Technical Report

I. Fliege

Documentation of micro protocols

Computer Science Department
University of Kaiserslautern
Erwin-Schrödinger-Str.
D-67663 Kaiserslautern, Germany

Technical Report 358/07

1 Introduction

This document is a tutorial describing the documentation of micro protocols specified in SDL. Additional formal comments in the SDL specification enable an automatic documentation of the protocol behavior, interfaces, data types and the provided service. The documentation algorithm analyses the state machine to extract a typical behavior of the protocol. Comments that are provided by the developers give further information and help to build a comprehensible description to the acquired scenarios. Additionally, the algorithm builds an accessibility graph by taking all possible states and messages into account. From this graph the provided service of micro protocols can be extracted.

The analysis of used packages, data types, signals and procedures generates a list of all provided definitions, the required definitions and the reason for the usage of all used package.

2 Mandatory comments

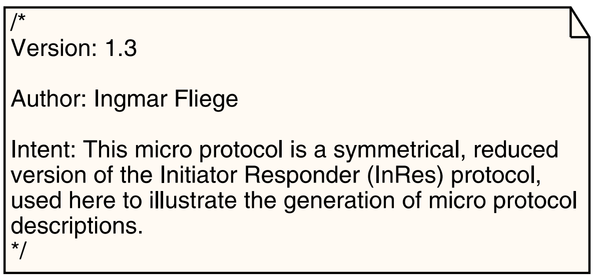
The documentation of micro protocols is possible for any of the following SDL container types:

- Packages (recommended)
- Processes
- Blocks

In order to automatically create a documentation of a micro protocol, a comment must be inserted in the protocol specification which must contain the following keywords:

- **Author:** providing the author(s) of a micro protocol
- **Version:** indicating the current version of this micro protocol
- **Intent:** describing the behavior and possible usage of the micro protocol.

This comment has to be inserted at the position in the SDL specification where the documentation should start. All elements within this scope will be analyzed and included in the documentation. For each comment a separate documentation is generated. Therefore, the position of the comment can affect the number and size of the created documentation.



```
/*
Version: 1.3

Author: Ingmar Fliege

Intent: This micro protocol is a symmetrical, reduced
version of the Initiator Responder (InRes) protocol,
used here to illustrate the generation of micro protocol
descriptions.
*/
```

Example 0.1: Essential comment

Example: If the comment is placed in a package definition, all blocks and processes within will be used for documentation. If the comment is placed within a process definition, the analysis will only cover this process without the surrounding scope.

Warning:
Any string used for the documentation of protocols will be reformatted!
The comment must be provided continuously! A separation may lead to a failure of detection. The parser uses any string behind the keywords, including newlines and separators. The end of the string is detected when the next keyword is found or the comment end is reached.

The comment can be augmented as follows:

- **Checklist:** providing a list of items that may help future developers to reuse the protocol. The character '-' is used to start a description of a checklist item.
- **Type:** specifying the type of protocol/component e.g. connection management, forward error correction, routing protocol.
- **Used:** specifying the project in which this component has been used

3 Basic comments

Important:
All comments described on the following pages must only describe one actual SDL action e.g. input, output, state or decision , and may not refer the context! In case of decisions an branches only the decision for this branch is given.

3.1 Triggers and Outputs

The automatic creation of the documentation is dependent on basic comments in the specified protocol. This includes the documentation of **inputs** and **outputs** of the state machine. In SDL this covers the consumption of **signals/timers** and the **output** of signals.

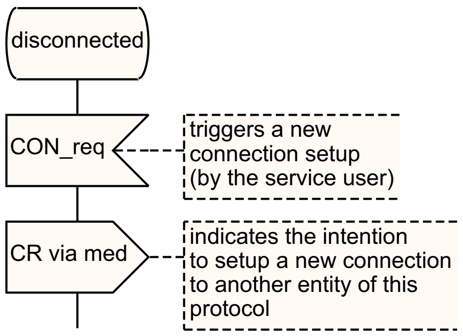
For each input trigger and output of a signal/timer a comment should be added, describing the causation of the action. The comment must complement the following sentence:

This signal/timer ....

Examples:

- "This signal indicates the request of a connection setup by the service user."
- "This timer fires due to absence of the signal x."

Warning: If the comment does not match the given sentence, the resulting documentation will contain grammatical errors and may be incomprehensible.

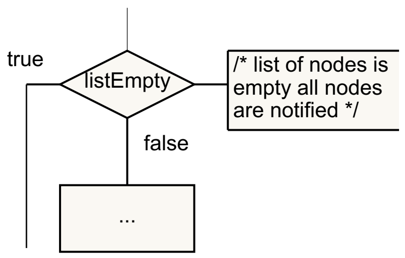

disconnected
CON_req
triggers a new connection setup (by the service user)
CR via med
indicates the intention to setup a new connection to another entity of this protocol

Example 0.2: inputs and outputs

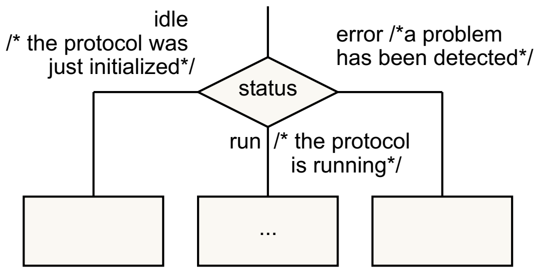
A missing comment is permitted. The algorithm will generate a default sentence describing the actions. This may be used if the name of a signal or timer is meaningful enough.

3.2 Decisions

SDL state machines allow the branching of transitions, allowing the specification of alternative behavior, which is done due to an internal decision of a protocol. Therefore any decision in SDL must be commented in order to provide a reasonable description. There are two possibilities to comment a decision in SDL:



Example 0.3: A boolean decision


idle /* the protocol was just initialized*/
error /*a problem has been detected*/
status
run
/* the protocol is running*/
...

Example 0.4: Multiple branches

1. The decision itself is provided with a comment describing the "question" itself in common speech and should describe the positive (true) part of a the transition. This comment should qualify a sentence such as:
   If ...., the signal X is sent..
   The comment is used to **begin a sentence** in the doumentation, describing the reason why a branch is chosen.

   Warning: This comment is only possible for boolean decisions.

2. The decision branch can be provided with a comment. This comment has to be added in the range condition of the path (where true/false/else or any other range is placed). It is also possible to place the comment in the immediate following empty task-box of the decision branch. This kind of comment may be more expressive and describes the choice of a branch more precisely (see example 0.4). Again, this comment must fulfill the following sentence:

   If ....,

3. Both possibilities may be combined, whereby ConTraST will select an appropriate description.

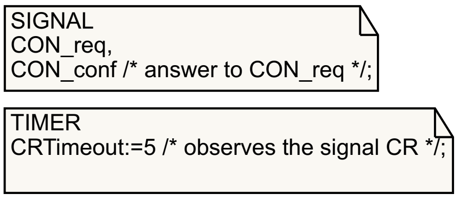
4 Optional recommended comments

Additional comments will improve the readability of the generated documentation:

4.1 Declarations

For declarations of signals and timers, an optional description can be provided. Signals can be related to each other, by giving a comment behind the declaration which includes **answer**, **response** or **reply** with one or more **signal name(s)** (see Example 0.5).

In Example 0.5, the signal CON_conf is amongst other signals the reply to the signal CON_req. An MSC of these signals is given in the example on page 11. In case of the SynchronousInquiry-Pattern the reply signal must be commented. In case of timers, the purpose e.g. supervision of an event can be specified. As shown in Example 0.5 you can add a comment which signal is **observed**.



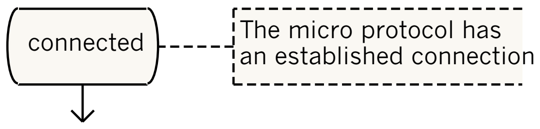
```
SIGNAL
CON_req,
CON_conf /* answer to CON_req */;
```

```
TIMER
CRTimeout:=5 /* observes the signal CR */;
```

Example 0.5: Declaration

4.2 States

States of a SDL specification may have a detailed description as expansion, which is used by the documentation algorithm as an introduction to applicable scenarios. The description is provided by adding a comment to the accordant state, as shown in example 0.6. The description must consist of (a) complete sentence(s), e.g. "The micro protocol has an established connection". In case of multiple descriptions of one state, any of them is selected at random.



Example 0.6: A description of states

Warning: The description of states only applies to actual states and not to next-state statements.

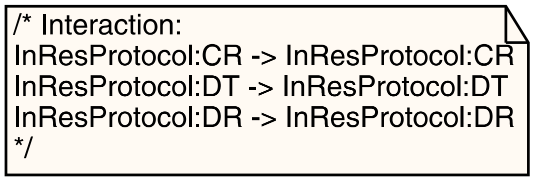
4.3 Interaction

It is possible to analyse the interaction behavior of micro protocols. By this, the provided service of the micro protocol can be extracted. This feature additionaly requires the description of related signals, as described in Section 4.1.

To analyse the interaction, an association between two or more micro protocols must be provided, by adding a comment with the following format:

SourceProtocol **:** *SourceSignal* **->** *TargetProtocol* **:** *TargetSignal*

This comment specifies the connection of the *SourceSignal* from the process *SourceProtocol* which is transfered to the *TargetProtocol* with a renaming to the signal *TargetSignal*. If no renaming is required, the *TargetSignal* must be the same as the *SourceSignal*. An example for this is given in Example 0.7.



```
/* Interaction:
InResProtocol:CR -> InResProtocol:CR
InResProtocol:DT -> InResProtocol:DT
InResProtocol:DR -> InResProtocol:DR
*/
```

Example 0.7: Description of interaction

5 Generating a documentation

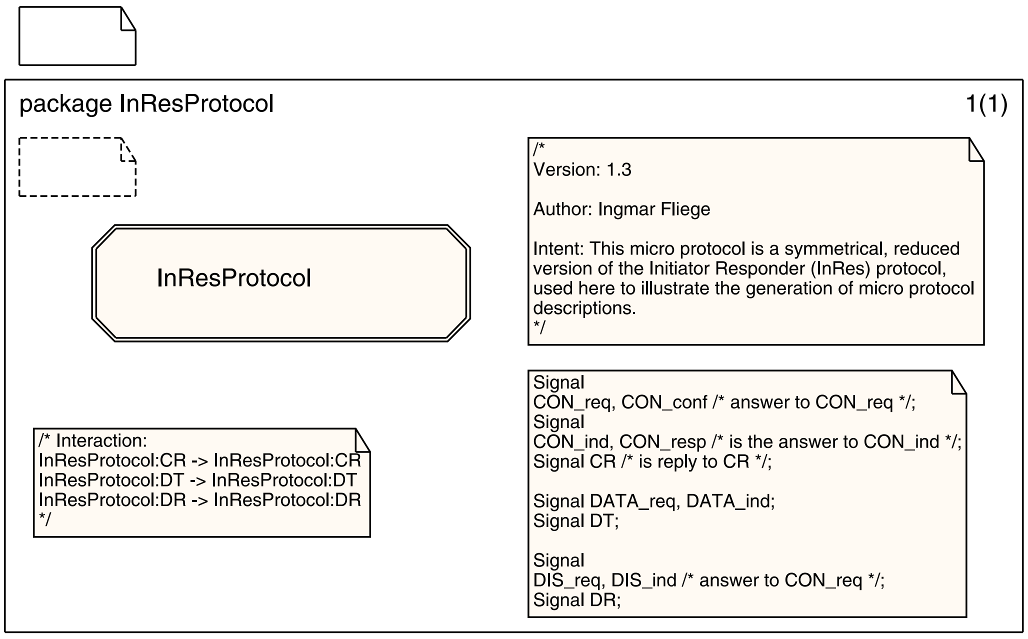
The input for the documentation algorithm is SDL/PR, the textual representation of SDL. Therefore a pr file must be created first, by analysing the given SDL system or package.

The automatic generation of a micro protocol is handled by ConTraST (Version 2.2 and above). The command line parameter "`-M`" or "`--document`" enable the documentation algorithm. The directory "`Doc`" will be created in the target directory (option "`-D`"), which contains separate LaTeX files for each protocol documentation.

The Telelogic Tau Organizer has a menu "Contrast", which has an entry "Create documentation". Also here, a prior analysis is required (menu item: "Analyse && Generate PR").

6 Resulting documentation

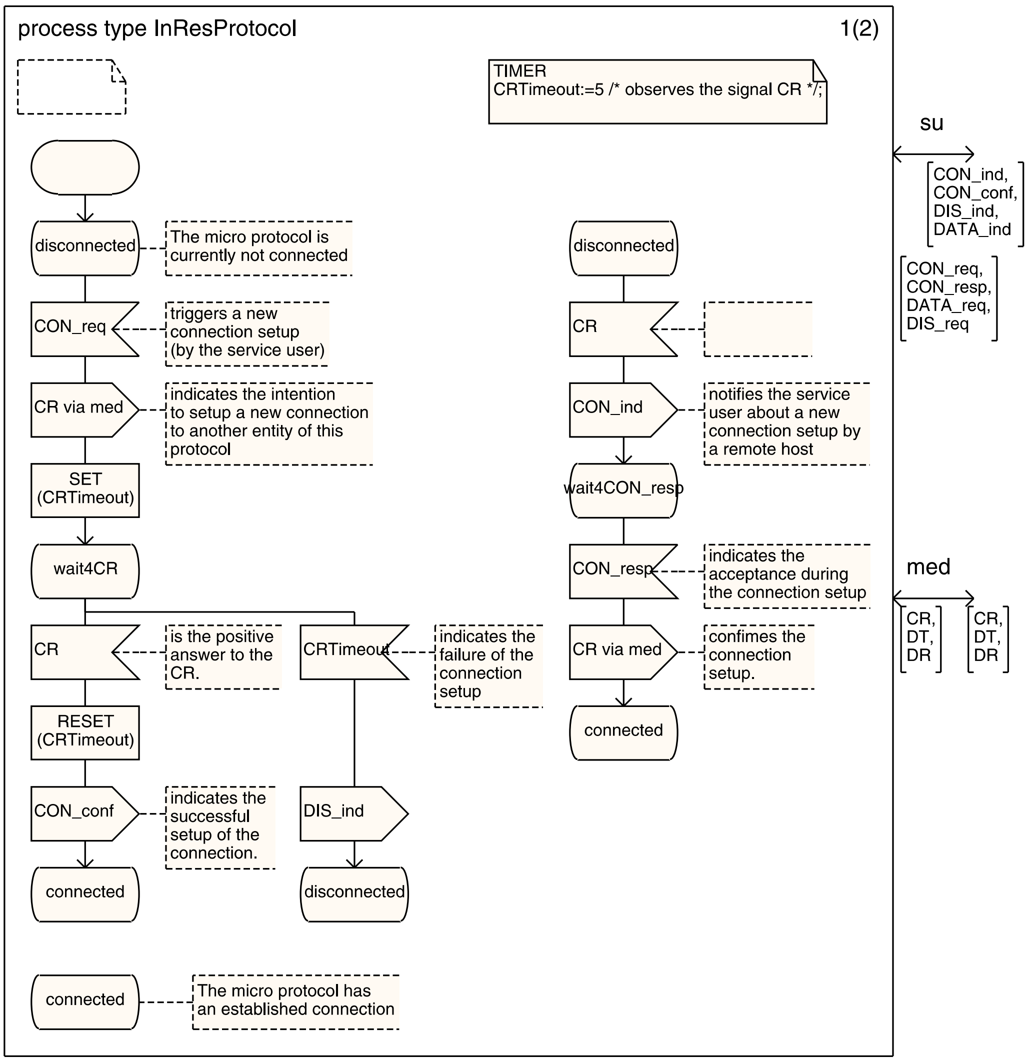
6.1 SDL specification with comments

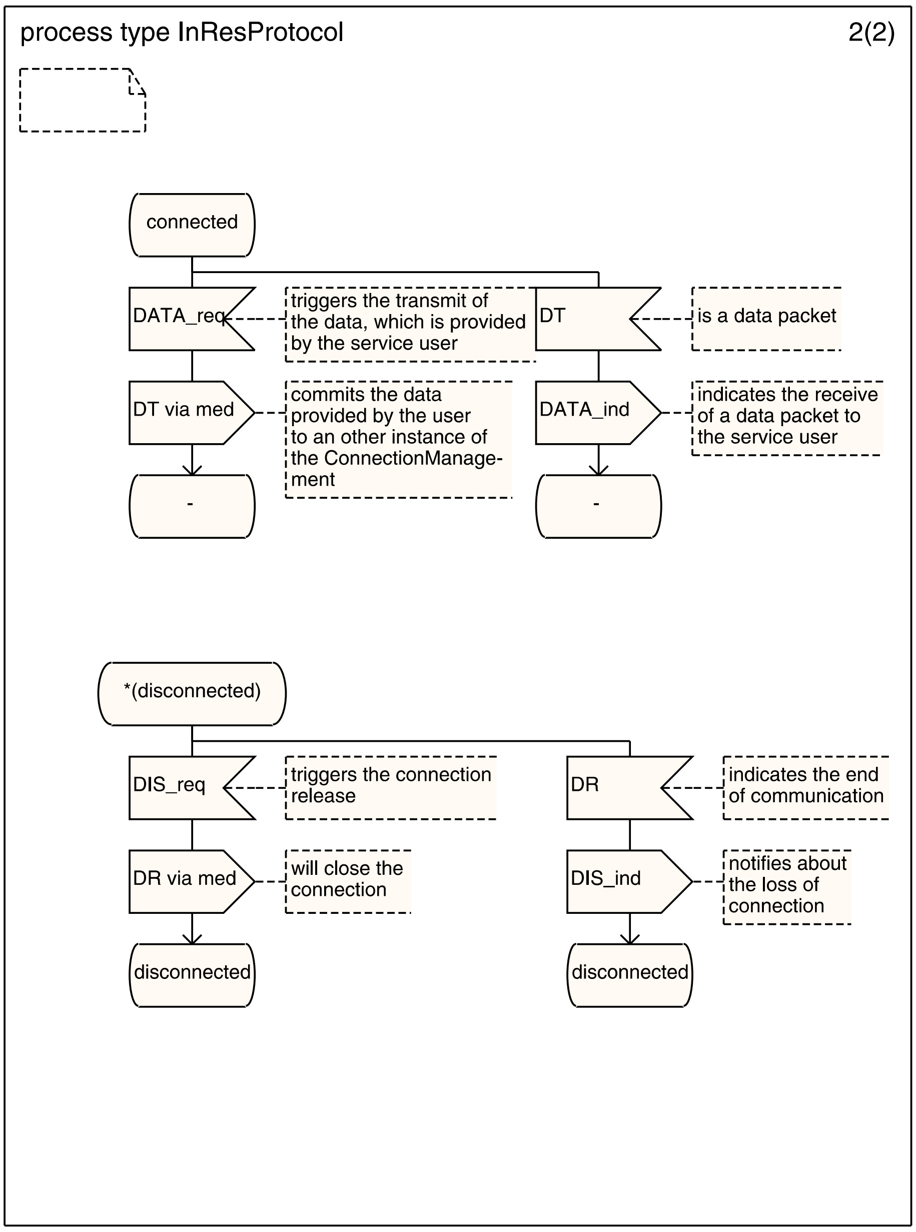


```
/*
Version: 1.3

Author: Ingmar Fliege

Intent: This micro protocol is a symmetrical, reduced
version of the Initiator Responder (InRes) protocol,
used here to illustrate the generation of micro protocol
descriptions.
*/
```

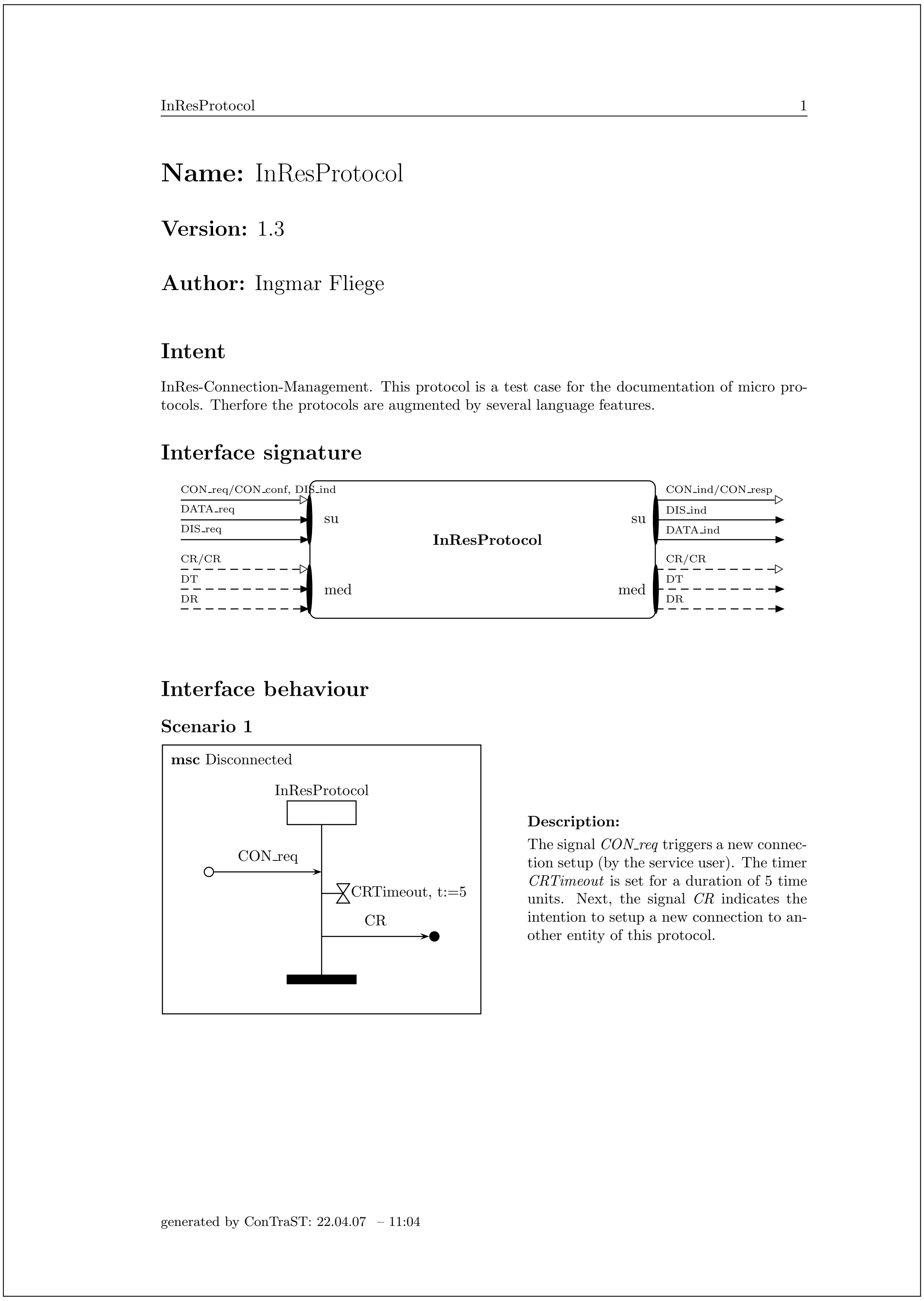
```
Signal
CON_req, CON_conf /* answer to CON_req */;
Signal
CON_ind, CON_resp /* is the answer to CON_ind */;
Signal CR /* is reply to CR */;

Signal DATA_req, DATA_ind;
Signal DT;

Signal
DIS_req, DIS_ind /* answer to CON_req */;
Signal DR;
```

```
/* Interaction:
InResProtocol:CR -> InResProtocol:CR
InResProtocol:DT -> InResProtocol:DT
InResProtocol:DR -> InResProtocol:DR
*/
```


process type InResProtocol
1(2)
TIMER
CRTimeout:=5 /* observes the signal CR */;
disconnected
The micro protocol is currently not connected
CON_req
triggers a new connection setup (by the service user)
CR via med
indicates the intention to setup a new connection to another entity of this protocol
SET (CRTimeout)
wait4CR
CR
is the positive answer to the CR.
CRTimeout
indicates the failure of the connection setup
RESET (CRTimeout)
CON_conf
indicates the successful setup of the connection.
DIS_ind
connected
disconnected
connected
The micro protocol has an established connection
disconnected
CR
CON_ind
notifies the service user about a new connection setup by a remote host
wait4CON_resp
CON_resp
indicates the acceptance during the connection setup
CR via med
confimes the connection setup.
connected
su
CON_ind, CON_conf, DIS_ind, DATA_ind
CON_req, CON_resp, DATA_req, DIS_req
med
CR, DT, DR
CR, DT, DR


process type InResProtocol
2(2)
connected
DATA_req
triggers the transmit of the data, which is provided by the service user
DT
is a data packet
DT via med
commits the data provided by the user to an other instance of the ConnectionManage-ment
DATA_ind
indicates the receive of a data packet to the service user
-
-
*(disconnected)
DIS_req
triggers the connection release
DR
indicates the end of communication
DR via med
will close the connection
DIS_ind
notifies about the loss of connection
disconnected
disconnected

6.2 Documentation in LaTeX



Name: InResProtocol

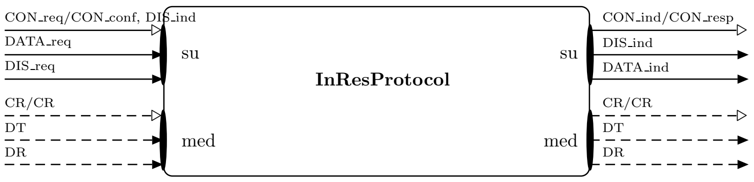
Version: 1.3

Author: Ingmar Fliege

Intent

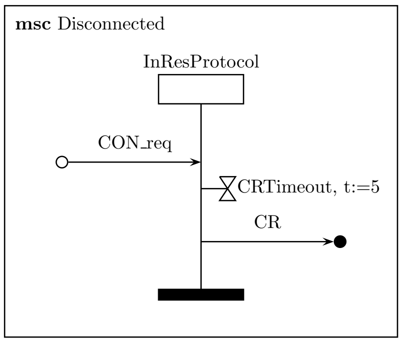
InRes-Connection-Management. This protocol is a test case for the documentation of micro protocols. Therfore the protocols are augmented by several language features.

Interface signature



Interface behaviour

Scenario 1

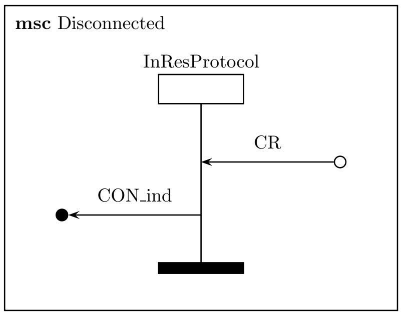


Description:

The signal *CON_req* triggers a new connection setup (by the service user). The timer *CRTimeout* is set for a duration of 5 time units. Next, the signal *CR* indicates the intention to setup a new connection to another entity of this protocol.

generated by ConTraST: 22.04.07 – 11:04

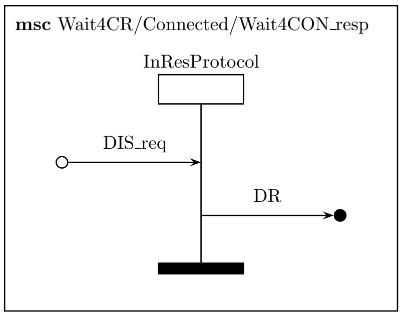
Scenario 2



Description:

The signal *CR* is consumed and the signal *CON_ind* notifies the service user about a new connection setup by a remote host.

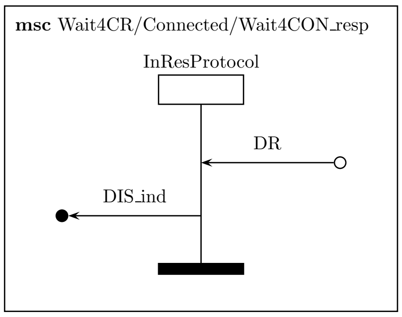
Scenario 3



Description:

The signal *DIS_req* triggers the connection release. The signal *DR* will close the connection.

Scenario 4

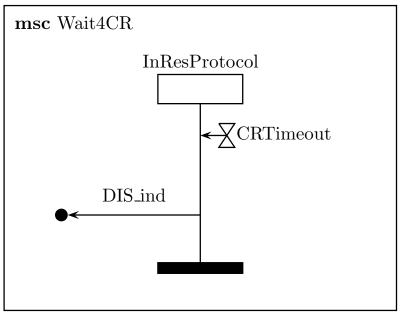


Description:

The signal *DR* indicates the end of communication. The signal *DIS_ind* notifies about the loss of connection.

generated by ConTraST: 22.04.07 – 11:04

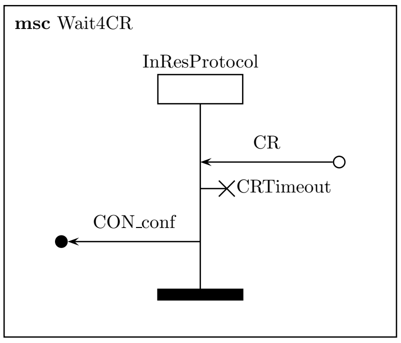
Scenario 5



Description:

The timer *CRTimeout* indicates the failure of the connection setup. The signal *DIS_ind* is sent.

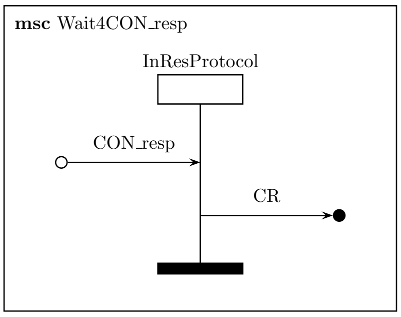
Scenario 6



Description:

The signal *CR* is the positive answer to the CR. The timer *CRTimeout* is reset. Next, the signal *CON_conf* indicates the successful setup of the connection.

Scenario 7

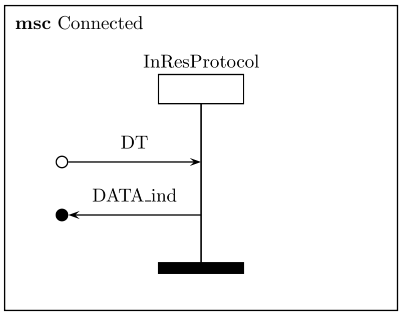


Description:

The signal *CON_resp* is consumed and the signal *CR* confimes the connection setup.

generated by ConTraST: 22.04.07 – 11:04

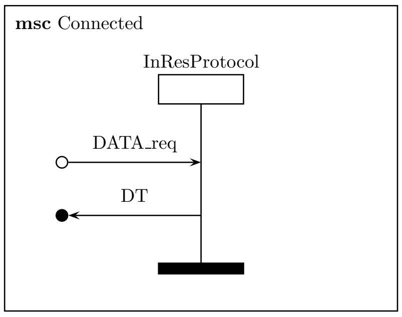
Scenario 8



Description:

The signal *DT* is a data packet. The signal *DATA_ind* indicates the receive of a data packet to the service user.

Scenario 9



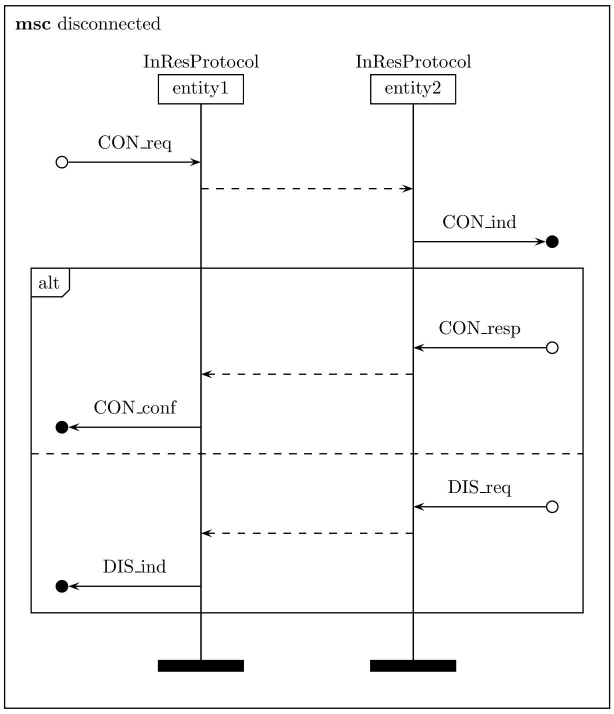
Description:

The signal *DATA_req* triggers the transmit of the data, which is provided by the service user. The signal *DT* commits the data provided by the user to an other instance of the InResProtocol.

generated by ConTraST: 22.04.07 – 11:04

Provided service

Scenario 1

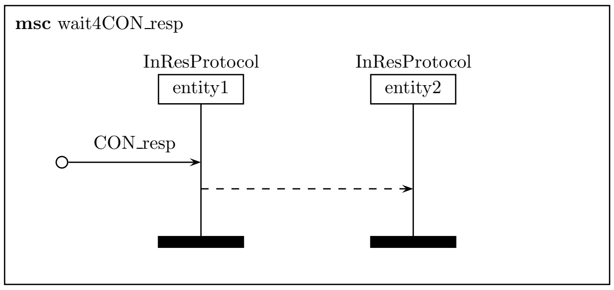


Description:

The signal *CON_req* triggers a new connection setup (by the service user). The signal *CON_ind* notifies the service user about a new connection setup by a remote host. Now, there are two alternatives:

- The signal *CON_resp* is sent. Next, the signal *CON_conf* indicates the successful setup of the connection.
- The signal *DIS_req* triggers the connection release. The signal *DIS_ind* notifies about the loss of connection.

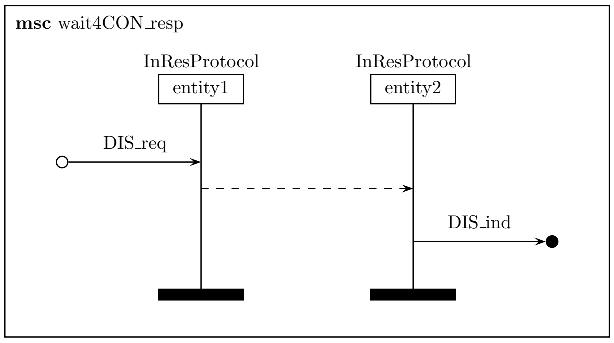
Scenario 2



Description:

The signal *CON_resp* is sent. This message is unexpected. Therefore no further operations are performed.

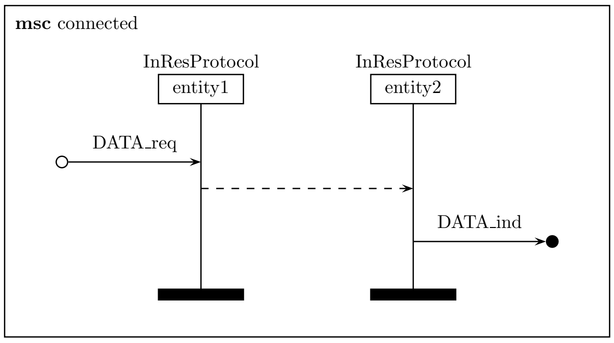
Scenario 3



Description:

The signal *DIS_req* triggers the connection release. The signal *DIS_ind* notifies about the loss of connection.

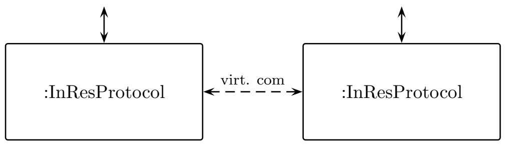
Scenario 4



Description:

The signal *DATA_req* triggers the transmit of the data, which is provided by the service user. The signal *DATA_ind* indicates the receive of a data packet to the service user.

Architecture



Imported and exported definitions

Used packages

– *none* –

Required definitions

– *none* –

Provided definitions

- Process type InResProtocol
- Signal CON_req, Signal CON_conf, Signal CON_ind, Signal CON_resp, Signal CR, Signal DATA_req, Signal DATA_ind, Signal DT, Signal DIS_req, Signal DIS_ind, Signal DR

Checklist

– *none* –

generated by ConTraST: 22.04.07 – 11:04

7 Restrictions

The following units are not supported and will be ignored:

- Procedures
- Services
- Systems