

# K2 – A Process Coordination Language Based on Coloured Petri Nets

Claus Assmann, Jan Lukoschus, Werner Kluge  
Department of Computer Science  
University of Kiel / Germany  
e-mail: wk@informatik.uni-kiel.d400.de

Preparing non-trivial application problems for concurrent computing is known to be a formidable organizational task. Program design generally requires explicit specifications pertaining to the creation (and termination) of processes, to their orderly cooperation (or communication), and to measures which ensure a stable overall system behavior which manifests itself in the existence of some essential invariance properties.

We present a process coordination language K2 based on a variant of coloured Petri-nets which cleanly separates the specification of processes and their communication structures from the algorithmic specifications (programs) to be executed by the individual processes, and defines clear-cut interfaces between these two levels. Communication among processes is limited to a small set of well-defined interaction schemes which ensures well-behaving systems largely by construction. Assertions that can be made in isolation about invariance properties of small subsystems are not corrupted in larger contexts since process communication and program execution are governed by the principle of context-free substitutions (or by the absence of side-effects across process boundaries). This principle also renders it possible to perform on both levels operations in a step-wise mode and in any selected context, and to inspect intermediate states of process communication or program execution for validation purposes. Large systems may thus be systematically constructed by hierarchical composition of small subsystems whose properties have been validated or verified individually.

Systematic net abstractions play a key role in the entire specification process. They may be used to specify process systems hierarchically over several levels of refinement (or nesting), to share subnets on a client / server basis, and to replicate net structures by recursive expansion. Recursiveness may be effectively employed to dynamically partition application problems into identical subproblems whose communication structures may, within the constraints imposed by the interaction schemes, be freely defined, including, for instance, interactions among peer processes that belong to different recursion levels.