INTERNATIONAL STANDARD

ISO 17356-1

First edition
2005-01-15

Road vehicles — Open interface for embedded automotive applications —

Part 1: General structure and terms, definitions and abbreviated terms

Véhicules routiers — Interface ouverte pour applications automobiles embarquées —
Partie 1: Structure générale et termes, définitions et termes abrégés

Reference number
ISO 17356-1:2005(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) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

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

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

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

ISO 17356-1 was prepared by Technical Committee ISO/TC 22, *Road vehicles*, Subcommittee SC 3, *Electrical and electronic equipment*.

ISO 17356 consists of the following parts, under the general title *Road vehicles — Open interface for embedded automotive applications*:

— *Part 1: General structure and terms, definitions and abbreviated terms*

— *Part 2: OSEK/VDX specifications for binding OS, COM and NM*

— *Part 3: OSEK/VDX operating system (OS)*

— *Part 4: OSEK/VDX communication (COM)*

— *Part 5: OSEK/VDX network management (NM)*

— *Part 6: OSEK/VDX implementation language (OIL)*

Road vehicles — Open interface for embedded automotive applications —

Part 1: General structure and terms, definitions and abbreviated terms

1 Scope

This part of ISO 17356 outlines the general structure of, and defines terms and abbreviations used in relation to, the specification of the software open interface for embedded automotive applications given by the other parts of ISO 17356.

2 Terms, definitions and abbreviated terms

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

2.1
acceptance filtering
mechanism which decides whether each received protocol frame is to be taken into account by the local node or ignored

2.2
activate
action of changing a task from the suspended to the ready state

NOTE The transition is achieved by a system service.

2.3
actual configuration
set of all operable nodes to which communication access is possible

NOTE See **operability of a node** (2.81).

2.4
address-related communication
type of communication between nodes using node addresses where each address-related communication message contains certain data and — either explicitly or implicitly — the node address of the transmitter and the receiver

NOTE 1 See **node addressing** (2.76).

NOTE 2 The communication of the network management is based completely on address-related communication.

2.5
alarm
association between a counter and a task, event or callback such that the task, event or callback occurs when a particular counter value is reached

NOTE 1 The expiry value can be defined relative to the current counter value or can be an absolute value.

NOTE 2 Alarms can be defined to be either single-shot or cyclic.

NOTE 3 An alarm is statically assigned at system generation time to one counter and a task, event or alarm callback routine.

2.6
alarm callback
short function, provided by the application, called when an alarm expires but before any task is activated or event set

2.7
alarm management
manipulation of an alarm's running/cancelled state, and the counter value at which it next expires

NOTE Alarm management is based on the counter concept. Alarms are a way of linking alarm callbacks, task activation or event setting to counter values.

2.8
alive message
message used to announce an initialized and operable node for integration in the actual configuration

NOTE 1 See **operability of a node** (2.81).

NOTE 2 A dedicated NM message is used for this purpose.

2.9
application program interface
API
description of the application's interface to the operating system, communications and network management functions

2.10
application error
error where the operating system cannot execute the requested service correctly, but assumes the correctness of its internal data, and calls centralized error treatment

2.11
arbitration
mechanism that guarantees that a simultaneous access made by multiple stations results in contention where one frame will survive uncorrupted

2.12
basic conformance class
BCC
conformance class of the operating system in which only basic tasks are permitted

NOTE Two basic conformance classes are distinguished: BCC1 and BCC2.

2.13
basic task
BT
task that can only release the processor when it terminates, when the operating system executes a higher-priority task or when an interrupt occurs

NOTE 1 A basic task can only enter the task states suspended, ready and running.

NOTE 2 It is not possible for a basic task to wait for an event.

2.14
broadcast
case of multicast whereby a single message is addressed to all nodes simultaneously

2.15
busOff
condition of switching off from the bus so that protocol frames can neither be sent nor received

2.16
callout
general mechanism, based upon function calls, allowing the behaviour of the interaction layer to be customized and enhanced

NOTE 1 Callouts are configured statically, are invoked in response to the passage of a message or I-PDU and cannot be changed at run-time.

NOTE 2 The prototype for a callout allows it to return a value that determines further treatment by the IL of the message or I-PDU.

2.17
controller area network
CAN
protocol originally defined for use as a communication network for control application in vehicles

2.18
certification
process of determining whether an implementation is consistent with a given reference model

NOTE The scope of the reference model has to be settled according to the objectives of the project and all constraints necessary to fulfil those objectives incorporated in the reference model.

2.19
COM-callback
short function, provided by the application, which can be called by the interaction layer as a notification mechanism (class 1)

NOTE No parameters are passed to a COM-callback routine and it does not have a return value. A COM-callback routine runs either on interrupt level or on task level.

2.20
communication layer
set of all entities and elements which constitute a communication layer based on the ISO/OSI reference model

NOTE For the basic model, see ISO 7498-1.

2.21
configurability
ability to set the parameters of a system in terms of static values

EXAMPLE Number of tasks, RAM size for stack, size of message buffer.

2.22
confirmation
service primitive via which a service provider informs a service user about the result of a preceding service request

NOTE The confirmation service primitive is defined by the ISO/OSI reference model (ISO 7498).

2.23
conformance class
CC
subset of services chosen by the application

NOTE 1 In each module (operating system, communication, network management), a pool of services is provided, with each of these being divided into a number of defined subsets. Applications can choose to use a particular subset of the services in order to reduce demands on the CPU and memory.

NOTE 2 The subsets are upwardly compatible and are described as conformance classes.

2.24
connection
logical communication channel between a transmitter and a receiver

NOTE A message is sent by exactly one transmitter and is received by exactly one receiver.

2.25
constructional element
definition and declaration services for system objects

2.26
counter
system object that registers recurring events such as time or angle

NOTE A counter is represented by a count and some counter-specific constants.

2.27
critical section
sequence of instructions where mutual exclusion is ensured

NOTE Such a section is called “critical” because shared data is modified within it.

2.28
data consistency
content of a given message correlating unambiguously to the operation performed on the message by the application such that no unforeseen sequence of operations may alter the content and thereby render it inconsistent with respect to its allowed and expected value

2.29
data link layer
communication layer, consisting of the communication hardware and the communication driver software, that provides services for the transfer of I-PDUs

2.30
deadlock
tasks that block one another so that further processing of the tasks concerned is no longer possible

EXAMPLE Each of two tasks waits for the reception of a message to be sent by the other task before sending its own message.

2.31
direct node monitoring
active monitoring of a node by another node in the network

NOTE For this purpose, the monitored node sends an NM message according to a dedicated and uniform algorithm. For the network-wide synchronization of NM messages, a logical ring is used.

2.32
deadline monitoring
informing of the application via the notification mechanism that a message has not been received from another node within a specified interval, or if a request to send an I-PDU has not been completed by the DLL within a specified interval

2.33
error handling
error service provided to handle errors detected by the operating system

NOTE The basic framework is predefined and has to be completed by the user, thus giving the user a choice of efficient centralized or decentralized error handling.

2.34
error hook
routine (ErrorHook) called when a system service returns a StatusType value not equal to E_OK or when an error is detected during task activation or event setting

2.35
event
method of task synchronization peculiar to extended task whereby a task may suspend its execution without terminating

NOTE The task suspends its execution by waiting for an event and continues when an appropriate event is set. Basic tasks cannot use events.

2.36
event mechanism
means of task synchronization using events

2.37
extended conformance class
ECC
conformance class of the operating system in which basic and extended tasks are permitted

NOTE Two extended conformance classes are distinguished: ECC1 and ECC2.

2.38
extended task
ET
task that is allowed to use additional operating system services, which may result in a waiting state

NOTE An extended task can enter the task state *suspended*, *ready*, *running* or *waiting*.

2.39
fatal error
error where the operating system can no longer assume correctness of its internal data

NOTE In this case, the operating system calls the centralized system shutdown.

2.40
frame
data unit determined according to the data link protocol specifying the arrangement and meaning of bits or bit fields in the data transferred across the transfer medium

2.41
full pre-emptive scheduling
scheduling where a task which is presently running may be pre-empted at any instruction by the occurrence of a trigger condition pre-set by the operating system that puts the running task into the ready state as soon as a higher-priority task becomes ready

NOTE The pre-emptee's context is saved so that it can be continued at the location where it was pre-empted.

2.42
group addressing
addressing of several receiver nodes, implemented using multicast connections, in a single address-related NM message

NOTE See **address-related communication** (2.4).

2.43
hook routine
user-defined function which will be called by the operating system only under certain circumstances and in a defined context

NOTE Hook routines may be used for tracing or application-dependent debugging purposes, user-defined extensions to context switches and in error handling. Most operating system services are not allowed in hook routines.

2.44
indication
service primitive where a service provider informs a service user about the occurrence of either an internal event or a service request issued by another service user

NOTE The indication service primitive is defined by the ISO/OSI reference model (ISO 7498).

2.45
indirect node monitoring
monitoring of a node by “listening” to dedicated application communication messages

NOTE Indirect node monitoring is based on monitored state messages which are sent periodically.

2.46
interaction layer
communication layer that implements the interface between the application and other potential communication layers such as the DLL and network layers

NOTE 1 The communication services of the interaction layer are independent of both microcontroller and network protocol.

NOTE 2 The interaction layer enables internal and network-wide communication by means of UnQueued messages and Queued messages.

2.47
internal communication
exchange of messages between tasks belonging to the same node

2.48
internal resource
resource which is not visible to the user and therefore cannot be addressed by the system functions GetResource and ReleaseResource

NOTE Internal resources are managed strictly internally within a clearly defined set of system functions.

2.49
interrupt
enforced suspension of the execution of the current program section

2.50
interrupt latency
time between the moment an interrupt occurs and the execution of the first instruction of the interrupt service routine

2.51
interrupt level
priority level provided by the CPU for ISRs

NOTE To keep the interrupt latency as short as possible, it is preferable that only absolutely indispensable actions be performed at interrupt level.

2.52
interrupt service routine
function that provides the main processing of an interrupt

2.53
intertask communication
mode of information interchange between tasks

NOTE In the course of intertask communication, messages are logically copied from the local area of a task (transmitter) to the local area of another task (receiver).

2.53
I-PDU
collection of messages for simultaneous transfer between nodes in a network

NOTE At the sending node, the interaction layer (IL) is responsible for packing messages into an I-PDU and then sending it to the underlying layer (transport layer or DLL) for transmission; at the receiving node, the DLL (or transport layer) rebuilds the I-PDU and passes it to the IL, which then unpacks the messages.

2.54
ISR category
trade-off between ISR response time and API complexity

NOTE Interrupt processing is subdivided into two categories of ISRs: Category 1 comprises all ISRs which do not use operating system services and are, therefore, typically faster for entry and exit than category 2 ISRs. Category 1 ISRs are only allowed to use a very restricted set of operating system services, whereas category 2 ISRs are allowed to use a less restricted set but are typically inherently slower.

2.55
latency time
time delay between the request of an activity and its execution

2.56
limp home
operating mode in NM which is entered in case of an error which cannot be corrected

2.57
limp home configuration
set of all nodes which cannot participate in direct node monitoring due to failure

2.58
limp home message
dedicated NM message used for notifying a node that the system has entered the limp home state

2.59
logical ring
structure imposed by software rather than physical arrangement that orders the nodes within a network such that every node has exactly one successor and one predecessor and a pathway exists from any node to any other node

NOTE 1 The nodes are arranged in terms of a ring. The logical ring is used for the network-wide synchronization of NM messages. In a logical ring, the communication sequence is defined independently of the network structure; therefore, each node is assigned a logical successor — the first logical node is the successor of the last logical node in the ring.

NOTE 2 A ring message is always sent from a node to its logical successor.

2.60
message
fundamental unit of data transfer between an application and a COM's IL and, therefore, also of intra- and inter-ECU communications

NOTE A message can be 0 or more bits long and could contain some application-specific data ranging from a bit to a large array or structure. Therefore, messages can support event and signal-based communication as well as more complex interfaces.