TECHNICAL REPORT

ISO/IEC TR 13066-3

First edition
2012-09-15

Information technology — Interoperability with Assistive Technology (AT) —

Part 3: IAccessible2 accessibility application programming interface (API)

Technologies de l'information — Interopérabilité avec les technologies d'assistance —
Partie 3: Interface de programmation d'applications (API) d'accessibilité IAccessible2

Reference number
ISO/IEC TR 13066-3:2012(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

In exceptional circumstances, the joint technical committee may propose the publication of a Technical Report, when the joint technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example).

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 13066 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC TR 13066-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 35, *User interfaces*.

ISO/IEC 13066 consists of the following parts, under the general title *Information technology — Interoperability with assistive technology (AT)*:

— *Part 1: Requirements and recommendations for interoperability*

— *Part 2: Windows accessibility application programming interface (API)* [Technical Report]

— *Part 3: IAccessible2 accessibility application programming interface (API)*

Introduction

Assistive technology (AT) is specialized information technology (IT) hardware or software that is added to or incorporated within a system that increases accessibility for an individual. In other words, it is special purpose IT that interoperates with another IT product enabling a person with a disability to use the IT product.

Interoperability involves the ability to add or replace AT to existing components of IT systems. Interoperability between AT and IT is best facilitated via the use of standardized, public interfaces for all IT components.

This part of ISO/IEC 13066 describes the IAccessible2 API that can be used as a framework to support software to software IT-AT interoperability on the Windows platform.

Information technology — Interoperability with Assistive Technology (AT) —

Part 3: IAccessible2 accessibility application programming interface (API)

1 Scope

This part of ISO/IEC 13066 provides an overview to the structure and terminology of the IAccessible2 accessibility API.

It provides:

— a description of the overall architecture and terminology of the API;

— further introductory explanations regarding the content and use of the API beyond those found in Annex A of ISO/IEC 13066-1;

— an overview of the main properties, including:

  — of user interface elements,

  — of how to get and set focus,

  — of communication mechanisms in the API;

— a discussion of design considerations for the API (e.g. pointers to external sources of information on accessibility guidance related to using the API);

— information on extending the API (and where this is appropriate);

— an introduction to the programming interface of the API (including pointers to external sources of information).

It provides this information as an introduction to the IAccessible2 API to assist:

— IT system level developers who create custom controls and/or interface to them;

— AT developers involved in programming "hardware to software" and "software to software" interactions.

2 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

2.1
accessible object
part of user interface object that is accessible by Microsoft Active Accessibility

NOTE An accessible object is represented by a pair of the `IAccessible` interface and `ChildId` identifier.

2.2
application programming interface
API
collection of invocation methods and associated parameters used by one piece of software to request actions from another piece of software

[ISO/IEC 18012-1:2004, definition 3.1.1]

2.3
application software
software that is specific to the solution of an application problem

[ISO/IEC 2382-1, definition 10.04.01]

EXAMPLE A spreadsheet program is application software.

2.4
assistive technology
AT
hardware or software added to, or incorporated within, a system that increases accessibility for an individual

EXAMPLE Braille display, screen reader, screen magnification software and eye tracking device.

[ISO 9241-171, definition 3.5]

NOTE Within this part of ISO/IEC 13066, where assistive technology (and its abbreviation AT) is used, it is to be considered as both singular and plural, without distinction. If it is to be used in the singular only, it will be preceded by the article "an" (i.e. an assistive technology). If it is to be used in the plural only, it will be preceded by the adjective "multiple" (i.e. multiple AT).

2.5
client
component that uses the services of another component

NOTE In this part of ISO/IEC 13066, client refers more specifically to a component that uses the services of either or both Microsoft Active Accessibility and/or UI Automation to access, identify, or manipulate the UI elements of an application.

2.6
Component Object Model
COM
object-oriented programming model that defines how objects interact within a single process or between processes

NOTE In COM, clients have access to an object through interfaces implemented on the object.

2.7
compatibility
capability of a functional unit to meet the requirements of a specified interface without appreciable modification

[ISO/IEC 2382-1, definition 01.06.11]

2.8
function
defined objective or characteristic action of a system or component

[IEEE Std. 610.12-1990, unnumbered definition]

EXAMPLE A system has inventory control as its primary function.

2.9
information/communication technology
ICT
technology for gathering, storing, retrieving, processing, analysing and transmitting information

[ISO 9241-20, definition 3.4]

EXAMPLE A computer system is a type of ICT.2.13.

2.10
interface
shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical interconnections, signal exchanges, and other characteristics, as appropriate

[ISO/IEC 2382-1, definition 01.01.38]

2.11
interoperability
capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units

[ISO/IEC 2382-1, definition 01.01.47]

2.12
Microsoft Active Accessibility
COM-based technology that improves the way accessibility aids work with applications running on Microsoft Windows

NOTE It provides dynamic-link libraries (DLLs) that are incorporated into the operating system, as well as a COM interface and application programming elements that provide reliable methods for exposing information about user interface elements.

2.13
operating system
OS
software that controls the execution of programs and that may provide services such as resource allocation, scheduling, input-output control, and data management

NOTE Although operating systems are predominantly software, partial hardware implementations are possible.

[ISO/IEC 2382-1, definition 01.04.08]

2.14
servers
servers in Microsoft Active Accessibility are components (applications, dlls, etc.) that have UI and expose information about the UI and/or allow it to be manipulated

NOTE The terminology ‘server’ in Microsoft Active Accessibility is a synonym of ‘providers’ in UI Automation.

2.15
service
functionality made available to a user electronically

[ISO/IEC 24752-1 URC, definition 4.27]

EXAMPLE Airline reservation service, currency translation services, weather forecasting, restaurant recommendations, etc.

2.16
simple element
<Microsoft Active Accessibility> UI element that shares an `IAccessible` object with other peer elements

NOTE A simple element relies on the shared `IAccessible` object (typically its parent in the object hierarchy) to expose its properties.

2.17
software
all or part of the programs, procedures, rules, and associated documentation of an information processing system

NOTE Software is an intellectual creation that is independent of the medium on which it is recorded.

[ISO/IEC 2382-1, definition 01.01.08]

2.18
system software
platform software
application-independent software that supports the running of application software

[ISO/IEC 2382-1, definition 01.04.02]

EXAMPLE An operating system, a Web browser, or a programming environment (e.g. Java) can be used as a platform for application software.

2.19
user interface
UI
mechanisms by which a person interacts with a computer system

NOTE The user interface provides input mechanisms, allowing users to manipulate a system. It also provides output mechanisms, allowing the system to produce the effects of the users' manipulation.

2.20
user interface element
user interface object
entity of the user interface that is presented to the user by the software

[ISO 9241-171 definition 3.38]

NOTE 1 User interface elements may or may not be interactive.

NOTE 2 Both entities relevant to the task and entities of the user interface are regarded as user interface elements. Different user interface element types are text, graphics and controls. A user interface element may be a representation or an interaction mechanism for a task object (such as a letter, a sales order, electronic parts, or a wiring diagram) or a system object (such as a printer, hard disk, or network connection). It may be possible for the user to directly manipulate some of these user interface elements.

EXAMPLE 1 User interface elements in a graphical user interface include such things as basic objects (such as window title bars, menu items, push buttons, image maps, and editable text fields) or containers (such as windows, grouping boxes, menu bars, menus, groups of mutually-exclusive option buttons, and compound images that are made up of several smaller images).

EXAMPLE 2 User interface elements in an audio user interface include such things as menus, menu items, messages, and action prompts.

EXAMPLE 3 User interface elements in tactile interfaces include such things as tactile dots, tactile bars, surfaces, knobs, and grips.

2.21
WinEvents
mechanism that allows servers and the Windows operating system to notify clients when an accessible object changes

3 General Description

3.1 General Description

IAccessible2 was developed by IBM in 2006 to complement Microsoft's earlier work on the Microsoft Active Accessibility (MSAA) API. MSAA provides accessibility services on the Windows® platform. The IAccessible2 set of interfaces allow application developers to leverage their investment in MSAA while also providing assistive technologies (AT), such as screen readers, reliable access to user interface features not previously supported by MSAA. ATs implementing only MSAA must use reverse engineering and heuristic techniques, such as screen scraping, to provide access to features such as rich document editing functions and tables in documents. While such AT implementations can work, they are customized for each application, expensive to develop, frequently unreliable, and often must be reworked when new versions of the applications are released. The additional function IAccessible2 provides over MSAA includes support for rich text, tables, relationships between objects, self-describing actions, application-specific information, and extensible object properties to support Web 2.0 applications.

A major requirement of many information technology (IT) vendors is the ability to efficiently create applications for multiple platforms. IAccessible2 has therefore been harmonized with the UNIX accessibility APIs, Accessibility Toolkit (ATK) and Assistive Technology Service Provider Interface (AT-SPI), to allow for efficient multi-platform development. Adding IAccessible2 events and interfaces to MSAA yields application semantics that are very similar on both Windows and UNIX.

These design principles guided the specification of the IAccessible2 interfaces which are based on the difference between MSAA and the UNIX Accessibility Toolkit, described in ISO/IEC 13066-4. Application vendors can incrementally add IAccessible2 interfaces as needed to their MSAA implementation providing improved accessibility to their application. And Windows assistive technology vendors can keep much of their MSAA design, only adding a check for these new interfaces before using their legacy heuristic code.

Note that IAccessible2 is the name used to refer to the “set” of interfaces and also a specific API within the set of interfaces. In this part of ISO/IEC 13066 “IAccessible2” and “IAccessible2 interfaces” refers to the set of interfaces and “`IAccessible2` interface” refers to the specific API.

3.2 Architecture

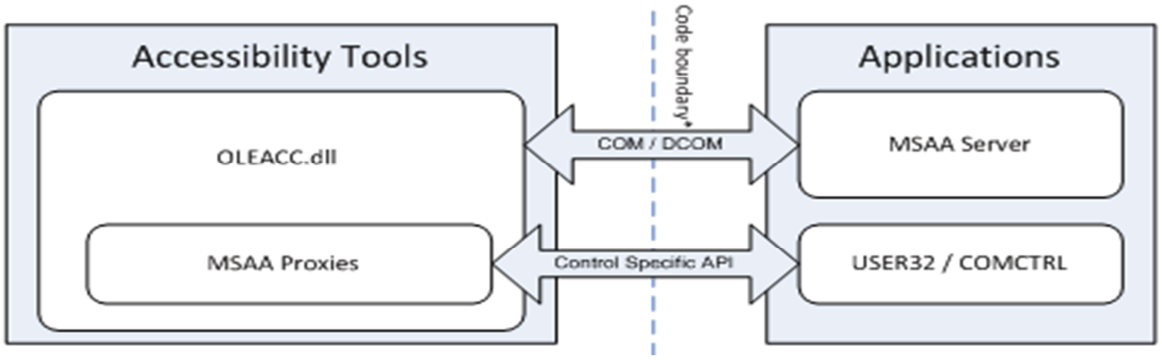
Since IAccessible2 is an extension of MSAA, described in ISO/IEC 13066-2, its architecture matches that of MSAA.

MSAA is based on the Windows Component Object Model (COM), which defines a common way for applications and operating systems to communicate. Applications are considered MSAA servers because they serve or provide information about their user interfaces (UI). Assistive technology products, such as screen readers, are called MSAA clients because they consume and interact with application UI information.

The accessible object is the central interface of Microsoft Active Accessibility and is represented by an `IAccessible` COM interface and an integer `ChildId`. MSAA clients use the interface to access, identify, and manipulate a server’s UI and register to be notified of changes to the server UI through event notifications known as WinEvents.

An accessible object can be of two types. A *parent* or *container* object is represented by a ChildID value of `CHILDID_SELF` (or 0 ‘zero’). The children of a *parent* or *container* object are simple elements that share the same `IAccessible` interface with their parent and are represented by a non-zero ChildID value (usually a positive sequential number beginning with 1). Simple elements cannot have children of their own. Multiple levels of accessible objects are used to represent UI elements that are composed of more than two levels of hierarchy.

As stated in ISO/IEC 13066-2, the system component of the Microsoft Active Accessibility framework, `Oleacc.dll`, aids in the communication between accessibility tools (clients) and applications (servers). The `Oleacc.dll` provides an implementation of the Microsoft Active Accessibility API, the accessibility system framework, and the proxy objects for the Windows Operating System standard controls. The implementation of the Microsoft Accessibility API is an implementation of the IAccessible interface for Windows Operating System controls. Additional implementations of IAccessible, external to Oleacc.dll, are provided by applications running on the Windows Operating System. The *code boundary* indicates the programmatic boundaries between applications that provide UI accessibility information and accessibility tools that interact with the UI on behalf of users. The boundary can also be a *process boundary* when Microsoft Active Accessibility clients have their own process.



* Also process boundary in case of out of process MSAA Clients

Figure 1

Windows application developers and assistive technology vendors are already familiar with MSAA and their COM implementations, and for the information provided, it is reliable and works well. So an important feature of IAccessible2 is that the implementation and semantics of MSAA supporting applications and assistive technologies are preserved. In fact, IAccessible2 builds on the existing COM foundation to expose new interfaces, roles, states, and events.

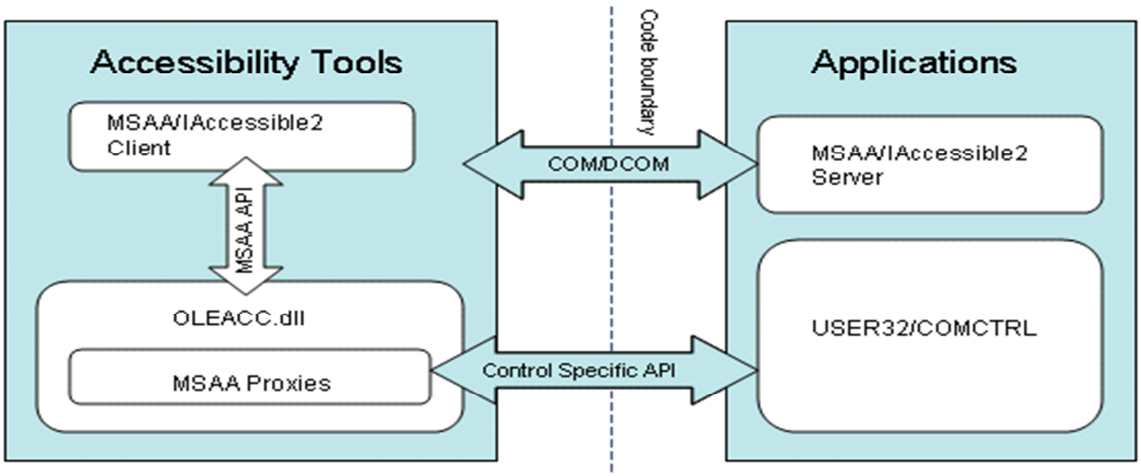


Figure 2

Alternatively, an application might choose to have two servers: one for MSAA and one for IAccessible2.