VARIAN
medical systems

Portal Dosimetry Scripting API Reference Guide

ARIA

CE 0086 | ISO 13485 REGISTERED | P1001365A | June 2013

Document ID	P1001365, Revision A
Abstract	*Portal Dosimetry Scripting API Reference Guide* provides basic information for writing scripts to access the data model in applications. The following product is used: ■ Portal Dosimetry, version 13.0 This publication is the English-language original.
Manufacturer	Varian Medical Systems, Inc. 3100 Hansen Way Palo Alto, CA 94304-1038 United States
European Authorized Representative	Varian Medical Systems UK Ltd. Oncology House Gatwick Road, Crawley West Sussex RH10 9RG United Kingdom
Notice	Information in this user guide is subject to change without notice and does not represent a commitment on the part of Varian. Varian is not liable for errors contained in this user guide or for incidental or consequential damages in connection with furnishing or use of this material. This document contains proprietary information protected by copyright. No part of this document may be reproduced, translated, or transmitted without the express written permission of Varian Medical Systems, Inc.
FDA 21 CFR 820 Quality System Regulations (CGMPs)	Varian Medical Systems, Oncology Systems products are designed and manufactured in accordance with the requirements specified within this federal regulation.
ISO 13485	Varian Medical Systems, Oncology Systems products are designed and manufactured in accordance with the requirements specified within the ISO 13485 quality standard.
CE	Varian Medical Systems, Oncology Systems products meet the requirements of Council Directive MDD 93/42/EEC.
EU REACH SVHC Disclosure	The link to the current EU REACH SVHC disclosure statement can be found at http://www.varian.com/us/corporate/legal/reach.html
HIPAA	Varian's products and services are specifically designed to include features that help our customers comply with the Health Insurance Portability and Accountability Act of 1996 (HIPAA). The software application uses a secure login process, requiring a user name and password, that supports role-based access. Users are assigned to groups, each with certain access rights, which may include the ability to edit and add data or may limit access to data. When a user adds or modifies data within the database, a record is made that includes which data were changed, the user ID, and the date and time the changes were made. This establishes an audit trail that can be examined by authorized system administrators.
WHO	ICD-O codes and terms used by permission of WHO, from: ■ International Classification of Diseases for Oncology, (ICD-O) 3rd edition, Geneva, World Health Organization, 2000. ICD-10 codes and terms used by permission of WHO, from:

- International Statistical Classification of Diseases and Related Health Problems, Tenth Revision (ICD-10). Vols 1–3, Geneva, World Health Organization, 1992.

CAUTION: US Federal law restricts this device to sale by or on the order of a physician.

Trademarks ARIA®, Clinac®, Varian® and VMS® are registered trademarks. Eclipse™ is a trademark of Varian Medical Systems, Inc. Microsoft®, Windows NT® and Windows® are registered trademarks of Microsoft Corporation. All other trademarks or registered trademark are the property of their respective owners.

Microsoft® is a registered trademark of Microsoft Corporation.

All other trademarks or registered trademarks are the property of their respective owners.

Copyright © 2013 Varian Medical Systems, Inc.

All rights reserved. Produced in Switzerland.

Contents

Chapter 1 Introduction

About this Manual

The Portal Dosimetry Scripting Application Programming Interface (Portal Dosimetry Scripting API) is a programming interface and a software library for Portal Dosimetry. It allows software developers to write scripts to access the data model in Portal Dosimetry. The scripts can be integrated into the Portal Dosimetry user interface, or they can be run as stand-alone executables.

Visual Cues Used in this Guide

This section presents the types of notes and precautionary notices used in the guide, along with their icons. The following notational conventions are used:

Note: *A note provides non-critical information, such as user requirements, computer messages, suggestions, and shortcuts, which can help users to obtain optimum performance from the equipment or software.*

CAUTION: Describes actions or conditions that could result in minor or moderate injury or damage to equipment or loss of data. All caution notices must be obeyed.

WARNING: Describes actions or conditions that could result in serious injury or death. All warning notices must be obeyed.

Bold Text

This guide uses bold text for:

- Menus and menu commands. Example: **Choose Edit** > **Delete** .
- Command buttons. Example: To confirm, click **Yes**.
- Option buttons. Example: In the dialog box, select the **3D option** button.

Quotation Marks

This guide uses quotation marks for:

- Internal cross-references, for example, see “Adding a New Patient” in this chapter.

- Text that displays in a window or dialog box, for example, the status bar text reads “Mean error 13”.

Italics

This guide uses italics for:

- References to other publications, for example, refer to *Using ARIA manual*.
- Emphasis, for example, *Always* verify plans visually.
- Workspaces. Example: Annotations can be introduced when in the *Acquire* workspace.

Who Should Read This Manual

This guide is written mainly for medical/technical personnel who wish to write custom scripts to be used in Portal Dosimetry. It is assumed that you are familiar with:

- Portal Dosimetry application
- Radiation oncology domain and concepts
- IMRT quality assurance concepts and methods
- DICOM
- Software engineering practices
- Microsoft Visual Studio development environment
- Microsoft Visual C# programming language and object oriented development
- English

CAUTION:

Before creating your own scripts, familiarize yourself with the Portal Dosimetry user documentation, especially any safety-related information, cautions and warnings found throughout the documentation.

Related Documentation

- Portal Dosimetry Reference Guide
- Portal Dosimetry Online Help

Contacting Varian Customer Support

Varian Customer Support is available on the internet, by e-mail, and by telephone. Support services are available without charge during the initial warranty period.

The my.varian.com website provides contact information, product documentation, and other resources for all Varian products.

Get Online Customer Support

You can browse the my.varian.com site without having a Varian account or logging in. However, you must have a Varian account to get online customer support and to access product information for products at your institution or clinic.

1. Go to http://my.varian.com.
2. Click Contact Us at the top of the window to display customer support and training options, and international e-mail addresses and telephone numbers.
3. Choose an option:
   - If you do not already have an account, click Create New Account and follow the instructions. Establishing an account may take a few days.
   - If you have an account, go to the next step.
4. Enter your user name and password.
5. Browse the information and then click the link that corresponds to what you want to do:
   - Fill out and submit a support request.
   - Find documents. Online documents in PDF format include customer technical bulletins (CTBs,) manuals, and customer release notes (CRNs).
   - Send an e-mail to Varian support. You can browse for international e-mail addresses and telephone numbers by geographic area, and for oncology-specific contacts such as for brachytherapy.
   - Find parts and services by geographical area.

E-Mailing Varian

Send e-mail inquiries through the my.varian.com website.

Alternatively, you can use a support e-mail address that corresponds to your location or interest:

Location	E-mail Address
North America	support-americas@varian.com
Latin America	soporte.al@varian.com
Europe	support-emea@varian.com
Australia and New Zealand	support-anz@varian.com
China	support-china@varian.com
Japan	support-japan@varian.com
South East Asia	support-sea@varian.com

Location	E-mail Address
Brachytherapy Systems	brachyhelp@varian.com

Ordering Documents by Phone

You can order documents by phone by calling Varian Medical Systems support.

Location	Telephone Number
North America	+ 1 888 827 4265 (Press 2 for parts)
Global	Call your local Varian office.

Chapter 2 The Portal Dosimetry Scripting API

About the Portal Dosimetry Scripting API

The Portal Dosimetry Scripting API is a Microsoft .NET class library that gives you read access to the data model and objects of the Portal Dosimetry application. It allows you to create scripts that let you retrieve data from the ARIA database and create your own dose evaluations using this data. You can integrate the scripts into Portal Dosimetry, or you can run them as stand-alone executables.

The Portal Dosimetry scripting API and scripting capabilities are very similar to the Eclipse and the Registration and Segmentation Scripting API. If you are already using one of these scripting APIs, many aspects of the Portal Dosimetry scripting API will already be familiar to you.

WARNING: The authors of custom scripts are responsible for verifying the accuracy and correctness of the scripts.

Features

By using the Portal Dosimetry Scripting API, you can:

- Write custom scripts and integrate them into the respective Portal Dosimetry user interface.
- Write stand-alone executable applications that leverage the Portal Dosimetry Scripting API.

Through the created scripts, you can access the following information in the ARIA database :

- Portal dose images, predicted dose images and composite images
- Portal Dosimetry Analysis, including tests and gamma analysis
- Portal Dosimetry analysis templates
- Plans, fields and sessions

As all ARIA applications that support scripting, the Portal Dosimetry scripting API provides read only access to the following RT domain base objects in the ARIA database:

- Field and control point data
- Treatment records
- Projection and volume images

- Annotations, contours and labels on projection images
- Volumetric structures on volume images
- Rigid and non-rigid spatial registrations

The Portal Dosimetry Scripting API provides you also the following:

- A wizard that makes it simple to create new scripts
- Patient data protection that complies with HIPAA
- Support for ARIA user authorization
- API documentation online help
- Example applications
- Creation of transient dose analysis

Note: *Transient means, that the analysis is not stored to the ARIA database.*

System Requirements

The basic system requirements of the Portal Dosimetry Scripting API are the same as those of Portal Dosimetry. For more information, refer to Portal Dosimetry Customer Release Note.

To create scripts with the Portal Dosimetry Scripting API, you need:

- Portal Dosimetry 13.0

Note: *Microsoft Visual Studio is not needed for creating scripts. However, some features described in this document assume that Microsoft Visual Studio 2010 has been installed.*

Compatibility of API Versions

The Portal Dosimetry Scripting API 1.0 is compatible only with Portal Dosimetry 13.0.

It is not guaranteed that the scripts written with the Portal Dosimetry Scripting API 1.0 are compatible with future Portal Dosimetry releases if there are major changes in Portal Dosimetry or in the ARIA platform.

Supported Script Types

Portal Dosimetry supports two types of scripts: Plug-ins and executable applications.

Plug-ins

Plug-ins are launched from the Portal Dosimetry user interface. After the launch, the plug-in gains access to the data of the currently open patient.

Portal Dosimetry supports two types of plug-ins:

- A single-file plug-in: A source code file that Portal Dosimetry reads, compiles on the fly, and connects to the data model of the running Portal Dosimetry instance.
- A binary plug-in: A compiled .NET assembly that Portal Dosimetry loads and connects to the data model of the running Portal Dosimetry instance.

Portal Dosimetry creates a Windows Presentation Foundation child window that the script code can then fill in with its own user interface components. The plug-in scripts receive the current context of the running Portal Dosimetry instance as an input parameter. The context contains the patient, plan, and image that are active in Portal Dosimetry when the script is launched. The plug-in scripts work only for one patient at a time in Portal Dosimetry.

Users of Portal Dosimetry scripts can define favorite plug-in scripts. These scripts are directly accessible in the tools menu of the application or optionally by a user defined shortcut key. Other plug-in scripts need to be started using the 'Scripts' dialog.

Executable Applications

A stand-alone executable is a .NET application that references the Portal Dosimetry Scripting API class library. It can be launched just like any Windows application.

Stand-alone executables can be either command-line applications, or they can leverage any .NET user interface technology available on the Windows platform.

While the plug-in scripts are restricted to work for one single patient opened in Portal Dosimetry, the stand-alone executable can scan the database and open any patient.

Chapter 3 Portal Dosimetry Scripting API Object Model

Portal Dosimetry Scripting API Data Model

The Portal Dosimetry data model is exposed in the Portal Dosimetry Scripting API as a collection of .NET classes with properties and methods. The class hierarchy is an abstraction over the ARIA data model and closely resembles the DICOM object model. None of the properties or methods makes any changes to the data in the ARIA database. This fact guarantees safety against any unintended or erroneous script code.

The consequence of this read-only approach is that it is not possible to create, delete or modify objects on the ARIA database by scripts. If a script needs to store data persistently, this needs to be done by another mechanism, for example files.

The classes of the object model hide all the details of interacting with the database and creating the in-memory representations of the Portal Dosimetry data. Because the Portal Dosimetry API is a .NET class library, all details of managing the memory and other low-level resources are also transparent to you when you create scripts.

Portal Dosimetry Scripting API Concepts

The most important concepts of the Portal Dosimetry Scripting API are described below.

Coordinate System and Units of Measurement

The Portal Dosimetry Scripting API uses the following coordinate systems and units of measurement.

Distances and Positions

In all methods and properties that work with distances and positions, the unit of measurement is millimeters. The positions in 3D space are returned using the DICOM coordinate system. Spatial registrations are always between two DICOM coordinate systems.

Image pixels are accessed using the pixel coordinate system as shown below, where the origin is in the center of the top, left pixel of the first image plane. Remember that pixel indices are zero based and that for projection images the only valid image plane index is zero. The scripting API provides conversion methods between the DICOM and the pixel coordinate systems.

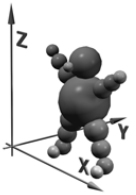

Figure 1 DICOM Coordinate System

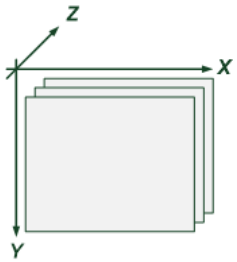

Figure 2 Pixel Coordinate System

For more information on the DICOM coordinate system, refer to the DICOM standard.

Dose Values

Raw integer pixel values of portal and predicted doses need to be converted to dose values using the following method of the Frame class: `public double VoxelToDisplayValue( int voxelValue )`

This delivers the dose values of the image in the unit as stored on the ARIA database, not considering dose normalization.

Normalization

Before comparing the dose values of predicted and portal doses in relative dose mode, both images must be normalized.

When the portal dose image has been acquired in CU mode, normalization is not needed. The predicted dose is stored in Gy/MU on the database. Its dose values must therefore be multiplied with the MUs of the beam before they can be compared with the CU values of the portal dose.

Note: *The currently active dose image passed in the script context to plug-in scripts contains already normalized dose values. Therefore in plug-in scripts the active portal dose and the active predicted dose can be compared directly.*

Treatment Unit Scales

All methods and properties of the Imaging and Registration Scripting API return the treatment unit and accessory properties in the IEC61217 scale. This feature makes it simpler to create scripts because the differences in scale interpretation between the treatment unit vendors do not need to be taken into account.

Images and Frames

The ARIA database supports multi-frame images. This influences also the Portal Dosimetry Scripting API. Every image consists of two parts:

- Container object of type 'Image'. The image contains data that is constant over all frames.
- Collection of 'Frame' objects, aggregated to the image object. The frames contain data that can vary from frame to frame. Most of the image data is contained in the frames, for example size, resolution and pixel data.

Note: *The images relevant for Portal Dosimetry typically only contain a single frame. Frame indices are zero based, therefore this frame has index 0.*

Scripting API Layering

The Portal Dosimetry scripting API classes can be grouped into two layers, represented by two namespaces:

- VMS.CA.Scripting namespace

  The classes in this namespace provide common objects and functionality for all ARIA applications that support scripting. The classes and the relations between the classes are similar – but not identical - to the classes defined by the Eclipse scripting API.
- VMS.DV.PD.Scripting namespace

  The classes in this namespace extend and enrich the base object model by classes specific to the Portal Dosimetry application. They provide a data model that is similar to the data representation in the user interface of the Portal Dosimetry application.

Some classes provided in the VMS.DV.PD.Scripting namespace extend classes from the core namespace by Portal Dosimetry specific aspects. This extension is implemented by aggregating the core namespace object in the extended class. Other classes in the VMS.DV.PD.Scripting namespace – like PDAnalysis – are unique to Portal Dosimetry and do not have a correspondence in the core namespace.

Note: *Extension classes in the VMS.DV.PD.Scripting namespace never duplicate existing functionality from the aggregated core class. Instead, the aggregated core object can be accessed using a property.*

Example: Access pixel data of an object of type 'PortalDoseImage'

```
PortalDoseImage
        pdImage = (get the image);
        Frame pdFrame = pdImage.Image.Frames[0];
        ushort[,] buffer = new ushort[pdFrame.XSize,
pdFrame.YSize];
        pdFrame.GetVoxels(0, buffer);
```

User Rights and HIPAA

The Portal Dosimetry Scripting API uses the same user rights and HIPAA logging features as Portal Dosimetry. If you have logged into Portal Dosimetry with certain user rights, and you execute a plug-in script, the script applies the same user rights as you have in Portal Dosimetry. If you execute a stand-alone executable script, the script code must provide a valid user name and password to authenticate itself to the system, or it can invoke the interactive ARIA login dialog.

According to HIPAA rules, a log entry is made for each patient opened by a standalone script. Additionally, the Portal Dosimetry Scripting API follows the rules of department categorization of ARIA.

Working with Several Patients

The plug-in scripts gain access to the context of the running Portal Dosimetry instance. They work only for the one patient that is selected in that context. In contrast, the stand-alone executables can open any patient in the database. However, only the object model of a single patient is available at a time. The previous patient data must be explicitly closed before another patient is opened. If you try to access the data of a patient that has been closed, an access violation exception is generated.

Overview of the Portal Dosimetry Object Model

The following class diagram gives an overview on the Portal Dosimetry scripting API object model, focusing on the objects specific to Portal Dosimetry:

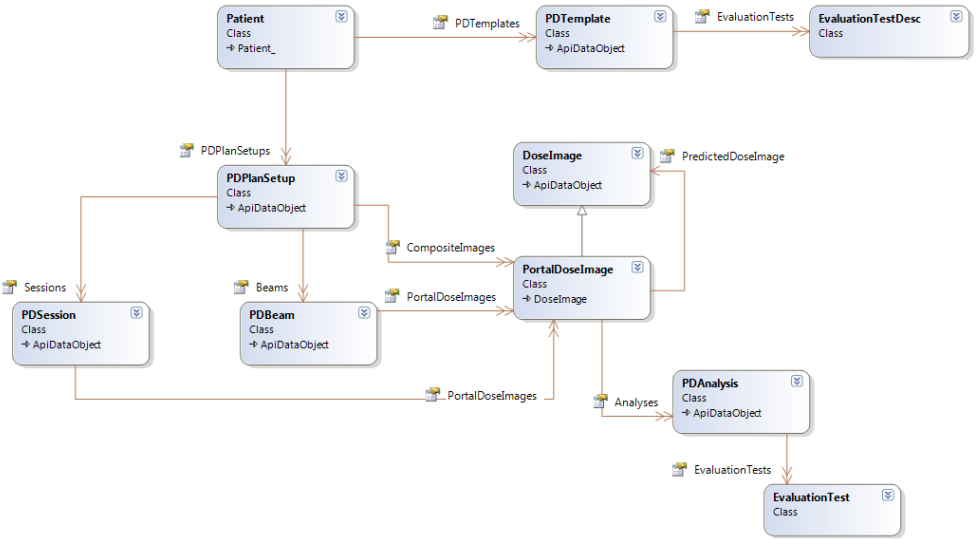

Figure 3 Portal Dosimetry Scripting API Object Model

The diagram contains the following objects:

- *Patient* having a collection of template and plan setup objects.
- *PDTemplate* having a collection of evaluation test description objects. Represents a template for performing a Portal Dosimetry analysis.
- *EvaluationTestDesc* representing a description of an evaluation test as part of a template.
- *PDPlanSetup* having a collection of session, beam and composite image objects. Represents a Portal Dosimetry plan setup object as shown in the user interface. Wraps a plan object of type *PlanSetup*.
- *PDSession* having a collection of portal dose images. Represents a session object in the Portal Dosimetry application.
- *PDBeam* having a collection of portal dose images. Represents a Portal Dosimetry beam object as shown in the user interface. Wraps a beam object of type Beam.
- *PortalDoseImage* having a collection of Portal Dosimetry analysis objects and an assigned predicted dose image. Wraps an image object of type *ImageRT*. Using the method CreateTransientAnalysis(), a new transient analysis can be created for this portal dose image. The analysis parameters are given by a template.

- *PDAnalysis* having a collection of evaluation test objects. Represents a Portal Dosimetry analysis, a comparison or a dose constancy check.

Overview of the Core Object Model

The following class diagram gives an overview of the *Image* related objects in the core scripting API object model:

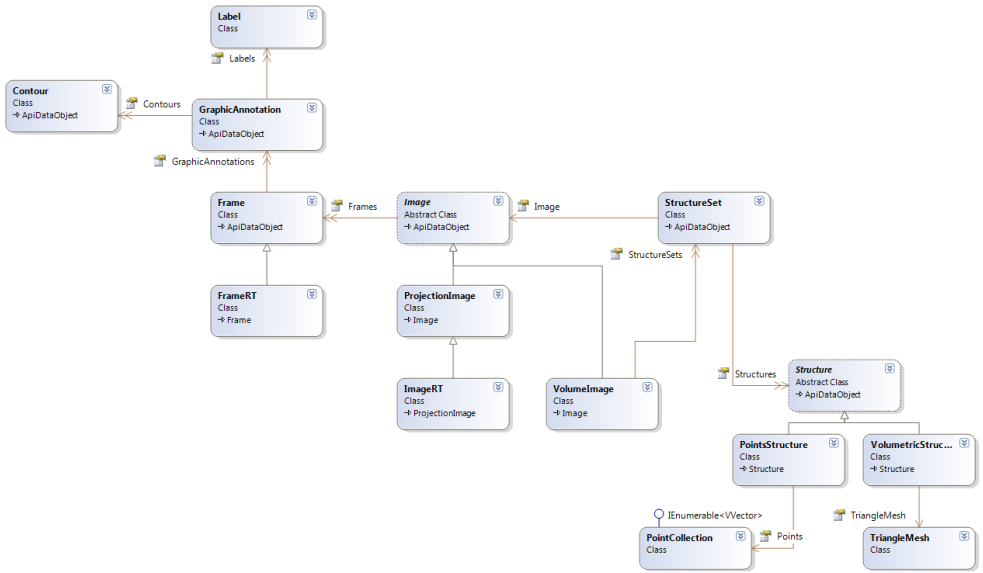

Figure 4 Image Data Model

The diagram contains the following objects:

- *Image* having a collection of frames. Represents a projection or a volumetric image.
- *ProjectionImage* representing a projection image (non-volumetric image).
- *ImageRT* representing an RT image.
- *VolumeImage* having a collection of structure sets. Represents a volumetric image or a slice of a volumetric image.
- *Frame* having a collection of graphic annotations. Represents a frame. A frame is always part of an image and contains the data that can vary from frame to frame in a multi-frame image.
- *FrameRT* representing a frame in an RT image.
- *GraphicAnnotation* having a collection of contours and labels.
- *Contour* representing a planar contour on a projection image.
- *Label* representing a text label on an image.

- *StructureSet* having a collection of structures.
- *Structure* representing a volumetric structure (segment, for example organ) or a point set.
- *PointsStructure* having a collection of points. Represents a point set – a collection of points in the volume image, for example markers.
- *VolumetricStructure* having a triangle mesh.
- *TriangleMesh* representing a triangle mesh.

Another important section of core scripting API is the model of Plan-related objects shown in the diagram below.

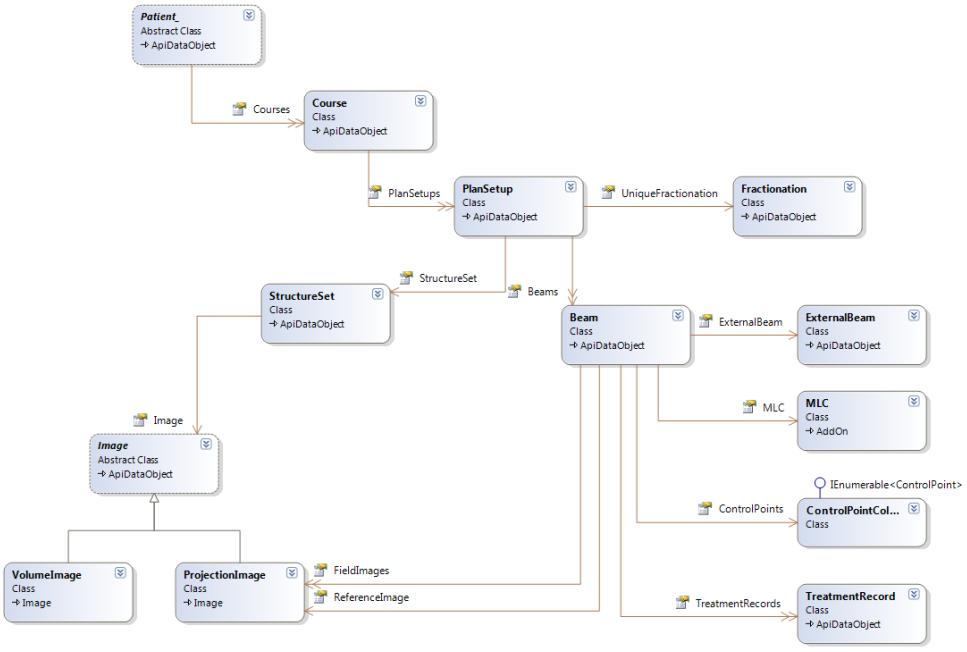

Figure 5 Plan Data Model

The diagram contains the following objects:

- *Patient* having a collection of courses.
- *Course* having a collection of plan setups.
- *PlanSetup* having a collection of beams, a structure set (representing the planning volume image if not *null*) and a fractionation.
- *Beam* having a collection of beam images, control points and treatment records, an assigned primary reference image, the external beam for treatment (for example a clinac) and an MLC.
- *ControlPoint* representing a point in a planned sequence of treatment beam parameters.

- *ExternalBeam* representing a machine or device for treatment.
- *MLC* representing a Multileaf Collimator.
- *TreatmentRecord* representing recorded data of a dose delivery performed.

The next diagram shows the objects related to the DICOM *Patient – Study – Series model*:

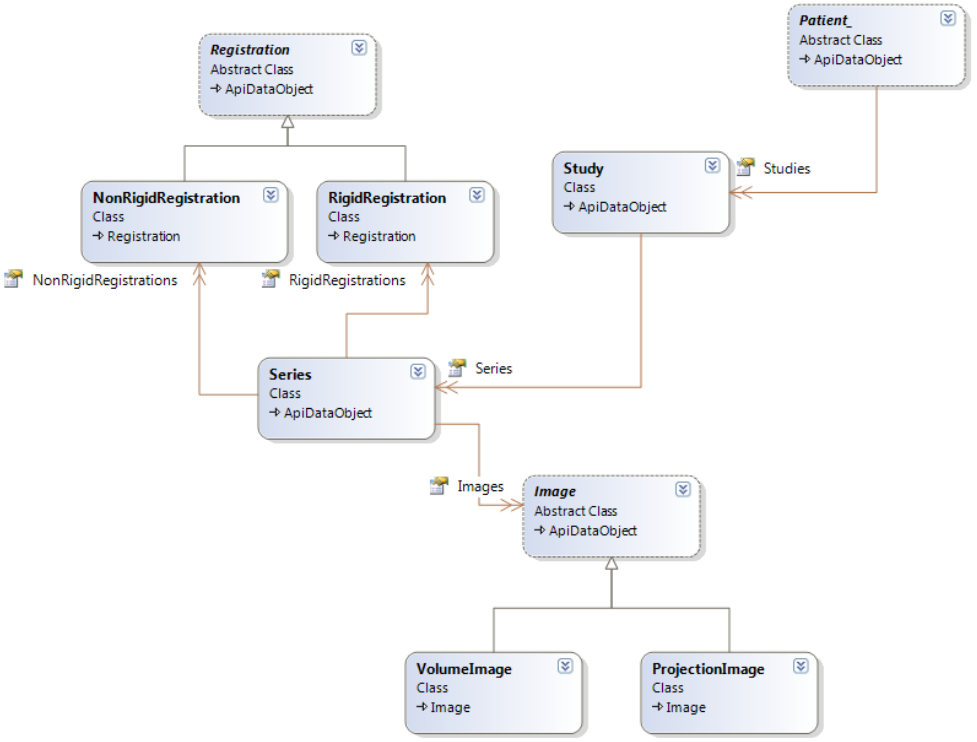

Figure 6 DICOM Patient-Study-Series Data Model

The diagram contains the following objects:

- *Study* containing a collection of studies.
- *Series* containing a collection of registrations (rigid and non-rigid) and images.

  A series always contains one single type of objects, for example only rigid registrations or only volume images.
- *Registration* representing a rigid or non-rigid spatial registration.
- *RigidRegistration* representing a rigid spatial registration, represented by a transformation matrix.
- *NonRigidRegistration* representing a non-rigid spatial registration, represented by a vector field.

Chapter 4 Getting Started

Getting Started with the Imaging and Registration Scripting API

To get quickly started with the Portal Dosimetry Scripting API, you can:

1. Copy the code shown below to a file.
2. Save the file with a *.cs* extension on the hard disk of your workstation.

```
using System;
using System.Linq;
using System.Text;
using System.Windows;
using System.Collections.Generic;
using VMS.CA.Scripting;

namespace VMS.DV.PD.Scripting {

  public class Script {

    public Script() {
    }
    public void Execute(ScriptContext context)
    {
      if (context.Patient != null)
      {
        MessageBox.Show("Patient id is " +
context.Patient.Id);
      }
      else
      {
        MessageBox.Show("No patient selected");
      }
    }
  }
}
```

3. In Portal Dosimetry, select **Tools** > **Scripts**.
4. To locate the script that you created, click **Change Directory** and navigate to the directory where you stored the script file.
5. In the **Scripts** dialog box, select the script from the list and click **Run**. The script displays a message box which contains the ID of the patient that is open in Portal Dosimetry.

Note: *It is recommended to use the Script Wizard to create a skeleton for a script. Refer to* Creating Scripts *on page 24 on instructions how to use the script wizard.*

Chapter 5 Using Examples

Using Example Scripts

The Portal Dosimetry Scripting API includes example scripts for each of the supported script types. You can use the Script Wizard to copy the example scripts:

1. From the **Tools** menu in Portal Dosimetry, select **Script Wizard**.
2. Click the **Copy Example Scripts** tab.
3. To select a location for copying the example scripts, click **Browse**.
4. Click **Copy**. The example scripts are copied to the specified location.
5. Open the Visual Studio project files, compile the examples, and launch them.

If you do not have Visual Studio available, you can compile the examples with the MSBuild program, which is included in the Microsoft .NET framework.

Use the following command line to compile the examples: C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe AnalysisStatistics.csproj /p:Platform=AnyCPU

Note: *It is recommended to install Microsoft Visual Studio 2010 (express edition or higher) for writing scripts. Visual Studio provides code completion and online help on methods and parameters as you type the code (IntelliSense). Microsoft Visual C# 2010 Express Edition is available free of charge from the Microsoft home page. It is not possible to use the Microsoft Visual C# 2012 Express Edition because it only supports Windows 8 style applications.*

Chapter 6 Creating Scripts

Creating Scripts

You can create scripts manually or by using the Script Wizard.

Creating Plug-in Scripts

The following sections give you step-by-step instructions on creating different types of plug-in scripts supported by the Portal Dosimetry Scripting API.

Creating Single-File Plug-ins with the Script Wizard

To create a single-file plug-in with the Script Wizard, follow these guidelines:

1. From the **Tools** menu in Portal Dosimetry, select **Script Wizard**.
2. Enter a name for the new script.
3. Select the **Single-file plug-in** option.
4. Select the **user interface library** to use in the script.
5. To select the location for storing the script, click **Browse**. By default, the script is stored in the *Documents/Portal Dosimetry Scripting API* folder.
6. Click **Create**.

   The Script Wizard creates the following folders in the location that you selected:
   - *Project* folder: Contains a script-specific subfolder where the Microsoft Visual Studio project file is stored. This project file is needed to provide IntelliSense support when the script code is edited in Microsoft Visual Studio.
   - *Plugins* folder: Contains the source code file for the single-file plug-in.
7. To edit the script, do one of the following:
   - Click **Open Visual Studio** to open the script in Visual Studio.

     The Script Wizard launches Microsoft Visual Studio. After the project is loaded in Visual Studio, double click on the script file in the Visual Studio solution explorer to open the file.
   - Click **Open folder** and open the script in a suitable program.
8. Edit the source code file according to your needs. You can use Visual Studio and its IntelliSense support for editing the file, but they are not required.
9. You do not have to compile the plug-in, because Portal Dosimetry compiles it automatically on the fly.

Note: *Do not add additional files to the Visual Studio project. Single-file plug-ins can only have one C# source code file. For projects requiring multiple source code files create a binary plug-in.*

Creating Binary Plug-ins with the Script Wizard

To create a binary plug-in with the Script Wizard, follow these guidelines:

1. From the **Tools** menu in Portal Dosimetry, select **Script Wizard**.
2. Enter a name for the new script.
3. Select the **Binary plug-in** option.
4. Select the **user interface library** to use in the script.
5. Click **Create**.
6. The Script Wizard creates the following folders in the location that you selected:
   - *Project* folder: Contains a script-specific subfolder where the Microsoft Visual Studio project file and source code file are stored.
   - *Plugins* folder: Contains the compiled plug-in DLLs. From this folder, the DLL can be loaded into Portal Dosimetry.
7. To edit the script, do one of the following:
   - Click **Open Visual Studio** to open the script in Visual Studio.

     The Script Wizard launches Microsoft Visual Studio. After the project is loaded in Visual Studio, double click on the script file in the Visual Studio solution explorer to open the file.
   - Click **Open folder** and open the script in a suitable program.
8. Edit the source code file according to your needs.
9. Compile the plug-in, for example, by using Visual Studio. The resulting plug-in DLL is saved into the *Plugins* folder. Note that you can also use the MSBuild tool to compile the binary plug-in. For more information about MSBuild, refer to Microsoft documentation.

Creating Single-File Plug-ins Manually

If you want to create a single-file plug-in without the Script Wizard, follow these guidelines:

1. Create an empty C# source code file.
2. Add the using statements for the `System` and `System.Windows` namespaces.
3. Add the using statements for the following namespaces:
   - `VMS.CA.Scripting`

4. Add a namespace called `VMS.DV.PD.Scripting`.
5. To the `VMS.DV.PD.Scripting` namespace, add a public class called `Script`.
6. To the `Script` class, add a constructor without parameters, and a method called `Execute`.
7. Define the return type of the `Execute` method as `void`.
8. To the `Execute` method, add the following parameters:
   - The context of the running Portal Dosimetry instance. The parameter type is `VMS.DV.PD.Scripting.ScriptContext`.
   - A reference to the child window that Portal Dosimetry creates for the user interface components (optional). The parameter type is `System.Windows.Window`.
9. You do not have to compile the plug-in, because Portal Dosimetry compiles it automatically on the fly.

Creating Binary Plug-ins Manually

If you want to create a binary plug-in without the Script Wizard, follow these guidelines:

1. Create the source code in the same way as for a single-file plug-in. For instructions, see Creating Single-File Plug-ins Manually on page 25.
2. Add references to the following class libraries of the Portal Dosimetry Scripting API:
   - `VMS.CA.Scripting.dll`
   - `VMS.DV.PD.Scripting.dll`

   On the basis of this information, the DLL can access the Portal Dosimetry Scripting API. These files are located in the installation directory of Portal Dosimetry.
3. Compile the plug-in into a .NET assembly (a DLL), for example, by using Visual Studio.

For more information on how to create a .NET assembly and add references to class libraries, refer to Microsoft documentation.

Debugging Binary Plug-in Scripts

To debug plug-in scripts, Microsoft Visual Studio is required. Make the following configuration in Visual Studio to debug a plug-in script:

Note: *This section describes the operations for Visual Studio 2010. In past or future releases, the steps might be different.*

1. Load the plug-in script project in Visual Studio.
2. In the solution explorer of Visual Studio, right click the plug-in project and select the **Properties** menu entry in the context menu. The project property pages are shown.
3. In the **Debug** tab, section **Start Action** choose the **Start external program** option and enter the full installation path of Portal Dosimetry.
4. In the **Start Options** section, enter "/standalone" as **command line argument** and enter the installation path of Portal Dosimetry as **working directory**.

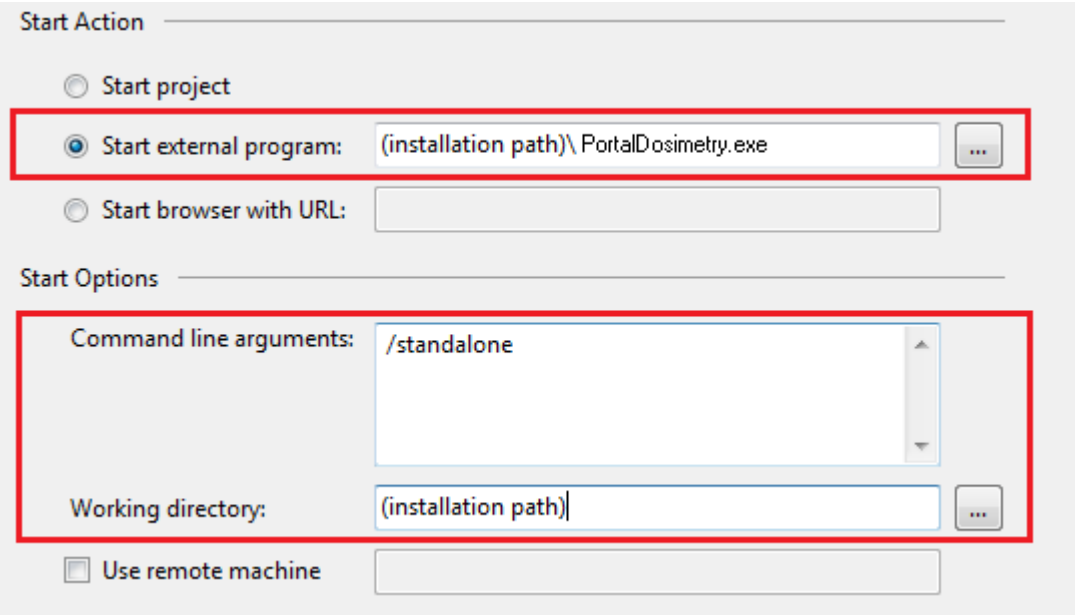

Figure 7 Debugging Options

5. Set breakpoints in the script as desired and start debugging mode with the F5 key. Portal Dosimetry starts up. In the **Tools-Scripts** dialog, select the script being debugged and click **Run**. Script execution stops at the defined breakpoints and the Visual Studio debugger is activated.

For more information on how to debug C# code, refer to Microsoft Visual Studio documentation.

Creating Stand-alone Executable Applications

The following sections give you step-by-step instructions on creating stand-alone executables supported by the Portal Dosimetry Scripting API.

Creating Stand-alone Executables with the Script Wizard

To create a stand-alone executable with the Script Wizard, follow these guidelines:

1. From the **Tools** menu in Portal Dosimetry, select **Script Wizard**.
2. Enter a name for the new script.
3. Select the **Standalone executable** option.
4. Select the **user interface library** to use in the script.
5. To select the location for storing the script, click **Browse**.
6. Click **Create**.
7. The Script Wizard creates a *Projects* folder in the location that you selected. The folder contains a script-specific subfolder where the Microsoft Visual Studio project file and source code file are stored.
8. To edit the script, do one of the following:
   - Click **Open Visual Studio** to open the script in Visual Studio.

     The Script Wizard launches Microsoft Visual Studio. After the project is loaded in Visual Studio, double click on the script file in the Visual Studio solution explorer to open the file.
   - Click **Open folder** and open the script in a suitable program.
9. Edit the source code file according to your needs.

For instructions how to compile the sources into an executable, refer to Microsoft Visual Studio documentation.

Creating Stand-alone Executables Manually

If you want to create stand-alone executables without the Script Wizard, follow these guidelines:

1. Create a new project file for the executable.
2. In the main method of the executable file, use the static `CreateApplication` method to create an instance of the `VMS.DV.PD.Scripting.Application` class. This class represents the root object of the data model. The `CreateApplication` method also initializes the Portal Dosimetry Scripting API.

3. Call the Dispose() method of the instance when the stand-alone executable exits to free the resources in the Portal Dosimetry Scripting API. For more information on disposing of objects, refer to Microsoft documentation of the IDisposable interface.
4. To the `CreateApplication` method, add the following parameters:
   - A user name and password for logging into the ARIA system. If you do not define the user name or password (values remain null), the system shows a log-in dialog requesting the user credentials.
5. Use a single-threaded apartment (STA) as the COM threading model of the executable. The Portal Dosimetry Scripting API must only be accessed from a single thread that runs in the default application domain. For more information about threading and application domains, refer to Microsoft documentation.

   The following is the code for a sample stand-alone executable in C# language:

```
using System;
using System.Linq;
using System.Text;
using System.Collections.Generic;
using VMS.CA.Scripting;
using VMS.DV.PD.Scripting;

namespace StandaloneExample {

    static class Program {
        [STAThread]
        static void Main(string[] args) {
            try {
                using (Application app =
Application.CreateApplication(null, null)) {
                Execute(app);
                }
            }
            catch (Exception e) {
                Console.Error.WriteLine(e.ToString());
            }
        }
        static void Execute(Application app) {
        string message =
        "Current user is " + app.CurrentUser.Id + "\n\n" +
        "The number of patients in the database is " +
        app.PatientSummaries.Count() + "\n\n" +
        "Press enter to quit...\n";
        Console.WriteLine(message);
        Console.ReadLine();
        }
    }
}
```

6. Insert an application configuration file (*app.config*) to the project. The application configuration file defines the location of the Portal Dosimetry binary files. The following shows the required settings. Remember to change the path according to the installation location of Portal Dosimetry on your workstation:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings>
    <add key="AssemblyPath"
         value="C:\Program Files (x86)\Varian
\PortalDosimetry\1.3\" />
  </appSettings>
</configuration>
```

7. Compile the project. The stand-alone executable is ready to be run.

   For more information on creating and compiling .NET applications, refer to Microsoft documentation.

Chapter 7 Launching Scripts

Launching Scripts

You can launch plug-in scripts from Portal Dosimetry, and stand-alone executables as any Windows application.

Launching Plug-in Scripts

To launch a plug-in script:

1. In Portal Dosimetry, select **Tools** > **Scripts**.

   The Scripts dialog box opens.
2. To locate the script that you want to run, select it from the list of scripts available in the scripts directory.
3. To change the script directory, click **Change Directory** and select a folder.

   All files with the .cs extension or with extension .dll become available on the list.
4. In the Scripts dialog, select the script file on the list.
5. Click **Run**.

Launching Stand-alone Executable Applications

You can launch a stand-alone executable like any Windows application on the workstation where Portal Dosimetry is installed. You can also debug the stand-alone executable using normal Windows debugging tools.

Index