EXTERNAL MODEL AND SYNCCHARTS DESCRIPTION

OF AN AUTOMOBILE CRUISE CONTROL

Bayart M.#, Lemaire E.#, Péraldi M-A.*, André C.*

#LAIL (URA CNRS 1440), Bat. P2, UFR IEEA,

Université des Sciences et Technologies de Lille, 59655 Villeneuve d'Ascq Cedex - France,

Tel. : (33) 3 20 43 45 65 - FAX : (33) 3 20 33 71 89, e-mail : bayart@univ-lille1.fr

**Laboratoire Informatique, Signaux, Systèmes (I3S)*

Université de Nice-Sophia Antipolis / CNRS, 41, bd Napoléon III - F - 06041 NICE Cedex

Tel. : (33) 4 93 21 79 56 - FAX: (33) 4 93 21 20 54, e-mail : {andre,map}@alto.unice.fr

Abstract : In this paper, the generic model developed for smart instruments and the synchronous model SYNCCHARTS are used to specify automated systems. The generic model provides us with an external description, which is the user's point of view, and the SYNCCHARTS model gives a behavioral model of the system. From those two models that provide complementary information upon a device, a method is proposed to obtain a coherent syncChart from a part of the external model. The classical example of an automobile speed cruise control system is used for illustration.

Keywords : Cruise Control, Formal Specification, Intelligent instrumentation.

1. INTRODUCTION

Various models are used to describe automated systems : functional, behavioral, object-based, internal or external models (Robert, *et al,* 1993; Staroswiecki and Bayart, 1994). The external model, using the concept of service offered to users and an organization based on operating modes, has led to a generic model description in a formal language (Bouras, 1997) that allows to specify and to qualify smart instruments and hybrid systems (Bayart and Lemaire, 1997).

Adding a behavioral model like SYNCCHARTS (André, 1996) to the external model leads to a more complete description of the equipment and offers extended facilities for simulation and validation.

In this paper, the external model concepts are firstly detailed then, an introduction to the SYNCCHARTS formalism is given. The third part concerns the obtaining of a SYNCCHARTS representation from a part of the external model is given in order to complete the equipment description. A cruise speed control system illustrates the proposed method.

2. THE EXTERNAL MODEL

The external model describes the device from the point of view of the services it is able to provide to external entities (operators, other field instruments, computers, ...). It introduces the following notions (Staroswiecki and Bayart, 1994; 1996) :

A **service** is defined as a procedure whose execution results in the modification of at least one datum in the instrument data base, or/and at least one signal on its output interface.

Services are required by the **users** who intervene on the equipment during its whole life cycle, i.e. not only during its exploitation (supervision, maintenance, technical management) but through out its life cycle, from its conception to its dismantling (initialization, configuration, ...).

The description of a service consists in the description of the result, which is produced by its execution.

In order to define the service, one will have to describe the computations which are done (algorithmic or sequential procedures, qualitative or fuzzy inferences, ...), the variables on which they are applied (inputs), the obtained results (outputs) and the required resources (hardware, software). Moreover, before it can be executed, a service must verify some activation conditions : i.e. conditions under which it can be executed in terms of accessibility and safety. So, a service is described by :

<Service>::=<Inputs, Outputs, Procedure, Activation Condition, Resources>

The following schema (Fig. 1) characterizes the structure of a service :

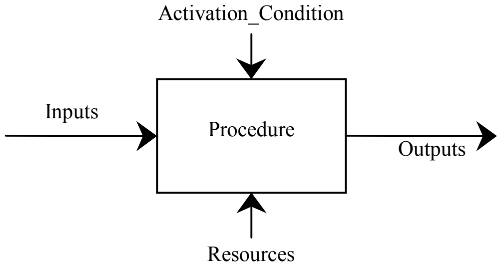

Fig. 1 : Structure of a service

The execution of a service can not be split and is obtained in response to a specific request. A request is defined by :

- its name which allows to identify it,

- its execution parameters, which allow to modulate the results. The set of all the parameterized requests the intelligent instrument recognizes, defines its supervisory language,

- its origin, which identifies the entity which produces it (control or maintenance operator, supervision device, control computer...),

- the communication link through which it is transmitted.

Associating the set of request, with the set of their authorized origins and communications links defines the supervisory protocol.

According to the resources state whose estimation is given by the Fault Detection and Isolation (FDI) algorithms that are implemented in the intelligent instrument, several versions of services (nominal and degraded) may be designed.

The services executions can be either dependent (precedence, mutual exclusion,...) or independent and concurrent. Likewise, the service can have a limited duration or can end on the occurrence of simple or complex events (operator request, emergency alarm,...). Finally, the services are organized according to user operating modes.

A **User Operating Mode** (USOM) is a coherent sub-set of services. It contains at least one service, and each service belongs at least to one USOM (Fig. 2). Moreover, in each USOM, there are services of the USOM that are implicitly executed (implicit request) as long as the system remains in the given USOM and other services which are the (requestable) services. This notion of USOM is, a priori, arbitrary since no reason gives us the right to form them. However, some of the USOMs are given by a general classification of the operating modes of industrial devices : off operation, configuration, manual or automatic mode, and so on.

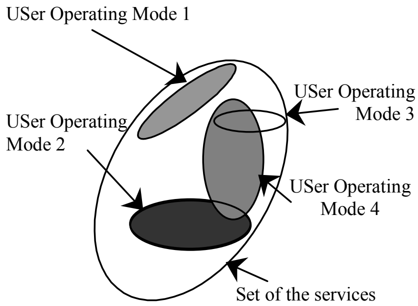

Fig. 2 : Service and USOMs

Obviously, for each USOM, there exists a specific service called "Changing USOM service" ; otherwise it would be impossible to leave any current USOM. This service is obtained in response to a request that must indicate the destination USOM. Of course all couple (origin, destination) can not be allowed, for security or operating reasons. So for each possible couple, the device definition has to include the conditions under which the passage from one mode to another one is possible.

The definition of the set of USOM and the transition conditions entirely specifies the intelligent device USOM management systems. A deterministic automaton (Fig. 3) can describe this system.

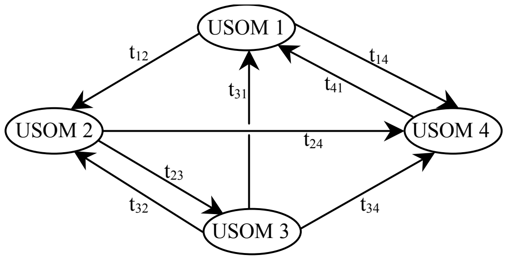

Fig. 3 : USOM deterministic automata.

From this description, one can obtain a formal specification of the intelligent equipment. However, the external model is usually not sufficient to validate the instrument behavior. So, the SYNCCHARTS formalism was chosen in order to complete and to be able to simulate and to validate the equipment running before realization.

3. THE SYNCCHARTS FORMALISM

"SYNCCHARTS" is an acronym for Synchronous Charts. SYNCCHARTS inherit from STATECHARTS (Harel, 1987) and ARGOS (Maraninchi, 1990). They are a new graphical representation of reactive behaviors based on the synchronous paradigm. They offer enhanced preemption capabilities and any syncChart can be translated into an equivalent Esterel program. Recall that Esterel (Boussinot and De Simone, 1991) is a powerful imperative synchronous language dedicated to reactive system programming.

Let us have a look at the hypotheses underlying the synchronous approach (André, 1996) :

- **Signals** : the system interacts with its environment through signals : input signals and output signals. A signal conveys two pieces of information : its *presence* status (a signal is either present or absent) and its *values* of a given type.

  The presence is transient (pulsed), whereas the value is persistent. The value of the signal may change only when it is present. Pure signals and sensors are special cases. A pure signal has no value, it is used to signal that some condition has become true. A *sensor* has no presence status ; the environment sets it, the reactive system can only read the value of a sensor.

- **Global Perception** : the synchronous approach assumes that all input and output signals are perceived simultaneously and that this perception is objective. So the model deals with tuples of signals. This hypothesis is called "the *perfect sampling* hypothesis".
- **Logical Time** : there is no physical time, but logical instants. This allows capturing the notion of *multiform time* : any input signal can be taken as a time reference, be it linked to a physical clock or any other physical phenomenon.
- **Zero-Delay Hypothesis** : internal operations are supposed to be executed in zero-delay. So the output signals are synchronous with the inputs that cause them.
- **Broadcasting** : the synchronous approach assumes that all signals are *instantaneously broadcast*. A consequence is that all the signals (including the output signals) have to be taken into account in order to determine the output signals to be emitted. This may induce surprising behaviors.

In SYNCCHARTS, preemption is a first class concept. There are two types of **preemption** : suspension and abortion. Abortion can be either weak or strong. The strong abortion one kills the process as soon as a given signal is present ; the killed process is not allowed to execute its "last wishes" when the abortion occurs. The weak abortion differs from the previous one in the fact the killed process executes its reaction at the current instant, before getting killed. SYNCCHARTS, like Esterel, deals with sequence, concurrency, preemption and communication in a fully deterministic way. SYNCCHARTS is endowed with a mathematically defined semantics, fully compatible with the ESTEREL one. The main differences between SYNCCHARTS and STATECHARTS are a stricter semantics and a richer preemption management for the former. Finally, SYNCCHARTS have their graphical representation. It is a state-based description of a reactive behavior. It supports states, hierarchy of states, concurrency and transitions of several types.

The basic block is the state or **star** (Fig.4) :

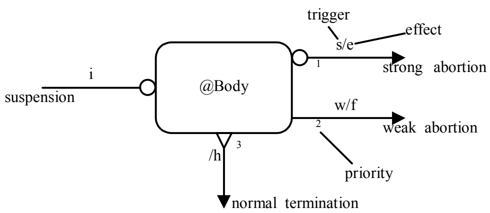

Fig. 4 : The star of a syncChart.

Stars are interconnected to make a **constellation**. Only one star at a time can be active in a constellation, so that a constellation can be seen as a classical state-graph (Fig. 5).

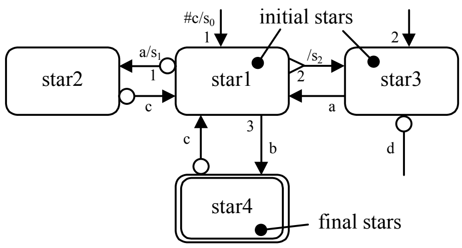

Fig. 5 : Constellation of a syncChart.

Each constellation has, at least, one initial star pointed to by an arrow. A parallel composition of constellations is a firmament or a **macro-state**. Dashed lines (Fig. 6) delimit concurrent constellations in a macro-state.

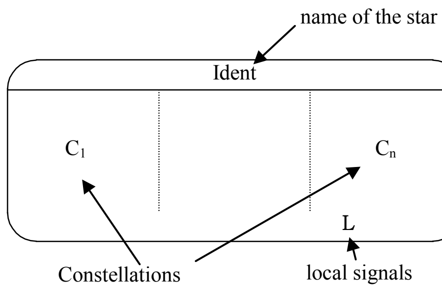

Fig. 6 : A macro-state of a syncChart.

The structure of a syncChart is defined recursively: the body of a star can be a macro-state (or even an Esterel module, i.e., a textual description). This feature is convenient for supporting hierarchical descriptions.

A prototype SYNCCHARTS editor/compiler is now available[1], a commercial version is under development. The SYNCCHARTS compiler generates an Esterel program equivalent to the syncChart. Thus, SYNCCHARTS can use the rich software environment developed for Esterel (compilers, links to proof systems, interactive simulation, efficient code generation).

4. EXTERNAL MODEL TO SYNCCHARTS DESCRIPTION

4.1. General method.

The problem is to complete the external model of a device by its syncChart description. In that sense, each USOM of the external model is defined as a macro-state. In each of those macro-states, the different constellations are formed by the different services present in the USOM. In this case, all the services are concurrent. The transitions from USOMs to USOMs are still the same in terms of syntax. However, the designer must specify if the transition is a weak abortion, a strong one or a normal termination. Finally, the suspension within each USOM for each service must be established according to the given specifications.

The obtained representation could be the final one. However, due to the large number of services in complex systems, the representation might be illegible. In that sense, a method has been developed in order to obtain a clearer internal representation.

[1] http://www.inria.fr/meije/esterel/syncCharts

4.2- Grouping of macro-states :

The first step of the method is to find the USOMs, which can be grouped according to the information provided by the external model.

Let M be the set of the user operating modes.

$M = \{m_j \,;\, j \in J\}$

Let S_{cm} be the set of "Changing USOM" services and T be the set of transition conditions.

$S_{cm} = \{(m_i, t_{ij}, m_j) \,/\, m_i \in M, m_j \in M, t_{ij} \in T\}$

t_{ij} indicates the logical condition required by the change from m_i to m_j.

Table 1 shows the activation conditions between USOMs.

USOM \ USOM	m_1	...	m_j	...
m_1	0			
...				
m_i			t_{ij}	
...				

From this table, one can deduce the USOMs that can be grouped. The "grouping condition" of two modes m_i and m_j expresses the fact that the resulting aggregated model should remain deterministic. For each entry in Table 1, one has to check :

$\forall m_i \in M, m_j \in M$, m_{ij} can be formed if and only if the following conditions hold ($\wedge$: logic AND) :

- $\forall k \in J \,/\, t_{ik} \neq 0, t_{jk} \neq 0$, $t_{ik} = t_{jk}$,
- $\forall k \in J, \forall l \in J \,/\, k \neq l$, if ($t_{jk} = 0$ or $t_{il} = 0$) then $t_{ik} \wedge t_{jl} = 0$
- $\forall k \in J, \forall l \in J \,/\, k \neq l$, if ($t_{ik} = 0$ or $t_{jl} = 0$) then $t_{il} \wedge t_{jk} = 0$.

This three conditions insure the deterministic behavior of the pair (m_i, m_j). The set of all the pairs of USOMs (m_i, m_j) which verify these above conditions is named K.

The second step consists in choosing, among the set K, the pairs which will be grouped according to the syncChart legibility. This one is increased if a macro-state contains USOMs that possess common services since those services will be expressed only once in the syncChart. So, the choice of pairs is linked to the number of common services they exhibit.

Let S be the set of services that the equipment can perform.

$S = \{s_1, s_2, ..., s_n\}$

Let Ls be the application, which associates to a USOM, the set of services, which are at disposal of users in this USOM. $\mathcal{P}(S)$ is the set of parts of S.

$Ls : M \rightarrow \mathcal{P}(S),$

$m_j \rightarrow Ls(m_j).$

The best macro-state obviously corresponds to the pair (m_i, m_j), in the set K, which shows the largest number of common services. Let K_1, a subset of K defined as :

$K_1 = \{(m_i, m_j) / Card(Ls(m_i) \cap Ls(m_j)) \geq Card(Ls(m_k) \cap Ls(m_l)), \forall(m_k, m_l) \in K\}$

Several cases have to be studied :

- case 1 : the pairs (m_i, m_j) of K_1 are independent, i.e. such as :

$\forall(m_i, m_j) \in K_1, \forall(m_k, m_l) \in K_1, m_i \neq m_k, m_i \neq m_l, m_j \neq m_k, m_j \neq m_l.$

The corresponding modes are then grouped in new ones.

- case2 : the pairs are not independent and several pairs can be grouped together, as example the pairs :

$(m_i, m_j), (m_i, m_k), (m_j, m_k)$

if $Ls(m_i) \cap Ls(m_j) = Ls(m_i) \cap Ls(m_k) = Ls(m_k) \cap Ls(m_k)$, then the mode m_{ijk} is formed

else, The USOM can not be grouped together and only one pair has to be considered. The choice has to take into account another criterion based on the E/S or on the resources used or on another specific characteristic.

After the formation of this "new" Macro-USOM m_{ij}, the set M has to be remolded : the macro-USOM "m_{ij}" must be added and the corresponding USOM m_i and m_j must be removed from M. Then the transitions have to be also modified :

- First step, the services (m_i, t_{ij}, m_j) and (m_j, t_{ji}, m_i) have to be "deleted" from S_{cm} : from outside of m_{ij} these services become transparent but still exist in m_{ij}.

- Second step, the transitions from m_{ij} to other USOM are specified. For each k :

if $t_{ik} \neq 0$ and $t_{jk} \neq 0$, then by construction $t_{ik} = t_{jk}$, the transition condition $t_{(ij)k}$ is formed with t_{ik} and the sum of the effects generated by t_{ik} and t_{jk} is added to $t_{(ij)k}$.

if $t_{ik} \neq 0$ and $t_{jk} = 0$, then the transition $t_{(ij)k}$ is formed by the conjunction of t_{ik} and the condition "In(m_i)".

if $t_{ik} = 0$ and $t_{jk} \neq 0$, then the transition $t_{(ij)k}$ is formed by the conjunction of t_{jk} and the condition "In(m_j)".

- Finally, the transitions to the macro-USOM m_{ij} may cause problems : which USOM (i.e. m_i or m_j) is activated upon the entrance in m_{ij} ? Three cases can arise :

if $t_{ki} \neq 0$ and $t_{kj} = 0$, the effect e_i is added to the transition $t_{k(ij)}$: it indicates that the mode m_i is activated by the changing USOM service on the transition $t_{k(ij)}$

if $t_{ki} = 0$ and $t_{kj} \neq 0$, the effect e_j is added to the transition $t_{k(ij)}$: it indicates that the mode m_j is activated by the changing USOM service on the transition $t_{k(ij)}$;

if $t_{ki} \neq 0$ and $t_{kj} \neq 0$, the two transitions have to been created $t_{k(ij)1}$ from t_{ki} associated with the signal e_i and $t_{k(ij)2}$ from the transition t_{kj} signal e_j.

In the syncChart description, if e_i and e_j are together generated for the macro-USOM, in the syncChart, an input arc label with the corresponding signal (#e_i for m_i and #e_j for m_j) is drawn for each USOM. If only one signal exists, an input arc label with the corresponding signal is drawn for the corresponding USOM and so the signal can be deleted from the specification.

For each obtained group of USOMs, the common services are then put as orthogonal constellations in the SYNCCHARTS formalism.

This "algorithm" is then reiterated with the "new" set M including the macro-USOMs until no grouping can be made.

5. EXAMPLE

As an example, the proposed design method of intelligent instrument is applied to the well-known example of the Automobile Cruise Control. This example provides several points of interest : on the one hand there is a continuous part relative to the control of the speed of the vehicle and, on the other hand a discrete part provided by the driver actions. This example of an Automobile Cruise Control was described in detail (Hatley, 1987) and used for the illustration of the SART method. For our specification, the application case described in Calvez (1990) is used.

5.1. Specification

The Cruise Control is an additional device that allows a driver to assign a constant speed set point for long Drives. The vehicle speed is controlled by an action on the electric valve that commands the fuel injection of the motor. When the appropriate speed is reached, the driver can engage the regulation. When the regulation is on, the driver can, at any time, take back the control of the vehicle by acting on the brake or on the accelerator :

- on an action on the accelerator, the speed of the vehicle increases and, at the end of the acceleration the speed is again regulated by the cruise control at the previous set point,
- on an action on the brake, the cruise control is deactivated; the driver can reactivate it by pressing the "Resume" button.

At the setting up, a calibration procedure determines the conversion factor between the wheel impulsion coder and the calculator. This procedure allows the cruise control to adapt to any type of wheel.

Some other functionnalities have been implemented in the example proposed by Calvez (1990). This functionnalities are :

- the follow-up of an average speed on a given drive : the driver set the start of the drive, and then can ask for the average speed with regard to the previous set point of the drive.
- the follow-up of an average consumption : on each filling up of the tank, the driver can follow the consumption of the vehicle in respect to the last filling up memorized.
- the production of maintenance information : they are produced to inform the driver of the maintenance phase necessity with regard to the vehicle's total mileage.

5.2. Description using the external model.

The specification of the Cruise Control begins with the enumeration of all the services available to the driver. Let S be this set of services.

S is composed of eleven services :

- Average_Speed_Elaboration, which calculates the average speed of the vehicle according to the memorized beginning of the drive,
- Consumption_Elaboration, which calculates the consumption of the vehicle,
- Distance_Elaboration, which calculates the current total distance covered by the vehicle,

- Maintenance_Information_Elaboration, which elaborate informations for the driver according to the maintenance phases.

- Maintenance_Writing, which is a service used to warn that a maintenance phase has been executed,

- Man_Valve_Control, which gives the action on the valve according to the pressure on the accelerator,

- Reg_Speed_Capture, which is the service that give the current regulation speed set point according to the driver's actions,

- Reg_Valve_Control, which gives the action on the valve according to the regulation speed set point,

- Speed_Calibration, which is the service that calibrate the services Distance_Elaboration and Speed_Elaboration,

- Speed_Elaboration, which calculates the current speed of the vehicle

- Start_Drive_Capture, which memorizes the beginning of the drive (set by the driver).

This set of external services is deduced from the specification given in natural language. Some other services exist in the specification for the designer. However, these external services need some other resources to fulfill their procedure. An external device can supply these resources. In this case, it was chosen to embed internal services, which produce the informational resources needed by the above external services. For the above set of services, a time base for the calculation of the speed and for the regulation valve control service is needed, and the "date" for calculating the average speed is also needed. There is two additional internal services to be added to the set S of services :

- *Clock*,

- *Hour and Date Production*.

The next step of the specification is the composition of the USOMs. In this case, five USOMs appear in the specification : Stop, Driver, Regulation, Brake in Regulation and Calibration. The composition of each USOM must be explicitly given. These can be done in a Backus Naur Norm form already used in Bouras, (1997):

USOMs_list ::=

(

Stop::= (Consumption_Elaboration, Maintenance_Information_Elaboration, Maintenance_Writing, *Clock*, *Hour_Date_Production*)

Driver::= (Average_Speed_Elaboration, Distance_Elaboration, Maintenance_Information_Elaboration, Man_Valve_Control, Speed_Elaboration, Start_Drive_Capture, *Clock*, *Hour_Date_Production*)

Regulation::= (Average_Speed_Elaboration, Distance_Elaboration, Maintenance_Information_Elaboration, Man_Valve_Control, Reg_Valve_Control, Reg_Speed_Capture, Speed_Elaboration, Start_Drive_Capture, *Clock*, *Hour_Date_Production*)

Brake_in_Regulation::= (Average_Speed_Elaboration, Distance_Elaboration, Maintenance_Information_Elaboration, Man_Valve_Control, Reg_Speed_Capture, Speed_Elaboration, Start_Drive_Capture, *Clock*, *Hour_Date_Production*)

Calibration::= (Maintenance_Information_Elaboration, Man_Valve_Control, Speed_Calibration)

)

When the set of USOMs, called M, is defined, the behavior of the Automotive Cruise Control has to be given. This can be done by defining the set of transitions T between the different USOM seen above. Here, a transition is considered as a disjunction of conjunctions of boolean terms. So, it is easy to verify the determinism of the transition graph (Atlee and Gannon, 1993). Furthermore, the connexity analysis of the graph and the lack of deadlock allow us to ensure its vivacity (Gondran and Minoux, 1990).

The obtained deterministic automata is given below (Fig. 7) with the set T of transitions between modes. In the transitions, the symbols used are logical and the part given after a "/" is a command that must be executed at the time the transition is taken.

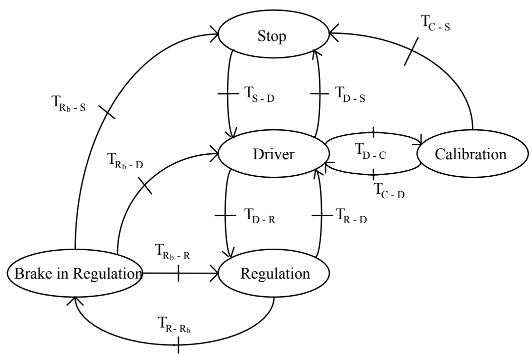

Fig. 7 : External model automata of the Automobile Cruise Control.

The transitions $T_{i\text{-}j}$ labeled S for Stop, D for Driver, C for Calibration, R for Regulation and Rb for Brake in Regulation, give first the source USOM and second the destination USOM. This transitions are :

- $T_{S\text{ - }D}$: (Motor Ignited)

- T_{D-S} : (¬Motor Ignited)

- T_{D-C} : Start Measurement KM / Caliber = 0

- T_{C-D} : Stop Measurement KM / Caliber = pulses

- T_{C-S} : (¬Motor Ignited)

- T_{D-R} : (Regulation Engaged)∧(S>50km/h)∧(¬brake)∧(Gear Lever Engaged)∧(Caliber ≠ 0) / RS= S

- T_{R-D} : Regulation Disengaged / RS = 0

- T_{R-Rb} : (Brake)∨(¬Gear Lever Engaged)

- T_{Rb-R} : ((¬Brake)∧(Gear Lever Engaged)∧(Resume)) ∨

  ((Regulation Engaged)∧(S>50km/h)∧(¬brake)∧(Gear Lever Engaged) / RS = S)

- T_{Rb-D} : Regulation Disengaged

- T_{Rb-S} : (¬Motor Ignited) / RS = 0

The deterministic automata gives us the behavior of the user operating modes of the Automotive Cruise Control.

The model proposed above is well fit to Hybrid Systems. It gives both a good and a clear representation of the general system behavior without describing it in details (i.e. giving a full internal description). In particular, it does not take into account the service organization. Let us now present the SYNCCHARTS representation to fit more closely to internal behavior and functionalities.

5.3. Application to the Cruise Control.

In order to obtain SYNCCHARTS description the method proposed above is used. First, the pair of USOM that can be grouped are searched for. By looking upon the transition conditions between the different USOMs, four possible pairs of USOMs that hold the above conditions (cf. §4.1.) can be obtained :

- (Driver, Brake in Regulation),
- (Driver, Calibration),
- (Driver, Regulation),
- (Regulation, Brake in Regulation).

Then the subsets of services belonging to several USOMs can be found. Here four great subsets appear :

- { Average_Speed_Elaboration, Distance_Elaboration, Maintenance_Information_Elaboration, Man_Valve_Control, Reg_Speed_Capture, Speed_Elaboration, Start_Drive_Capture, *Clock, Hour_Date_Production* } in Regulation and Brake in Regulation.

- {Average_Speed_Elaboration, Distance_Elaboration, Maintenance_Information_Elaboration, Man_Valve_Control, Speed_Elaboration, Start_Drive_Capture, *Clock*, *Hour_Date_Production*} which belong to the three USOMs : Driver, Regulation and Brake in Regulation,

- {Manual Valve Control, Maintenance_Information_Elaboration, *Clock*, *Hour_Date_Production*} which belong to four USOMs : Driver, Brake_in_Regulation, Regulation and Calibration,

- { Maintenance_Information_Elaboration, *Clock*, *Hour_Date_Production*} which belong to all the USOMs of the cruise control.

The formation of the new syncChart proceeds as follows :

- From the first subset and from the pair (Regulation - Brake in Regulation), a macro–state is formed that can be called "Cruise_Control_Engaged".

- the first subset is also contained in the second one. Moreover all conditions hold to group the set of three USOMs { Driver - Regulation - Brake in Regulation}. In this case, the macro-state will contain the USOM Driver and the macro-state "Cruise Control Engaged" and it will be called "Running".

- once more, this second subset is contained in the third subset. The macro-state that contains the macro-state "Running" and the USOM Calibration is also created. This macro-state will be called "Motor Ignited"

- Finally, the USOM "Stop" is contained in the macro-state "SPEED CRUISE CONTROLLER" which is the top macro-state of our syncChart of this example.

The obtained SYNCCHARTS representation from the external model is given below (Fig. 8). All the five USOMs described in the external model can be found in this representation.

For an efficient use of SYNCCHARTS some adaptations are desirable. Even if SYNCCHARTS may deal with conditions (predicates), event-driven descriptions are preferable. Thus, the following events are introduced :

- Ignition_ON (Ignition_OFF, resp.) is the event that causes the motor to enter (to leave, resp.) the state " Motor_Ignited".

- Regulation_ON (Regulation_OFF, resp.) is the event that causes the regulation to enter (to leave, resp.) the state "Cruise_Control_Engaged".

For sake of clarity the boolean function B is defined : B = (¬brake)∧(Gear Lever Engaged). Then the transition's condition between the different USOMs and macro-states become the following :

- the USOM "Regulation" is engaged on entrance in the macro-state "Cruise_Control_Engaged" and the transitions are :

$T_{Reg\text{ - }Rb}$: [not B] ;

$T_{Rb\text{ - }Reg}$: Resume [B and (?S>50)] ;

- the USOM "Driver" is engaged on entrance in the macro-state "Running" and the transitions are :

$T_{D\text{ - }CCE}$: Reg_ON [B and (?caliber>0) and (?S>50)] / RS(?S) ;

$T_{CCE\text{ - }D}$: Reg_OFF ;

- the macro-state "Running" is engaged on the entrance in the macro-state "Motor_Ignited" and the transitions are :

$T_{Running\text{ - }C}$: Start_KM [In(Driver)] / caliber(0) ; the condition [In(Driver)] specifies that the USOM Calibration is only reachable from the USOM "Driver" ;

$T_{C\text{ - }Running}$: Stop_KM ;

- the USOM "Stop" is engaged on entrance in the state "SPEED CRUISE CONTROLLER" which is logical and the transition is Ignition_ON (resp. Ignition_OFF) from Stop to "Motor_Ignited" (resp. from "Motor_Ignited" to Stop).

The obtained SYNCCHARTS representation from the external model is given below (Fig. 8). All the five USOMs described in the external model can be found in this representation. One can see that the services are now clearly visible and classified.

The hierarchical description supported by SYNCCHARTS makes them different from automata, which are « flat » models. The use of SYNCCHARTS leads to a reduction of the number of arcs (e.g., transitions $T_{Rb\text{ - }D}$ and $T_{R\text{ - }D}$ in Fig. 7 are factorized in Fig. 8).

Last but not least, clever use of preemption may result in more concise description (e.g., note the self-loop arc on the "Cruise Control Engaged" macro-state. It is used to instantaneously restart the regulation with a new reference speed).

6- CONCLUSION :

The external model allows to describe an intelligent equipment from the point of view of the users. It provides a functional description, which is not sufficient to validate entirely the instrument behavior. In that sense, a SYNCCHARTS description was proposed to complete the external model. The interest of this method rests on a hierarchical representation of user operating mode and a clear presentation of concurrent services in each mode.

In this paper, a method to derive a SYNCCHARTS description from the external model has been proposed. At the present time, several SYNCCHARTS descriptions can be found from the same specification.

REFERENCES

André, C. (1996). Representation and analysis of reactive behaviors : a Synchronous Approach, *Invited paper CESA'96*, pp. 19-29, Lille, France.

Atlee, J.M., and J. Gannon J. (1993). State-Based Model Checking of Event-Driven System Requirements., *IEEE Transactions on Software Engineering*, **Vol. 19 n°1**, pp. 24-40.

Bayart M., M. Staroswiecki M. (1993). A Generic Functional Model of Smart Instrument for Distributed Architectures, In : *IMEKO, Intelligent instrumentation for remote and on-site measurement*, Brussels, Belgium.

Bayart, M., E. Lemaire (1997). Intelligent Formal Description Applied to Hybrid Systems, In : *IFAC-SICICA'97 :* , pp. 715-720, Annecy, France.

Bouras, A. (1997). Contribution à la conception d'architectures réparties : modèles génériques et interopérabilité d'instruments intelligents., PhD thesis, Université de Lille I, France.

Boussinot, F. and De Simone, R. (1991). The Esterel language., *Proceeding of the IEEE, 79*, pp. 1293-1304.

Calvez J.P. (1990). Spécification et Conception des Systèmes", pp. 61-98, Masson, Paris.

Gondran, M. and M. Minoux (1990). Graphes et algorithmes, Eyrolles, Paris.

Harel, D.(1987). StateCharts, a Visual Formalism for Complex Systems, Sc. of Computer Prog., **Vol 8**, pp. 231-274.

Hatley, D.J. and I. Pirbhai (1987). Strategies for Real-Time System Specification, Dorset House, NY.

Maraninchi, F.(1990). ARGOS : un langage pour la conception, la description et la validation des systèmes réactifs., PhD thesis, Université Joseph Fourier, Grenoble I, France.

Robert, M., M. Marchandiaux, M. Porte (1993) Capteurs Intelligents et Méthodologie d'Evaluation, Hermès, Paris.

Staroswiecki, M. and M. Bayart (1994). Actionneurs intelligents, Hermès, Paris.

Staroswiecki M. and M. Bayart (1996). Models and Languages for the Interoperability of Smart Instruments, *Automatica*, **Vol. 32**, No6, pp. 859—873.

Ward, P., Jensen and Al (1988). ESML : An Extended Systems Modeling Language Based on the Data Flow Diagram, *ACM Software Engineering Notes*, **Vol. 13**, pp. 1-19.

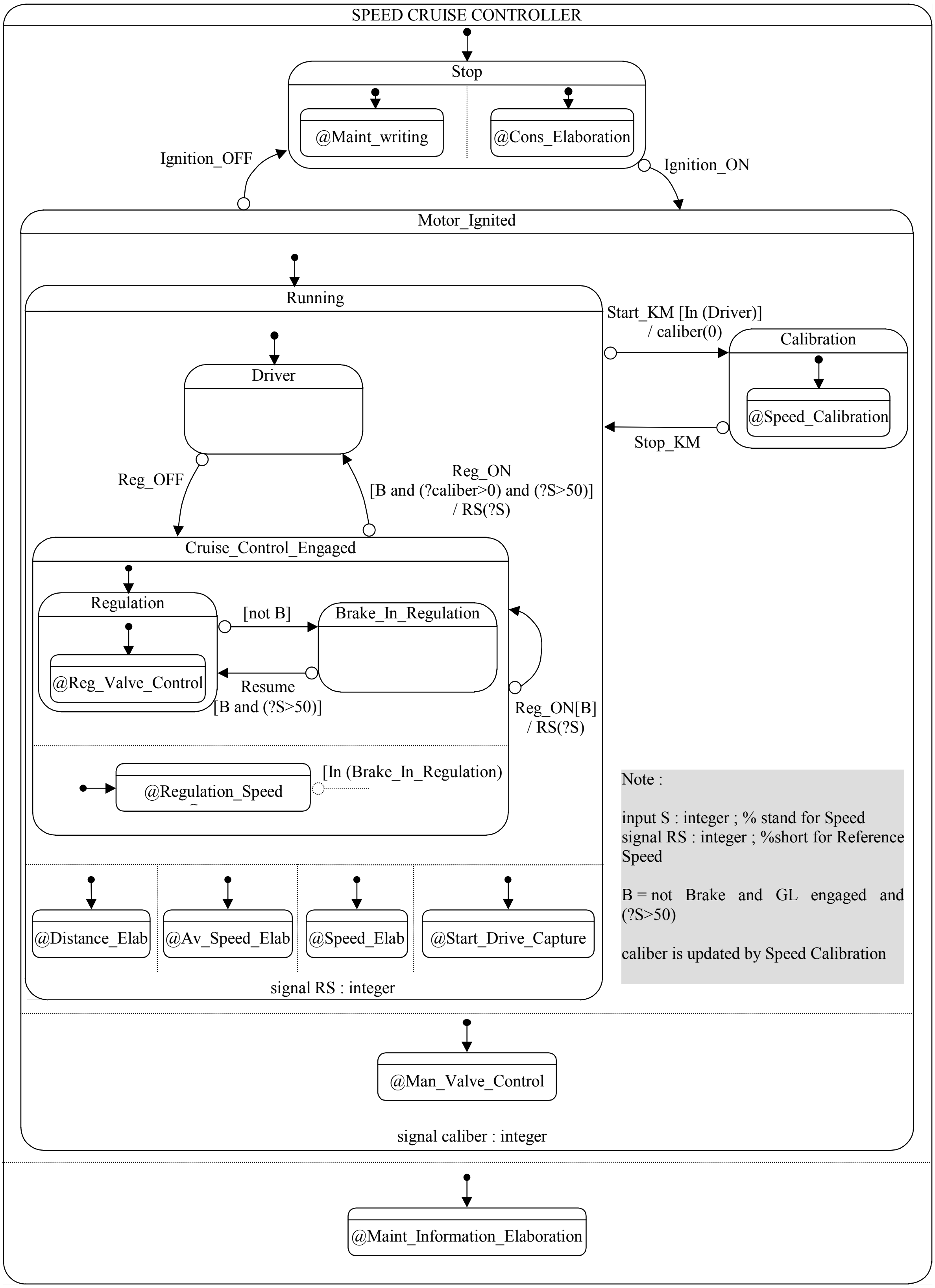

Fig. 8 : The syncChart of the Automobile Cruise Control.