Nòmos 3: Designing Sociotechnical Systems (Group B)

Healthcare Scenario

Whenever Alice takes ill, Bob her parent takes her to a hospital. Bob also works under Steve, his employer, who expects him to work during work hours. One day, Alice fell sick during Bob's work hours. Bob disregards his employer's expectation in light of the medical emergency involving his ward.

Background

Finite State Machine

To model the behavior of a system, we use *state machines*. State machines indicate different states of a system at different times. The *state* of a system is considered as behavior of a system at any given time. A system can change from one state to another by triggering an action or an event. We call this action or event as a *transition*. Examples of state machines include vending machines, elevators, traffic lights, combination locks, and so on.

A finite state machine has the following properties

- The system must have finite number of states (S)
- The system has a particular initial state (s_0)
- The system must have finite number of inputs or events that can trigger transitions between states ($\sum$)
- The behavior of a system at the given time depends on the current state and the input or event that occur at that time
- The system has a set of final states (F)
- The state transition function is δ: S $\times$ $\sum$ $\rightarrow$ S

Consider an example of a turnstile that has two states: S = $\langle$Locked, Unlocked$\rangle$. There are two inputs: $\sum$ = $\langle$coin, push$\rangle$. Coin indicates putting a coin in the slot. Push indicates pushing the arm. Consider the initial state as the locked state. In the locked state, the arm of the turnstile cannot be moved irrespective of how many times the arm is pushed (δ: Locked $\times$ push $\rightarrow$ Locked). However, when a coin is put, the state of machine moves

from the locked state to the unlocked state (δ: Locked $\times$ coin $\rightarrow$ Unlocked). Putting an additional coin doesn't change the state (δ: Unlocked $\times$ coin $\rightarrow$ Unlocked). Now, when the arm is pushed, the state again moves from the unlocked state to the locked state (δ: Unlocked $\times$ push $\rightarrow$ Locked). Figure 1 represents the state diagram for a turnstile with its states, inputs, and state transitions.

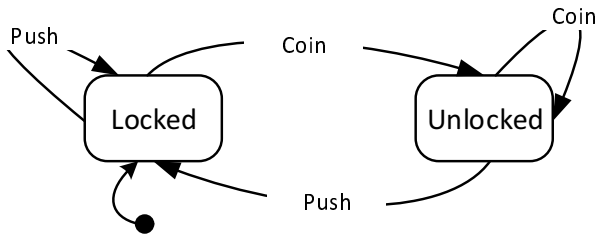

Figure 1: State diagram for a turnstile.

Nòmos 3

Nòmos 3 is a normative modeling language to represent legal provisions such as laws and regulations. In addition, it provides guidance to users to reason about the compliance of different requirements. The requirements are captured as *Goals*.

Nòmos contains the following primitives

- A *Situation* represents a proposition describing the state of affairs
- A *Goal* represents a particular Situation desired by a role in the domain
- A *Domain assumption* represents a specific Situation that hold in the domain
- A *Norm* is a right or a duty that a role has
- A *Role* is someone who desires to achieve a Goal (Social Role) or someone who is addressed by a Norm (Legal Role)

The relationships between the primitives are

- Situations make a Norm applicable or not applicable (active or block)
- Situations make a Norm or a Goal satisfied or not satisfied (satisfy or break)
- Goals bring-about Situations (brings-about)
- Situations can be brought about by a specific role
- Social roles want Goals (wants) and play Legal Roles (play)
- Legal Roles are in charge of satisfying norms (holds)
- Norms make other norms applicable, not applicable or complied (endorse, derogate, imply)

Nòmos 3: Approach Outline

Defining Normative Model

- Identify subscenarios in the scenario
- Identify all actors involved in a subscenario
- Abstract actors as legal and social roles in a subscenario
- Identify goals of each social roles
- Identify norms associated to legal roles
- Identify situations that activates or blocks a norm
- Identify situations that satisfy or break a norm

Designing State Machines

- Identify possible states, inputs, and transitions from the situations, goals and norms related to each role
- From identified states, inputs and transitions create state machines for each subscenario
- Create a final state machine by combining all state machines of individual subscenarios

Solution

Solution: Normative Model

The subscenarios in the healthcare scenario are

- Bob takes Alice to a hospital, when Alice is sick
- Bob works for Steve during business hours
- Bob disregards Steve's expectation by taking Alice to a hospital during business hours

The actors involved in the scenario are Bob, Alice, and Steve. In the first scenario, we can abstract Bob as GUARDIAN and Alice as CHILD. In the second scenario, we can abstract Bob as EMPLOYER and Steve as EMPLOYEE. In the third scenario, the roles are same as the first scenario.

In the first scenario, CHILD has a Goal G_1 to receive the treatment from GUARDIAN whereas in the second scenario EMPLOYER has a Goal G_2 to receive the completed work from EMPLOYEE. The third scenario discusses about a specific Situation instead of Goals. We consider both CHILD and EMPLOYER as social roles.

In the first scenario, GUARDIAN holds a Norm (duty) N_1 to take its CHILD to hospital. In the second scenario, EMPLOYEE holds a Norm (duty) N_2 to work for its EMPLOYER. Again the third scenario doesn't contain any Norm since it discusses about a specific Situation.

The first scenario has two Situations: *sick* and *bringToHospital*. The second scenario has two Situations: *workhours* and *work*. The third scenario has two Situations: *workhours* $\wedge$ sick, *bringToHospital*.

We can associate Situations to Norms and Goals. In the first scenario, the Situation *sick* activates GUARDIAN's Norm N_1 to take its CHILD to hospital. When GUARDIAN takes its CHILD to hospital, N_1 is satisfied. In the second scenario, the Situation *workhour* activates EMPLOYEE's Norm N_2 to work for its EMPLOYER. When EMPLOYEE work for its EMPLOYER, N_2 is satisfied.

Figure 2 represents the normative model constructed for the scenario using the primitives and relationships given by Nòmos 3.

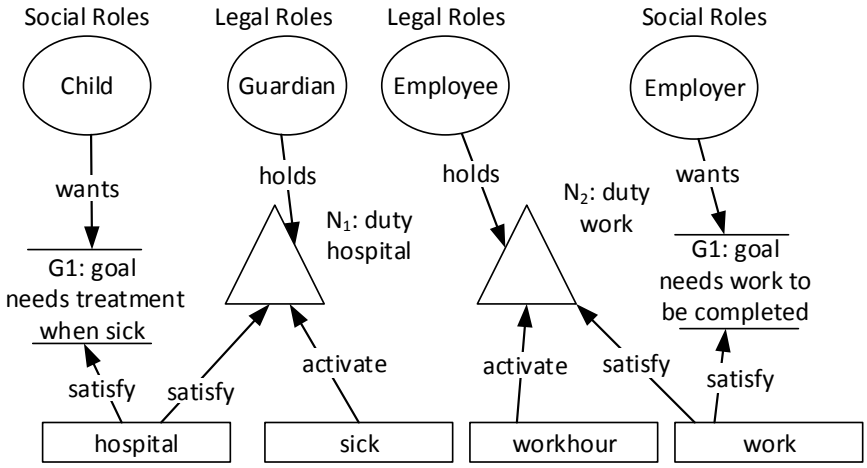

Figure 2: The normative model for the scenario constructed using Nòmos 3.

Solution: State Machines

For each subscenario, we can come up with the following state machines

- The state machine from the first subscenario has the following: $S = \langle S_1, S_2, S_3 \rangle$, $s_0 = S_1$, $\sum = \langle$sick, hospital$\rangle$, δ: $S_1 \times$ sick $\rightarrow S_2$, $S_2 \times$ hospital $\rightarrow S_3$. S_1 indicates N_1 is inactive, S_2 indicates that N_1 is active, and S_3 indicates N_1 is satisfied.

- The state machine from the second subscenario has the following: $S = \langle S_1, S_4, S_5 \rangle$, $s_0 = S_1$, $\sum = \langle$workhour, work$\rangle$, δ: $S_1 \times$ business hours $\rightarrow S_4$, $S_4 \times$ work $\rightarrow S_5$. S_1 indicates N_2 is inactive, S_2 indicates that N_2 is active, and S_3 indicates N_2 is satisfied.

- The state machine from the third subscenario has the following: $S = \langle S_1, S_6, S_7 \rangle$, $s_0 = S_1$, $\sum = \langle$business hours $\wedge$ sick, hospital$\rangle$, δ: $S_1 \times$ sick $\wedge$ business hours $\rightarrow S_6$, S_6

$\times$ hospital $\rightarrow$ S_7. S_1 indicates N_1 and N_2 are inactive, S_6 indicates N_1 and N_2 are inactive, S_7 are N_1 is satisfied and N_2 is broken.

Figure 3 represents the overall state machine for the scenario.

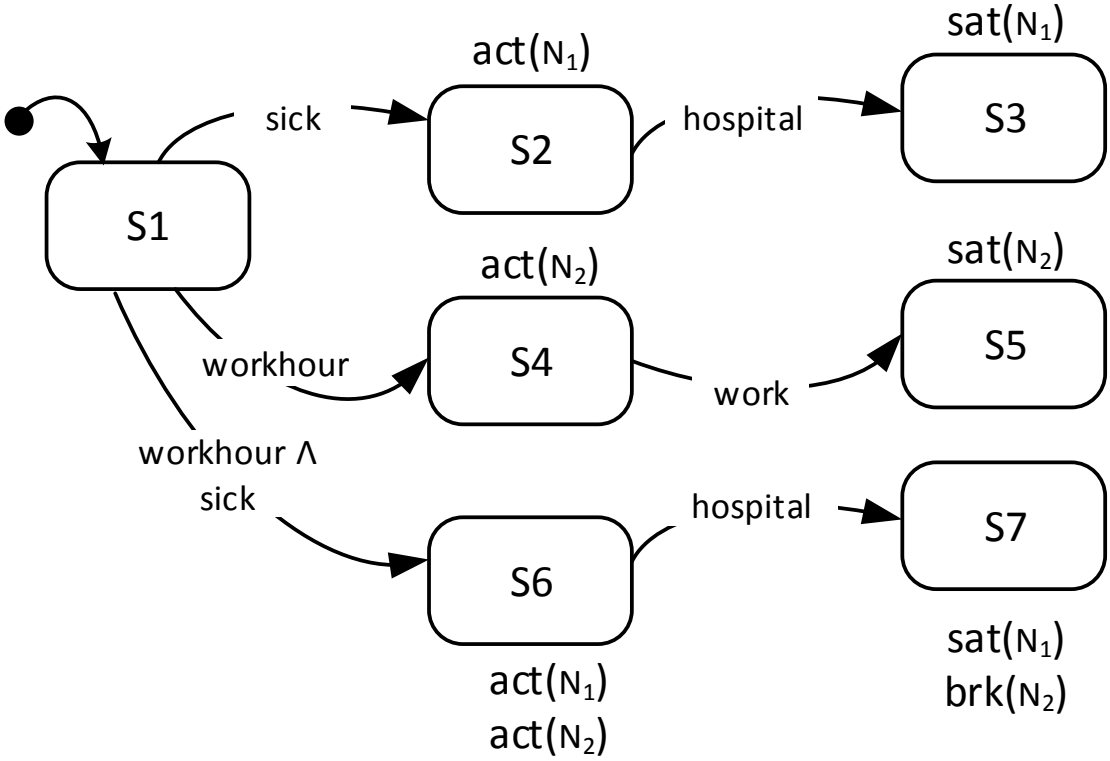

Figure 3: State machine for the scenario.

References

[1] Silvia Ingolfo, Alberto Siena, John Mylopoulos. Nòmos 3: Legal Compliance of Roles and Requirements *Proceedings of the 33rd International Conference on Conceptual Modeling (ER)*, 275–288, October 2014.