

IPC-HERMES-9852

The Global Standard for Machine-to-Machine Communication in SMT Assembly

Version 1.5.1

Contributing companies:

4IR.UK
6TL Engineering
Achat Engineering GmbH
allSMT
ASM Assembly Systems GmbH
Asscon
ASYS Automatisierungssysteme GmbH
BESI
Bright Machines
BTU
CKD
CTI Systems
CTS
CYBEROPTICS
Digitaltest
ECD
Essemtec
EUNIL
Eunil Co., Ltd.
Exelsius
Famecs
FlexLink
GKG

GÖPEL electronic GmbH
Hanwha
Heller Industries
Innomelt
IPTE
IBL-Löttechnik GmbH
ITW EAE
JAPAN UNIX Co. Ltd.
JOT Automation Kft.
Keysight Technologies
KIC
KOH YOUNG Technology Inc.
kolb Cleaning Technology GmbH
Kulicke & Soffa
Kurtz Ersä
Magic Ray Technology
MIRTEC
MYCRONIC
Nordson ASYMTEK & MATRIX
Nutek Europe B.V.
OMRON Corporation
OSAI
PARMI

Pemtron
Rehm Thermal Systems GmbH
Rejoint
RG Elektrotechnologie
SEICA SpA & SEICA Automation
SAKI Corp
Scheid IT
SEHO Systems
SICK AG
SMT-Wertheim
SolderStar
Sonic Technology
SPEA S.p.A.
SYNEO
Test Research, Inc. (TRI)
Takaya
Universal Instruments
VISCOM AG
ViTrox
YJ Link Co., Ltd.
YXLON



Contents:

1	SCOPE	5
2	TECHNICAL CONCEPT	6
2.1	Prerequisites	6
2.2	Board IDs	6
2.3	Machine-to-Machine Communication (Horizontal Channel)	6
2.3.1	Topology	6
2.3.2	Connecting, Handshake and Detection of Connection Loss	7
2.3.3	Normal Operation.....	9
2.3.4	Transport Error Handling	10
	Scenario U1a	11
	Scenario U1b	12
	Scenario U2	13
	Scenario U3	14
	Scenario D1	15
	Scenario D2	16
	Scenario D3	17
2.3.5	Handling of BoardForecast	18
	Scenario 1	18
	Scenario 1 (error handling)	20
	Scenario 2.....	21
2.3.6	Protocol States and Protocol Error Handling	22
2.3.7	Handling of Attribute 'Route'	23
2.3.8	Handling of Attribute 'Action'	23
2.4	Remote Configuration	23
2.4.1	Topology	23
2.4.2	Remote Configuration	24
2.5	Communication With Supervisory System (Vertical Channel)	24
2.5.1	Topology	24
2.5.2	Connecting, Handshake and Detection of Connection Loss	24
2.5.3	Protocol States and Protocol Error Handling	27
3	MESSAGE DEFINITION	28
3.1	Message Format	28
3.2	Root Element.....	28
3.3	CheckAlive	29



3.4	ServiceDescription	29
3.5	Notification.....	31
3.6	BoardAvailable	31
3.7	RevokeBoardAvailable	35
3.8	MachineReady	35
3.9	RevokeMachineReady	36
3.10	StartTransport.....	36
3.11	StopTransport	37
3.12	TransportFinished.....	37
3.16	BoardForecast	37
3.17	QueryBoardInfo	40
3.18	SendBoardInfo	41
3.19	SetConfiguration	43
3.20	GetConfiguration	45
3.21	CurrentConfiguration	45
3.22	SupervisoryServiceDescription.....	46
3.23	BoardArrived.....	46
3.24	BoardDeparted	50
3.25	QueryWorkOrderInfo	53
3.26	SendWorkOrderInfo	54
3.27	ReplyWorkOrderInfo	57
3.28	Command	58

4 Appendix..... 60

4.1	Special Scenarios.....	60
4.1.1	Board Tracking When Board Is Torn Out From the Line	60
4.1.2	Board Tracking When Board Is Temporarily Removed From the Line	61
4.1.3	Board Tracking When Board Was Transferred Without Data.....	62
4.1.4	Oven Error Loop	64
4.1.5	Request Pause / Confirm Pause and Resume Operation	65
4.1.6	Board Removal at Downstream Conveyor	67
4.1.7	Reversal Transportation to a Flipping Unit Located Downstream a Process Machine ...	68
4.1.8	Reversal Transportation to a Flipping Unit Located Upstream a Process Machine	69
4.1.9	Board Routing Within a Production Line by Predefined Routes	70
4.1.10	Board Routing Within a Production Line Towards Target Locations	70



4.2	Glossary / Abbreviations	71
4.3	References	72
4.4	History	72



1 SCOPE

The aim of this specification is to create a state-of-the-art communication protocol for handling board transfers and associated data at surface-mount technology (SMT) production lines. Therefore, this new communication protocol has to cope with the following:

- Replace the electrical SMEMA interface as specified in IPC-SMEMA-9851
- Extend the interface to communicate:
 - Unique identifiers for the handled printed circuit boards (PCBs)
 - Equipment identifiers of the first machine noticing a PCB
 - Barcodes
 - Conveyor speed and intended board route
 - A lightweight digital twin of the product containing e.g.,
 - Product type identifier
 - Length
 - Width
 - Thickness
 - Board state

With respect to version numbers The Hermes Standard adheres to the rules of Semantic Versioning 2.0.0 [SemVer_2.0.0].

Hints on naming:

- Wherever a feature is described by the word “**shall**” it is mandatory.
- The word “machine” is used for any equipment which can be found in a SMT production line (e.g., printers, placement machines, ovens, AOIs, transport modules, shuttles, stackers).
- The term “PCB” may also refer to carriers transporting PCBs.
- The word “Hermes” is used as abbreviation for “The Hermes Standard”.
- “The Hermes Standard” and IPC-HERMES-9852 are synonymys for the standard specified in this document and might be used interchangeably.



2 TECHNICAL CONCEPT

2.1 Prerequisites

This specification is based on the prerequisite, that any application implementing this protocol has to provide connectivity based on Internet Protocol (IP) [IETF_RFC_791] / [IETF_RFC_2460] via Transmission Control Protocol (TCP) [IETF_RFC_793] (ISO / OSI model [ISO_7498-1] layer 3) to the adjacent machines machines and for communication with supervisory systems.

2.2 Board IDs

Board individuals are identified by board IDs. These must be Globally Unique Identifiers (GUIDs) according to [ITU-T_REC_X.667], e.g., 123e4567-e89b-12d3-a456-426655440000. They are generated by the first machine in a consecutive row of machines implementing the Hermes protocol. The board ID is passed from machine to machine. If a machine in a line does not implement the Hermes protocol, the board ID is lost and a new one will be generated by the next machine implementing Hermes.

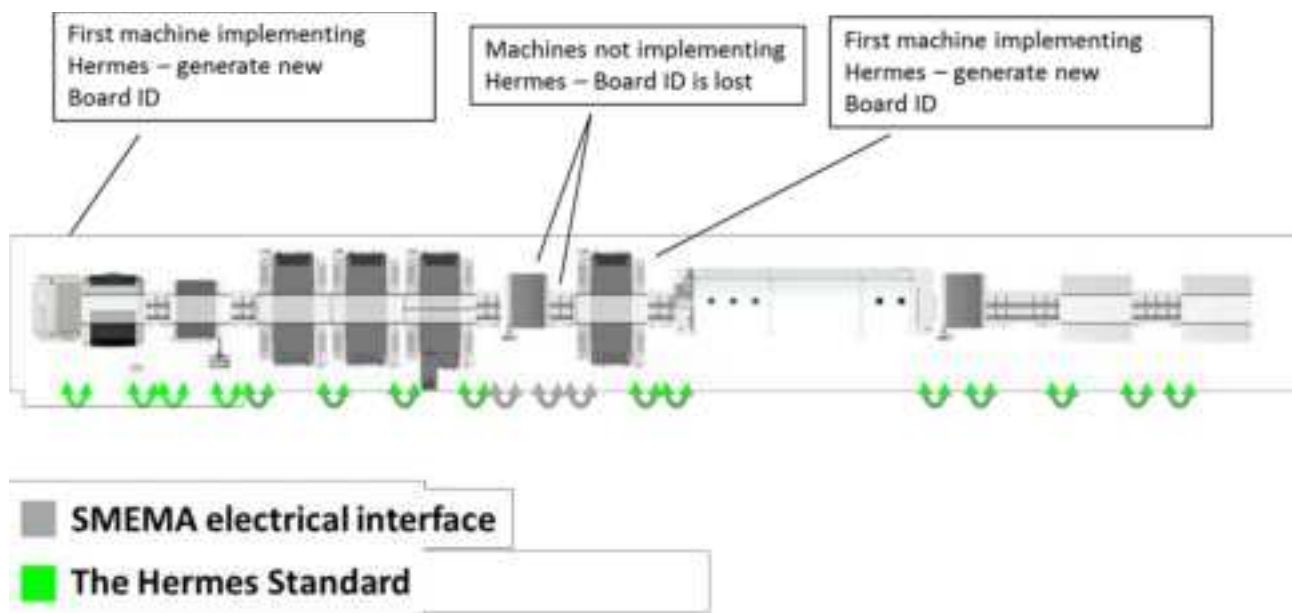


Figure 1 Generation of Board IDs

2.3 Machine-to-Machine Communication (Horizontal Channel)

2.3.1 Topology

Any machine in a line offers one TCP server per lane on its downstream side. Further servers per lane might also be necessary (e.g., if reverse transportation is supported). The TCP port number is not specified but can be configured by the user. The recommended port numbers are 50100 plus lane identifier (ID) with lanes being enumerated looking downstream from right to left beginning with 1 (i.e., for the left lane of a dual lane machine,



the upstream machine server accepts connections on port 50102). For every further server, plus 10 is recommended to be added to the port number.

The downstream machine opens one connection for every lane and every transportation interface on its upstream side to the upstream machine(s). So every PCB handover point corresponds to one TCP connection per exchange direction.

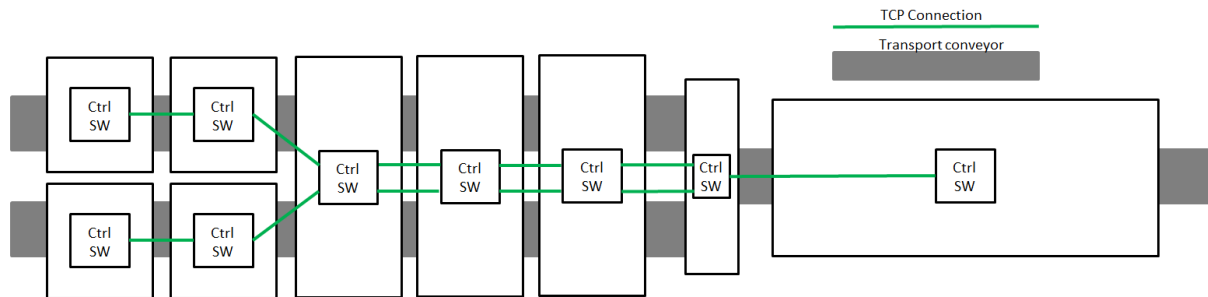


Figure 2 TCP Connections in a Line

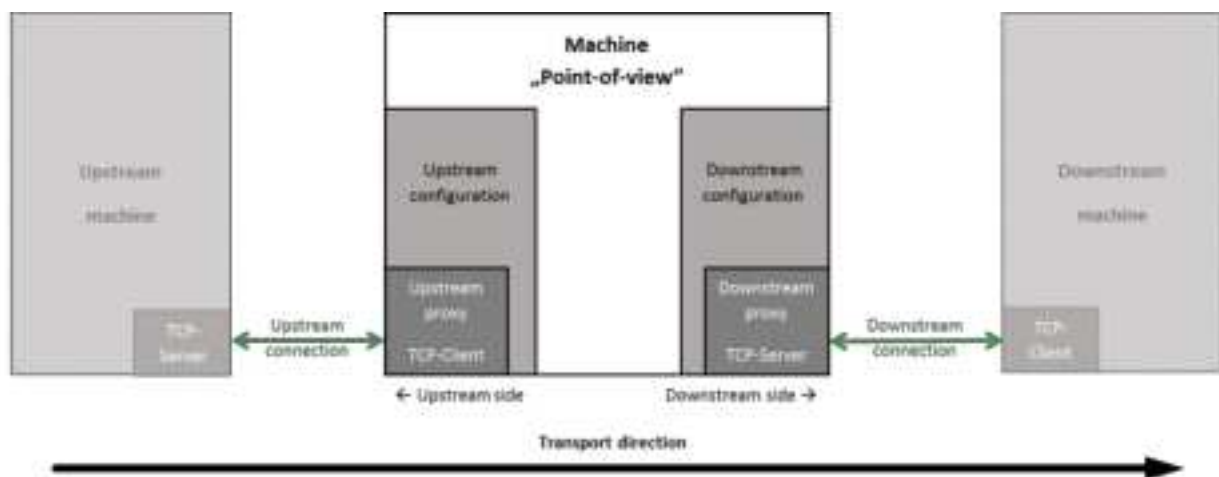


Figure 3 Upstream and Downstream From the Perspective of the Machine

2.3.2 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 machine. The features of the Hermes specification version 1.0 have to be supported by any implementation.

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 2.3.3).

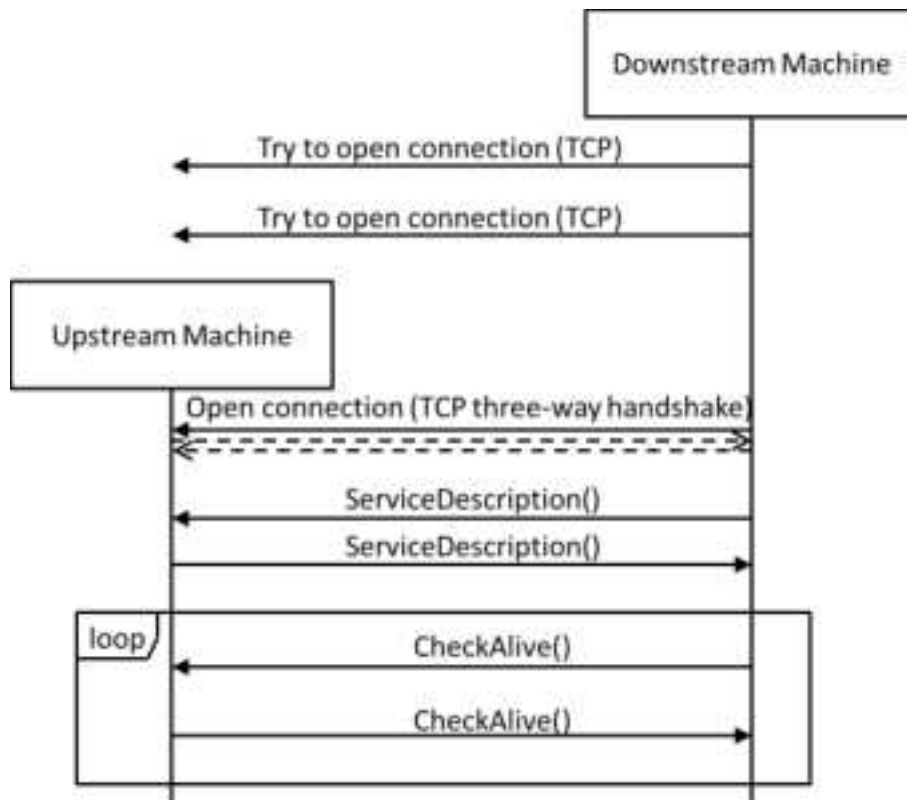


Figure 4 Connection, Handshake and Connection Loss Detection on Horizontal Channel

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 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 Figure 5.

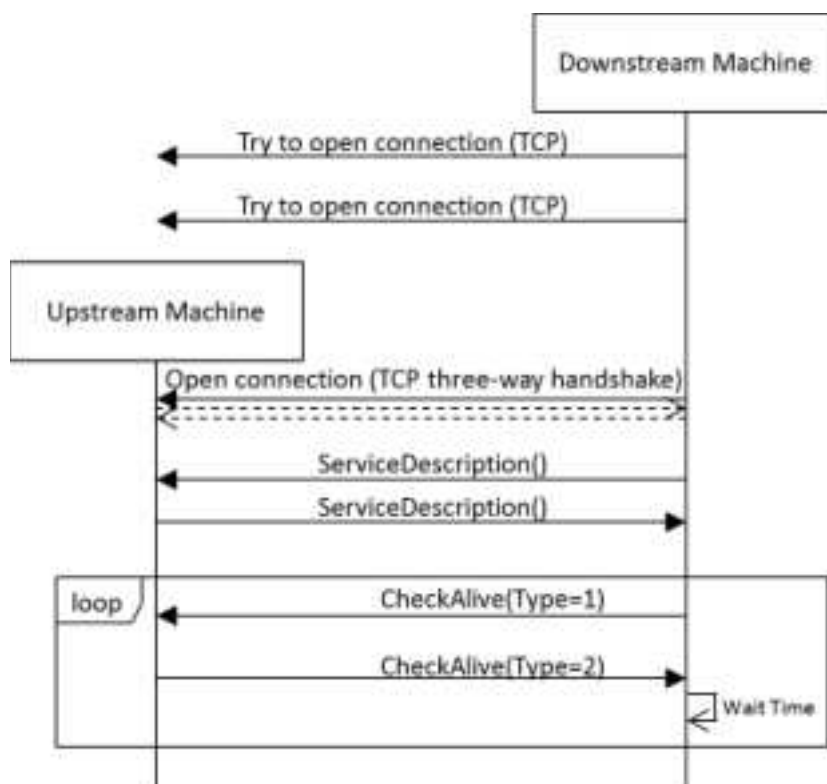


Figure 5 Example for Connection Loss Detection With FeatureCheckAliveResponse on Horizontal Channel

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.

2.3.3 Normal Operation

When an upstream machine has a PCB available for handover, it sends a BoardAvailable message, while a downstream machine ready to accept a PCB sends a MachineReady message. The naming of these messages is inspired by the electrical SMEMA interface. However, the messages do not represent the state of a machine's interface directly but are events for initiating a PCB handover.



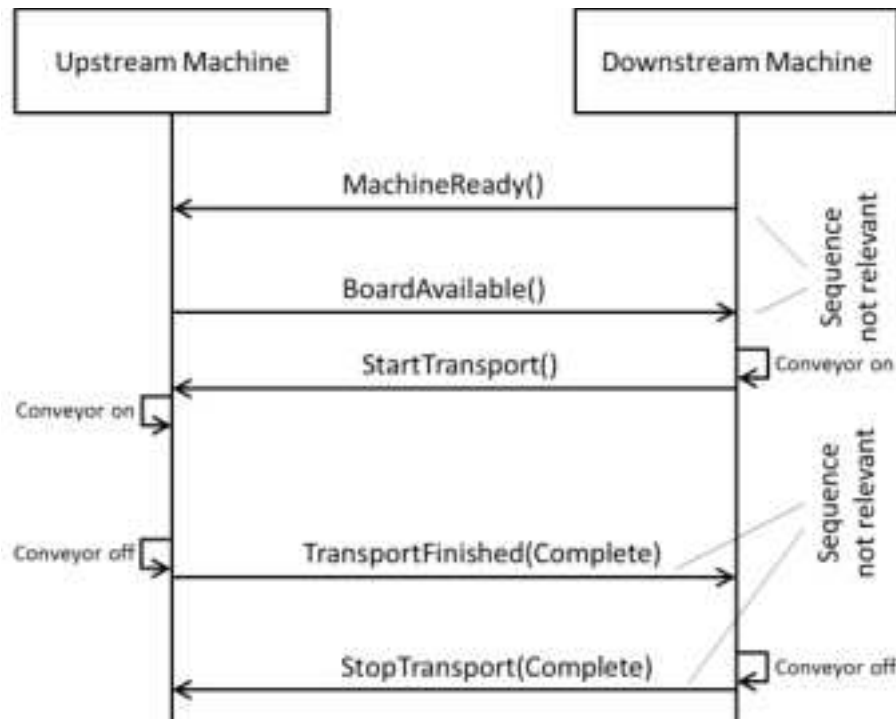


Figure 6 Communication Sequence for Board Transport

When both machines have indicated their readiness to handover the PCB, the downstream machine initiates the transfer by switching on its conveyor and sending the **StartTransport** message. Upon receiving this message, the upstream machine switches on its conveyor and the PCB moves into the downstream machine.

When the upstream machine is able to state that the PCB has fully left the machine, it sends the **TransportFinished** message. When the downstream machine has full control of the board, it sends the **StopTransport** message. The handover of a PCB is finished and is ready to start over.

If the upstream machine receives a **StopTransport** message and has not sent the **TransportFinished** message yet, it has to stop its conveyor and send the **TransportFinished** message.

The **MachineReady** message does not trigger an action on one of the machines directly. However, it still is necessary to realize machines like e.g., shuttles which have to react to the availability of their downstream machines.

2.3.4 Transport Error Handling

To keep this protocol hardware independent, the handling of transport errors is described based on a very simple model of the board handover. The handover process is structured into the three phases:

- NotStarted:** The board is fully inside the upstream machine.
- Incomplete:** The board is partly inside both machines.
- Complete:** The board is fully inside the downstream machine.



Any state or event which prevents one or both machines from handing over a PCB is interpreted as an error. An error may be detected by any of the machines in any of the three handover phases. It is up to the application how to detect the current handover phase, how to detect errors and how to solve them eventually (e.g., sensors, model based prediction, timeouts, user interaction).

The following sequence charts give an overview of the communication within this protocol depending on the machine which detects the error and the phase in which it is detected. The point in the sequence where the error is detected is marked by the following symbol: 

Scenario U1a

- Error detected by the upstream machine
- PCB fully inside the upstream machine
- Error detected before StartTransport has been received

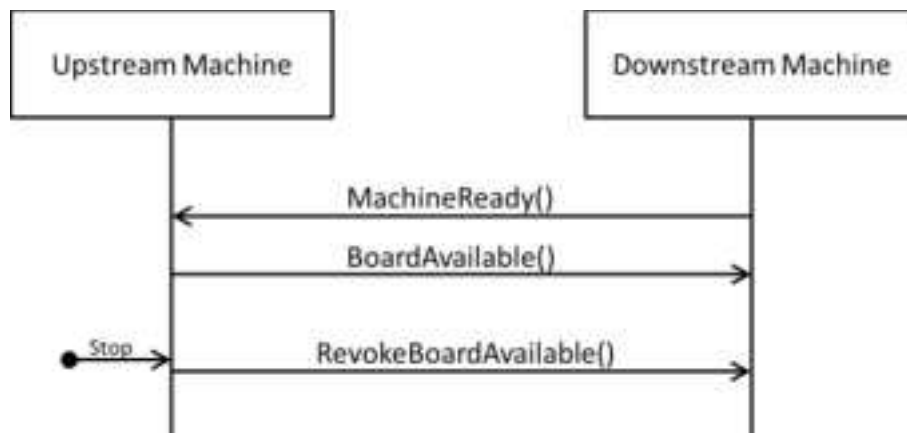


Figure 7 Communication Sequence in Scenario U1a

Error detection: The error is detected before any transport started.

Reaction on upstream machine: The upstream machine sends a RevokeBoardAvailable message.

Reaction on downstream machine: None.

Resolution: After the error is solved, the regular transport sequence can start from the beginning.

Scenario U1b

- Error detected by the upstream machine
- PCB fully inside the upstream machine
- Error detected after StartTransport has been received

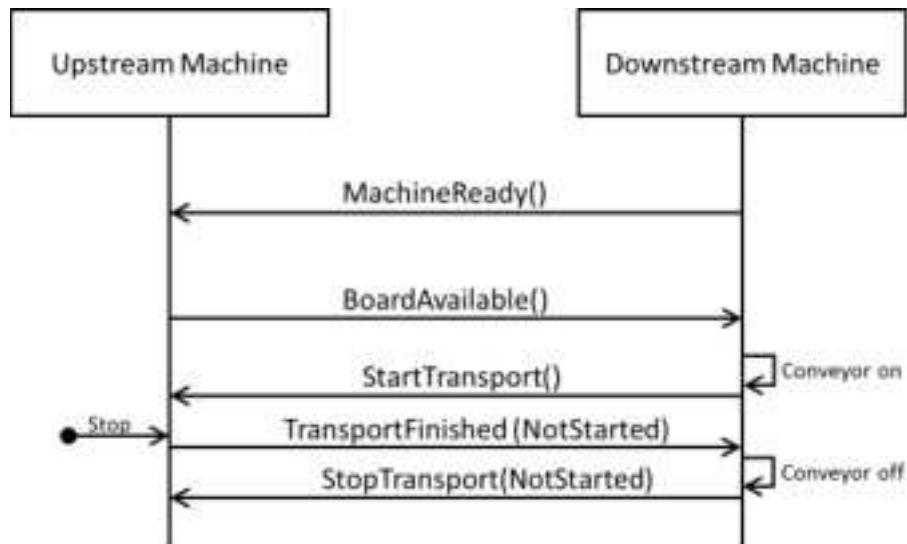


Figure 8 Communication Sequence in Scenario U1b

Error detection: The error is detected after the downstream machine started its conveyor and has sent the StartTransport message.

Reaction on upstream machine: The upstream machine sends a TransportFinished message indicating that it has not started the transport.

Reaction on downstream machine: Upon the TransportFinished message, the downstream machine stops its conveyor and sends a StopTransport message indicating that no transport has started.

Resolution: After the error is solved, the regular transport sequence can start from the beginning.

Scenario U2

- Error detected by the upstream machine
- PCB partly inside both machines

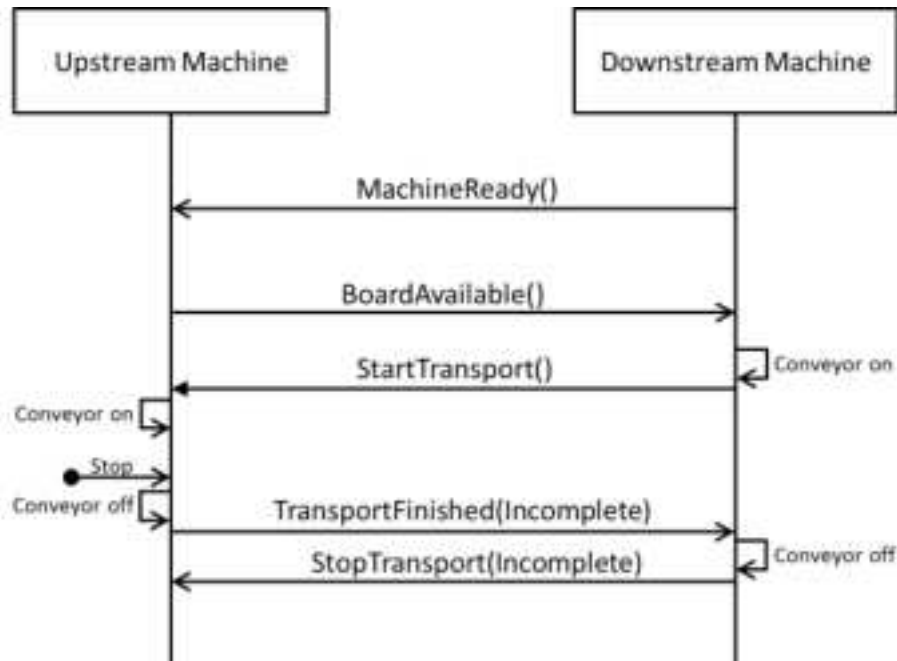


Figure 9 Communication Sequence in Scenario U2

Error detection: The error is detected after both machines started their conveyors. The upstream machine assumes that the PCB may have partly entered the downstream machine.

Reaction on upstream machine: The upstream machine sends a TransportFinished message indicating that the PCB might be located between the machines.

Reaction on downstream machine: Upon the TransportFinished message, the downstream machine stops its conveyor and sends a StopTransport message indicating the state of the PCB handover. Note that in Figure 9 the StopTransport message is represented with parameter "Incomplete". However, in this scenario, the downstream machine could send any of the allowed transport states.

Resolution: After the error is solved, the regular transport sequence can start from the beginning. The regular transport message sequence also applies to a PCB located between the two machines.



Scenario U3

- Error detected by the upstream machine
- PCB fully inside the downstream machine

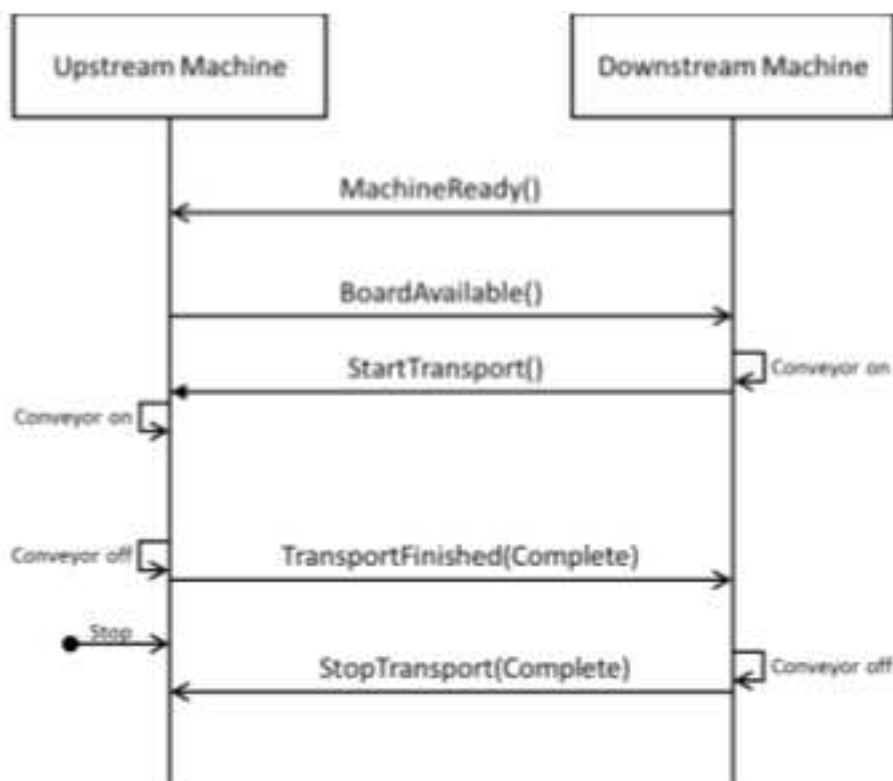


Figure 10 Communication Sequence in Scenario U3

Error detection: The error is detected after the PCB is fully inside the downstream machine.

Reaction on upstream machine: None. Although the machine detected an error, it is irrelevant for the handover process.

Reaction on downstream machine: None. The downstream machine is not aware of any error.

Resolution: This scenario is irrelevant for the Hermes protocol. It is just listed for completeness.



Scenario D1

- Error detected by the downstream machine
- PCB fully inside the upstream machine
- Error detected before StartTransport has been sent

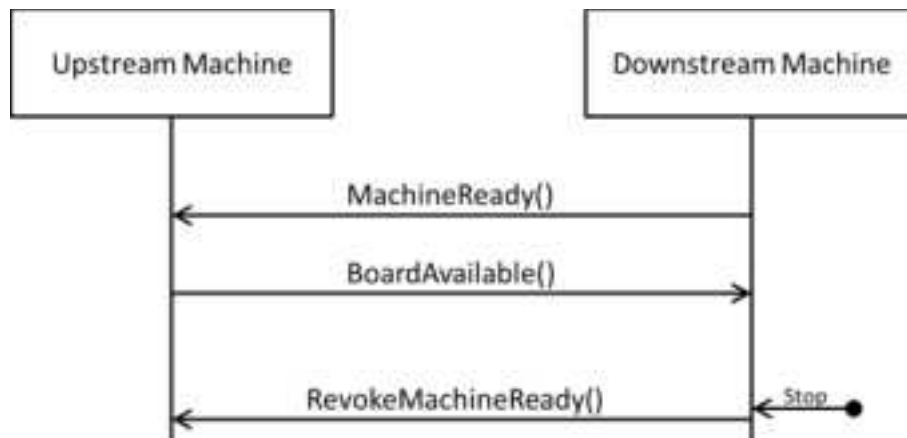


Figure 11 Communication Sequence in Scenario D1

Error detection: The error is detected before any transport started.

Reaction on upstream machine: None.

Reaction on downstream machine: The downstream machine sends a RevokeMachineReady message.

Resolution: After the error is solved, the regular transport sequence can start from the beginning.

Scenario D2

- Error detected by the downstream machine
- PCB partly inside both machines

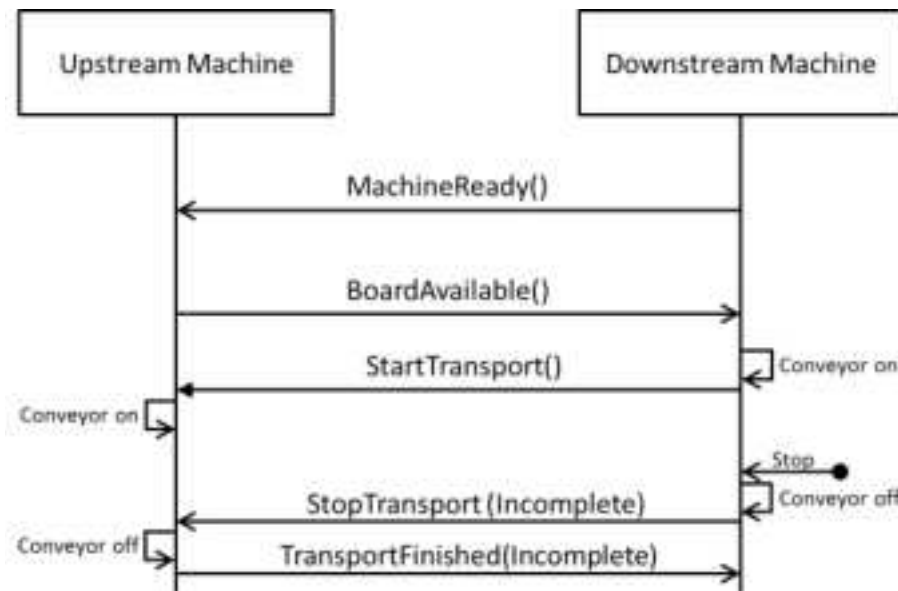


Figure 12 Communication Sequence in Scenario D2

Error detection: The error is detected after both machines started their conveyors. The downstream machine assumes that the PCB may already have entered its conveyor.

Reaction on upstream machine: Upon the StopTransport message from the downstream machine, the upstream machine stops its conveyor and sends a TransportFinished message indicating the state of the PCB handover. Note that in Figure 12 the TransportFinished message is represented with parameter "Incomplete". However, in this scenario, the upstream machine could send any of the allowed transport states.

Reaction on downstream machine: The downstream machine stops its conveyor and notifies the upstream machine of the error by sending a StopTransport message indicating an incomplete PCB handover.

Resolution: After the error is solved, the regular transport sequence can start from the beginning. The regular transport message sequence also applies for a PCB located in between the two machines.



Scenario D3

- Error detected by the downstream machine
- PCB inside the downstream machine

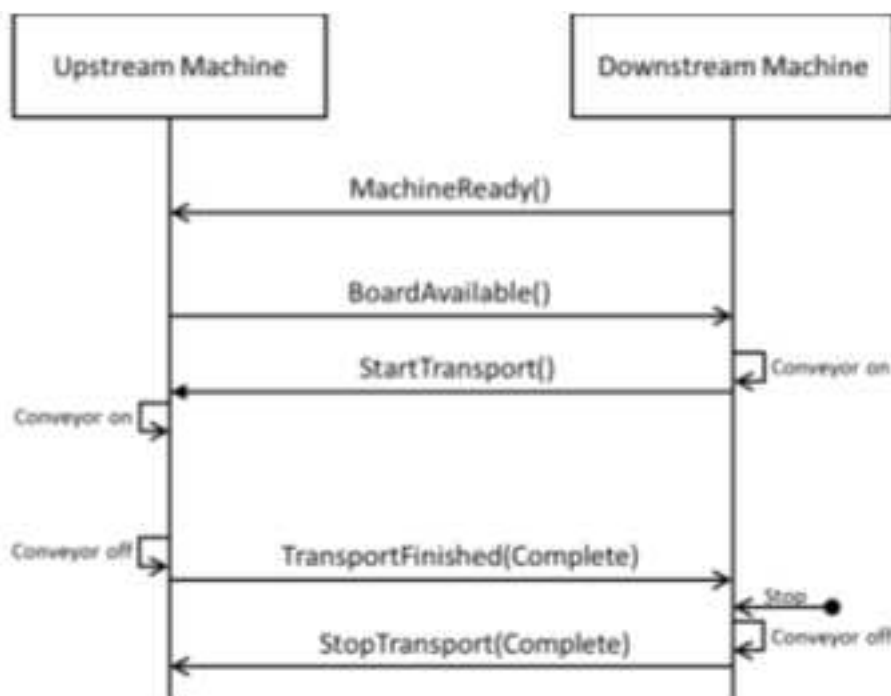


Figure 13 Communication Sequence in Scenario D3

Error detection: The error is detected after the PCB is fully inside the downstream machine.

Reaction on upstream machine: None. The upstream machine is not aware of any error.

Reaction on downstream machine: None (at least in the scope of this protocol).

Resolution: This scenario is irrelevant for the Hermes protocol. As transport sequences are always initiated by the downstream machine sending StartTransport, trouble-shooting (possibly including running the conveyor of the downstream machine) can be executed independently from the upstream machine.



2.3.5 Handling of BoardForecast

Among others the BoardForecast may be used in following scenarios:

- Scenario 1: Anticipating a product change without a board (e.g., because upstream machine does not have stoppers / conveyor that can be stopped).
- Scenario 2: Sending an estimated time to downstream machine until a board will be available (e.g., to allow downstream machine to choose between several upstream machines to get next available board).

Scenario 1

Upstream machine is processing a changeover (new product type) and wants to ensure that the downstream machine is simultaneously also processing a changeover. Upstream machine also needs to check that this actually happens. It sends a BoardForecast with a (forecast-)ID, to which the downstream machine at some point must respond with a MachineReady with the same ID. Upon receiving this MachineReady, the upstream machine can assume that the product change was successful.

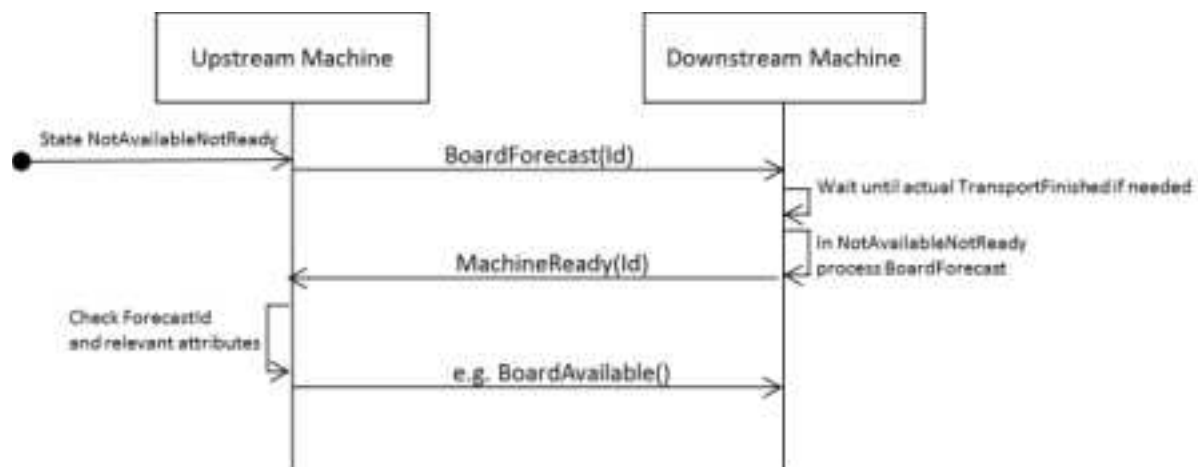


Figure 14 Example of Communication Sequence for BoardForecast

Note: If starting the BoardForecast handling in the state MachineReady, the downstream machine must send a RevokeMachineReady message (see Figure 15).

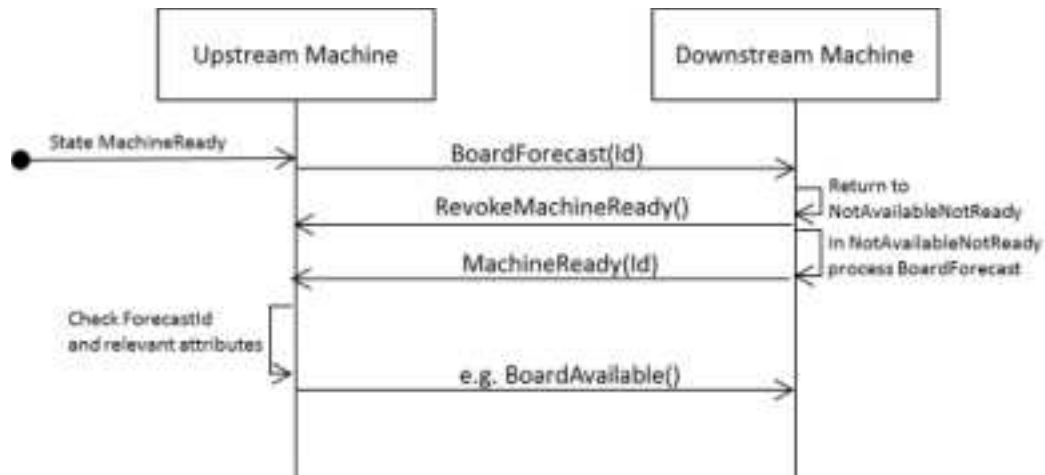


Figure 15 Example of Communication Sequence for BoardForecast With RevokeMachineReady

If several BoardForecast messages (e.g., with different ProductTypeId) are sent in a short delay, the downstream machine may process only the last BoardForecast message:

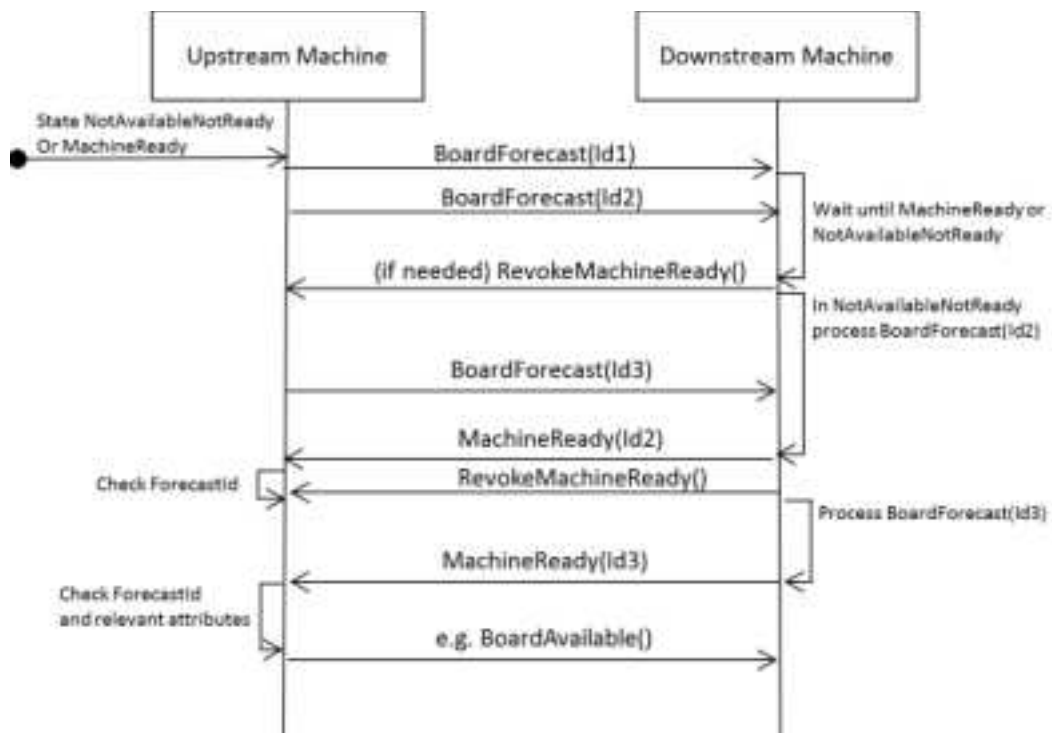


Figure 16 Example of Communication Sequence With Several BoardForecast



Scenario 1 (error handling)

If the downstream machine cannot accept the product exchange (e.g., unknown ProductId or width is physically impossible in machine) it will respond after a `RevokeMachineReady` with a notification of type "BoardForecastError". The upstream machine must then do some error handling (e.g., ask operator if machine should retry the BoardForecast or if the operator wants to remove the board).

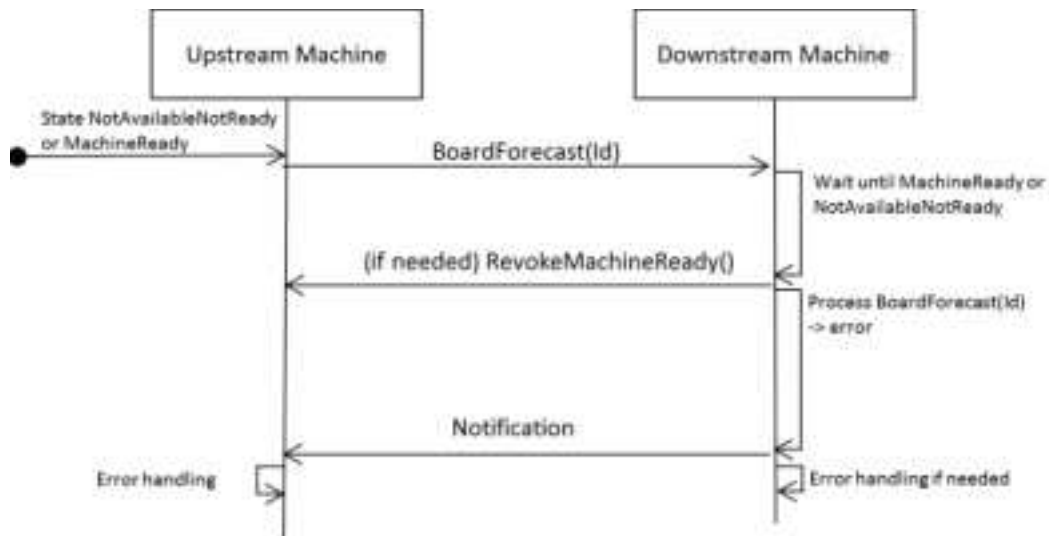


Figure 17 Example of Communication Sequence in Case With Error Handling

Scenario 2

As BoardForecast in that case usually only gives some information to the downstream machine, several BoardForecast may be sent. However, error handling or checking are not needed on the side of the downstream machine. In that scenario ForecastId will not be sent.

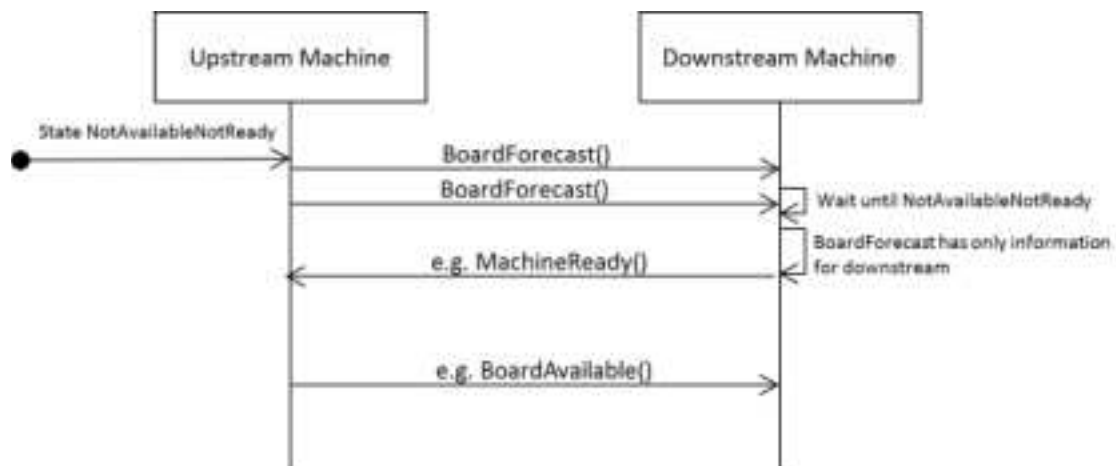


Figure 18 Example of Communication Sequence BoardForecast Without Product Change

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



2.3.6 Protocol States and Protocol Error Handling

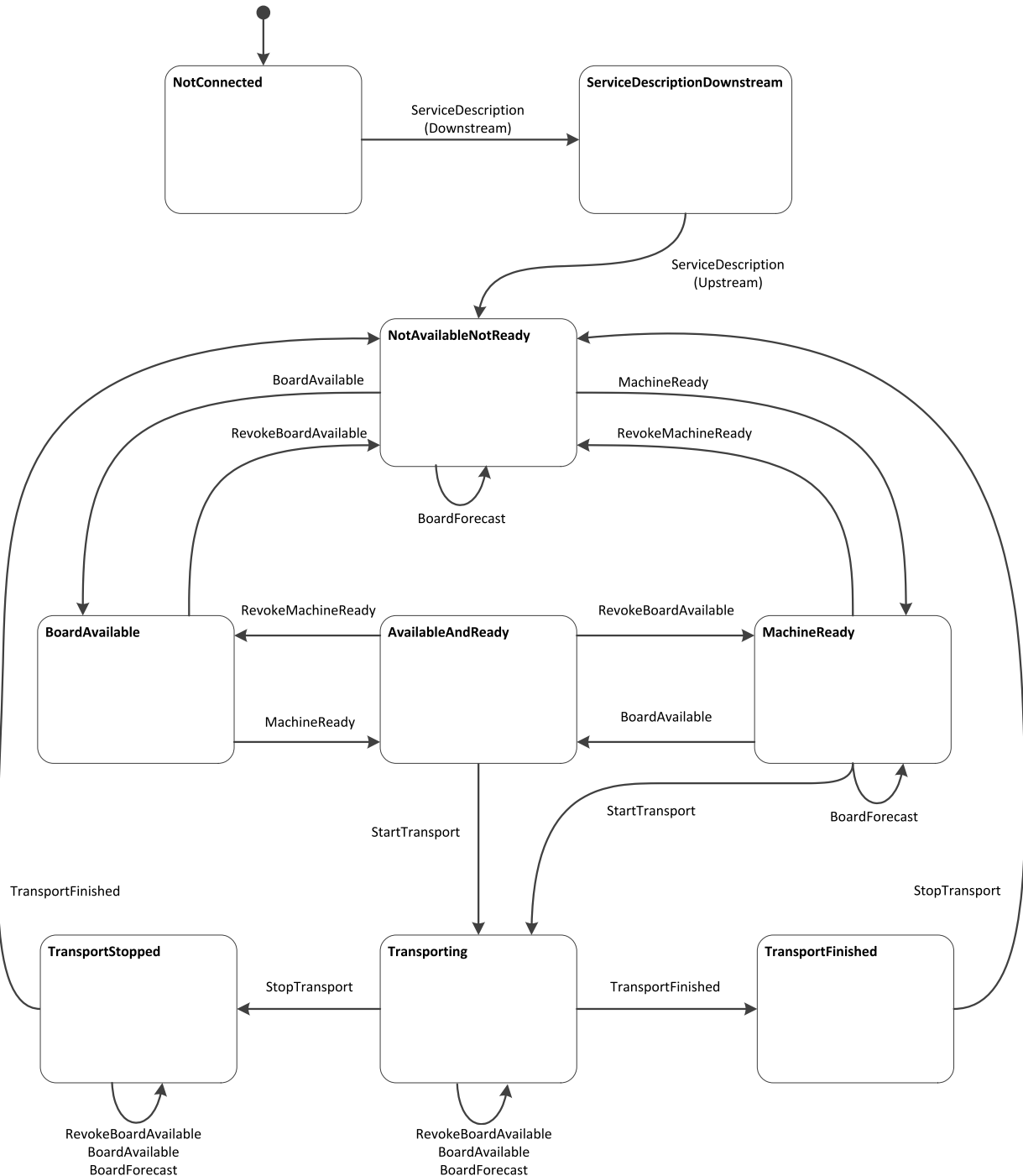


Figure 19 Hermes Interface States on Horizontal Channel

Figure 19 lists all states and transitions of a Hermes interface corresponding to the machine-to-machine (M2M) communication. The state is the comprehensive state of the interface rather than the state of one of the involved machines.



The messages may only be sent if they trigger the corresponding transition shown in the state chart. Any message defined in this standard, except "Notification", "CheckAlive", "QueryBoardInfo", "SendBoardInfo" and "Command", which is received not triggering a transition is interpreted as a protocol error (e.g., a MachineReady message when the interface is in the state Transporting). In case of a protocol error, any running transport **shall** be stopped and the connection is terminated. The interface may start over with a new connection. To keep upward compatibility, any message or attribute unknown by an implementation shall be ignored and discarded. Note that due to race conditions, a RevokeBoardAvailable message may overlap with a StartTransport message or even a StopTransport message, so this **shall** not be treated as a protocol error (transition from MachineReady to Transporting and self-transitions on Transporting and TransportStopped).

2.3.7 Handling of Attribute 'Route'

Usually a machine in the production line knows its task by manual configuration, by analyzing the properties of the next coming board, or by an ERP/MES system (vertical communication). However, in some cases it is necessary for process machines to control adjacent transport facilities. The horizontal M2M-communication consequently provides a way to forward such instructions along with the board transfer. This option also makes it possible to keep transport machine simple.

Route is an optional attribute inside the messages 3.6 BoardAvailable, 3.18 SendBoardInfo, 3.23 BoardArrived, 3.24 BoardDeparted and 3.26 SendWorkOrderInfo. A Route number can guide the course of the board within a production line either on a set path or towards a specific destination machine. The concept for board routing must always be defined as part of a production line planning in collaboration with all machines involved. Special cases like reverse transportation or manual removal of the board are also possible. Some common scenarios are described in the appendix 4.1.

If **Route** is received from upstream and insignificant or unknown to the machine it **shall** be forwarded unaltered downstream. But the machine can react to it or set a new route, dependent on its own available function and settings.

2.3.8 Handling of Attribute 'Action'

Action is an optional attribute inside the messages 3.6 BoardAvailable, 3.23 BoardArrived and 3.24 BoardDeparted. It is also intended to control adjacent simple transport facilities. Action refers to a real board transfer and therefore it is not included in messages of data-without-board-transfer. The concept for being initiator or executor of actions must always be defined as part of a production line planning in collaboration with all machines involved. Some common scenarios are described in the appendix 4.1.

If **Action** is received from upstream and insignificant or unknown to the machine it **shall** be forwarded unaltered downstream. When an **Action** is executed by a machine, it **shall** not be forwarded downstream or it **shall** be passed with 0. However, the machine can initiate a new **Action** to be performed downstream if needed.

2.4 Remote Configuration

2.4.1 Topology

Although a machine may offer the possibility to configure the Hermes TCP port(s) and the IP address(es) of its upstream machine(s) locally (e.g., via a graphical user interface of the machine controller), every machine implementing this protocol **shall** offer a possibility to configure these properties remote via TCP. Therefore,

