IVI-5.0: Glossary

June 7, 2016 Edition
Revision 1.1

Important Information

Notice

IVI-5.0: Glossary is authored by the IVI Foundation member companies. For a vendor membership roster list, please visit the IVI Foundation web site at `www.ivifoundation.org`.

The IVI Foundation wants to receive your comments on this specification. You can contact the Foundation through the web site at `www.ivifoundation.org`.

Warranty

The IVI Foundation and its member companies make no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The IVI Foundation and its member companies shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Trademarks

Product and company names listed are trademarks or trade names of their respective companies. No investigation has been made of common-law trademark rights in any work.

Revision History

This section is an overview of the revision history of the IVI-5.0: Glossary.

Table 1. IVI-5.0: Glossary

Revision Number	Date of Revision	Revision Notes
Revision 0.1	August 8, 2000	Original draft
Revision 0.11	March 8, 2001	Draft modifications from Spring 2001 meeting
Revision 0.2	May 24, 2001	Draft to be reviewed at Paris meeting
Revision 0.26	June 25, 2001	Draft with modifications from Paris meeting
Revision 0.3	September 6, 2001	Draft to be reviewed at Boston meeting. Includes MSS terms submitted by Agilent.
Revision 0.4	November 13, 2001	Last draft not reviewed in Boston. This draft to be reviewed in 11/16/01 conference call. Includes installer terms.
Revision 0.5	December 1, 2001	Draft posted before Dec. Austin meeting..
Revision 0.6	April 4, 2002	Changes from 4/4/02 conference call
Revision 1.0	June 28, 2002	Accepted changes. Posting for vote.
Revision 1.01	August 22, 2003	Added a term and definition for *Configurable Initial Settings* as approved by IVI Foundation vote of IVI Glossary change document.
Revision 1.02	October 22, 2007	Removed reference to IVI Event Server
Revision 1.03	November 17, 2008	Added “bitness” a term used to identify the distinction between 32-bit and 64-bit operating systems,

Table 1. IVI-5.0: Glossary

		applications, and drivers. Editorial change to update the IVI Foundation contact information in the Important Information section to remove obsolete address information and refer only to the IVI Foundation web site.
Revision 1.04	March 6, 2013	Added minor change for Windows 7 and Windows 8
Revision 1.04	August 6, 2015	Editorial change to remove Windows 2000 and add Windows 10 as supported operating system
Revision 1.1	June 7, 2016	Minor change to remove support for Windows XP and Windows Vista

Definitions of Terms and Concepts

ADE	Application Development Environment. An environment which provides users various software development tools for the development of automated test systems. Examples of ADEs include National Instruments LabVIEW, Agilent VEE, Microsoft Visual C++, Microsoft Visual Basic, and National Instruments LabWindows/CVI.
Agilent VEE	A graphical programming ADE developed by Agilent Technologies.
API	Application Programming Interface.
asset	See *hardware asset*.
attribute	In the IVI specifications, this term generally refers to either a C attribute or a COM property. Attributes are used for hardware configuration and software control. Generally, each instrument setting is associated with a hardware configuration attribute. These attributes allow the user to set and retrieve values of the associated instrument settings. Software control attributes control how the instrument driver works rather than representing particular instrument settings.
base class capabilities	The minimum set of functions, attributes, and attribute values that an IVI driver must implement to claim compliance with an instrument class specification. For example, the base class capabilities of the oscilloscope class have functions and attributes that configure an edge-triggered acquisition, initiate an acquisition, and return the acquired waveform. An IVI class-compliant specific driver implements all the base capabilities for a particular class. For a complete description of the base capabilities for a particular class, refer to individual class specifications. Refer Section 2.6, *Capability Groups*, of *IVI 3.1: Driver Architecture Specification*, for more information on base class capabilities.
behavior model	A diagram indicating an instrument's possible states, the functions that cause it to transition between states, and the attributes that affect its behavior in each state.
bitness	A reference to the number of bits in the memory addresses in operating systems, applications, and drivers. Windows 7 (32-bit), Windows 8 (32-bit), and Windows 10 (32-bit) are examples of operating systems with a bitness of 32. Windows 7 (64-bit), Windows 8 (64-bit), and Windows 10 (64-bit) are examples of operating systems with a bitness of 64. Applications and drivers with a bitness of 32 or 64 can run on Windows 7 (64-bit), Windows 8 (64-bit), and Windows 10 (64-bit).
C class driver	See *IVI-C class driver*.
C specific driver	See *IVI-C specific driver*.

capability group	A set of functions, attributes, and attribute values defined in an IVI Foundation specification. There are four different types of instrument capabilities – Inherent Capabilities, Base Class Capabilities, Class Extension Capabilities, and Instrument Specific Capabilities.
channel string	An instrument specific string that refers to a particular channel of a device. An IVI specific driver that implements channels defines the channel strings that it recognizes. A channel string is an example of a repeated capability identifier.
channel	One of multiple physical inputs or outputs to an instrument. A set of channels is a type of repeated capability.
class extension capabilities	Groups of functions, attributes, and attribute values that an instrument class specification defines to represent instrument class features that are more specialized than the features that the Base Class Capabilities represent. A driver implements an extension capability only if the instrument being controlled by the driver supports the specialized features of the particular extension capability group. For example, the IviScope specification defines extension capability groups for various trigger modes, such as glitch triggering and TV triggering. IVI class-compliant specific instrument drivers are not required to implement extension groups, but should implement all class extensions that the instrument hardware supports. For a complete description of the class extension capabilities for a particular class, refer to individual class specifications. Refer Section 2.6, *Capability Groups*, of *IVI 3.1: Driver Architecture Specification*, for more information on class extension capabilities.
class-compliant interface	An interface of an IVI-COM driver that implements the inherent capabilities and the methods and properties that one of the IVI class specifications defines. Users can achieve a useful degree of instrument interchangeability in their test programs by restricting their instrument driver calls to the class-compliant interface of an IVI-COM driver.
coercion	See *value coercion*.
completion code	A value that an instrument driver returns after successful execution to provide additional information to the user. For example, if an instrument does not support a self test operation, calling the Self Test function will return a Self Test Not Supported code. Completion codes are often referred to as warning codes.
configurable initial settings	The set of attribute values that the user specifies in the driver session of the IVI configuration store that the IVI specific driver applies during driver initialization. Refer to Section 2.11.2, *Configurable Initial Settings*, of *IVI 3.1: Driver Architecture Specification*, for an overview of the use of Configurable Initial Settings. Refer to Section 6.6.5.1, *Defining Configurable Initial Settings in the IVI Configuration Store*, of *IVI 3.1: Driver Architecture Specification*, for more information on the structure of Configurable Initial Settings in the IVI configuration store. Refer to Sections 10.1, *Configurable Initial Settings*, and 14.1, *Configurable Initial Setting*, of *IVI 3.5: Configuration Server Specification*, for more information on how the IVI Configuration Server implements configurable initial settings.

configuration store	See *IVI configuration store*.
DLL	Dynamic Link Library
driver	A software module that controls a hardware device. See also *IVI driver.*
driver session	A session for an IVI driver.
driver session configuration	An item the user configures in the IVI configuration store to associate an IVI driver with initial settings and possibly one or more hardware assets.
error code	A value returned by the instrument driver when execution does not successfully complete.
error coordinator	The person responsible for subdividing the status code range for IVI shared components and standard cross-class capabilities. The error coordinator can be reached via email at error@ivifoundation.org.
extension capability	See *class extension capabilities*.
function	In the IVI specifications, this term generally refers to either a C function or a COM method.
function hierarchy	A hierarchy of the functions exported by an IVI-C driver. The function hierarchy provides users with an organized view of the instrument driver API. Typically, the function hierarchies of IVI class-compliant instrument specific drivers reflect the hierarchies defined in the class specifications.
GPIB	General Purpose Interface Bus. The common name for the communications interface system defined in ANSI/IEEE Standard 488.1-1987.
handle	A unique identification that allows access to an object or a session.
hardware asset	A physical device that performs a measurement or stimulus function. The term hardware asset includes instruments, but may include other types of hardware as well, such as switch matrices and probes.
hardware configuration attribute	An attribute that allows the user to set and retrieve the value of an instrument setting.
I/O resource descriptor	See *resource descriptor*.
instrument	A type of hardware asset.
inherent capabilities	The set of functions, attributes, and attribute values that all IVI drivers are required to implement. For a detailed description on these inherent capabilities, refer to *IVI-3.2: Inherent Capabilities Specification*.

instrument class	A way to typify instruments. Instruments with common capabilities can be considered to be of the same class. For example, “Function Generator” is an instrument class typified by an ability to generate voltage signals.
instrument driver	A software module used for controlling instruments.
instrument interchangeability	The ability to exchange hardware assets in a test system with few or no changes to the test program. Any changes required depend on the system components being used and the extent to which the application requires the same behavior.
instrument specific capabilities	Functions, attributes, and attribute values that represent features not defined by an instrument class specification. For example, some oscilloscopes have special features such as jitter and timing analysis, that are not defined in the IviScope class specification. The functions, attributes, and attribute values necessary to access the jitter and timing analysis capabilities of an oscilloscope are considered instrument specific capabilities. Instrument specific capabilities are beyond the scope of an instrument class.
interoperability	The ability of drivers and their required libraries to be compatible with each other and other system components. Interoperability allows users to install and use components implemented by a variety of suppliers. Components include drivers, interactive interfaces to the IVI Configuration Store, and vendor-specific implementations of VISA-C and VISA-COM.
IVI	Interchangeable Virtual Instruments
IVI class driver	An IVI driver that exposes a class-compliant API and serves as a pass-through layer to IVI class compliant specific drivers. For example, an IviScope class driver exports the functions, attributes, and attribute values defined in the IviScope class specification. When an application program calls an IviScope class driver, the IviScope class driver calls an IVI class-compliant specific driver that communicates with an oscilloscope. IVI class drivers are necessary for instrument interchangeability when using IVI-C class-compliant specific drivers.
IVI class-compliant specific driver	An IVI specific driver that complies with one of the IVI class specifications. For example, an IVI class-compliant specific driver for an oscilloscope exports the API defined by the IviScope class specification. Typically, an IVI class-compliant specific driver also provides instrument specific capabilities.
IVI class specification	A specification that defines the standard set of interfaces for an instrument class.
IVI Configuration Server	The shared component that accesses the IVI configuration store. Refer to *IVI-3.5: Configuration Server Specification*.

IVI configuration store	A data file that contains IVI configuration information. An IVI configuration store contains logical names and IVI session configuration information such as resource descriptors, software driver module paths, virtual channel name mappings, role control modules, and initial settings.
IVI configuration utility	An application program that allows users to modify the information in the IVI configuration store.
IVI custom class driver	A driver that is similar to an IVI class driver, except that the instrument class specification to which it complies is not one of the approved IVI class specifications. IVI custom class driver APIs are defined outside of the IVI Foundation and are not approved by the IVI Foundation.
IVI custom specific driver	An IVI specific driver that implements the inherent capabilities but does not implement any of the instrument class APIs defined by the IVI Foundation. IVI custom specific drivers are typically created for use with specialized instruments.
IVI driver	A software module that controls a hardware device and that complies with the IVI Foundation specifications. For IVI Foundation compliance requirements, refer to *IVI-3.1: Driver Architecture Specification*.
IVI driver installer	An IVI installer that installs IVI drivers.
IVI Floating Point Services	The IVI shared component that generates standard floating point values that represent positive infinity, negative infinity, and not-a-number.
IVI Foundation, Inc.	Interchangeable Virtual Instruments, Inc., a non-profit Delaware Corporation, composed of instrument end-users, instrument and software suppliers, and system integrators, chartered to define software standards that promote instrument interchangeability.
IVI inherent capabilities	Functions and attributes that all IVI drivers implement. For details on these requirements, refer to *IVI-3.2: Inherent Capabilities Specification*.
IVI installer	An installation program that implements the requirements specified by the IVI Foundation for the purpose of installing IVI drivers or IVI shared components.
IVI shared component	A software module defined, developed, and supported by the IVI Foundation. IVI software modules use IVI shared components as support libraries to ensure interoperability with other IVI software modules.
IVI shared component installer	The IVI installer created and distributed by the IVI Foundation that installs all the IVI shared components.
IVI software module	A conceptually cohesive set of software executables that is registered through the IVI Configuration Server. For example, all forms of IVI drivers are software modules, and both IVI-MSS servers and role control modules are software modules. For more information, refer to *IVI-3.5: Configuration ServerSpecification*.

IVI specific driver	An IVI driver that contains information for controlling a particular instrument or family of instruments and communicates directly with the instrument hardware. For example, IVI specific drivers control message-based instrument hardware by sending command strings and parsing responses.
IVI-C class driver	An IVI class driver that has a C API.
IVI-C class-compliant specific driver	An IVI Class-compliant specific driver that has a C-API.
IVI-C Shared Components	The IVI shared component that provides services for session management, thread locking, and error handling.
IVI-C specific driver	An IVI specific driver that has a C API.
IVI-COM class driver	An IVI class driver that has a COM API.
IVI-COM class-compliant specific driver	An IVI class-compliant specific driver that has a COM-API and has at least one class-compliant interface.
IVI-COM Session Factory	The IVI shared component that can dynamically load an IVI-COM software module without requiring the application program to identify the IVI software module directly. The IVI-COM Session Factory binds a logical name to the appropriate software module and hardware asset at run time. Client programs use the IVI-COM Session Factory to achieve instrument interchangeability when using an IVI-COM class compliant specific driver.
IVI-COM specific driver	An IVI specific driver that has a COM API.
IVI-MSS	IVI Measurement and Stimulus Subsystems architecture. IVI-MSS is an IVI architecture that builds on the IVI driver architecture to provide a higher degree of interchangeability. IVI-MSS takes into account such factors as aggregation of instruments and differences in measurement techniques among instruments. For more information, refer to *IVI-3.10: Measurment and Stimulus Subsystems Specification*.
IVI-MSS hardware asset interface	The programming interface to a hardware asset. This may be implemented in an external driver or the hardware asset.
IVI-MSS server	A software module that implements an IVI-MSS solution interface.
IVI-MSS solution	A solution that includes one or more measurement or stimulus servers and the associated role control modules.
IVI-MSS solution interface	An IVI-MSS role that is identified to the user as the primary interface for interacting with an IVI-MSS solution.
IVI-MSS role	A published API or interface that exposes well-defined measurement or stimulus functionality in the context of an IVI-MSS solution. Roles are the key interfaces for IVI-MSS interchangeability.
IVI-MSS role control module (RCM)	A software module that implements one or more IVI-MSS roles. Among other tasks, role control modules manage access to hardware assets in the IVI-MSS architecture.

LabVIEW	A graphical programming ADE developed by National Instruments.
LabWindows/CVI	A C-based ADE developed by National Instruments.
logical name	An item the user configures in the IVI configuration store to identify a particular session configuration. Application programs use logical names to avoid direct references to software modules and hardware assets.
physical identifier	See *physical repeated capability identifier*.
physical repeated capability identifier	An instrument specific string that refers to a particular instance of a repeated capability. An IVI specific driver that implements repeated capabilities defines the repeated capability identifiers that it recognizes.
published API	An API that is designed for implementation across several software modules, and as such is published independently of any one software module that implements it. Examples are IVI instrument class APIs and IVI-MSS role APIs.
range checking	The functionality of a driver that validates parameter and attribute values against published instrument limits. Range checking is a configurable behavior that can be enabled or disabled by the user.
repeated capability	An instrument capability for which multiple instances can be configured independently. Examples of repeated capabilities include channels on an oscilloscope, traces on a spectrum analyzer, and modulators on a RF signal generator.
repeated capability identifier	A virtual or physical repeated capability identifier.
repeated capability name	The name published by an IVI driver or class specification to refer to a particular repeated capability. For example, the IviScope class specification defines “Channel” as a repeated capability name.
repeated capability selector	The value that a user passes to a driver function to identify a repeated capability instance or a set of repeated capability instances.
resource descriptor	A string, such as a VISA resource descriptor, that specifies the I/O address of a hardware asset.
resource name	See *resource descriptor*.
session	A run-time instance that provides context for communicating and interacting with a particular physical instrument or collection of physical instruments. For IVI drivers, a session maintains configuration information and instrument state information from one IVI driver call to another and across threads. When using an IVI-COM driver, the user creates a driver session by either instantiating the main class of the IVI-COM specific driver or by using the IVI-COM Session Factory. When using an IVI-C driver, the user creates a driver session by calling the Initialize function of the driver. When creating a session in an application program, the user may reference a session configuration from the IVI configuration store.

session configuration	An item the user configures in the IVI configuration store to associate an IVI software module with initial settings and possibly one or more hardware assets.
shared component	See *IVI shared component*.
simulation	A required feature of IVI drivers that allows users to develop application code even when the instrument is not available. In simulation mode, the instrument driver does not perform I/O on the instrument.
software control attribute	An attribute that controls how the instrument driver works. Examples of software control attributes include Simulate and Range Check.
specific driver	See *IVI specific driver*.
state caching	An optional behavior of IVI specific drivers that tracks the state of instrument settings at run time. When the user enables state caching on an IVI driver that implements state caching, the instrument driver performs instrument I/O only when the current state of an instrument setting is different from what the user requests. State caching can improve performance of a test program by preventing the driver from sending redundant commands to the instrument.
status code	An error code or completion code.
value coercion	Occurs when an IVI specific driver alters the value that the user specifies for an attribute or parameter to a value that the specific driver or instrument can accept. If the specific driver coerces an attribute value, the specific driver returns the coerced value when the user reads the value of the attribute. For attributes that represent a continuous range of values, a driver may coerce the value that the user requests to a value that is more appropriate for the instrument. IVI class specifications may define rules for coercing values.
virtual identifier	See *virtual repeated capability identifier*.
virtual repeated capability identifier	An alias the user defines in a driver session in the IVI configuration store to represent a physical repeated capability identifier. The IVI configuration store contains the mappings between virtual repeated capability identifiers and physical repeated capability identifiers. Users striving for instrument interchangeability should use virtual repeated capability identifiers in their application programs.
VISA	Virtual Instrument Software Architecture. VISA provides an I/O API for communicating over a variety of bus interfaces, including GPIB, VXI, and serial interfaces.
Visual Basic	An ADE developed by Microsoft.
VPP	VXI*plug&play*.
VXI	VMEbus Extensions for Instrumentation, or IEEE 1155.VME eXtensions for Instrumentation.

VXI*plug&play* Systems Alliance	An organization whose members share a common commitment to end-user success with open, multivendor test and measurement systems. To view specifications defined by this alliance, refer to `www.vxipnp.org`.
warning code	See *Completion Code*.