EUROPEAN ORGANIZATION FOR NUCLEAR RESEARCH
ORGANISATION EUROPÉENNE POUR LA RECHERCHE NUCLÉAIRE

CERN - ST Division

CERN-ST-2001- 058
20 February 2001

ACCESSING THERMODYNAMIC FLUID PROPERTIES IN LABVIEW

Steffen Grohmann, ST-CV

Abstract

LabVIEW users that are experimenting with fluid circuits and want to have an online evaluation of measuring data need to import thermodynamic properties of the working fluid. This document describes how CRYODATA's software package *GASPAK* can be accessed from LabVIEW via ActiveX Automation. An interface server has been developed to enable inter-application communication. The installation of the server is explained and its function calls are demonstrated in a set of example VIs.

1 INTRODUCTION

Experimenting with fluid circuits in cryogenics and refrigeration is common business at CERN, where National Instruments *LabVIEW*[1] is often used for data acquisition. When performing experiments, users often face the problem that they have to convert sensor readings offline into thermodynamically meaningful data to know about the present state of their system.

One tool to calculate fluid properties from thermodynamic equations is CRYODATA's *GASPAK*[2], which provides fluid state and transport properties of 38 working fluids including all noble gases. Both *LabVIEW* and *GASPAK* are available as site licenses at CERN.

Direct access to thermodynamic fluid properties in LabVIEW simplifies operation and control of fluid circuits. Some examples shall illustrate the benefit:

- monitoring of the saturation temperature as function of a pressure sensor signal for calibration of temperature sensors,
- monitoring and control of the cooling capacity (calculation of enthalpies from temperature and pressure readings together with mass-flow data),
- monitoring and control of two-phase flow, e.g. inlet and outlet qualities of the fluid passing through a heat exchanger.

In order to access GASPAK's thermodynamic equations in LabVIEW, one has to establish interaction between the two, with LabVIEW as client calling the server application GASPAK. Windows[3] 32-bit operating systems provide a facility called *ActiveX Automation* to solve this task.

ActiveX is a diverse set of technologies based on the Component Object Model (COM) for inter-application communication. The standard allows developers to create code and applications from any languages, and build a defined interface to that code, making it easily accessible by other applications. LabVIEW supports the *ActiveX Automation* and *ActiveX Container* technologies, where only the *ActiveX Automation* is needed in this case.

2 SYSTEM REQUIREMENTS

The software has been developed and tested on Windows NT[3]. However, it should run on any Windows 32-bit operating system with the developed interface server being installed and registered in the system directory of the operating system (see chapter 5).

3 SOFTWARE RELEASES

3.1 LabVIEW

The inter-application communication between LabVIEW and GASPAK has been developed and tested with LabVIEW 5.1 and all example VIs (virtual instruments) in 'chapter 6' are written in LabVIEW 5.1.

For details on LabVIEW's general functionality and its ActiveX features please refer to the LabVIEW user manuals and the National Instruments online documentation.

3.2 GASPAK

The GASPAK Version 3.30 used here is an integrated package of thermodynamic equations for properties of 38 fluids, including all noble gases and a number of common refrigerants. The equations

[1] LabVIEW is a trademark of National Instruments Corporation.
[2] GASPAK software is protected by USA copyright 8 1999 Cryodata Inc. All rights reserved.
[3] Windows and Windows NT are trademarks of Microsoft®

are valid from the triple point and melting line through the compressed liquid and the two-phase region to well above room temperature and several hundred bar of pressure (upper temperature and pressure limits vary from fluid to fluid). For details on standard reference data, accuracy and limits of the individual fluids, the list of fluids and properties and the conventions on function calls please refer to the GASPAK user's guide [1].

GASPAK 3.30 provides a 32-bit ActiveX server DLL (dynamic link library), which is called *GaspakGd5.dll*. This server utilizes the GASPAK properties engine *Gaspkdll.dll*. Both DLLs are installed in the system directory of the computer. Coefficient files (**.gas*) are needed for each working fluid. These files are installed in the installation directory of the GASPAK application. In principle one could access the *GaspakGd5.dll* directly from LabVIEW. However, it has some significant drawbacks that make its use inconvenient and troublesome for safe operation.

The first disadvantage is that a routine to check for errors is not implemented in the server DLL, but has to be programmed externally. This involves an additional call to the server with some input parameters deviating from the parameters specified by the user (see example for macro programming in Microsoft Excel[4] in [1]). The intermediate result has to be evaluated in a certain procedure. Programming of this error handling directly in LabVIEW would lead to a complicated and confusing block diagram.

The major drawback, however, is the way in which the fluid coefficients are retrieved from the coefficient files (*.gas). These files always have to be in the present working directory, i.e. the directory from where *GaspakGd5.dll* is called. This means in practice that copies of all fluid files needed are to be placed in every single directory where users store their LabVIEW VIs. As one changes the working directory, e.g. during the development of a project to open another subVI, the VI calling *GaspakGd5.dll* would not work any longer. Furthermore, it is not possible to store the coefficient files within a LabVIEW library and one would need to distribute them separately.

In order to facilitate safe access to GASPAK in LabVIEW, an interface ActiveX server has been developed that solves all the problems mentioned before. Its features are explained below.

4 THE INTERFACE SERVER GaspakLV

The ActiveX server *GaspakLV.dll* has been developed to provide easy access for LabVIEW users to GASPAK. The code is written and compiled in Microsoft Visual Basic[5] 5.0.

The interface library calls CRYODATA's *GaspakGd5.dll* and performs the error check routine (see [1]) internally. During execution of the code the working directory is first changed to the fluids coefficient file directory *FldFileDir* and set back to the original directory at the end. *FldFileDir* is passed through the function calls. If it points at the GASPAK installation directory there is no need to scatter the fluid coefficient files (*.gas) around the hard disk. The change of directories takes, of course, some time. However, time-sensitive LabVIEW applications are not affected since ActiveX Automation runs in a different thread on multithreaded platforms.

The *GaspakLV.dll* contains one COM-based class, which is called *FluidProps*. The class has two public functions, *FConst* and *FProp*, and the public property *ErrorNumber*.

4.1 The Function Call FConst

The function call *FConst* has the following structure:

```
FConst(FluidID, ConstID, FldFileDir)
```

[4] Excel is a trademark of Microsoft®
[5] Visual Basic is a trademark of Microsoft®

Name	Description	Data Type
FConst	Function call (result)	DBL
FluidID	Fluid index	I16
ConstID	Fluid constant index	I16
FldFileDir	Fluid coefficient file directory	String

For the list of fluids and constants please refer to the GASPAK user's guide [1]. Unlike the function call *FProp*, fluid constants are not in SI units but consistent with the units set /3/ of the GASPAK user's guide [1]!

4.2 The Function Call FProp

The function call *FProp* has the following structure:

```
FProp(FluidID, In1ID, In1Val, In2ID, In2Val, OutID, PhaseID, FldFileDir)
```

Name	Description	Data Type
FProp	Function call (result)	DBL
FluidID	Fluid index	I16
In1ID	1st input property	I16
In1Val	Value of the 1st input property	DBL
In2ID	2nd input property	I16
In2Val	Value of the 2nd input property	DBL
OutID	Output index	I16
PhaseID	Phase index	I16
FldFileDir	Fluid coefficient file directory	String

For the list of fluids, input, output and phase indices and conventions on valid input property pairs please refer to the GASPAK user's guide [1]. All fluid properties are in SI units (units set /1/ in [1]).

4.3 The Property ErrorNumber

The property *ErrorNumber* returns the result of the error handling routine and is used to verify correct processing of *FProp*. *ErrorNumber* supplies an integer number (I16), which has the following meanings:

0:	No Error,
-1 to –9:	Invalid coefficient file,
-10 to –99:	External coding error,
-100 to –199:	Input out of range,
-200 to –299:	Iteration error.

5 INSTALLATION OF THE INTERFACE SERVER GaspakLV.DLL

For inter-application communication between LabVIEW and GASPAK it is assumed that both programs are properly installed on the computer. Their installation is not a subject of this note.

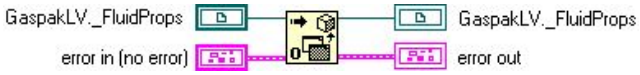
The 32-bit ActiveX server library *GaspakLV.dll* has to be copied in the system directory (e.g. "C:\Wnt\System32" on Windows NT). Afterwards it has to be registered by hand (right mouse click to open the sub-menu below).

6 LABVIEW APPLICATION EXAMPLES

The LabVIEW application examples consist of the following files stored in the LabVIEW library *Fluid Properties.llb*:

- GaspakLV Initialize.vi,
- GaspakLV FProp.vi,
- GaspakLV FConst.vi,
- GaspakLV Example1.vi,
- GaspakLV Example2.vi,
- GaspakLV Close.vi.

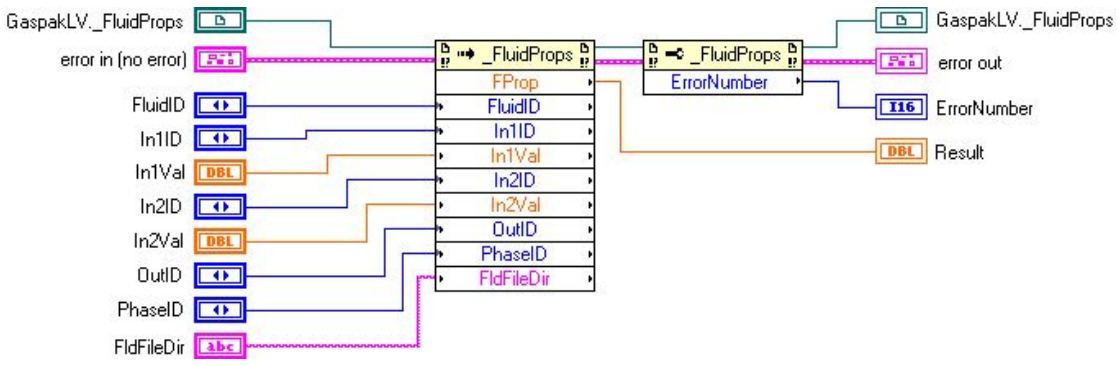
6.1 GaspakLV Initialize.vi



The *GaspakLV Initialize.vi* opens an *Automation refnum* that creates and returns a reference to an ActiveX object. It means that an instance of the class *FluidProps* is created in the memory and can be used for any kind and sequence of calculation (an instance is an item that conforms to a particular definition, class or template; the instance of a class is an object).

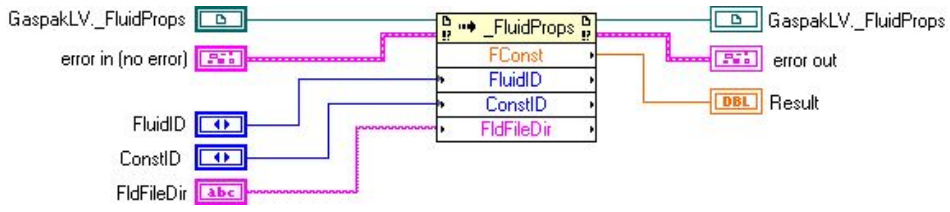
At the end of a LabVIEW application VI the instance has to be removed from the memory using the *GaspakLV Close.vi* (see 6.5).

6.2 GaspakLV FProp.vi



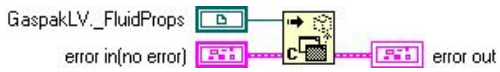
The subVI *GaspakLV FProp.vi* uses the LabVIEW ActiveX *Invoke Node* to call the function *FProp* of the Automation refnum created before. Afterwards a *Property Node* is used to check the result of the internal error handling routine. The subVI can only be used after an instance of the class *FluidProps* has been created in the memory.

6.3 GaspakLV FConst.vi



The subVI *GaspakLV FConst.vi* uses the LabVIEW ActiveX *Invoke Node* to call the function *FConst* of the Automation refnum. The property *ErrorNumber* has no meaning in this case. The subVI can only be used after an instance of the class *FluidProps* has been created in the memory.

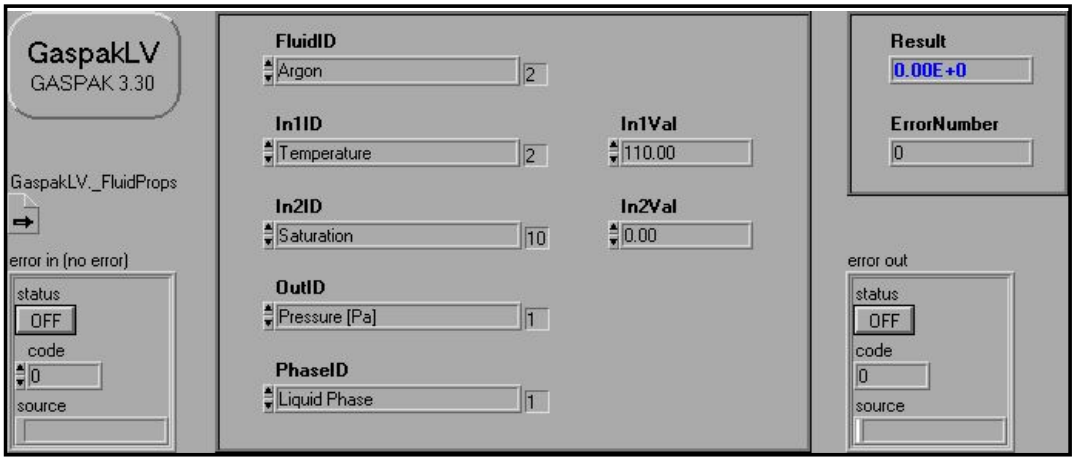
6.4 GaspakLV Close.vi

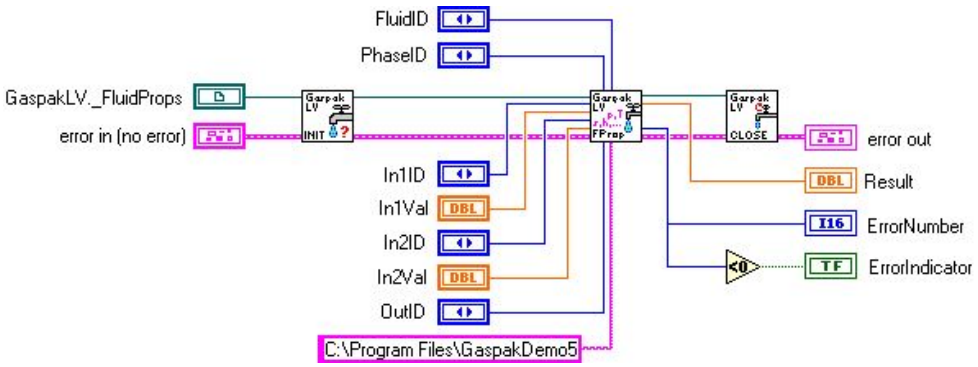


The *GaspakLV Close.vi* closes the *Automation refnum* that has been created with *GaspakLV Initialize.vi* and removes the instance of *FluidProps* from the memory.

6.5 GaspakLV Example1.vi

The *GaspakLV Example1.vi* demonstrates the use of the GaspakLV subVIs in order to access GASPAK fluid properties in LabVIEW. The *GaspakLV Example2.vi* is similar but calls the *GaspakLV FConst.vi* instead of *GaspakLV FProp.vi*.





REFERENCES

[1] User's guide to GASPAK, Version 3.30. Cryodata Inc., February 1999