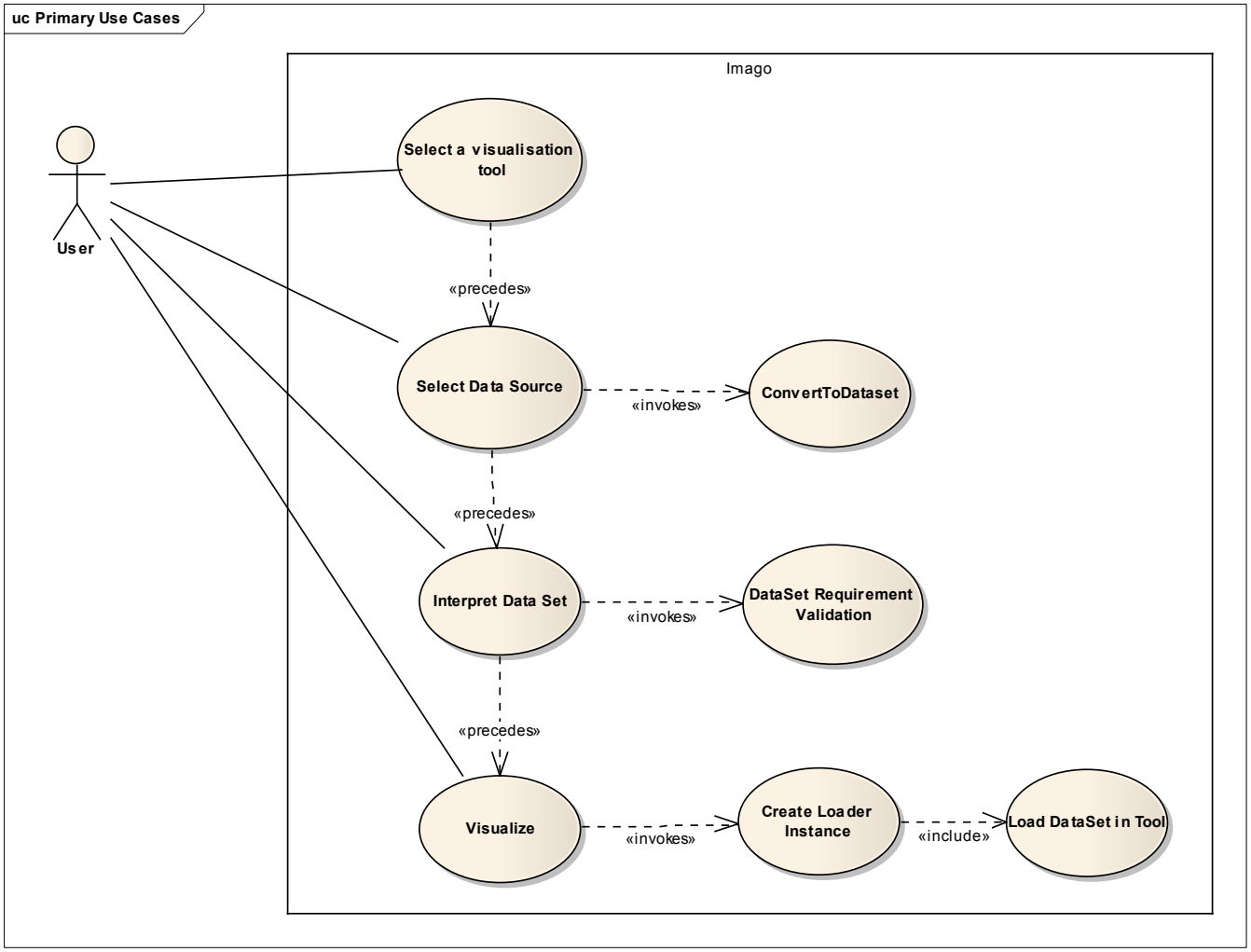
### 3.3.1 Overview of the Functionality

This is the final step of the process of previewing and also the only mandatory one, the system need to create the dynamic instance of the visualisation tool to preview and load it up with the selected data set. Since those visualisation tool does not form any sort of standard as far as API and Interoperability is concerned we can only provide a basic mechanism to initiate the loading step but each of the tool will need custom code to be fully integrated.

### 3.3.2 Actors

**User**: the Imago user who in the process of previewing a visualisation tool wants to test it, he will now use the tool once its dynamic component as been loaded up as part of the Imago client.

### 3.3.3 High-Level Use Cases




### 3.3.4 USE CASE: Visualize

This last action of the user could be thought of as the final confirmation of the user before loading the tool. It is the user triggered hook into the internal load process of the tool.

### 3.3.5 Implementation Overview

This general hook in Imago is the last generic call in the preview pattern. It calls a factory to obtain the loader instance and call the methods to load the tool with the data.

### 3.3.6 USE CASE: Create Loader Instance

This is an internal operation done by a factory based pattern.

### 3.3.7 Implementation Overview

Based on a logical key obtained from the tool instance, the factory will create an instance of a tool specific loader class. The class once created will be in charge of:

- creating the bridge to the tool API;
- load the tool GUI;
- load the data set into the tool;
- Release the API once the user as finished the preview.

This is where the design becomes very tool specific and we could control the activation of the preview function based on the existence of the loader class. No loader class for a tool mean a preview not possible. From an implementation perspective, we could create the loader class much sooner and put into it any tool specific behaviour or information that we need. The whole preview process should be an application of the Template Method Design Pattern (http://en.wikipedia.org/wiki/Template_method).

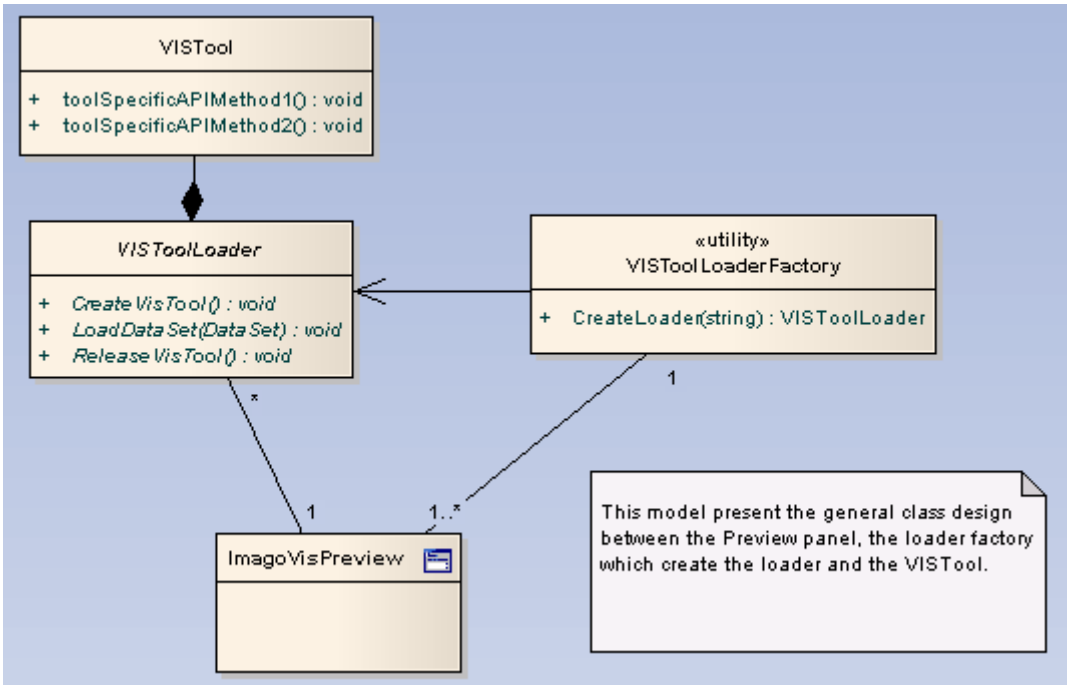



**Figure 3-6: Loader Creation Preview**

# 4 TECHNOLOGIES ASSESSMENT

## 4.1 Existing Infrastructure

### 4.1.1 Imago

Prior to launching the development of the preview function we should prototype the integration of various type of API from a java standpoint:

- other java applications;

- COM based application;
- .NET interface;
- Etc…

Those prototypes would act as a proof of feasibility for typical integration and we should be able to draw guidelines from those prototype.

### 4.1.2 LiveSpaces

As we understand it, the LiveSpaces technology would not help nor hinder the preview in any way.

## 4.2 Other Technologies

When it comes to integrating visualisation capabilities, most viewers will need to be installed side by side with the Imago client on the user computer. Some of these tools might be able to run server-side but they would be exception. The deployment mechanism of these tools will depend on the type of application. This is a very difficult and probably mostly manual process. To be able to formulate any kind of recommendation on deployment, the current visualisation tool needs to be explored and analysed.

## 4.3 User Interface

To be determined

# 5 RECOMMENDATION

Seeping from the analysis commanded by this study are the following recommendations;

- Before committing to any development, evaluate the quantity of tools that “can” be integrated;
- Evaluate the possibility of replacing the real tools by small dynamic java “mock-up” that could have a standard interface rather then integrating the real application;
- Evaluate the possibility and pertinence of giving the user a remote desktop connection to a machine that hold those products rather then deploying and integrating the tools directly in Imago;
- Evaluate development costs by taking into account the 2 pitfalls of deployment and the tool specific loader class;

## 5.1 Other Consideration

The technical solution presented by this document assumes that the licensing of the tools to be deployed are covered and paid for. Although this is a simple assumption, it is a very difficult one to verify and predict as the number of Imago users is not defined at this time. There is also legal ramification if the licensing of these tool where to be violated. Also, the licensing mechanism is something that can be quite different from one tool to another; leading to more configuration management problems along the way.

## DOCUMENT CONTROL DATA

| 1. ORIGINATOR (name and address) | 2. SECURITY CLASSIFICATION (Including special warning terms if applicable) |
|---|---|
| Defence R&D Canada Valcartier<br>2459 Pie-XI Blvd. North<br>Val-Bélair, QC<br>G3J 1X8 | Unclassified |

3. TITLE (Its classification should be indicated by the appropriate abbreviation (S, C, R or U)

MISR Visualization Experimental Environment - System Architecture Document (SAD) - Visualization Tool Preview and Data Set Integration (U)

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)

Thales Canada Inc.

| 5. DATE OF PUBLICATION (month and year) | 6a. NO. OF PAGES | 6b .NO. OF REFERENCES |
|---|---|---|
| March 2008 | 15 | 0 |

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. Give the inclusive dates when a specific reporting period is covered.)

Contractor Report

8. SPONSORING ACTIVITY (name and address)

| 9a. PROJECT OR GRANT NO. (Please specify whether project or grant) | 9b. CONTRACT NO. |
|---|---|
| 11he | W7701-7-1670 |

| 10a. ORIGINATOR'S DOCUMENT NUMBER | 10b. OTHER DOCUMENT NOS |
|---|---|
| CR 2008-129 | N/A |

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

- ☒ Unlimited distribution
- ☐ Restricted to contractors in approved countries (specify)
- ☐ Restricted to Canadian contractors (with need-to-know)
- ☐ Restricted to Government (with need-to-know)
- ☐ Restricted to Defense departments
- ☐ Others

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected.)

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

(U)

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

MISR, Maritime, Intelligence, Surveillance, Reconnaissance, Testbed, Experimental Environment, Visualization, Evaluation (U)

**Defence R&D Canada**

Canada's Leader in Defence
and National Security
Science and Technology

**R & D pour la défense Canada**

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

**www.drdc-rddc.gc.ca**