The Hermes Standard
for "M-to-M" in SMT Assembly

The Hermes Standard Change Proposal

Clarification version number and supported features

Voting meeting:
28th of January 2019 (APEX / San Diego)

Requesting company:
Hermes Moderator

Version change:

Revision

Affected versions:

1.0.2 and 1.1

Service description tag:

-

Description:

Added some clarification regarding the meaning and usage of version number and supported features in ServiceDescription message.

Use cases:

-

Functionality / communication sequences:

-

New / changed XML messages:

-

Proposed changes to standard:

2 Technical concept

…

2.3 Connecting, handshake and detection of connection loss

After booting, the downstream machine starts cyclic connection attempts to the configured upstream machines. When a connection is established, the downstream machine starts sending a ServiceDescription message whereupon the upstream machine answers with its own ServiceDescription. This ServiceDescription message contains the lane ID and interface ID (optional) of the sending machine related to this TCP connection. It also contains the implemented version and a list of all optional features and additional features of a higher version which are implemented by the ~~client~~ machine. The features of the Hermes specification version 1.0 have to be supported by any implementation ~~and shall not be included explicitly.~~

If a downstream machine is already connected to the lane and the transportation interface, this connection will be retained. A Notification message shall be sent to the new connection before it is closed.

After exchanging the handshake messages, both machines may begin to send BoardAvailable/ MachineReady messages (see section 2.4).

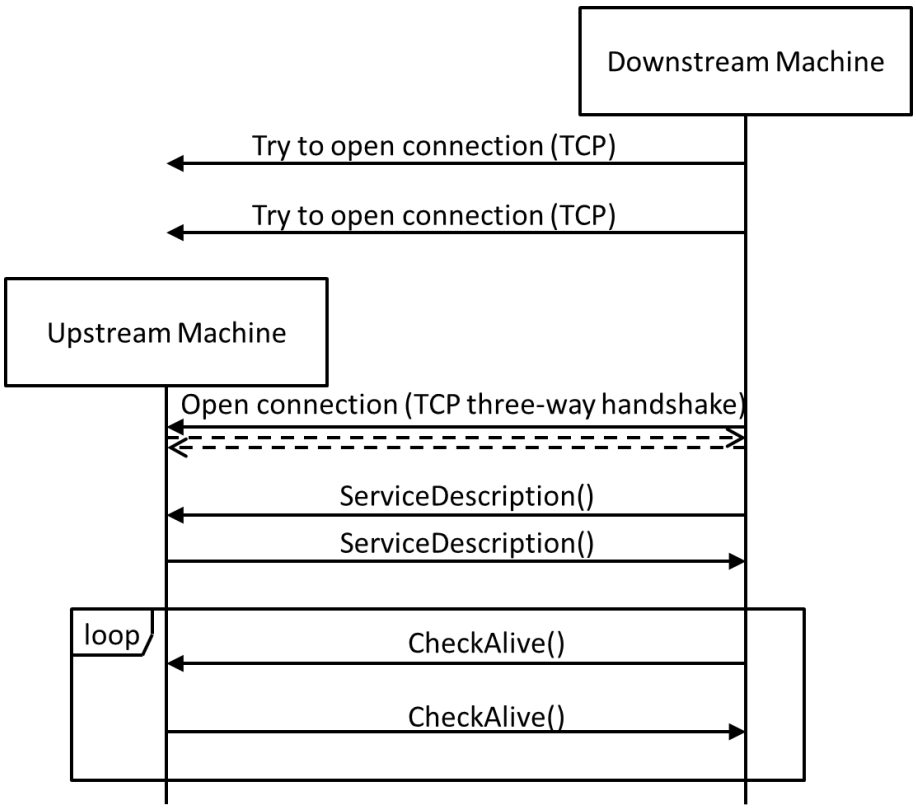

Fig. 3 Connection, handshake and connection loss detection

The connections are kept open all the time. As TCP by itself does not detect connection losses ("Half-open connections" caused by e.g. process-/computer crash, unplugged network cables …) both sides of a connection have to send cyclic CheckAlive messages. Those messages do not have to be answered by the remote side – the TCP stack will detect a connection loss when trying to send the packet. If the server detects a connection loss, it ends the connection and waits for a new connection by the client. If the client detects a connection loss, it ends the connection and re-starts with the cyclic connection attempts.
As not all TCP stacks recognize correctly the loss of connection when sending messages it is possible to extend the implementation of this functionality to an exchange of CheckAlive messages. Machines which have implemented this function do have the tag FeatureCheckAliveResponse in the ServiceDescription.
The exchange of CheckAlive messages then works like shown in Fig. 4.

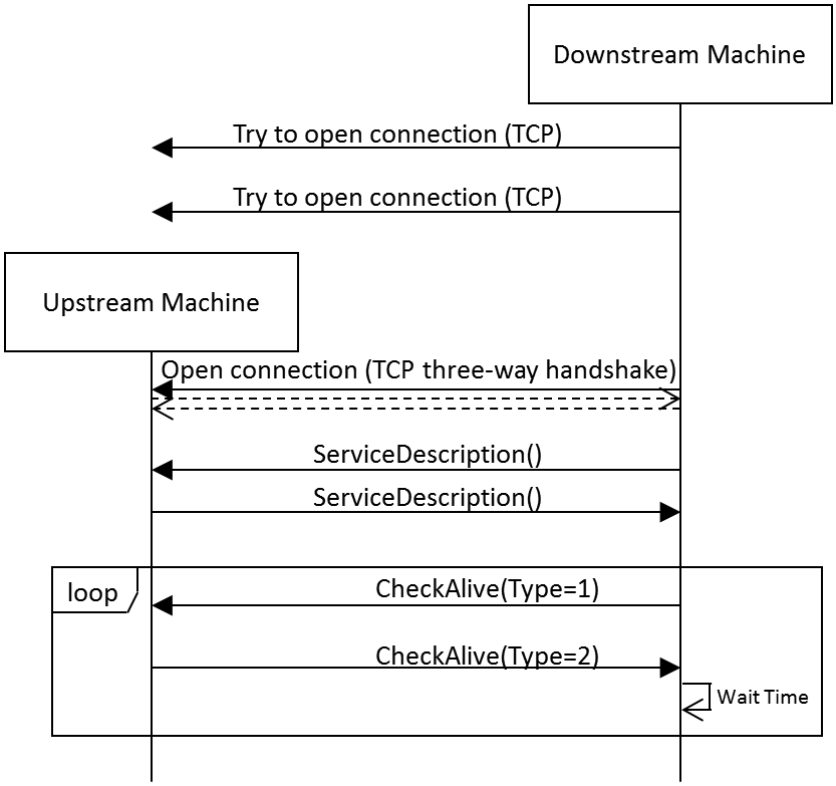

Fig. 4 Example for connection loss detection with FeatureCheckAliveResponse

One of the machines (in the figure the downstream machine but it could be also the upstream machine) sends a (ping) CheckAlive message, that is a CheckAlive message with the attribute Type set to 1. The peer machine then responds immediately with a (pong) CheckAlive message, that is a CheckAlive message with the attribute Type set to 2 and the Id matching the Id of the (ping) CheckAlive message.
A missing response (it is recommended to wait for 3 seconds.) indicates a connection loss.

3 Message definition

…

3.3 CheckAlive

The CheckAlive message is used to detect connection losses. It therefore does not have to transport data and can be ignored by the receiver. Accordingly there is no response.
However, if a machine supports the FeatureCheckAliveResponse, it must answer CheckAlive messages with Type set to 1 with a CheckAlive message with Type set to 2 and the same Id as the received CheckAlive message.

Note: The function of CheckAliveResponse is optional. If FeatureCheckAliveResponse is specified in the ServiceDescription, it must be fully supported. Otherwise it can be ignored.

CheckAlive	Type	Range	Optional	Description
Type	int	1..2	yes	Ping / Pong message type
Id	string	any string (minimum supported length: 80 bytes)	yes	Idenfier of the message

Type may be one of the following values:

1 Ping: CheckAlive request

2 Pong:CheckAlive response

The machine sending CheckAlive message with Type set to 1 chooses a unique for Id (e.g. GUID or time stamp). The machine responding with CheckAlive message with Type set to 2 has to answer using the same Id.

3.4 ServiceDescription

The ServiceDescription message is sent by both machines after a connection is established. The downstream machine sends its ServiceDescription first whereupon the upstream machine answers by sending its own ServiceDescription.

ServiceDescription	Type	Range	Optional	Description
MachineId	string	any string (minimum supported length: 80 bytes)	no	ID/name of the sending machine for identifying it in a Hermes enabled production line.
LaneId	int	1 .. n	no	The sending machine's lane to which this connection is relating to. Lanes are enumerated looking downstream from right to left beginning with 1
InterfaceId	string	any string (minimum supported length: 80 bytes)	yes	The ID of the sending machine's transportation interface to which this connection is relating to.
Version	string	xxx.yyy (7 bytes)	no	The implemented interface version of the machine
SupportedFeatures	Feature []		no	List of supported features (empty for version 1.0)

Feature	Type	Range	Optional	Description
FeatureCheckAliveResponse	FeatureCheckAliveResponse		yes	Indication of CheckAliveResponse function implementation
FeatureBoardForecast	FeatureBoardForecast		yes	In the upstream role: Machine emits BoardForecast messages
FeatureQueryBoardInfo	FeatureQueryBoardInfo		yes	Indication of QueryBoardInfo function implementation
FeatureSendBoardInfo	FeatureSendBoardInfo		yes	Indication of SendBoardInfo function implementation

xxx.yyy must match the regular expression
`[1-9][0-9]{0,2}\.[0-9]{1,3}`

The features specified in version 1.0 of this protocol have to be provided by any implementation and thus are not listed in the SupportedFeatures list of the ServiceDescription explicitly. The same applies for all mandatory features of the version specified in the Version attribute. All optional features or additional features of an higher version supported by a machine need to be listed in the SupportedFeatures list to indicate there availability.