Assumption: one job (multiple tasks), starting from C, platform instance fixed

When?	Platform	What?	Remarks
DT	PI	Determine Task Granularity / Parallelism	
DT	PI	Algorithm Selection (1)	Fit for different qualities
DT	PI	Data Structures (Granularity of Communication / Computation) (1)	General / coarse-grain at application level
DT	PI	Quality Level Selection (1)	Only identification of which quality levels exist Includes determining definition of quality: (frame rate, jitter, #wavelets per object, …)
DT	PI	Scenario Detection (1)	Only identification of which scenarios exist (based on PATH analysis only)
DT	PI	Separation Control – Stream (1)	Only identification of what is code related to control and what is related to stream
DT	PI	Functional Validation	Application level
DT	PI	Deadlock Detection (1)	
DT	PD	Algorithm Selection (2)	Optimize for architecture (Might imply backloop!)
DT	PD	Data Structures (Granularity of Communication / Computation) (2)	Refine for architecture
DT	PD	Separation Control – Stream (2)	Separate as much as necessary at mapping step
DT	PD	Task Assignment	Part of mapping
DT	PD	Task ordering /arbitration (1)	To ensure predicatability
DT	PD	Memory Allocation	Assumption: we know the size of data a priori
DT	PD	Functional Verification	Taking mapping choices and architecture limitations into account
DT	PD	Connections Assignment	
DT	PD	Deadlock Detection (2)	
DT	PD	Buffer Sizing	
DT	PD	Throughput analysis	
DT	PD	Task (WC)ET analysis	Compilation based?
DT	PD	Model based analysis for soft-real time	BDF modeling, channel model, remote memories, …
RT	PD	Memory budget reservation	
RT	PD	Task ordering / arbitration (2)	
RT	PD	Processor budget reservation	
RT	PD	Connection budget reservation	
RT	PD	Quality level selection (2)	Actual selection of previously identified quality levels
RT	PD	Scenario selection (2)	Actual selection of previously identified scenarios