

A Modeling and Analysis Tool: Times

Jong-Hoon Lee

Dependable Software Laboratory

Konkuk University

Index

- What is *Times*?

- Features of *Times*
 - Main features
 - Graphical Editor
 - Simulator
 - Verifier
 - Code Generator

- Applied Simple Examples
 - Examples in Tutorial

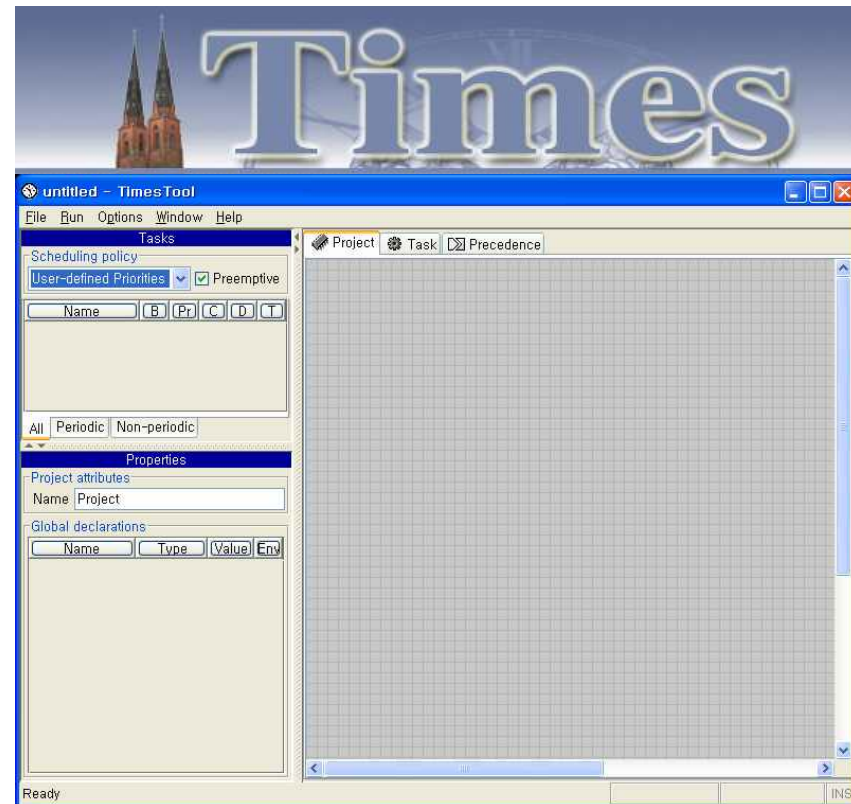
A modeling and analysis tool: Times

What is Times?

What is Times?

- A modeling and schedulability analysis tool
 - Features for real-time embedded Systems
 - Verifying the system based on model

- Developed at Uppsala Univ.
 - *UPPAAL*






A modeling and analysis tool: Times

Features of Times

Features of Times - Environment

- Runnable Platform
 - Windows
 - 32bit / 64bit?
 - Linux
 - Solaris

	Platform	Without JVM	Instructions
	Windows	Download (4.2M)	View
	Linux	Download (4.0M)	View
	Solaris	Download (4.0M)	View

Features of Times – Main Features

- Graphical Editor
 - For extended Timed Automata(task model)
 - User can specify tasks with parameter such as deadline, priority, etc.

- Simulator
 - Validate the dynamic behavior of the system
 - See how the tasks execute according to task parameter and a given scheduling policy

- Verifier for Schedulability
 - Can check all task instances meet their deadlines
 - Based on the DBM techniques and implemented based on the *UPPAAL*
 - Verification tool and Schedulability analysis tool

- Code Generator
 - Generate C code from the model
 - For brickOS
 - For platform-independent

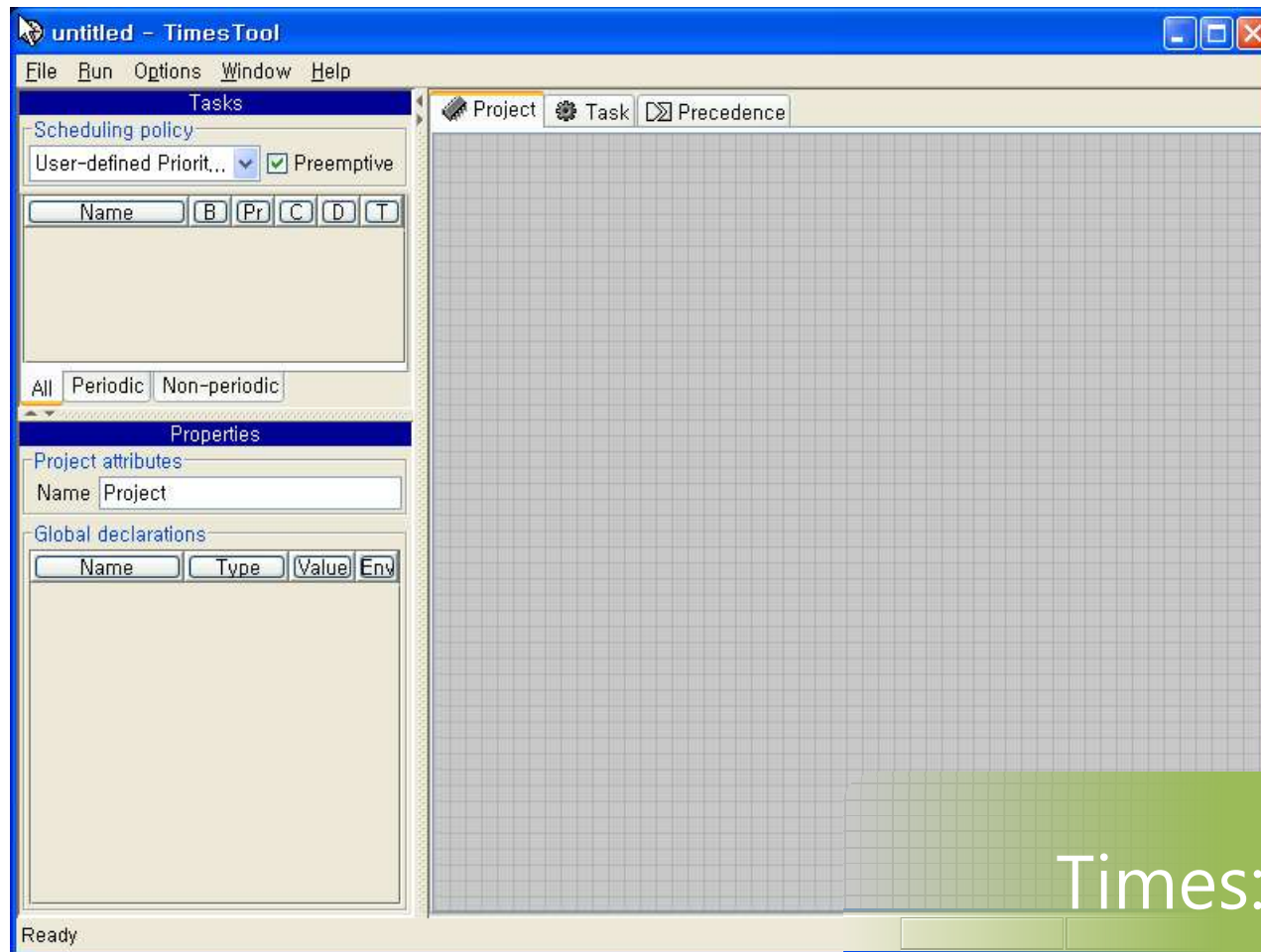
Features of Times – Main Features

- DBM Technique & *UPPAAL*
 - DBM(Difference Bound Matrices)
 - Data structures to represent clock constraints in timed automata
 - Used in *UPPAAL*
 - *UPPAAL*
 - Tool for modeling, simulation and verification of real-time embedded system
 - Simulator
 - Model-Checker
 - Commercial tool

A modeling and analysis tool: Times

Examples

Examples



Times: Editor

Examples – Editor

- Creating new project
 - Creating tasks

The screenshot shows the TimesTool interface with the following task configuration table:

Name	B	Pr	C	D	T
task1	S	0	0	inf	0
task2	P	0	0	inf	0

Annotations in the image:

- Behavior of tasks:** Points to the 'B' column in the task table.
- Attributes of tasks:** Points to the 'D' and 'T' columns in the task table.

Attributes of tasks listed in the annotation box:

- Behavior
- Priority
- Execution time
- Deadline
- Period(time interval)

Examples – Editor (Cont'd)

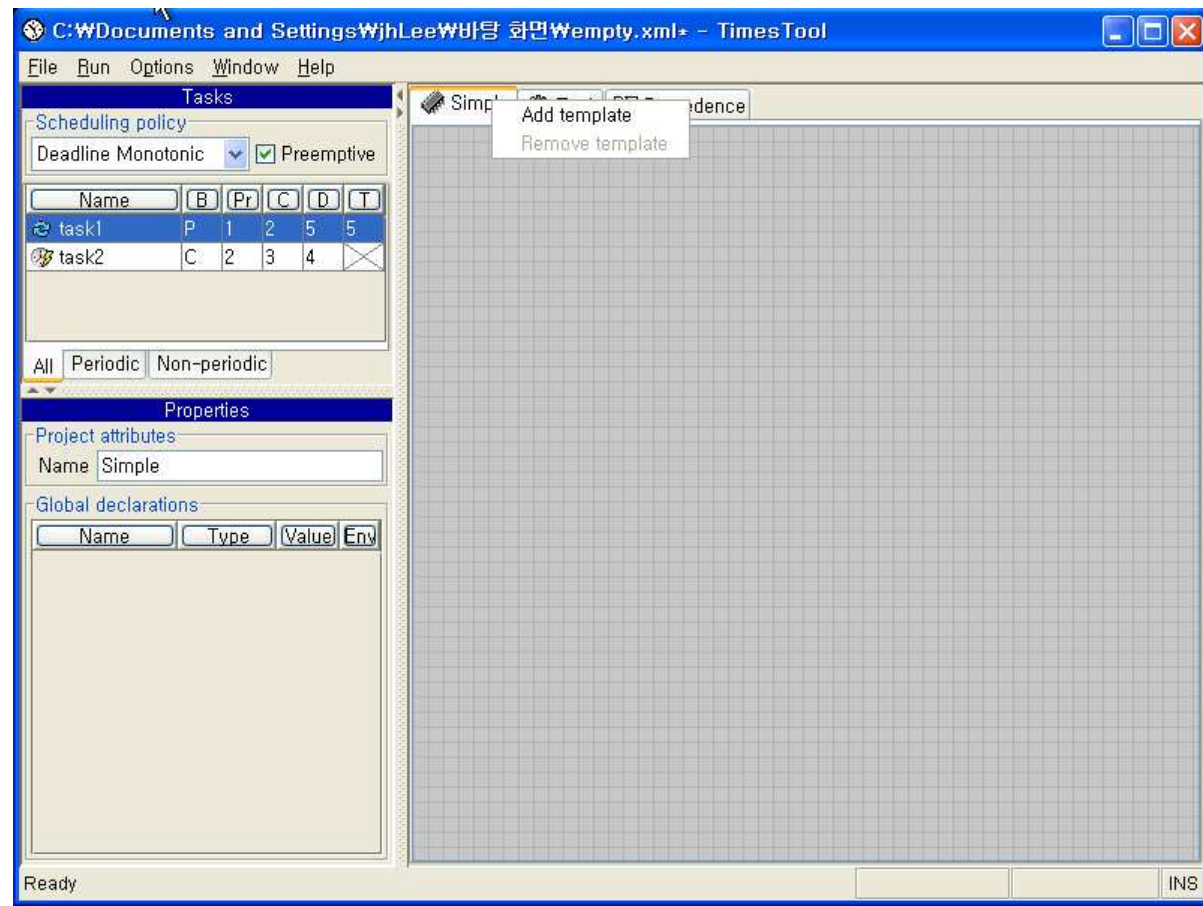
- Setting task property

The screenshot shows the TimesTool application window. The 'Tasks' panel on the left lists 'task1' and 'task2' with their respective attributes. The 'Attributes' table below it provides detailed settings for 'task1'. The 'Task code editor' on the right is currently empty.

Attribute	Value
Name	task1
Behaviour	Sporadic
Priority	1
Computing time	0
Deadline	inf
Period	0
Offset	0
Max # of tasks	1

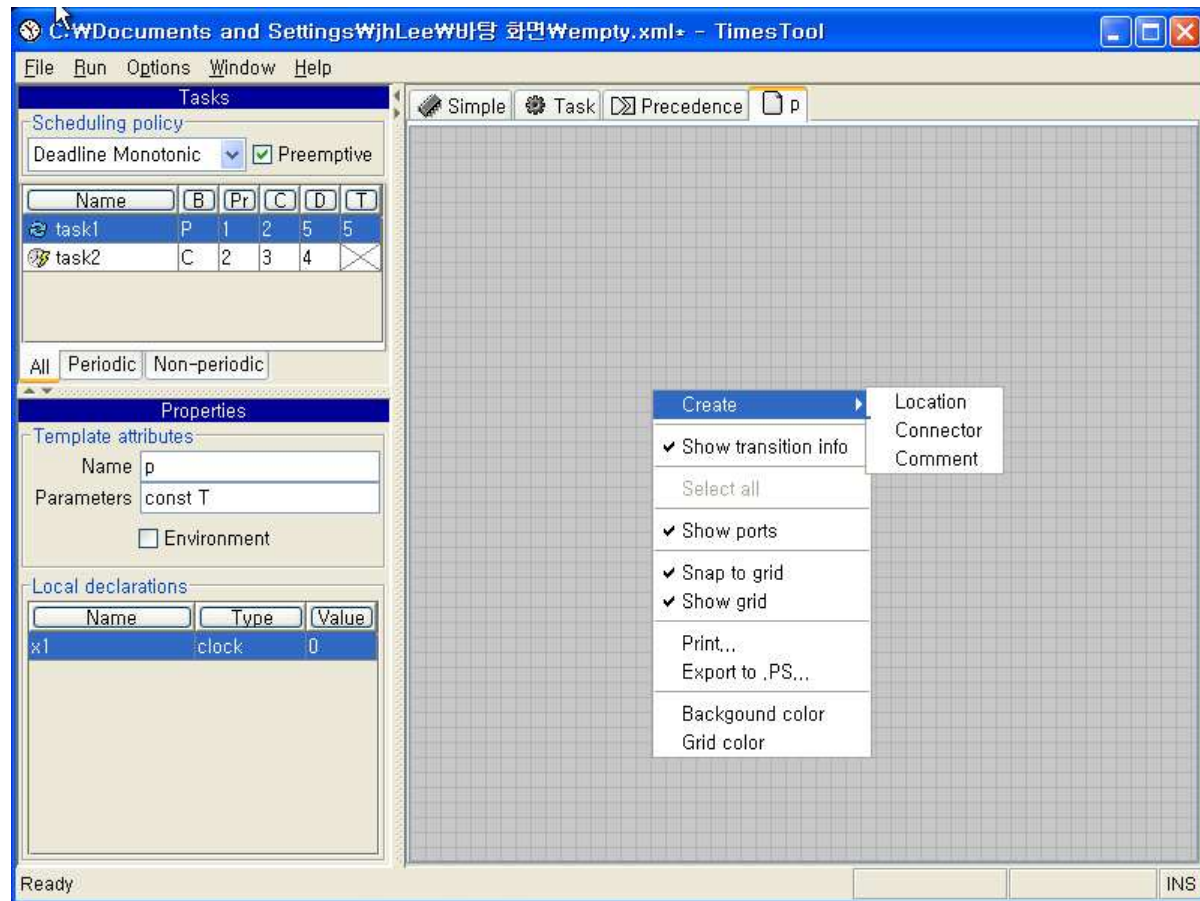
Examples – Editor (Cont'd)

- Creating template



Examples – Editor (Cont'd)

- Creating automaton



Examples – Editor (Cont'd)

- Creating automaton
 - Making locations
 - Associate the location with task

The screenshot shows the TimesTool application window. The title bar reads "C:\Documents and Settings\WjhLee\바탕 화면\Empty.xml - TimesTool". The interface is divided into several panels:

- Tasks Panel:** Shows a table of tasks and their scheduling parameters.

Name	B	Pr	C	D	T
task1	P	1	2	5	5
task2	C	2	3	4	⊗
- Properties Panel:** Shows template attributes for a selected task.

Name	p
Parameters	const T
<input type="checkbox"/> Environment	
- Local declarations Panel:** Shows a table of local variables.

Name	Type	Value
x1	clock	0
- Diagram Area:** Displays an automaton diagram on a grid. It features a starting node (black dot) with an arrow pointing to a box labeled "Location_1". Below it, another box labeled "Location_5" is shown with a dropdown menu set to "task2".

Examples – Editor (Cont'd)

- Creating automaton
 - Making transitions

- Guard: clock constraints and/or predicate on data variables
- Sync: a! and a?
- Assign: assignment

The screenshot shows the TimesTool interface with the following components:

- Tasks Table:**

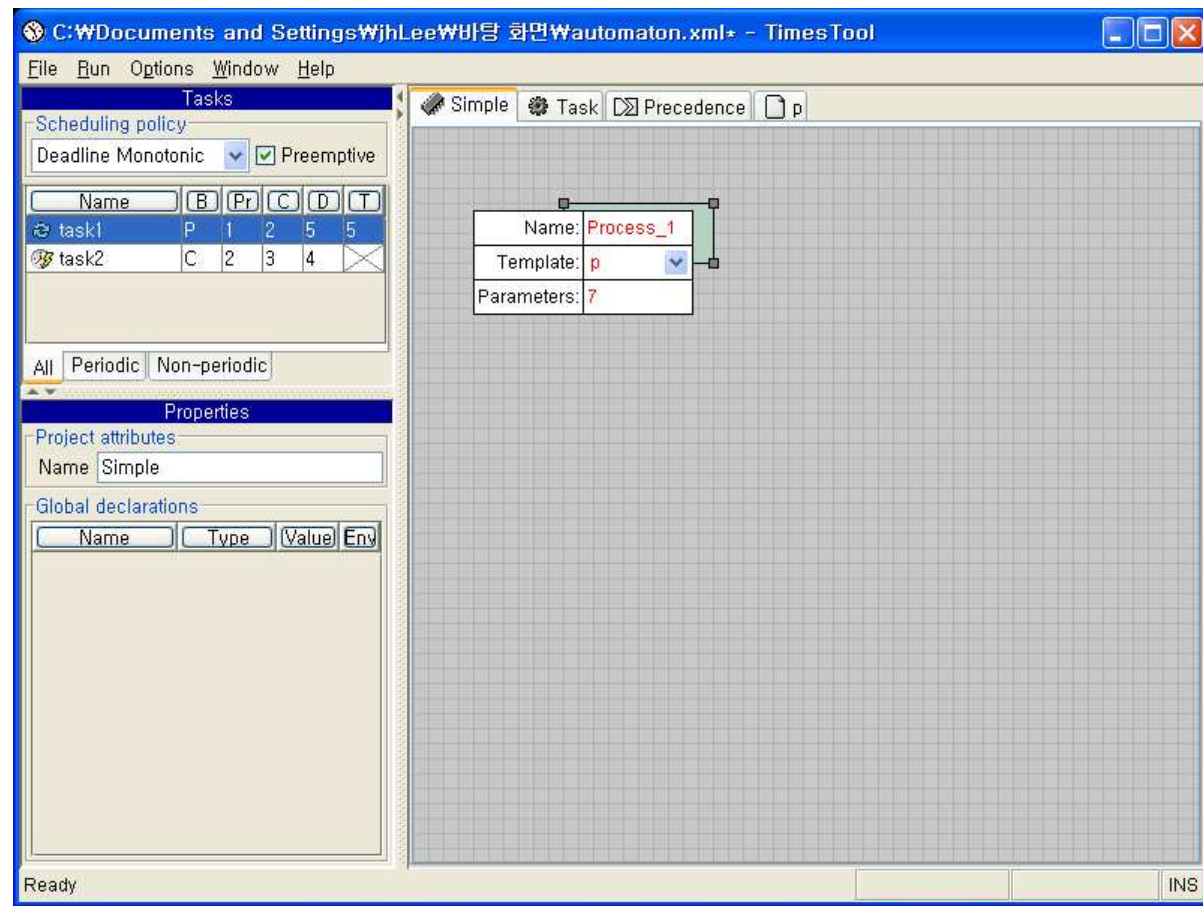
Name	B	Pr	C	D	T
task1	P	1	2	5	5
task2	C	2	3	4	⊗
- Properties:**
 - Template attributes: Name: p, Parameters: const T, Environment:
 - Local declarations:

Name	Type	Value
x1	clock	0
- Diagram:** A state transition diagram with a state named "Location_1". A red box highlights the state's properties:

Name:	
Guard:	$x1 \geq T$
Sync:	<input type="checkbox"/>
Assign:	
Probability:	

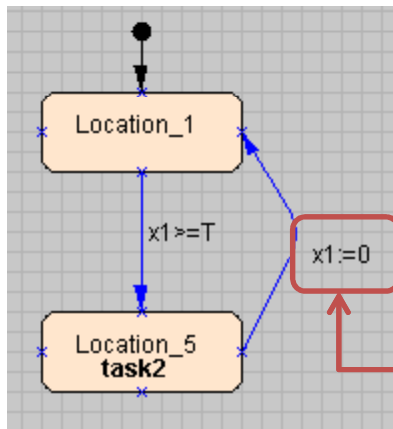
Examples – Editor (Cont'd)

- Creating process



Examples – Editor (Cont'd)

- Syntax checking



C:\Documents and Settings\WjhLee\바탕 화면\Wautomaton.xml+ - TimesTool

File Run Options Window Help

- Syntax checking
- Simulation
- Schedulability analysis
- Verification...
- Code synthesis
- Compile

Simple Task Precedence p

Process_1 p(7)

Instantiator errors (1/1)

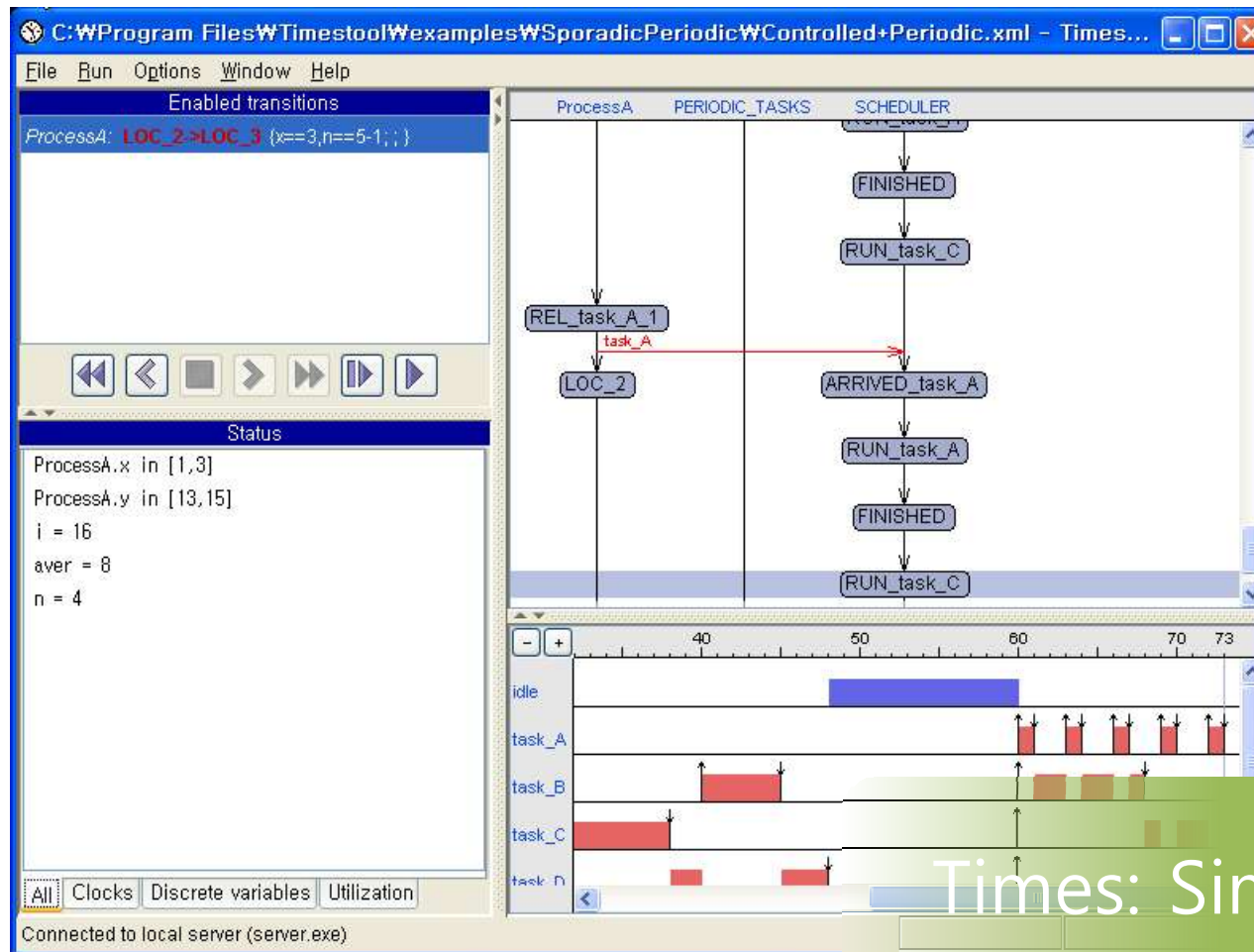
Invalid guard 'x1:=0' in template 'p'.
Encountered ':=' but was expecting one of:

- "[" ...
- "<" ...
- ">" ...
- "<=" ...
- ">=" ...
- "=" ...
- "<=" ...
- ">" ...

Close << Previous Next >> Show

Ready INS

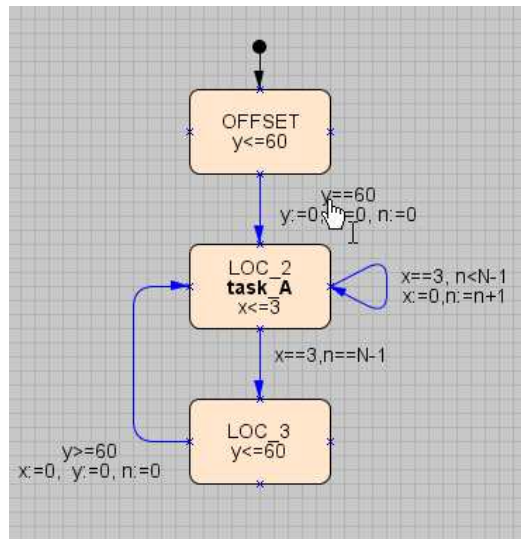
Examples



Times: Simulator

Examples – Simulator

- Example for Simulator



Tasks

Name	B	Pr	C	D	T
task_A	C	4	1	3	
task_B	P	3	5	20	20
task_C	P	2	8	28	30
task_D	P	1	5	30	30

Properties

Project attributes
Name: SporadicPeriodic

Global declarations

Name	Type	Value	Env
i	int	0	<input type="checkbox"/>
aver	int	0	<input type="checkbox"/>
n	int	0	<input type="checkbox"/>

Ready

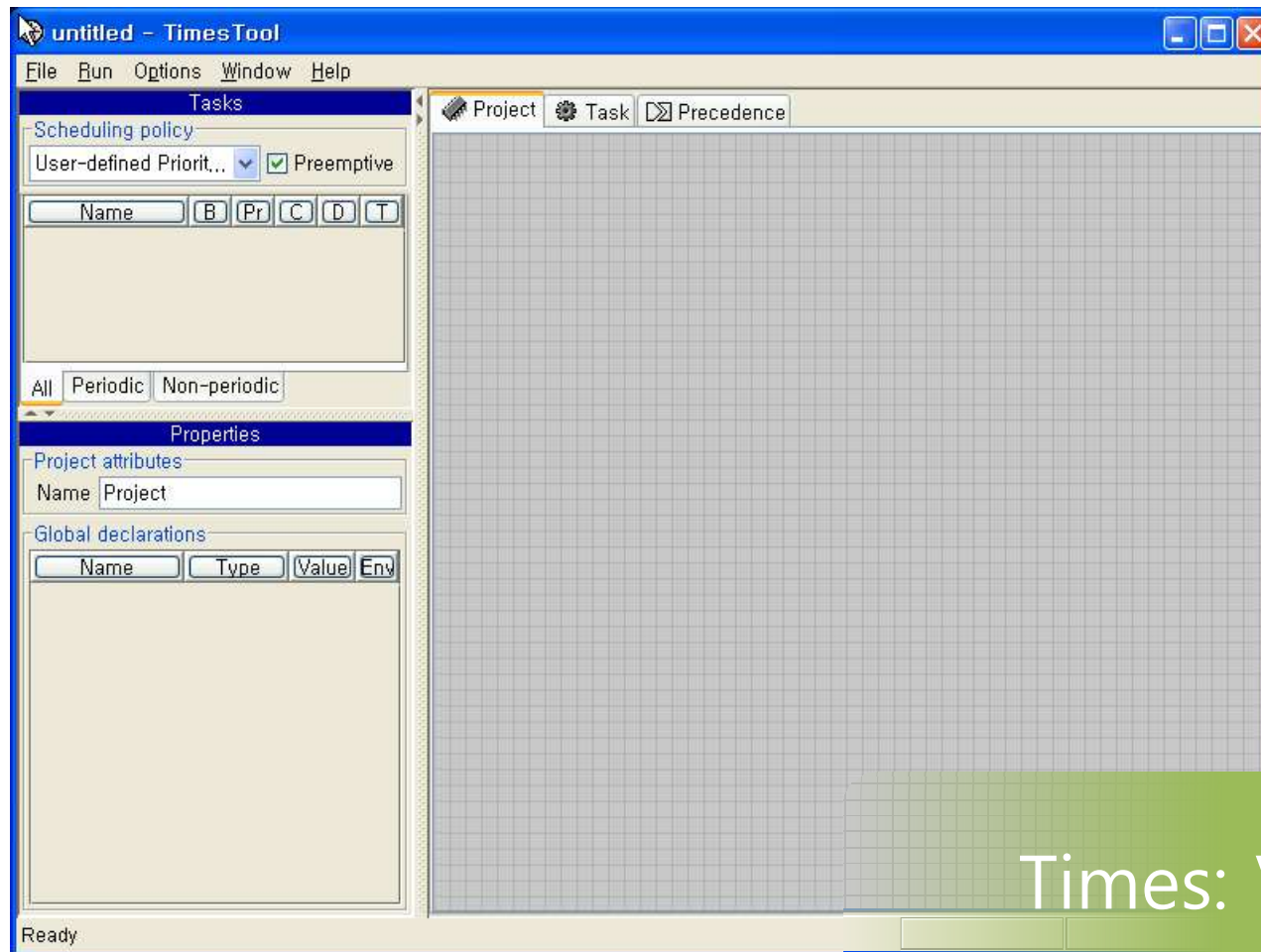
Examples – Simulator (Cont'd)

- Simulation

The screenshot shows a simulation window titled "C:\Program Files\Timestool\examples\SporadicPeriodic\Controlled+Periodic.xml - Time...". The interface includes several key components:

- Enabled transitions:** A list at the top left showing "SCHEDULER: FINISHED->RUN_task_D".
- Message Sequence Chart:** A central diagram showing the state transitions of "ProcessA" and "SCHEDULER". States include "OFFSET", "START", "IDLE", "ARRIVED_task_D", "RUN_task_D", "ARRIVED_task_B", "RUN_task_B", "ARRIVED_task_C", and "RUN_task_B". Transitions are labeled "task_D", "task_B", and "task_C".
- Simulation Control:** A set of navigation buttons (back, forward, stop, etc.) located below the message sequence chart.
- Status:** A text area displaying current simulation variables: "ProcessA.x = 13", "ProcessA.y = 13", "i = 1", "aver = 0", and "n = 0".
- Execution Gantt-Chart:** A timeline at the bottom showing the execution duration of "task_A", "task_B", "task_C", and "task_D" as red bars.

Examples

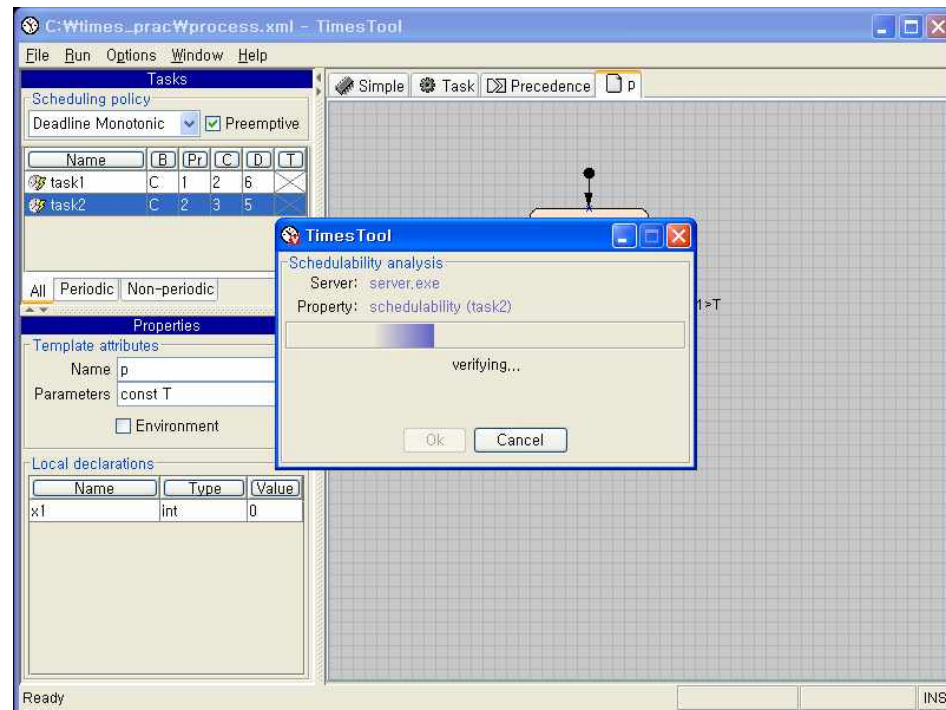


Times: Verifier

Examples – Verifier

- Verifier
 - Schedulability Analysis
 - All the tasks in the system managed to meet their deadlines in all possible execution traces
 - If the answer is satisfied, observe worst-case response time of each task.

- Verifying Model Properties
 - Checking the Model against specified properties



Examples – Verifier (Cont'd)

- Schedulability analysis

The screenshot displays the TimesTool interface for a schedulability analysis. The main window shows a 'Schedulability analysis' window with the following details:

- Server: server.exe
- Property: schedulability
- Result: **SATISFIED**
- Buttons: Ok, Show WCRT

The 'Show WCRT' button is highlighted with a red box, and a red arrow points from it to the 'WCRT Analysis' window. The 'WCRT Analysis' window displays the following table:

Name	C	WCRT	D
task_A	1	1	3
task_B	5	8	20
task_C	8	18	28
task_D	5	28	30

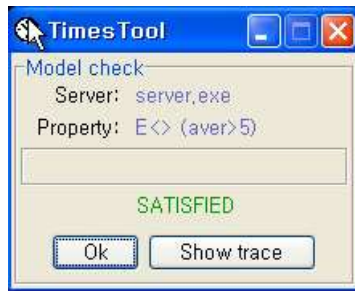
The main TimesTool window also shows a 'Tasks' table with columns Name, B, Pr, C, D, T:

Name	B	Pr	C	D	T
task_A	C	4	1	3	X
task_B	P	3	5	20	20
task_C	P	2	8	28	30
task_D	P	1	5	30	30

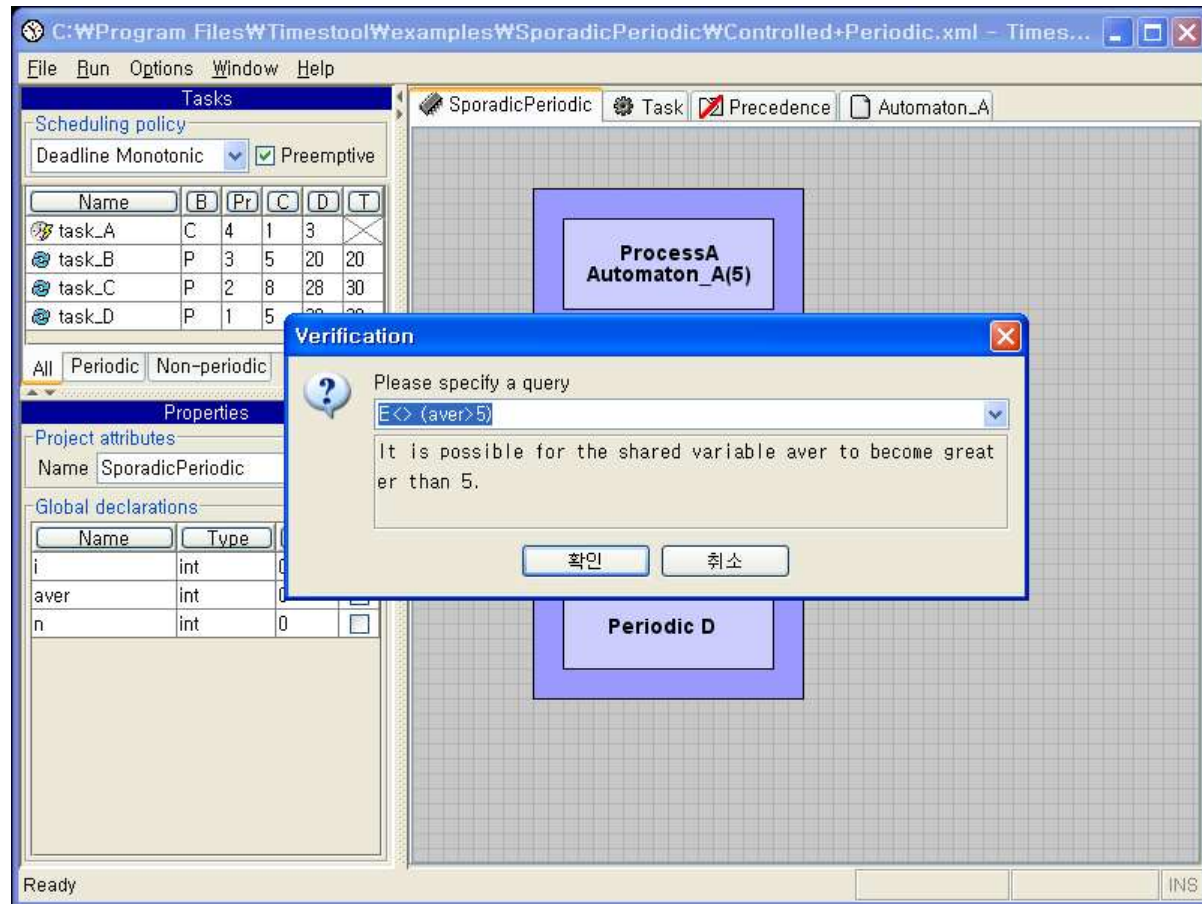
Below the tasks table, there are sections for 'Properties' (Project attributes: Name: SporadicPeriodic) and 'Global declarations' (table with columns Name, Type, Value, Env).

Examples – Verifier (Cont'd)

- Verifying Model Properties



If properties are not satisfied, show counterexample



Examples

The screenshot shows the Times Tool interface for configuring a task set. The 'Tasks' table is as follows:

Name	B	Pr	C	D	T
task_A	C	4	1	3	×
task_B	P	3	5	20	20
task_C	P	2	8	28	30
task_D	P	1	5	30	30

The 'Code writer' dialog box displays the following message:

Code generated successfully in:
 Controlled+Periodic.c
 Controlled+Periodic.h
 new Makefile created

The dialog also shows a '확인' (OK) button. In the background, a state transition diagram is visible with nodes like 'OFFSET y<=60' and 'LOC_3 y<=60', and transitions labeled with conditions like 'x==60', 'x:=3, n<N-1', and 'x:=0, n:=n+1'.

Times: Code Generator

Thank You!

A modeling and analysis tool: Times

Installation

