Design and Analysis of Distributed, Interacting Systems (DIS)

Tutorial 08
Raphael Pham
Raphael.Pham@inf.uni-hannover.de
Run: a, b, c, d, ... we are now in c₃ and we get:

..., a, ...

1. **iterative**: c₄
2. **invariant**: c₄ (go on iteratively), c₁ (start new machine, land directly in c₁)

..., a

1. **iterative**: cold viol. in c₄
2. **invariant**: hot viol. in c₁
Assignmt 7, Ex 2b,c

• Give an automaton that accepts the same language as the MSD and uses
  – iterative interpretation.
  – invariant interpretation.
One of your solutions

iterative?

invariant?
One of your solutions

iterative? invariant?
Consider the coffee machine example of the lecture.

- The user can start (s), shutDown (sD) the machine, press the Button (pB), reset the machine (rM). The machine has an internal error signal (eS). Additional signals are given in the MSD.
- State $\Sigma, \Sigma_{env}, \Sigma_{sys}$. 

Raphael Pham, DIS-Tutorial 08
The coffee machine example of the lecture is enhanced by:

- The brewer unit now has to confirm that the water is correctly boiled – otherwise no coffee can be made…

**Enhanced MSD:**

Raphael Pham, DIS-Tutorial 08

- Give the enhanced MSD.
Enhanced coffee machine

- MSD PrepareCoffee
  - u: User
  - c: Controller
  - bu: BrewerUnit
  - pressButton (c/m)
  - boilWater (h/e)
  - waterReady (h/m)
  - prepareCoffee (h/e)
  - pourCoffee (h/e)
• Model an env. controller, that allows all sequences of events.

\[
\Sigma = \{pB, bW, prC, poC, s, SD, rM, eS\}
\]
Synchrony Assumption

• Be $C_e$ and $C_s$ the controllers for the environment resp. system objects, the we assume
  – if an environment event occurs, the system can take finitely many steps before it listens for the next environment event, i.e.
  – the environment must never block system events
  – but the system must always eventually listen for environment events

• Model $C_s$ so that its composition with $C_e$ is in accordance to MSD.
Synchrony Assumption

- active, inactive states:
  - active states: one or more system events going out (finite machine steps!)
  - inactive states: env. events are going out
- inactive states are eventually visited
  - synchrony assumption, no blocking of env

env. controller:

\[ \Sigma = \{ pB, bW, prC, poC, s, SD, rM, eS \} \]