A Flexible Micro Protocol Framework

Ingmar Fliege

Computer Science Department – Networked Systems

University of Kaiserslautern

What are Micro Protocols?

- component as structuring unit (code-oriented)
- single (distributed) protocol functionality using a specific mechanism
- required collaboration
- well-defined interfaces (operators)
- self-contained and ready-to-use
- not decomposable

Micro Protocols

- can be composed yielding macro protocols
- examples:
  - protocol phases
    - connection setup, data transfer, neighbour sensing
  - protocol functionalities
    - flow control, loss control, error control, addressing, en/decoding, authentication, message forwarding, reservation, multiplexing, message sequencing, segmentation and reassembly, …

Why do we use micro protocols?

- structuring is essential
  - functionality, phases, entities, layers
- reuse plays a key role
  - components, subsystems, frameworks
- reduce development effort
- improvement quality and reliability

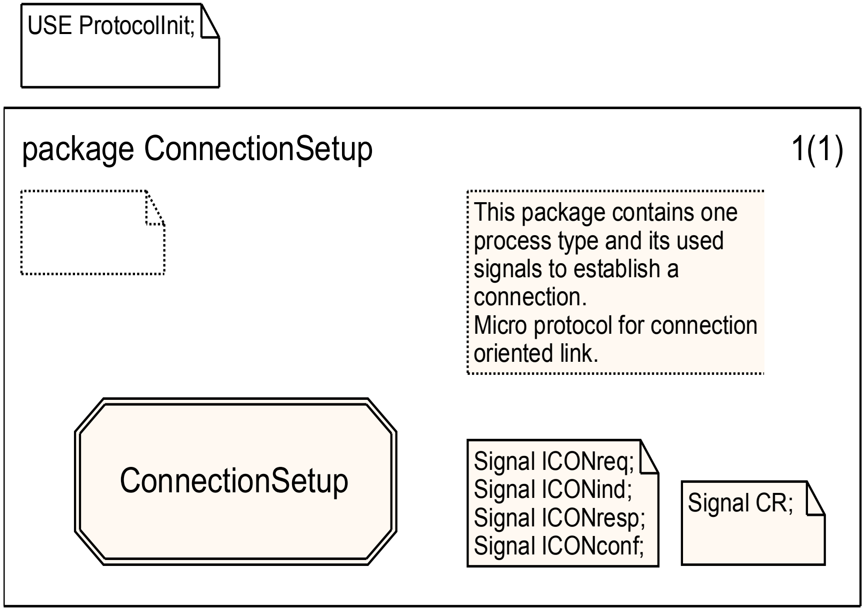
→ develop customized protocols

Micro Protocol Library

- each micro protocol is described as separate entity
  - summary of functionality and behaviour
  - structural description
  - interface definitions
  - subsets of collaborations (behaviour)
  - used data types
  - different realisations of composition
  - different languages (SDL, C++)

Packages

- micro protocols are encapsulated in SDL-packages.
- e.g. signals for communication between micro protocols are defined in separate packages



Framework

- skeleton of a system that has to adapted by a system developer
  - example: layered communication architectures
    adaptation by adding specific protocols
- A generic micro protocol framework is a set of general principles and rules for the composition of micro protocols
  - generic composition: concurrent, in sequence, hierarchical
  - problem specific composition: exceptions, notifications, calls

Operators

>> passing the thread of control to another micro protocol

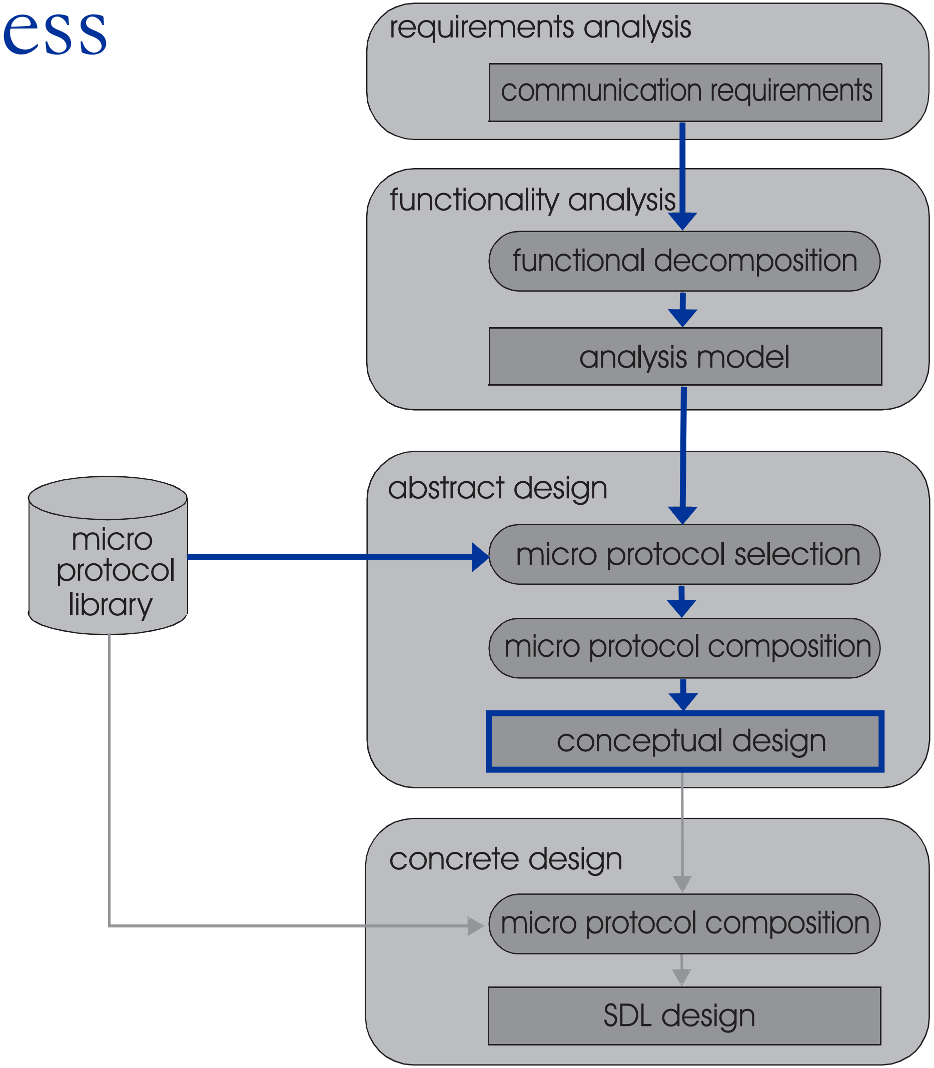
[> taking the thread of control (interruption)

@> signalling the occurrence of an event or exception

→ data flow (with direction indicated)

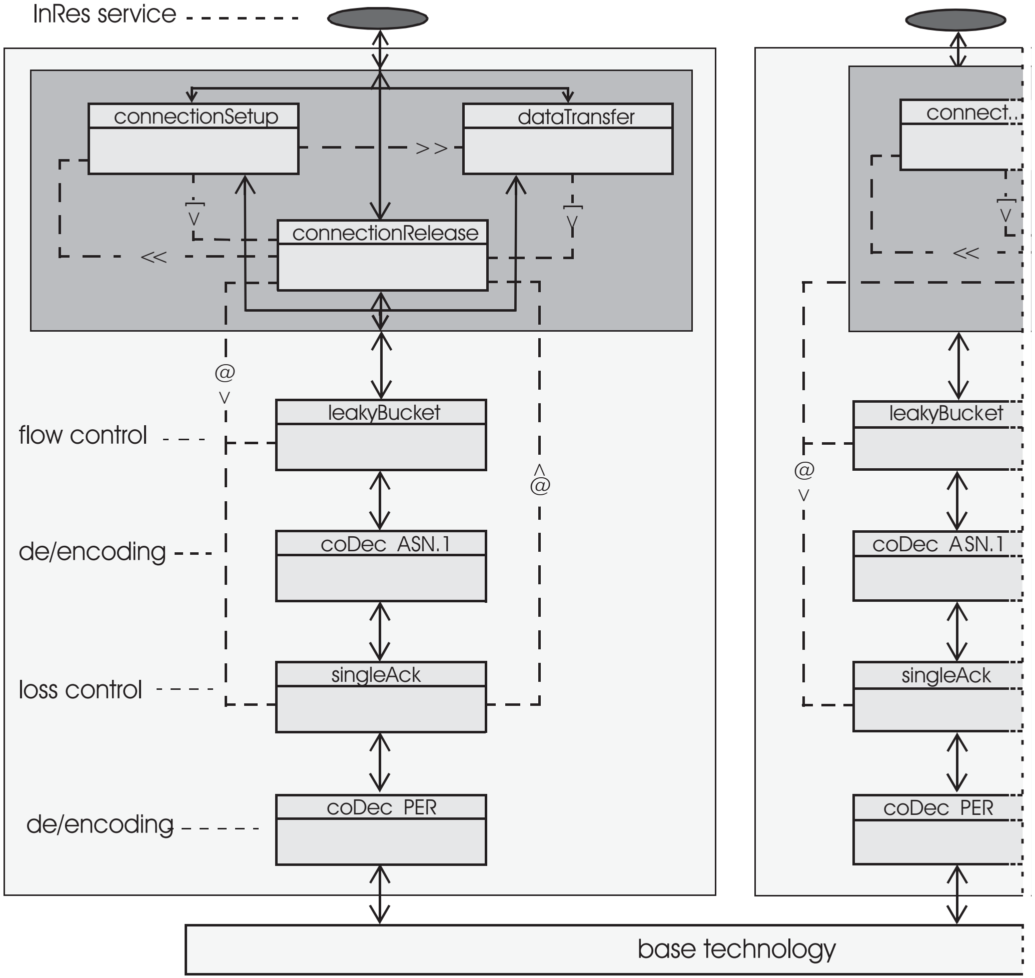
Development Process

- functionality analysis allows selection of micro protocols from the micro protocol library.
- micro protocols are composed using a conceptual design.



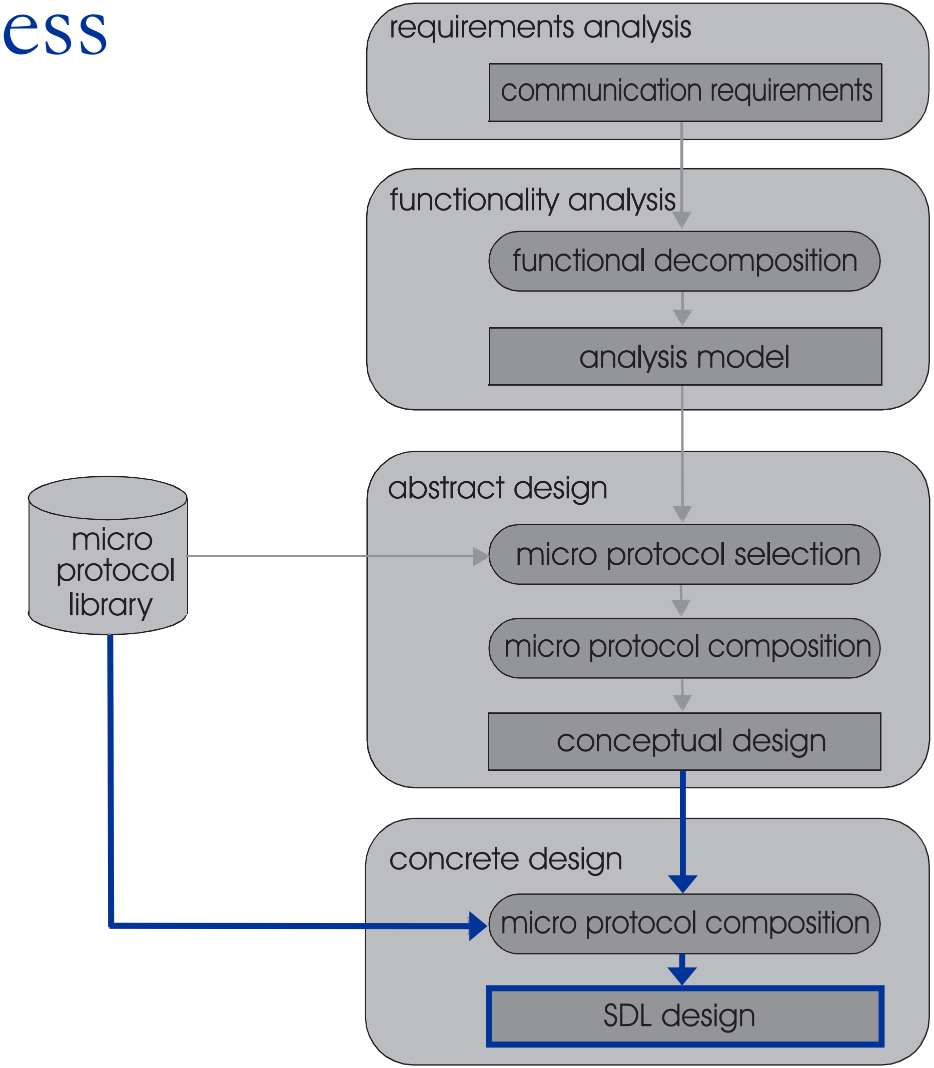
Conceptual Design

- example: InRes-communication system
- service of the base technology is extended by loss and flow control
- operators are describing the composition of micro protocols



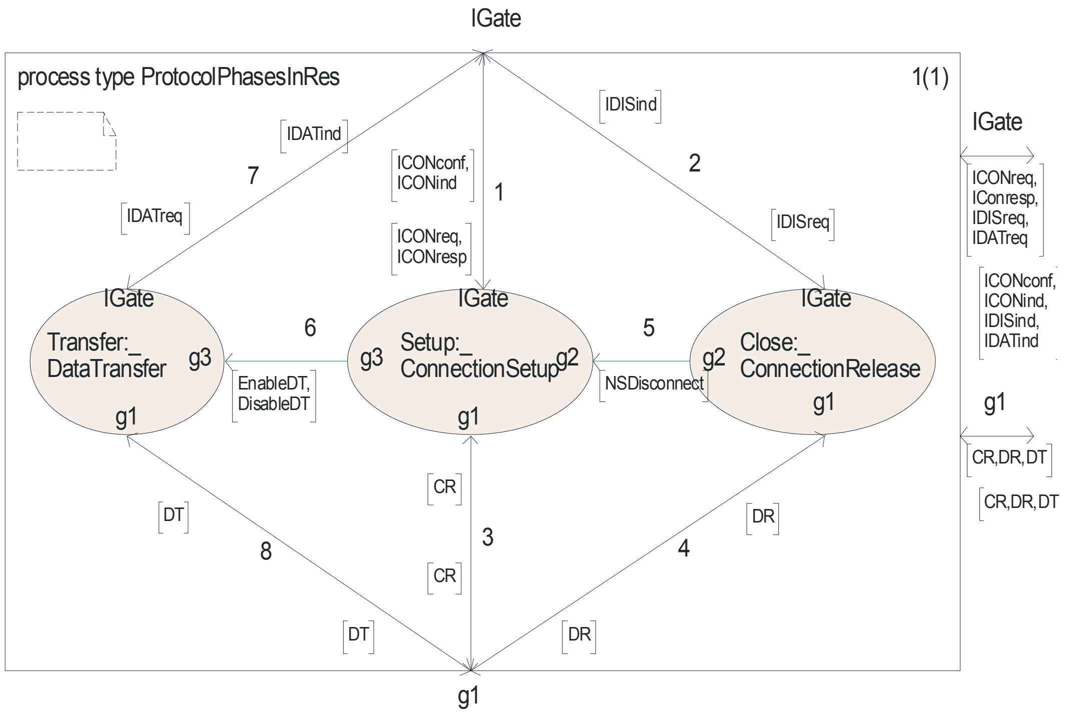
Development Process

- one SDL solution is used for the concrete design
- several possibilities to realize the operators using SDL
  - ❑ signals
  - ❑ variables
  - ❑ inheritance
  - ❑ procedures



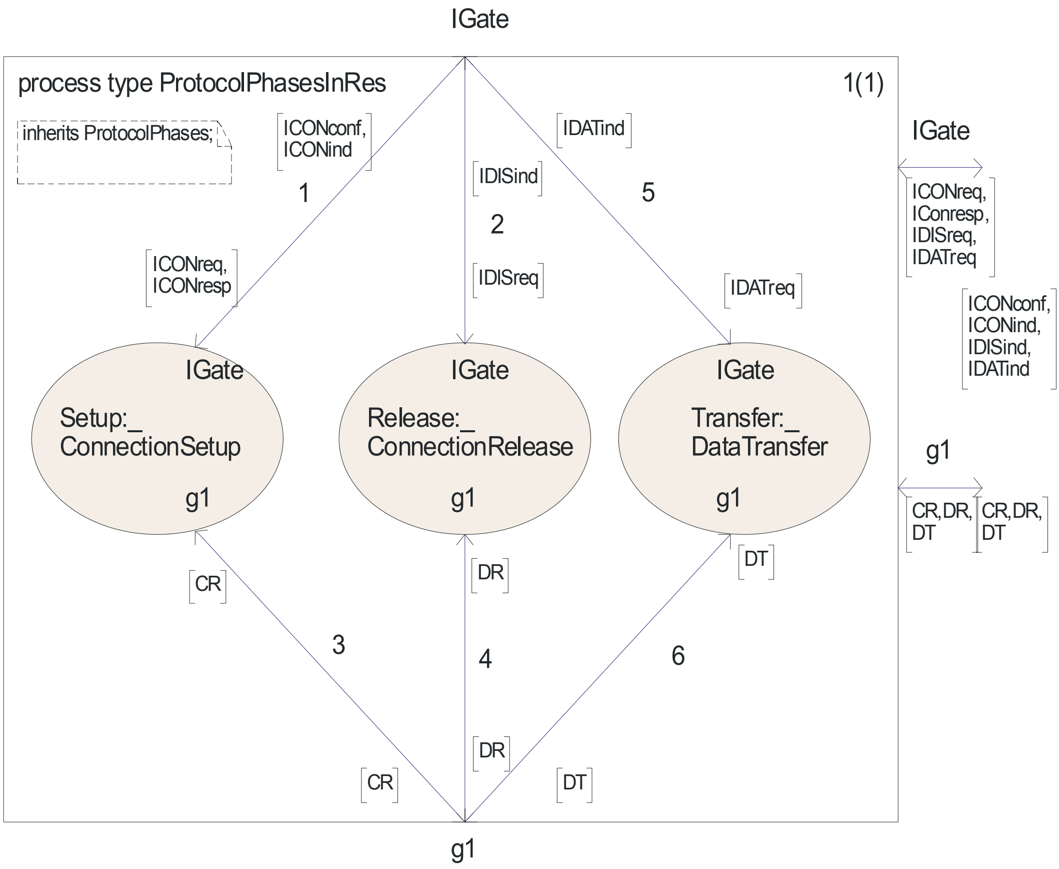
SDL-Solution – distributed signal exchange

- signals used for synchronisation
- knowledge about micro protocols to synchronize with → gates



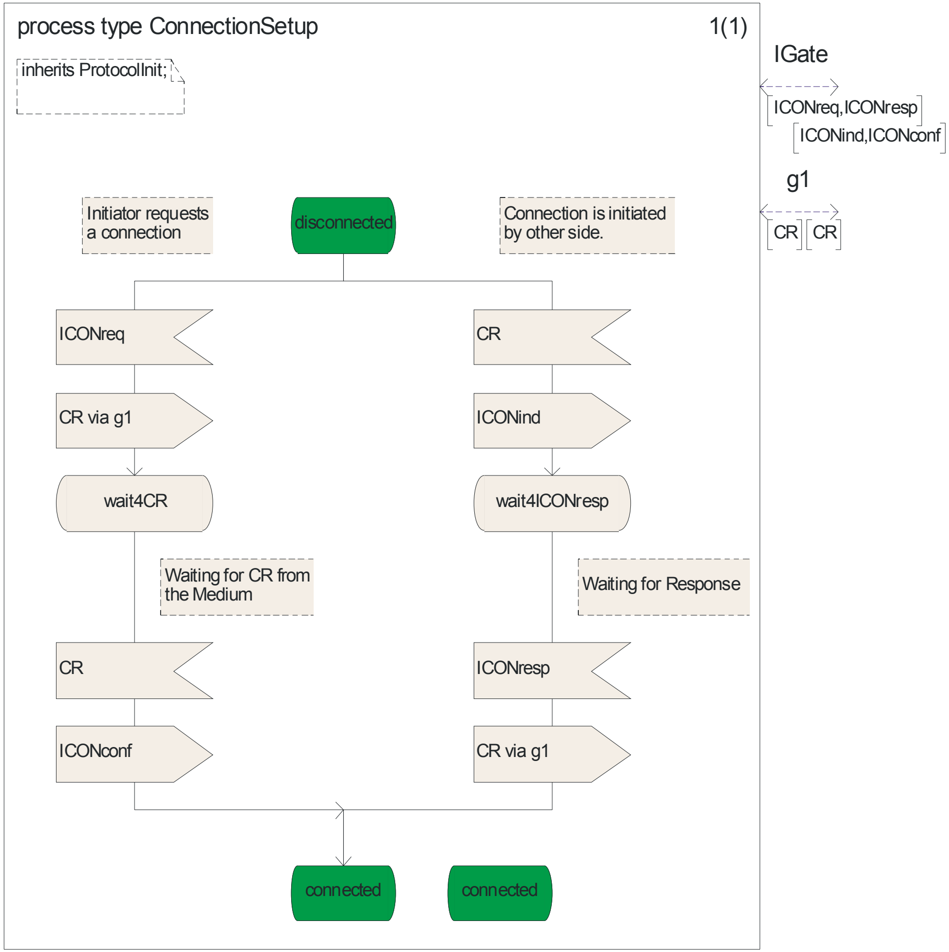
SDL-Solution – services with shared context variables

- shared variables used for synchronisation
- knowledge of the shared variables
- only possible within one process → use services



SDL-Solution – process inheritance with shared states

- states used for synchronization
- knowledge of the shared states necessary



process type ConnectionSetup

1(1)

inherits ProtocolInit;

Initiator requests a connection

disconnected

Connection is initiated by other side.

ICONreq

CR via g1

wait4CR

Waiting for CR from the Medium

CR

ICONconf

CR

ICONind

wait4ICONresp

Waiting for Response

ICONresp

CR via g1

connected

connected

IGate

ICONreq,ICONresp

ICONind,ICONconf

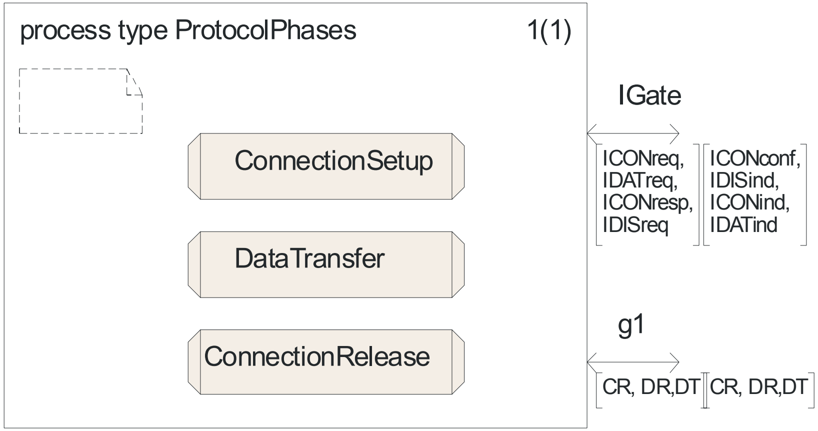
g1

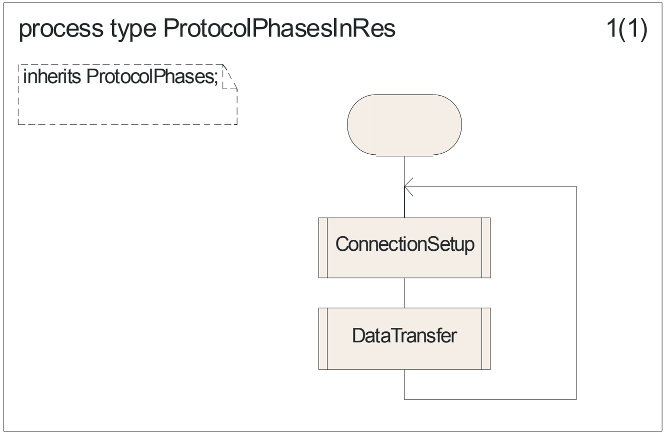
CR

CR

SDL-Solution – procedures and inheritance

- each micro protocol is encapsulated in one procedure.
- passing thread of control is realized by termination of a procedure call
- disruption (connection release) is possible by using inheritance


process type ProtocolPhases
1(1)
ConnectionSetup
DataTransfer
ConnectionRelease
IGate
ICONreq, IDATreq, ICONresp, IDISreq
ICONconf, IDISind, ICONind, IDATind
g1
CR, DR,DT
CR, DR,DT



Summary & Outlook

- several difficulties providing generic and self contained solutions → tool support
- compositional testing and validation
  - each component tested
  - resulting system tested for composition faults
- identify more micro protocols
  - routing, quality of service
- SDL 2000 offers new mechanisms for composition
  - exceptions, composite states

Thank you
for your attention!

Questions?