MODELING WITH THE TIMING DEFINITION LANGUAGE (TDL)

Wolfgang Pree
## Contents

- Modeling of timing & concurrency with TDL
  - scientific foundations
  - developer perspective
  - run-time guarantees
  - future research challenges
- Case study: Active Rear Steering system
Scientific foundations of TDL

- Giotto abstractions and concepts (UC Berkeley):
  - logical timing
  - separation of timing and functionality
  - modes and mode switches
  => software properties:
  - time & value determinism, portability
- component model (Univ. Salzburg): module as
  - unit of parallel composition
  - unit of distribution
  => modeling of
  - multi-mode & multi-rate & multi-program systems
Developer perspective (I)

TDL: timing

Simulink: functionality

TDL: VisualCreator in Simulink

Simulink editor

Simulink Simulation Environment
Developer perspective (II)

TDL: timing

Simulink: functionality

TDL: VisualCreator in Simulink

Simulink editor

TDL code

C code
Developer perspective (III)

platform mapping: platform specification & TDL module-to-ECU-assignment

TDL:VisualDistributor

TDL configuration file
run-time guarantees

- observable timing and functional behavior of all TDL modules does not change
  - no matter on which ECU the TDL modules execute
  - if TDL modules are added or removed

requires:
- sophisticated bus schedule algorithm (note that all TDL modules switch their modes independently)
- TDL:Machine as basis of portability
  - identical implementation on Simulink and other RT-OS
  - lean: for example, ca. 15 KB on OSEK
Future research challenges

- improve control engineering aspects
  - eg, minimize delays by supporting 10:1 rule and advance calculations (=> compound TDL tasks)
- additional modeling language features, such as
  - generic modules
- formal verifications of
  - bus schedule generation
  - TDL:Compiler and TDL:Machine
TDL case study: Active Rear Steering system (TDL tool demo)