# Modeling Reactive Systems Using Cooperating Adaptive Devices

José Maria Novaes dos Santos and João José Neto
Escola Politécnica da Universidade de São Paulo (USP)
Av. Prof. Luciano Gualberto, travessa 3, n$^{o.}$ 380.
CEP 05508-010 – São Paulo – SP - Brasil
zemaria.novaes@gmail.com, jjneto@gmail.com

## ABSTRACT

Some complex problems can be modeled using more than one type of device thus having some interaction between them to represent their behavior. From this perspective, we do not have a common formulation to represent both the formalism and its interaction. The purpose of this paper is to fill in this gap by proposing a formulation that represents, for a group of devices, their behavior and interactions. We call this formalism as "Cooperating Adaptive Devices". For illustration purposes, we presented an application where its behavior can be modeled using this formalism.

## KEYWORDS

Reactive systems modeling, adaptive devices, rule-driven formalisms, self-modifying machines, adaptive automata, cooperating adaptive devices, behavior modeling, Decision Table.

## 1 INTRODUCTION

Reactive systems at a high level of abstraction can be considered as black boxes that take inputs and in response provide appropriate outputs [1]. The reactive systems are present in several fields, like computer science [1], [2], [3], biology [4], [5], [6], [7], [8] and social science [9], [10], [11].

Some formalisms are characterized by having a set of rules that define its behavior, like the classical Petri Nets, Finite State Machine and Statechart [12]. They can be used in modeling reactive systems behavior, where the environment stimuli change their configuration. They have the initial configuration and change such configuration in response to the environment changes according to its rules set. They have been used to model reactive system behavior in Artificial Intelligence and Natural Language Process.

Neto introduced the adaptive formalism in [13] where the main property is to change its rule set dynamically in response to environment behavior changes. A self-modifying device approach applied in games can be found in [14]

Some modeling issues require simultaneous usage of different types of devices having some communication between them and, to the best we know, we do not have any formalism to represent both the formalism and its communication. The purpose of this paper is to propose a single formulation to fill in this gap thus expanding the fields of the adaptive concept. This formulation is called Cooperating Adaptive Devices (CAD).

In section 2, we briefly describe the static rule-driven devices and introduce the concept of adaptivity. In section 3 we present our general formulation to Cooperating Adaptive Devices. In section 4, we exhibit an application to illustrate our proposal. Section 5 presents some related works holding the adaptivity concept and in section 6, we finish this paper by expressing the conclusion and some comments.

## 2 ADAPTIVE RULE-DRIVEN DEVICES

Some formalisms are characterized by a configuration that can change in response to an environment event according to their rule sets. A configuration of the device consists of all elements that hold information of the current status of the device.

A rule-driven device is described for a rule set that specifies its behavior in response to environment changes. Each rule determines a configuration change in response to conditions or stimuli occurred in the environment. The device starts in the initial configuration and then follows the rules changes configuration in response to the events that occurred.

If there is only one next configuration after all input stimuli, we say the device is deterministic, otherwise we say the device is non-deterministic. In other words, a non-deterministic device has a set of rules that maps at least one possible configuration into two or more next configurations. The deterministic devices are usually more effective than the non-deterministic equivalent device.

In the standard formulation, we define ND = (C, NR, S, $c_o$, A, NA) where ND is a rule-driven device, which operation is given by a set of rules NR, C is the set of all possible configurations, $c_o$ ($c_o \in C$) is the initial configuration and "S" is the finite set of all possible events that correspond all valid input stimuli for ND, with the null event belonging to S. The subset A coincides with all accepting configurations of the device, $A \subseteq C$ and the set NA is composed of all possible symbols outputted by ND as side-effects of the application of the rules in NR. NR is defined by a relation NR $\subseteq$ C x S x C x NA. Rules from NR have the shape r = ($c_i$, s, $c_j$, z), meaning that in reaction to any input event s ($s \in S$) the device changes the current configuration $c_i$ to $c_j$, consuming "s" and producing "z" as output, $z \in NA$, as side effects.

A formal rule-driven device is said to be adaptive if its behavior may change dynamically. The concept of adaptivity applies to any device that it is able to change its own behavior. In particular, when the behavior is determined by a set of rules, adaptivity is easily achieved by changing the set of rules that define the device's behavior.

The adaptive formalism has been used for modeling environment changes because its formulation is appropriate for these problems by having a clear expression.

The general formulation for rule-driven adaptive devices can be thought as adaptive layer placed around the original subjacent non-adaptive device (standard rule-driven device).

Conceptually we can identify two major components in the adaptive device: an equivalent underlying device, typically similar to those devices described in the beginning of this section and an adaptive mechanism responsible for the adaptivity, which has the feature of modifying the rules set.

One notation elaborated for representing adaptive rule-driven devices in a way as similar as possible to its original non-adaptive underlying formulation was presented in detail in [13].

In brief, each rule has two new components, which they have the shape r = (ba, $c_i$, s, $c_j$, z, aa). The first new component, defined as "ba", implies in executing a function associated to the rule before the beginning of rule execution. The second component is the "aa", performed after the rule execution. If a rule has both "ba" and "aa" as null set, it is performed in the same way as a rule is performed in the underlying device definition.

The elaboration of adaptive device formulation encouraged its expanding notation to represent the communications between devices, as presented in details in the next section.

## 3 COOPERATING ADAPTIVE DEVICES - FORMULATION

We have briefly mentioned the adaptive rule-driven device´s main idea in the previous section. In this section, we present the formulation for Cooperating Adaptive Devices, enhancing the power of its usage in modeling real problems, especially in reactive system behavior.

### 3.1 Introduction

The Cooperating Adaptive Devices are a finite group consisting of rule-driven devices, of heterogeneous types having common communication mechanisms. We can have communication between any pair of devices that belong to the group being managed by a mechanism in order to ensure only one concurrent communication. We call this mechanism as MCM mechanism. The main consequence of the communication between the devices is that the behavior of the devices can be changed by themselves.

The concept of Cooperating Adaptive Devices can be briefly defined as the property of any member in the group being able to modify the behavior of another device in the group. Concisely, any member in the group can send a message that results in rule set change of another device thus modifying its behavior.

The communications between devices is comprised of a group of standard protocol messages, named CP protocol and is detailed in section 3.4.

Any action for changing rules belonging to any other device is started by performing a communication from a source device to some target device by using a common adaptive mechanism as an intermediary agent of such interactions.

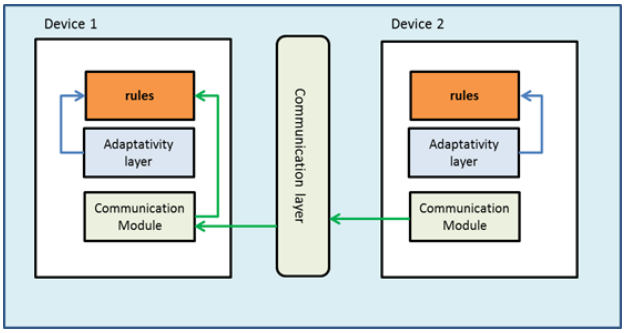


**Figure 1 -** Illustration of cooperating adaptive devices.

Figure 1 illustrates the Cooperating Adaptive Devices and its communication. Device 2 starts the communication sending a message to device 1 through the communication layer. The interaction between devices is represented in the figure 1 by the green arrow.

### 3.2 Definition

We will reference an adaptive device as AD. The formalism Cooperating Adaptive Devices (CAD) represents a group of adaptive device and the communication between them.

The Cooperating Adaptive Devices are formed by "m" adaptive devices (m > 1, where "m" is an integer number), a common communication mechanism (CCM), and a mechanism responsible for the coordination and management of the messages communication (MCM).

$$CAD = ( \{AD^r\}, CCM, MCM ) \quad (1)$$
$$(\text{for } r = 1, .., m)$$

Thus, the Cooperating Adaptive Devices consist of:

- $\{AD^1, AD^2, \ldots, AD^m\}$ – m adaptive devices,

• CCM - communication mechanism between the devices and
• MCM – mechanism to manage the messages.

So, we can express:

$$CAD=(\{AD^1,AD^2,AD^3,..,AD^m\}, CCM, MCM) \quad (2)$$

The group of adaptive devices $AD^r$ (for r = 1, .., m) can be denoted GADM, thus:

$$CAD = (GADM, CCM, MCM) \quad (3)$$
$$GADM = \{AD^1, AD^2, ... , AD^m\} \quad (4)$$

### 3.3 Cooperating Adaptive Devices and the Common Communication Mechanism

Similar to the behavior of adaptive devices, a built-in counter T2 is defined for Cooperating Adaptive Devices, with initial value 0 and automatic increments by 1 whenever a non-null communication message is executed (an action belonging to CCM). Thus each name of a group during a step tk ($tk \geq 0$) is identified for each value assumed by T2.

Thus, the Cooperating Adaptive Devices can be described as:

$$CAD_{tk}=(GADM_{tk} , CCM , MCM ) \quad (5)$$

$$CAD_{tk}=(\{AD^1_{tk},AD^2_{tk},..,AD^m_{tk}\},CCM, MCM) \quad (6)$$

$CAD_{tk}$ is said Cooperating Adaptive Devices when for all operation, for every step tk ($tk \geq 0$), each element of the set of devices follows the behavior of the corresponding element of $GADM_{tk}$ until the execution of some non-null message communication, in the common communication mechanism (CCM), when the current step tk terminates and the next one (tk+1) starts.

Similar to adaptive actions in the Adaptive Device, each step increment is composed of two message communication components. The first one is called “before-communication” and has its instructions performed before the rule execution. The second component is called “after-communication” and has its instruction performed after rule execution. For a non-null communication message, at least one of the components must be non-null. A communication message component is formed by elementary standard messages and its execution may result in multiple additions and/or multiple deletions in some device´s rules.

Cooperating Adaptive Devices start its operation at some known initial shape for all m devices (m > 1) of the CAD, ($AD^1_0$, $AD^2_0$,.., $AD^m_0$), in the perspective of the communication between devices.

As defined in [13], an adaptive device can changes its own rules through an adaptive action. Thus, a built-in counter “T” is defined for each adaptive devices with initial value 0 and automatic increments by 1 whenever a non-null adaptive action is executed. In each step “k” ($k \geq 0$), the device has a different rule set. Thus, for each configuration of the rule set, for each step “k”, the device is referenced as $AD_k$.

Using both definition we have described, an adaptive device from CAD can be referenced $AD^r_{kr,tk}$. The “r” indicates the respective device of the CAD. The subscript “kr” indicates the adaptivity step of the device associated to the number “r”, while the subscript “tk” indicates step of the communication message in the CAD. Observe that “tk” is the same for all device of the CAD.

So, we can express the CAD initial configuration ($CAD_0$) as:

$$CAD_0 = (GADM_0, CCM, MCM) \quad (7)$$

$$CAD_0 = (\{AD^1_{k1,0} , AD^2_{k2,0} ,.., AD^m_{km,0}\}, CCM, MCM) \quad (8)$$

In formula (8), each $AD^r_{kr,0}$ (for r=1,…,m and kr=k1,…,km; all kr ≥ 0) means a device at some internal step "kr" of the adaptivity and the initial step of the communication message, which is indicated by the subscript "0" (tk = 0). So, in this configuration, no communication message has occurred. However, each device may have changed its own rules through adaptivity, if "kr" is greater than 0.

At step "tk" (tk≥0), an input stimulus always changes the Cooperating Adaptive Devices if and only if any non-null communication message is performed. Then, in any combination of step "kr" for all "m" devices (kr=k1, k2, …, km) and the step "tk", every device can be represented in the form $AD^r_{kr,tk}$. In this formulation "kr" indicates the step of its adaptivity while "tk" indicates the step of the communication message for all devices.

$$(AD^r)_{tk} = AD^r_{kr,\,tk} \quad (9)$$
(for r = 1, ... , m)

Formula (9) represents the configuration of each device of the Cooperating Adaptive Device.

$$GADM_{tk} = \{AD^1_{k1,tk}\ ,\ AD^2_{k2,tk}\ ,\ \ldots\ ,\ AD^m_{km,tk}\} \quad (10)$$

In formula (10) we have the configuration of the m devices at communication message step "tk", each one within its own step kr of adaptivity.

$$AD^r_{kr,\,tk} = (C^r_{kr,tk},\ IAR^r_{kr,tk},\ S^r,\ c^r_{kr,tk},\ A^r,\ NA^r,\ BA^r,\ AA^r,\ IBA,\ IAA) \quad (11)$$
(for r=1,..,m; kr=k1,…,km; tk ≥ 0)

In formula (11), we have:

- $C^r_{kr,tk}$ is the set of all possible configurations of device r, for steps "tk" and "kr" ( tk ≥ 0 e kr ≥ 0 , for r=1...m ).

- $S^r$ (for r=1,…,m) is a finite set of all possible events considered valid input stimuli for $AD^r$, containing the null event ( $\varepsilon \in S^r$ ).

- The input stimulus $w^r$ is:
$w^r = w_1^r\ w_2^r\ w_3^r\ w_4^r \ldots\ldots w_{nr}^r$ ( $w^r \in S^r$) (for r=1,...,m and nr ≥ 0).

- $c^r_{kr,tk}$ belongs to $C^r$ and is the initial device configuration ($c^r_{kr,tk} \in C^r_{kr,tk}$), for r = 1 .... , m; kr = k1 ...,km and tk ≥ 0. Before the occurrence of the first adaptive action ( kr = 0 ) and the first communication message ( tk = 0 ), $c^r_{0,0}$ is the initial configuration of the device "r".

- $A^r$ is the subset of its accepting configurations ( acceptance ) of the device "r", $A^r \subseteq C^r$ (for r=1,...,m)

- $NA^r$ is a finite set of output symbols of the device r (for r = 1, ... , m).

- $IAR^r_{kr,tk}$ is the finite set of all possible CAD rules, given by a relation IAR ⊆ IBA x $BA^r$ x $C^r$ x $S^r$ x $C^r$ x $NA^r$ x $AA^r$ x IAA. The rules of $IAR^r_{0,0}$ (for r=1,...,m) define the initial performance of the CAD rule set. A rule of a device containing communication message group changes another device rules set, by adding and/or deleting rules. The rules $IAR^r_{kr,tk}$ (r=1, .., m) have the form $iar^r$ = (iba, $ba^r$, $c_i^r$, $s^r$, $c_j^r$, $z^r$, $aa^r$, iaa), meaning that, in response to some stimulus $s^r \in S^r$, initially performs the "before-communication" message group $iba^r$, then the adaptive rule $ar^r$ = ($ba^r$, $c_i^r$, $s^r$, $c_j^r$, $z^r$, $aa^r$), and finally perform the "after-communication" message group $iaa^r$. The adaptive rule execution, $ar^r$, is performed as described in [13].

- $BA^r$ and $AA^r$ are the finite sets of adaptive action of the "r" device, which can modify its own rule set. This property is defined in details in [13].

- IBA and IAA are the finite sets of all communication message groups, both containing the null message ε (ε ∈ IBA ∩ IAA).

- CCM ⊆ IBA x BA x NR x AA x IAA, defined for a particular CAD is a communication mechanism to be applied to any rule in $IAR_{kr,tk}$, where the rule iar = (iba, ar, iaa), ar ∈ AR and AR ⊆ BA x NR x AA. In brief, the communication mechanism can have a communication message group belonging to IBA, which must be performed before the adaptive rule execution. Then, the adaptive rule is executed and followed by the communication message execution, which belongs to IAA.

- MCM is described in the next section.

### 3.4 MCM Mechanism and Communication Protocol (CP)

Each device in the Cooperating Adaptive Devices can modify the set of rules of some device by sending messages. After receiving a request message, the MCM mechanism sends an answering message. All communication between the devices and the MCM mechanism is executed by standard messages of the communication protocol.

Messages from the CP can be classified in four categories. The first one is the initial group and is responsible for configuring the device and the communication protocol, naming the devices with an association number.

The second one consists of configuration functions for setting the device information in a standard way that any device can reference the input stimulus, configuration states, rules and any information on other device's configuration. For example, any $c^r \in C^r$ (r=1,…, m) will be referenced as a number to be identified, if necessary, by a communication message of another device. After these associations, any device can reference any other device´s property.

The third group has the purpose of performing synchronization between the two devices involved in the communication, ensuring that only one adaptive rule with communication message is executed at any time by all devices.

The fourth group provides messages to finalize the interaction among the Cooperating Adaptive Devices and reset the protocol communication.

### 3.5 Coordination and Management Mechanism of Communication Messages

The MCM mechanism is responsible for preventing concurrent execution of communication messages. For example, if a device named "A" needs to perform a communication at the same time as another device tries the same, the mechanism MCM authorizes just one request of communication, keeping the other one in a waiting queue.

### 3.6 Communication Message Group

Communication messages group are defined as abstractions called communication function, in a similar way we have the function calls and declarations in a usual programming language. Communication messages correspond to a specific communication function call, which are generic abstraction.

A specification of communication message group has the following parts:

- *name*: a symbolic name used for referencing communication messages.

- *parameters*: a set of symbolic names used for referencing values passed as arguments to an communication function at the time it is

called. Once they are filled in with the values of their associated arguments, they may not be any further modified during the execution of the function.

- *variables*: symbolic names used for holding values resulting from the application of some standard message. They are currently used in association with another device´s property like actual configuration, events, rules, etc.

- *body*: the main part of a communication function encodes all instructions and standardize messages of the CP protocol needed to make the desired changes to the current set of some device's rules.

## 4 EXAMPLE

Our example problem consists of a distance learning course applied in a student´s class. As each student has different skills and different levels of related subjects learned, we need to represent all possible sequence of learning the topics and the real sequence adopted by each student.

To illustrate our formulation of Cooperating Adaptive Devices, an illustrative example is showed consisting of only two devices: a Finite State Machine and a Decision Table.

We will use these two formalisms to model a course of some subject, as mathematic for example, and the topics sequence chosen by each student. The course option is represented by the Decision Table while the individual choice of lessons and example learned is represented by the Finite State Machine. So, we must have a different Finite State Machine for each student and this information is very important to teacher´s analysis to comprehend how each student learned the available topics in a course.

The Decision Table has the ability to recognize ten unit lessons and ten unit tests. The end of the course is represented by the symbol " ├".

As the Decision Table recognizes a valid lesson or test, it sends a message to the State Machine, causing equivalent recognizing rule. So, the State Machine has the capability to recognize the same sequence of lessons and tests learned in the State Machine. From this perspective, the State Machine represents a student sequence of learning, while the Decision Table represents all possible sequence of the learning strategies, each one with its own characteristics. This modeling can be very useful to represent the differences of each student in distance education.

In the Cooperating Adaptive Devices example, as described before, we have a device with the ability to recognize some information (pattern) while another device holds the knowledge already acquired by the first one. The ability to modify the State Machine configuration is implicit in the communication message group, which is associated to a rule of the Decision Table.

Hereafter, we present the definitions and the interaction between these devices from our example.

### 4.1 Device 1 - Decision Table

The configuration of the Decision Table is shown in figure 2. The second row indicates the type of column. A code "I", as referenced in rows 6 and 7 indicates that column represents information, or more precisely, a final condition that could be an acceptance state or not, indicated respectively as "*OK*" or "*Not OK*" in the last two rows. If the second row contains "R", the respective column indicates a rule of the Decision Table. The rows 3 and 7, in yellow, represent the "before-communication" and "after-communication" communication function.

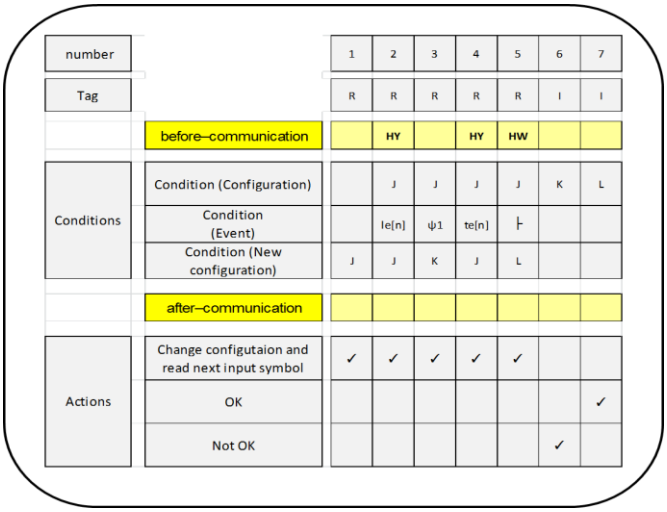


**Figure 2 -** Decision table – configuration.

According to the definition and making number association using standards messages from CP, we have:

The initial configuration (k=0):

$$AD^1{}_0 = (C^1, EAR^1, S^1, \mathbf{c^1_0}, A^1, NA^1, BA^1, AA^1, EBA^1, EAA^1) \quad (12)$$

$$C^1 = \{ J, K, L \} \quad (13)$$

Standard association:

- J : 1
- K: 2
- L : 3.

$$\mathbf{c^1_0} = \{ J \} \quad (14)$$

$$NA^1 = \{ \text{reject, accept} \} \quad (15)$$

Standard association:

- reject : 1
- accept : 2

$$S^1 = \{le1, le2, le3, le4, le5, te1, te2, te3, te4, te5, \vdash\} \quad (16)$$

Standard association:

- le1 : 1
- le2 : 2
- le3 : 3
- le4 : 4
- le5 : 5
- te1 : 6
- te2 : 7
- te3 : 8
- te4 : 9
- te5 :10
- ├ : 11

$$IAR^2 = \{rule^1(1), rule^1(2), rule^1(3), rule^1(4), rule^1(5)\} \quad (17)$$

where:

$rule^1(1)$: (Ø, Ø, Ø, Ø, J, Ø, Ø, Ø)
$rule^1(2)$: (HY(1,2), Ø, J, le[n], J, Ø, Ø, Ø)
$rule^1(3)$: (Ø, Ø, J, ψ1, K, reject, Ø, Ø)
rule1(4): (HY(1,2), Ø, J, te[n], J, Ø, Ø, Ø)
$rule^1(5)$: (HW(1,2), Ø, J, ├, L, accept, Ø, Ø)

- ψ1 is not a valid input, ψ1 ∉ S.
- le[n] → le1 or le2 or le3 or le4 or le5
- te[n] → te1 or te2 or te3 or te4 or te5
- HY and HW are group of communication message, called communication function.

The communication function HY creates a new rule in the state table by associating the current configuration to a new one, also generated by HY, by consuming the event "le[n]" or "te[n]", an event associated to a valid input stimulus, which represents the lessons and tests available. The "le[n]" or "te[n]" event represents in the Decision Table the same "lesson[n]" and "test[n]" as represented in the State Machine.

Similarly, the communication function HW creates an association in the State Machine between its current configuration and the final acceptance configuration.

In order to clarify our main idea of the Cooperating Adaptive Device, we omitted the

adaptive representation in the figure 2, but this feature is showed in details in [13].

### 4.2 Device 2 – Finite State Machine

Graphically, the State Machine initial configuration is represented in figure 3.

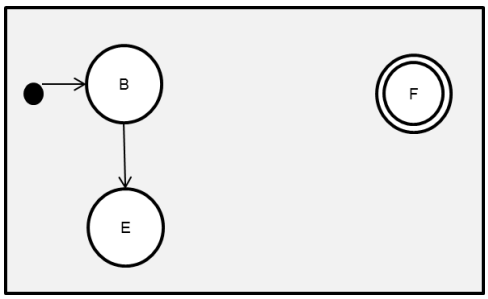


**Figure 3 –** State Machine - initial configuration.

According to the definition and making number association through standard associations from CP, we have:

$$C^2 = \{ B, E, F \} \quad (18)$$

Standard association:

- B : 1
- E : 2
- F : 3

$$c^2_0 = \{ B \} \quad (19)$$

$$NA^2 = \{ OK, NOT\_OK\} \quad (20)$$

Standard association:

- OK : 5
- NOT_OK : 10

$$S^2 = \{lesson1, lesson2, lesson3, lesson4, lesson5, test1, test2, test3, test4, test5, end\} \quad (21)$$

Standard association:

- lesson1 : 1
- lesson2 : 2
- lesson3 : 3
- lesson4 : 4
- lesson5 : 5
- test1 : 6
- test2 : 7
- test3 : 8
- test4 : 9
- test5 : 10
- end : 11

$$IAR^2 = \{ rule^2(1) , rule^2(2) \} \quad (22)$$

where:

- $rule^2(1)$: (∅, ∅, ∅, ∅, B, ∅, ∅, ∅)
- $rule^2(2)$: (∅, ∅, B, ψ2, E, reject, ∅, ∅)

-ψ2 is not a valid input, ψ2 ∉ S.

### 4.3 Interaction between the Devices

In this section, we present the interaction behavior of the pair of devices during the handling of simulation input "le1 le2 te1 ├ " in the Decision Table.

After handling the first element, the starting "le1", the Decision Table executes its $rule^1$ (2), which has the HY ("before–communication") communication function. The HY adds a new state in the State Machine, assign to "lesson1" event. The figure 4 shows the State Machine after the Decision Table has handled the input "le1 le2 te1 ├".

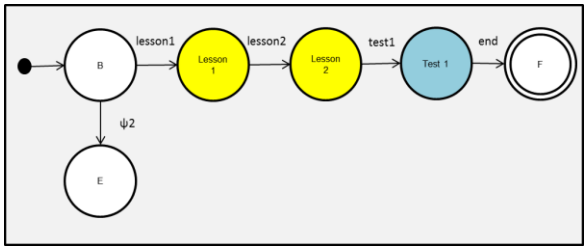


**Figure 4 –** State Machine final configuration, after Decision Table handling "le1 le2 te1 ├".

Sets $S^1$ and $S^2$ are different, but there is an implicit relationship between them.

In this final configuration, the State Machine has the following rules set:

$$IAR^2 = \{ rule^2(1), rule^2(2), rule^2(3), rule^2(4), rule^2(5), rule^2(6)\} \quad (23)$$

where:

- $rule^2(1)$: (∅, ∅, ∅, ∅, B, ∅, ∅, ∅)
- $rule^2(2)$: (∅, ∅,B, ψ2, E, reject, ∅, ∅)
- $rule^2(3)$: (∅, ∅, B, 1, Lesson 1, ∅, ∅, ∅)
- $rule^2(4)$: (∅, ∅, Lesson 1, 2, Lesson 2, ∅,∅, ∅)
- $rule^2(5)$: (∅, ∅, Lesson 2, 6, Test 1, ∅, ∅, ∅)
- $rule^2(6)$: (∅, ∅, Test 1, 11, F, ∅, ∅, ∅)

-ψ2 is not a valid input, ψ2 ∉ S.

The last element of our input stimulus is the end symbol, which is handled by the Decision Table rule (5), within the HW ("before–communication") communication function. When the automaton applies rule (5), this causes a response from the new rule (6) in the rule set of the State Machine. The Decision Table reaches a final state of acceptance and halts. The State Machine connects the current configuration to the final acceptance state as illustrated in figure 4. In this configuration, the State Machine has the ability to recognize the entry sequence "lesson1 lesson2 test1 end".

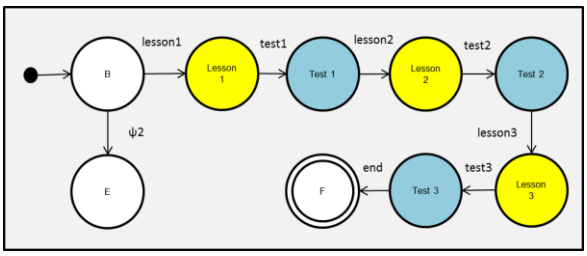


**Figure 5 –** Another possible State Machine final configuration**.**

If another student chooses another sequence to study, we will have a different final configuration of the State Machine like the example in figure 5.

## 5 ADAPTIVITY-RELATED WORK

Adaptive technology refers to techniques and methods involved in applications of adaptive devices. Historically, adaptive devices emerged from automata theory and most early applications were in the fields of formal languages, and later, of computer languages. Some of those early works are described in [15], [16], [17]. Adaptive automata were initially formulated as a supporting formalism for the representation of context-sensitive languages [18].

Afterwards, works were developed in the field of reactive systems [19]. The adaptive technique was applied in other fields such as robotic [20], [21], decision-taking problems [22] and decision-making systems [23]. Several other examples of the evolution of the adaptive technology are surveyed in [24].

## 6 CONCLUSION

Concluding, this work´s main result is to show an approach in representing reactive systems behavior using Cooperating Adaptive Devices. This approach can be used in any problem that needs two or more types of devices to represent the behavior and its dependence according the environment stimuli.

As showed, its usage is appropriate for situations where there is a device showing all possible alternatives of the phenomenon represented and the second device is used to show all alternatives performed by the first one. In short, one device represents all alternatives of the phenomenon while the second one

represents the history of the stimuli occurred in the environment causing rules changes.

As presented, the formulation used has a clean notation that can be applied to modeling adaptive phenomena, where the underlying device focus on its main usage and the communication messages on modeling the external influence of each device based on environment changes.

We hope this work will assist in the conception and construction of Cooperating Adaptive Devices for a new and cleaner perspective in activities involving complex problem solving with self-modifying formalisms.

## REFERENCES


[1] A Ingólfsdóttir, K. G. Larsen and J. Srba, Reactive systems: modelling, specification and verification, vol. 8, Cambridge, Cambridge University Press, 2007.

[2] J. Thyssen, and H. Benjamin, “Behavioral specification of reactive systems using stream-based I/O tables”, Software & Systems Modeling 12, no. 2, pp. 265-283, 2013.

[3] A. Vogelsang, S. Eder, M. Feilkas, and D. Ratiu, “Functional Viewpoint”, Model-Based Engineering of Embedded Systems, Springer Berlin Heidelberg, pp. 69-83, 2012.

[4] J. F. Woolley, J. Stanicka, and T. G. Cotter, "Recent advances in reactive oxygen species measurement in biological systems", Trends in biochemical sciences 38, n° 11, pp. 556-565, 2013.

[5] L. Hood, and D. J. Galas, "Systems biology and emerging technologies will catalyze the transition from reactive medicine to predictive, personalized, preventive and participatory (P4) medicine", Interdisciplinary Bio Central, vol 1, no. 6, pp. 1-5, 2009.

[6] E. Emma, J. P. Golding, and J. B. Phillips, "A versatile 3D culture model facilitates monitoring of astrocytes undergoing reactive gliosis", Journal of tissue engineering and regenerative medicine 3, no. 8, pp. 634-646, 2009.

[7] P. Antony, R. Balling, and N. Vlassis, "From systems biology to systems biomedicine", Current opinion in biotechnology 23, no. 4, pp. 604-608, 2012.

[8] P. M. A. Antony, C. Trefois, A. Stojanovic, A. S. Baumuratov, and K. Kozak. "Light microscopy applications in systems biology: opportunities and challenges." Cell Communication and Signaling 11, no. 1, pp. 1-19, 2013.

[9] Y. Shang, "An agent based model for opinion dynamics with random confidence threshold", Communications in Nonlinear Science and Numerical Simulation 19, no. 10, pp. 3766-3777, 2014.

[10] Y. Shang, "Modeling epidemic spread with awareness and heterogeneous transmission rates in networks", Journal of biological physics 39, no. 3, pp. 489-500, 2013.

[11] M. A. Louie, and K. M. Carley. "Balancing the criticisms: Validating multi-agent models of social systems", Simulation Modelling Practice and Theory 16, no. 2, pp. 242-256, 2008.

[12] D. Harel, and M.Politi, “Modeling reactive systems with statecharts: the STATEMATE approach”, McGraw-Hill, Inc., 1998.

[13] J. J. Neto, “Adaptive rule-driven devices-general formulation and case study”, Implementation and Application of Automata, Springer Berlin Heidelberg, pp. 234-250, 2002.

[14] M. Swiechowski, and J. Mandziuk, “Self-Adaptation of Playing Strategies in General Game Playing”, Computational Intelligence and AI in Games, IEEE Transactions on , vol. PP, no. 99, 2013.

[15] B. Burshteyn, "Generation and recognition of formal languages by modifiable grammars", ACM SIGPLAN Notices 25, no. 12, pp. 45-53, 1990.

[16] S. Cabasino, P. S. Paolucci, and G. M.Todesco, "Dynamic parsers and evolving grammars." ACM Sigplan notices 27, no. 11, pp. 39-48, 1992.

[17] R. S. Rubinstein, and J. N. Shutt, "Self-modifying finite automata: An introduction", Information processing letters 56, no. 4, pp. 185-190, 1995.

[18] J. J. Neto, “Adaptive automata for context-dependent languages”, ACM Sigplan Notices 29, no. 9, pp. 115-124, 1994.

[19] J. J. Neto, and J. R. Almeida Junior, and J. M. N. dos Santos, "Synchronized statecharts for reactive systems", in Proceedings of the IASTED International Conference on, 1998.

[20] M. A. A. Sousa, and A. R. Hirakawa, “Robotic mapping and navigation in unknown environments using adaptive automata”, Adaptive and Natural

Computing Algorithms, Springer Vienna, pp. 345-348, 2005.

[21] L. C. Barros Neto, and A. R. Hirakawa, "Data Structures in Robot Navigation Optimized by Adaptive Straightness", International Journal of Computer Applications 62, no.11, pp. 1-9, 2013.

[22] T. Pedrazzi, and A. H. Tchemra, and R. L. A. Rocha, "Adaptive Decision Tables A Case Study of their Application to Decision-Taking Problems", Adaptive and Natural Computing Algorithms, Springer Vienna, pp. 341-344, 2005.

[23] R. S. Okada, and J. J. Neto, "Hybrid Decision-Making using Adaptive Technology", International Journal of Computer Applications 45, pp. 38-45, 2012.

[24] J. J. Neto, "Um levantamento da evolução da adaptatividade e da tecnologia adaptativa", IEEE Latin America 5, no.7, pp. 496-505, 2007 (in portuguese)