Dantec™ KEYPOINT®

Integration Services

Reference Manual
© 2018, Natus Medical Incorporated. All rights reserved.

The contents of this manual are the property of Natus Medical Incorporated. Any reproduction in whole or in part is strictly prohibited.

At the time of printing / transferring to the CD-ROM/DVD, this manual correctly described the device and its functions. However, as modifications may have been carried out since the production of this manual, the product package may contain one or more addenda to the manual.

This manual including any such addenda must be thoroughly read before using the product.

The following situation voids any guarantee(s) and obligations for Natus Neuro:
- The product is not used according to the enclosed manuals and other accompanying documentation.

Natus Neuro is a division of Natus Medical Incorporated.

Natus Neuro and Keypoint are registered trademarks of Natus Neuro in the U.S. and in other countries.

Before using the product, make sure you read the instructions for use and fully understand how to use the product. Pay special attention to all Warnings and Cautions in this manual and in any other documentation accompanying the product.

For any further questions regarding the use of this product, please contact your customer support representative.
1 Introduction

Keypoint Integration Services (KPIS) provides a consistent interface for interaction with the Keypoint.NET application. Using Keypoint Integration Services, a programmer can create integration software to control and query Patient Study information in Keypoint.NET.

The interface to Keypoint Integration Services is implemented as a web service based protocol, following the Microsoft way of doing integration between disparate systems.

To maximize the number of potential integration platforms, Keypoint Integration Services is written as loosely coupled as possible. There is no need for compiling and linking to existing software modules and other systems can easily exchange data with Keypoint.NET using this interface.

For integration a with Hospital Information System (HIS) or Electronic Medical Record system (EMR), additional middleware is required in order to transform and exchange data between the HIS/EMR and Keypoint.NET. For such integrations a full HL7 Gateway product is offered which includes the required middleware. Please contact your product support representative for details.

This document is intended for developers planning to write middleware that integrates with Keypoint.NET v2.31 and newer. For integration with Keypoint.NET v2.21 and earlier, use related Keypoint Integration Services Reference Manual (9033M1466).

Along with this description of the interface, a simple application is available, as an example of how integration can be done using Microsoft C# .NET. Notice that the example software and related material is for demonstration purpose only.

2 Safety Requirements

This product is intended to be used by qualified personnel, knowledgeable in the field of information technology and with the appropriate education and training.

To prevent unauthorized monitoring of network traffic, a suitable layer of encryption must be used between HTTP and TCP. This could be accomplished using the HTTPS scheme.

Consideration should be given when enabling Keypoint.NET Automation (as described in chapter Automation), since this may alter the behavior of the Keypoint.NET Workstation.
## 3 System Requirements

<table>
<thead>
<tr>
<th>Component</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keypoint.NET</td>
<td>Keypoint Integration Services is a part of the Keypoint.NET product and can be enabled as required. For integration with Keypoint.NET v2.21 and earlier, use related Keypoint Integration Services Reference Manual (9033M1466).</td>
</tr>
</tbody>
</table>
| System                   | This product is designed for installation on a:  
- Keypoint.NET Workstation G3, G4 or Focus  
- Keypoint.NET Review Station  
- Servers meeting Keypoint.NET system requirements  
Refer to Keypoint.NET Release Note for system requirements. |
| Remote Control (Automation) | To enable Automation for remote control of a Keypoint.NET client, Microsoft Message Queue (MSMQ) must be installed.  
MSMQ is a part of the Microsoft Operating System and can be installed from the Control Panel – Programs and Features.  
Special configuration might be required as described in the Automation chapter. |
| Keypoint Data Center (Networking) | Networked solutions must be using the Keypoint Data Center product for central data storage.  
Keypoint.NET v2.31 and forward does not support networked solutions based on the older Polygram Business/Database Server. |
The Keypoint Integration Services is a part of the Keypoint.NET product and can be enabled as required. The enabling of Keypoint Integration Services depends on the networking solution and the planned usage of the provided Services.

The typical and recommended usage is to install Keypoint.NET and enable Keypoint Integration Services on the same computer also running the middleware.

The local Keypoint.NET client must be configured to use the shared Keypoint Data Center. This will provide a single point of entry for Keypoint Integration Services, to all Patient Study information from all the Keypoint.NET clients connected to the same Keypoint Data Center.

The following shows examples of different integration solutions.

### 4.1 Simple Networking
For a simple network as sketched on the figure, the Keypoint Review station is also used for hosting:
- Keypoint Data Center
- The middleware
- Keypoint Integration Services

### 4.2 Standard Networking
If 24/7 operation is required, the product should be installed on a Server. This would typically be required for HIS/EMR integration solutions. In this example the server is used for hosting:
- Keypoint Data Center
- The middleware
- Keypoint Integration Services
4.3 Remote Control

Keypoint.NET Automation allows outside clients to remotely control a Keypoint.NET Workstation to e.g. load a specific Patient Study or navigate to the Homepage.

Keypoint.NET Automation requires that the Microsoft Message Queue (MSMQ) is installed on the Keypoint.NET Workstation to be controlled.

Keypoint.NET Automation does not require that KPIS is enabled.

Refer to the Automation chapter for details about this functionality.
5 Installation

Before starting the installation ensure that:

- The specified System Requirements has been met
- The network topology as specified by the Integration Solution chapter has been considered
- The current user has Administrator rights on the local machine

To install Keypoint Integration Services, run the Setup.exe program located in the root of the installation media. Then follow the instructions provided on the screen.

For Keypoint.NET v2.31 and newer, the Keypoint Integration Services is installed as part of the Keypoint.NET product.
6 Configuration
Keypoint Integration Services is hosted as a Windows Service. The Service Name is ‘KeypointNET Integration Service’ and disabled (not running) by default.

6.1 Troubleshooting tool
For easy configuration of Keypoint Integration Services and the related Windows Service, start the Troubleshooting tool accessible from the Windows Start Menu under Keypoint.NET\Service Tools\KPIS:

Use the tool for automatic:
- Configuration and start of the Windows Service
- Modification of Service Base address
- Test using specified Endpoint Name
- Test of KPIS access to the Keypoint.NET database

The tool will update the KeypointNetServer.exe.config file as required.

6.2 Network check in mode
Keypoint Integrations Services acts as any other Keypoint.NET client is relation to data access. So in order for a Service to modify a given Patient Study, it has to be reserved/checked out to ensure exclusive access and ensure that other Keypoint.NET clients are not modifying the same Patient Study.

When using the provided Services, the Patient Study is automatically reserved/checked out, if not reserved by another client. After a period of 30s, the Patient Study is automatically released/checked in to the server again. The 30s delay is a caching mechanism to optimize performance when multiple Services are requested on the same Patient Study.

The release/check in process is normally handled by the running Keypoint.NET application. However for Server based solutions, this must be handled by the KeypointNET Windows Service, so this process is independent of having the
Keypoint.NET application running.

The *Network check in mode* can be set on the Networking tab in Keypoint.NET Administrative Tools.

Notice that after changing the *Network check in mode*, the KeypointNET Windows Service, must be restarted. This is done be rebooting the computer or by manually restarting the KeypointNET Service using Windows Administrative Tools.

### 6.3 Auto delete of Un-processed orders

Using Keypoint Data Center for the central data storage also provides the option for auto deletion of obsolete Patient Studies (Un-processed orders). A stored procedure named ScheduleCleanUpEmptyStudies deletes all Studies without performed Tests older than the specified date range. Refer to the Keypoint Data Center Reference Manual for details.
7 Interface Overview

Interfacing with the Keypoint.NET software can be accomplished using a web service based protocol as described in this section.

7.1 Parameter Restrictions

In Keypoint.NET every Study has a patient attached to it. The only way to perform another Study on the same patient returning to the clinic/hospital is to create a new Patient Study. To be able to differentiate among the studies, the external system utilizing the Keypoint Integration Services must provide a unique ID of the Patient Study.
Every operation of the Integration Services requires the ID identifying the combined Patient Study.

**NOTE: External system must provide a unique ID for every Patient Study**

In addition, all textual parameters should be escape encoded as specified in the HTML 4.0 standard, meaning that a `<` should be represented as `&lt;` and so forth. For a complete list of the escape sequences, consult the following W3C recommendation: [http://www.w3.org/TR/REC-html40/sgml/entities.html](http://www.w3.org/TR/REC-html40/sgml/entities.html).

7.2 WCF Service

Keypoint Integration Services (KPIS) is exposed through three WCF Endpoints hosted as a Windows Service. The Service Name is ‘KeypointNET Integration Service’ and disabled (not running) by default.

The base port is defined by the KeypointNetServer.exe.config file. The default port number is 8731. If the port number is changed the service must be restarted.

<table>
<thead>
<tr>
<th>EndPoint Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asmx</td>
<td>Provides backwards compatibility to previous versions of KPIS. It uses the WCF basicHttpBinding that uses SOAP 1.1. The default endpoint address is: <a href="http://localhost:%3Cport%3E/KPIIntegrationService/asmx">http://localhost:&lt;port&gt;/KPIIntegrationService/asmx</a></td>
</tr>
<tr>
<td>ws</td>
<td>wsHttpBinding - SOAP 1.2</td>
</tr>
<tr>
<td>web</td>
<td>webHttpBinding for addressing through a web browser. Primarily for testing purposes. This endpoint will not be described or supported.</td>
</tr>
</tbody>
</table>

For easy configuration of Keypoint Integration Services and the related Windows Service, start the KPIS Troubleshooting tool as described in the Configuration chapter.
The tool will update the KeypointNetServer.exe.config file as required.

7.3 Authentication

Keypoint Integration Services is dependent on access to both the local Polygram database and Keypoint Data Center where used.
Access to the local database is using Windows Authentication while access to Keypoint Data Center supports SQL authentication as well. Database access is configured using Keypoint Administrative Tools.
Keypoint Integration Services supports impersonation, which allows the user credentials from the calling application (middleware) to be used when accessing required components.

Pay attention to impersonation limitations which will be determined by the lowest combined level between the calling user account and the user account specified for the Windows Service.

By default Keypoint Integration Services (the Windows Service) is running under the Local System account which should be sufficient.

Similar when using Services that generates a file in a specified target, sufficient credentials must be available for the Service to access and generate a file in the specified folder location.

Remote access to Keypoint Integration Services from another machine is also possible. This also requires that the calling user account can provide sufficient credentials to access the databases.

### 7.4 WSDL interface

The Web Services Description Language (WSDL) for Keypoint Integration Services is available using the following URL:

http://<ComputerName>:<port>/KPIntegrationService?wsdl

To be able to access the services, use your preferred web service development tool (i.e. Visual Studio, WebSphere) and fetch the WSDL interface specification from the server by specifying the URL within the development tool.

The tool will then generate an interface, as described in the next chapter.

Using Microsoft Visual Studio, C#.NET the interface can be generated by providing a reference to the running service:

![Visual Studio screenshot](image)

Subsequently services can be accessed from your code like displayed below.
7.5 Logging

For debugging the service, logging can be enabled. In the KeypointNETServer.exe.config file please find:

```xml
<applicationSettings>
  <KPISWcfServiceLibrary.Properties.Settings>
    <setting name="LogLevel" serializeAs="String">
      <value>None</value>
    </setting>
  </KPISWcfServiceLibrary.Properties.Settings>
</applicationSettings>
```

Change the value of the setting named “LogLevel” by editing the value of the enclosed `<value>` node from “None” to “Extended”. When the service is restarted, it will begin writing log entries into a file named KPISService.log in the binaries folder when the service methods are called.

7.6 Testing

For testing the Services, start the Test Integration Services tool accessible from the Windows Start Menu under Keypoint.NET\Service Tools\KPIS:

The tool allows execution of each of the provided Services and displays the result returned by the Service.
8 Description of Services

For integration with Hospital Information System (HIS), the HL7 protocol is commonly used. The services available using Keypoint.NET Integration Services has been grouped in the following HL7 message categories:

- Admitting, Discharge, and Transfer Transaction (ADT)
- Detailed Financial Transaction (DFT)
- Medical Document Management (MDM)

To ensure backward compatibility, interfaces for existing Services always remain unchanged. If new functionality is required for an existing Service, a new interface version of the Service is added.

8.1 Overview

The following table lists the available Services including the version where the Service was added:

<table>
<thead>
<tr>
<th>Category</th>
<th>Service</th>
<th>Added / Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADT</td>
<td>CreatePatientStudy</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>CreatePatientStudy2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CreatePatientStudy3</td>
<td>v2.11 – Additional parameters</td>
</tr>
<tr>
<td></td>
<td>StudyExists</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>DeletePatientStudy</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetIsStudyComplete</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetIsStudyLocked</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetListOfPatientStudyParameters</td>
<td>v2.11- Possible parameters and labels</td>
</tr>
<tr>
<td></td>
<td>GetPatientStudyParameter</td>
<td>v2.11- Reading specific parameter</td>
</tr>
<tr>
<td></td>
<td>UpdatePatientStudyParameter</td>
<td>v2.11- Update of parameters</td>
</tr>
<tr>
<td></td>
<td>GetStudyConclusion</td>
<td>v1.10/v2.11 – now RTF format</td>
</tr>
<tr>
<td>DFT</td>
<td>GetStudyList</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetStudyList2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>GetStudyList3</td>
<td>v2.11- XPath + additional info</td>
</tr>
<tr>
<td></td>
<td>GetListOfPerformedTests</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetReviewData</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetStudyData</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetCompletedStudies</td>
<td>v2.20</td>
</tr>
<tr>
<td>MDM</td>
<td>GetListOfAvailableReports</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetListOfAvailableReports2</td>
<td>v2.11 – To support PDF files</td>
</tr>
<tr>
<td></td>
<td>GetListOfAvailableReportTemplateNames</td>
<td>v2.11 – To support PDF files</td>
</tr>
<tr>
<td></td>
<td>GetFirstPatientStudyReport</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetPatientStudyReportByName</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>GetPatientStudyReportByName2</td>
<td>v2.11 – To support PDF files</td>
</tr>
<tr>
<td>Other</td>
<td>Ping</td>
<td>v1.10</td>
</tr>
<tr>
<td></td>
<td>Version</td>
<td>v1.10</td>
</tr>
</tbody>
</table>

Details about each Service are described in the following sections.
8.2 Create Patient Study

The CreatePatientStudy service creates a new Patient Study in the Keypoint.NET database:

- `string CreatePatientStudy(<Parameter Set>)`
- `string CreatePatientStudy2(<Parameter Set>)`
- `string CreatePatientStudy3(<Parameter Set>)`

The different services provide the same functionality, but with different set of input parameters, having v3 supporting the maximum parameter set:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Name</th>
<th>Req.</th>
<th>Example</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>dateofstudy_day</td>
<td>Y</td>
<td>31</td>
<td>1..31</td>
</tr>
<tr>
<td>int</td>
<td>dateofstudy_month</td>
<td>Y</td>
<td>12</td>
<td>1..12</td>
</tr>
<tr>
<td>int</td>
<td>dateofstudy_year</td>
<td>Y</td>
<td>2006</td>
<td>Valid year</td>
</tr>
<tr>
<td>string</td>
<td>Studyid</td>
<td>N</td>
<td>&quot;a-zA-Z0-9&quot;</td>
<td>Max 256 char's</td>
</tr>
<tr>
<td>string</td>
<td>Patientid</td>
<td>N</td>
<td>&quot;a-zA-Z0-9&quot;</td>
<td>Max 256 char's</td>
</tr>
<tr>
<td>string</td>
<td>Studyuniqueid</td>
<td>N</td>
<td>&quot;a-zA-Z0-9&quot;</td>
<td>HIS system external ID, Max 256 char's</td>
</tr>
<tr>
<td>string</td>
<td>Firstname</td>
<td>N</td>
<td>&quot;John&quot;</td>
<td>Max 256 char's</td>
</tr>
<tr>
<td>string</td>
<td>Middlename</td>
<td>N</td>
<td>&quot;Fabricius&quot;</td>
<td>Max 256 char's</td>
</tr>
<tr>
<td>string</td>
<td>Lastname</td>
<td>N</td>
<td>&quot;Jensen&quot;</td>
<td>Max 256 char's</td>
</tr>
<tr>
<td>int</td>
<td>dateofbirth_day</td>
<td>Y</td>
<td>31</td>
<td>1..31</td>
</tr>
<tr>
<td>int</td>
<td>dateofbirth_month</td>
<td>Y</td>
<td>12</td>
<td>1..12</td>
</tr>
<tr>
<td>int</td>
<td>dateofbirth_year</td>
<td>Y</td>
<td>1950</td>
<td>1870..2200</td>
</tr>
<tr>
<td>bool</td>
<td>IsMale</td>
<td></td>
<td>True, False</td>
<td></td>
</tr>
<tr>
<td>float</td>
<td>Height</td>
<td>Y</td>
<td>186</td>
<td>Unit = cm</td>
</tr>
<tr>
<td>float</td>
<td>weight</td>
<td>Y</td>
<td>80</td>
<td>Unit = Kg</td>
</tr>
<tr>
<td>bool</td>
<td>IsRightHanded</td>
<td></td>
<td>True, False</td>
<td></td>
</tr>
<tr>
<td>string</td>
<td>Technician</td>
<td>N</td>
<td>&quot;Mrs. Tech&quot;</td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>Physician</td>
<td>N</td>
<td>&quot;Mr. Phys&quot;</td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>Code</td>
<td>N</td>
<td>&quot;C27&quot;</td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>Reffering Physician</td>
<td>N</td>
<td>&quot;Phys X1&quot;</td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>Reffering Depart.</td>
<td>N</td>
<td>&quot;Dep X1&quot;</td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>diagnosis1</td>
<td>N</td>
<td></td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>diagnosis2</td>
<td>N</td>
<td></td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>diagnosis3</td>
<td>N</td>
<td></td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>diagnosis4</td>
<td>N</td>
<td></td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>Comment</td>
<td>N</td>
<td></td>
<td>Max 4000 char's</td>
</tr>
<tr>
<td>string</td>
<td>Conclusion</td>
<td>N</td>
<td></td>
<td>Max 4000 char's</td>
</tr>
<tr>
<td>string</td>
<td>Memo1</td>
<td>N</td>
<td></td>
<td>Max 4000 char's</td>
</tr>
<tr>
<td>string</td>
<td>Memo2</td>
<td>N</td>
<td></td>
<td>Max 4000 char's</td>
</tr>
<tr>
<td>string</td>
<td>Memo3</td>
<td>N</td>
<td></td>
<td>Max 4000 char's</td>
</tr>
<tr>
<td>string</td>
<td>CaseID</td>
<td>N</td>
<td></td>
<td>Max 50 char's (HIS)</td>
</tr>
<tr>
<td>string</td>
<td>OrderID</td>
<td>N</td>
<td></td>
<td>Max 50 char's (HIS)</td>
</tr>
<tr>
<td>string</td>
<td>mpField1</td>
<td>N</td>
<td></td>
<td>Max 50 char's</td>
</tr>
<tr>
<td>string</td>
<td>mpField2</td>
<td>N</td>
<td></td>
<td>Max 50 char's</td>
</tr>
</tbody>
</table>

The string data type is a Unicode string.

These services return a unique identifier, referred to as the Internal Unique ID for that Patient Study. The Internal Unique ID can be used to refer to that particular Patient Study on subsequent calls. Alternatively, the supplied Studyuniqueid (External ID) parameter can serve this purpose.

The advantage of using the return value is that this is guaranteed to be unique. There is nothing in the Keypoint.NET that ensures a unique Studyuniqueid parameter as input to the methods above. This is to be controlled externally ("placer-id" in HL7 terms).
To determine which existing Patient Study a client is referring to, the Keypoint.NET system does the following matching:

**First attempt:**
Match `Studyuniqueid` to the internal identifier (Internal Unique ID).

**Second attempt:**
Match `Studyuniqueid` to the externally supplied identifier (`Studyuniqueid`).

### 8.3 Study Exists

This Service checks if a Patient Study with the specified ID exists in the Keypoint.NET database:

```csharp
bool StudyExists(string studyuniqueid);
```

### 8.4 Delete Patient Study

This Service deletes a Patient Study in the Keypoint.NET database:

```csharp
void DeletePatientStudy(string studyuniqueid, bool Force);
```

Note that the `Force` parameter has the following meaning:

- `false` : Patient Study will not be deleted if it contains captured data
- `true` : Delete Patient Study regardless of captured data

### 8.5 Study State

Use these Services to query the current status of a Patient Study. Two separate status bits are available.

```csharp
bool GetIsStudyComplete(string studyuniqueid);
bool GetIsStudyLocked(string studyuniqueid);
```

The “complete” status can be altered in the Keypoint.NET user interface, whereas the “locked” status is automatically set whenever a Patient Study is opened in the Keypoint.NET application, and reset when the user leaves the Patient Study. If a Patient Study is “locked” by another user it cannot be modified.

### 8.6 Patient Study Parameters

This Service returns a list of available Parameters for a Patient Study. It also includes the label used within the Keypoint.NET application. Notice that labels can be modified by the user in the Keypoint.NET application.

```csharp
XmlDocument GetListOfPatientStudyParameters();
```

For a given Patient Study each of the available parameters can be received and updated. The purpose is to allow update of existing Patient Study data as required to support HL7 ADT messages:

```csharp
bool GetPatientStudyParameter( string ParameterName, string studyuniqueid)
bool UpdatePatientStudyParameter( string ParameterName, string Value, string studyuniqueid)
```
8.7 **Patient Study Conclusion**

The Patient Study conclusion is available through this Service:

```csharp
string GetStudyConclusion(string studyuniqueid)
```

Note that previously the conclusion was returned in HTML format. This has been changed to an RTF formatted string. For proper presentation of this, some kind of RTF parsing or rendering is needed.

8.8 **Patient Study List**

These Services return an XML document with a list of the Patient Studies located in the Keypoint.NET database.

```csharp
XmlDocument GetStudyList()
XmlDocument GetStudyList2(string XPathFilter)
XmlDocument GetStudyList3(string XPathFilter)
```

V2 & V3 allow filtering of the returned Patient Studies based on the provided XPath search filter. This makes it possible to query for Patient Studies meeting specific criteria’s. E.g. “//*[@CreatedDateXpath=20110203102753]” where the number represents the Patient Study creation date.

Note that filtering is only possible on the fields available in the resulting XmlDocument.

8.9 **List of Performed Tests**

This Service returns a list of Keypoint.NET IDs describing the Tests that have been performed for a given Patient Study:

```csharp
string[] GetListOfPerformedTests(string studyuniqueid).
```

The GetReviewData Service returns an xml document, containing the same information as can be found in the Review panel in the Keypoint.NET application. This includes the following information for each Test performed:

- Unique ID of Test type
- Name
- Anatomy information

```csharp
XmlDocument GetReviewData(string studyuniqueid)
```

8.10 **Patient Study Report**

These Services relate to Patient Study reports, which have been generated and saved as part of a Patient Study in the Keypoint.NET application.

A list of reports saved for a given Patient Study can be retrieved through the method:

```csharp
string[] GetListOfAvailableReports(string studyuniqueid)
```
string[] GetListOfAvailableReports2(string studyuniqueid)

For testing purpose, the interface can return first found report for a given ID. The sorting order is database specific. The destinationfullpath must include the file name.

void GetFirstPatientStudyReport (string studyuniqueid, string destinationfullpath)

If the name of a report is known, that report can be saved to a specified location on disk using the function (full path must include the file name).

void GetPatientStudyReportByName ( string studyuniqueid; string reportname; string destinationfullpath)

void GetPatientStudyReportByName2(string studyuniqueid; string reportname; string destinationfullpath string reportType)

Using v2 of this Service is required to retrieve PDF reports generated by the Keypoint.NET application.

8.11 Get Study Data

Given a Patient Study ID and the name of a report template present in the Keypoint.NET system, Patient Study results corresponding to the setup in the report template will be generated and returned as XML.

XmlDocument GetStudyData(string studyuniqueid, string reportTemplateName)

Using this Service, it is possible to extract specific parameter values from performed Tests, for further processing.

Refer to the appendix for an overview of the data which can be extracted using this method.

To get a list of the report templates available in Keypoint.NET use this Service:

XmlDocument GetListOfAvailableReportTemplateNames()

8.12 Get Completed Patient Studies

This service returns a list of Patient Studies marked as completed after the specified point in time. The parameter must be specified in the W3C datetime format:

- YYYY-MM-DDThh:mm:ss.

Using this Service, it’s possible to poll for changes and react if Patient Studies have been changed since last poll.

XmlDocument GetCompletedStudies(string startdatetime)
8.13 Other

This Service can be used to verify the state of the web service. Both the access rights and the database access are verified with this method. If the ping succeeds, a value of ‘true’ is returned.

bool Ping()

The following Service returns the version number of the installed Keypoint.NET Integration Services:

string Version()
9 Automation

Keypoint.NET Automation allows outside clients to remotely control a Keypoint.NET Workstation to navigate to different pages (e.g. from Homepage to Study List, etc.). This can be accomplished by posting a message to the Keypoint.NET workstation on a Microsoft Message Queue.

Whenever the Keypoint.NET application is running, a private message queue is created, in which the application listens for new messages. When a message arrives, the application will attempt to execute the command found in the message.

Note that there is no guarantee that the application will execute the command. If the application is busy or locked, the command will not be executed. Furthermore, the command needs to be well formed.

As of now, there is no acknowledgement of messages received in the application. Monitoring of requests can be done through the event viewer in the operating system, where all incoming commands are logged.

9.1 The Message Queue

The name of the message queue is `private$\ KPIntegrationServicesQueue`. If Automation has been enabled and the application is running, it can be verified on the Workstation using the tool found in Computer Management on the local machine:

It is the responsibility of the Keypoint.NET application to create and destroy this queue.

The following illustration shows the Message Queue Path in the computer management console.
9.2 Enabling Automation
To enable automation, it is necessary to create a key in the registry. This key is named `HKEY_LOCAL_MACHINE\SOFTWARE\Medtronic\ConnectivityServices\KPISAutomationEnabled`, and should have the value ‘1’.

Special configuration of the Firewall might be required. — E.g., for Windows Vista the TCP Port 135 must be opened if the firewall is turned on.

9.3 Messages
Messages contain commands in the form of text strings. The following commands are available:

LOADSTUDY [studyuniqueid]
This will load the Patient Study with the given ID, and display the Patient data page.

NEWSTUDY [studyuniqueid]
This will create a new Patient Study with the given ID and display the Patient data page.

GOTO [Home | Studylist]
This will display either the Homepage or the Studylist.
10 Appendix: The Test Application

To illustrate how Keypoint.NET Integration Services may be used, a simple test application is available. The application has been developed to act as a simple hospital information system, allowing for handling of Patient Study data. The source code, written in C#.NET is available in the KPIS example folder on the installation media.

The following illustration shows the main form of the test application.

A list of Patient Studies is displayed in the top panel (randomly generated data), and the bottom panel holds a log, displaying the interaction with Keypoint.NET Integration Services.

To configure the test application, select the ‘Options’ menu item from the ‘File’ menu.
The name of the server where Keypoint.NET Integration Services is installed should be specified. If no server is specified, localhost:8731/KPIntegrationService/asmx is assumed.

The options menu suggests a wider range of protocols (i.e. file based or HL7). None of these protocols are implemented in the current version of Keypoint.NET Integration Services; however, they may be available on a future release.

When the name of the server has been specified, patient data can easily be transferred to Keypoint.NET. Simply, select a patient in the list of patients, and select ‘Patient data’ -> ‘Create study’.
The XML data returned using the GetStudyData Service is based on a Keypoint.NET report template. The following table describes the available report elements:

<table>
<thead>
<tr>
<th>Ref</th>
<th>Keypoint.NET Report Element</th>
<th>Supported</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Patient Data</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>Memo Field</td>
<td>Yes</td>
</tr>
<tr>
<td>3</td>
<td>NC Data Table</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>Report EMG Table</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>Report EMG Table Item</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>Report EP Table</td>
<td>Yes</td>
</tr>
<tr>
<td>7</td>
<td>RNS Table</td>
<td>Yes</td>
</tr>
<tr>
<td>8</td>
<td>Impedance</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>Blink Reflex Table</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>EMG Findings</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>Stim SF EMG</td>
<td>No</td>
</tr>
<tr>
<td>12</td>
<td>Voluntary SF</td>
<td>No</td>
</tr>
<tr>
<td>13</td>
<td>Report Image</td>
<td>No</td>
</tr>
<tr>
<td>14</td>
<td>Page Breaks</td>
<td>No</td>
</tr>
<tr>
<td>15</td>
<td>RNS bar/trend plot</td>
<td>No</td>
</tr>
<tr>
<td>16</td>
<td>EMG Plot</td>
<td>No</td>
</tr>
<tr>
<td>17</td>
<td>Curve Item (NC plots)</td>
<td>No</td>
</tr>
<tr>
<td>18</td>
<td>Blink Reflex table and curves</td>
<td>No</td>
</tr>
<tr>
<td>19</td>
<td>EMG Epochs</td>
<td>No</td>
</tr>
<tr>
<td>20</td>
<td>EMG MUPs</td>
<td>No</td>
</tr>
<tr>
<td>21</td>
<td>F Data &amp; F Curves (F-Wave data)</td>
<td>No</td>
</tr>
<tr>
<td>22</td>
<td>Separate Tests</td>
<td>No</td>
</tr>
<tr>
<td>23</td>
<td>EMG Monitor</td>
<td>No</td>
</tr>
<tr>
<td>24</td>
<td>RR</td>
<td>No</td>
</tr>
<tr>
<td>25</td>
<td>SF EMG</td>
<td>Yes</td>
</tr>
<tr>
<td>26</td>
<td>M-Increment Data and Curves</td>
<td>No</td>
</tr>
<tr>
<td>27</td>
<td>Macro EMG</td>
<td>No</td>
</tr>
</tbody>
</table>

(*) The presence of these report elements in the used report template is not allowed and could potentially cause an empty XML report to be returned.

In general, Plot data & Curve data are not available.

The ExportManagerSchema.xsd file is used for validation of the generated XML report. The file is located in the KPIS\WebService\bin folder.

A successful validation requires that Result tables only include data from a single Test type. — E.g. the NC Data Table must not contain data from both Motor and Sensory Tests. Instead, multiple instances of the NC Data Table element can be defined in the report template.
12 Troubleshooting

For configuration and troubleshooting of Keypoint Integration Services it’s recommend to use the Troubleshooting tool as described in the Configuration chapter. For manual troubleshooting follow the step below.

1. • Start Keypoint.NET  
   • Create a new Study  
   If this fails, there might be a database issue.  
   Please refer to KeypointNET troubleshooting.

2. • Open the Windows Service Manager  
   (Windows 7: In Start Menu search box enter Services.msc)  
   • Locate the KeypointNET Integration Services  
   • Check that start type is Automatic and service is started  
   If the service is not started, right click the service and press Start.

3. • Start the Test Integration Services tool  
   (Windows Start Menu under Keypoint.NET\Service Tools\KPIS)  
   • Select “Ping” Service and press Execute  
   • “True” must appear. See screen dump below.
4. Enable logging and restart service:
Find and open KeypointNETServer.exe.config file. This can be found in the KeypointNET Installation Folder – ex.: C:\Program Files(x86)\KEYPOINT.NET\Common\MDT\Binaries\KeypointNETServer.exe.config.

Locate:

```xml
<applicationSettings>
    <!--KPISWcfServiceLibrary.Properties.Settings-->
    <setting name="LogLevel" serializeAs="String">
        <value>None</value>
    </setting>
    <!--/KPISWcfServiceLibrary.Properties.Settings-->
</applicationSettings>
```

Change the value of the setting named “LogLevel” by editing the value of the enclosed `<value>` node from None to Extended. When the service is restarted (using the Windows Service Manager see Troubleshooting step 2), it will begin writing log entries into a file named KPISService.log in the binaries folder when the service methods are called.

In the test app (see Troubleshooting step 3) click Ping”.

Open the log file (KPISService.log) and see if and entry with a new timestamp and the text Ping() has been written. If it is written and Keypoint works, Keypoint Integration Services will work.

5. In the the KeypointNET Installation Folder – ex.: C:\Program Files(x86)\KEYPOINT.NET\Common\MDT\Binaries\KeypointNETServer.exe.config please find the text that looks similar to:

```
<host>
    <baseAddresses>
        <add baseAddress="http://localhost:8731/KPIntegrationService" />
    </baseAddresses>
</host>
```

Write down the port number – in this case 8731.

Open the KeypointIntegrationServicesTest.exe.config file. This can be found in the KeypointNET Installation Folder – ex.: C:\Program Files(x86)\KEYPOINT.NET\Common\MDT\Binaries\KPISTestApp\KeypointIntegrationServicesTest.exe.config

Find the text:

```
<endpoint address="http://localhost:8731/KPIntegrationService/ws" behaviorConfiguration="ClientEndpointCredentialBehavior" binding="wsHttpBinding" bindingConfiguration="WSHttpBinding_KPIntegrationServices" contract="KPIntegrationServiceReference.IKPIntegrationServices" name="WSHttpBinding_KPIntegrationServices" />
```

It is important, that the portnumber (in this case 8731) is identical to the port number written down previously. If it is not, change the port number accordingly and save the file.

(Re)start the KeypointIntegrationServicesTest.exe test program and click Ping”.
Now, the right section should show “True”.
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 6. | Click GetStudyList.  
Check that the text in the right section starts with “<StudyList xmlns=""””, Keypoint  
Integration Services is running. |
| 7. | Otherwise, copy all text from the right section and contact your product support  
representative. |
Please consult www.natus.com for your local sales & service office.

Natus Manufacturing Limited
IDA Business Park Gort, Co.
Galway Ireland
Fax: +45 4457 9010

Rx only

Reg. No. 9033M1467 Rev 03