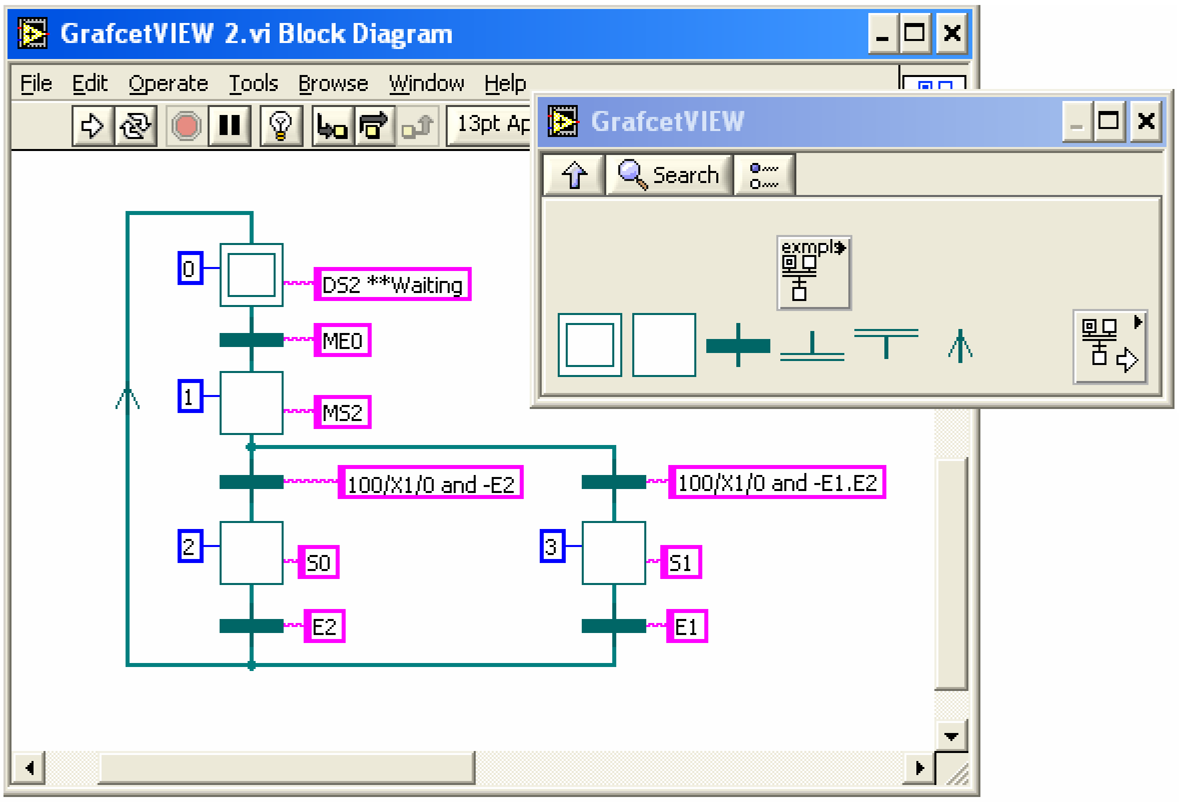


# GrafcetVIEW v.1.2

## Reference Manual

TecAtlant

july 2009

TecAtlant
Kervarail N°11
56470 Saint Philibert
FRANCE
Phone : + 33 2 97 55 10 62

# Warranty Limitation

The media on which you receive TecAtlant software are warranted not to fail to execute programming instructions, due to defects in materials and workmanship, for a period of 90 days from the date of shipment, as evidenced by receipts or other documentation. TecAtlant will, at this option, repair or replace software media that do not execute programming instructions if TecAtlant receives notice of such defects during the warranty period. TecAtlant does not warrant that the operation of the software shall be uninterrupted of error free.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. TecAtlant will pay the shipping costs of returning to the owner parts which are covered by warranty.

TecAtlant believes that the information in this document is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical error exists. TecAtlant reserves the rights to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult TecAtlant if errors are suspected. In no event shall TecAtlant be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, TECATLANT MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIM ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. COSTUMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF TECATLANT SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE COSTUMER. TECATLANT WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCT, OR INCIDENTAL OF CONSEQUENTIAL DAMAGES, EVENT IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of TecAtlant will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against TecAtlant must be brought within one year after the cause of action accrues. TecAtlant shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow TecAtlant installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

# Table of contents

# About this manual

The *GrafcetVIEW reference manual* describes the user interface of the **GrafcetVIEW** library.

To benefit fully from it, it is preferable that you are familiarised with MS-DOS, MS-WINDOWS and with the programming language LabVIEW.

## Conventions

The following convention appears in this manual:

| | |
|---|---|
| » | The symbol » leads you through nested menu items and dialog box options to a final action. The sequence **File»Page Setup»Options** directs you to pull down the **File** menu, select the **Page Setup** item, and select **Options** from the last dialog box. |
| ✍ | This icon denotes a note, which alerts you to important information. |
| ⚠ | This icon denotes a caution, which advice you of precaution to take to avoid injury, data loss, or a system crash. |
| **Bold** | Bold text denotes items that you must select or click in the software, such as menu items and dialog box options. Bold text also denotes parameter names, controls and buttons on the front panel, dialog boxes, selection of dialog boxes, menu names, and palette names. |
| *Italic* | Italic text denotes variables, emphasis, a cross reference, or an introduction to a key concept. This font also denotes text that is a placeholder for a word or value that you must supply. |
| ***Bold italic*** | Bold and italic text denotes a note, an advice or a warning. |

## Contacts with customers

You will find forms of technical support and informative at the end of this manual, in the appendix “Contact with customers”. Thanks to complete this forms and to return it indicating your comments and notes on this product and its handbook.

You can also description which you will develop with this product to obtain more information and to help you to solve possible problems which you can encounter.

# Chapter 1.
# Installation of GrafcetVIEW

This part describes the requirements to use GrafcetVIEW and its installation procedure.

## Matériel requis

GrafcetVIEW, as LabVIEW, is a multi-platform product.

⚠ One version of GrafcetVIEW is available on each platform that is supported by LabVIEW:

| | Minimum | Recommended |
|---|---|---|
| **Windows** | | |
| OS | Windows NT/2000/XP (NT 4,0 SP6 or more recent) | Windows NT/2000/XP (NT 4,0 SP6 or more recent) |
| Processor | Pentium III, Celeron 600 MHz or equivalent | Pentium IV or equivalent |
| Memory | 128 Mo | 512 Mo |
| Disk space | 10 Mo | 10 M0 |
| **Mac OS** | | |
| OS | Mac OS X (Version 10,2) | Mac OS X (Version 10,2) |
| Processor | G3 | G4 |
| Memory | 128 Mo | 256 Mo |
| Disk space | 10 Mo | 10 Mo |
| **Linux** | | |
| OS | All Linux distributions with GNU C version 2.1.3 or later, including Red Hat 7.0, SuSE Linux 7.1, Debian Linux 3.0 and later. | |
| Processor | Pentium III, Celeron 600 MHz or equivalent | Pentium IV or equivalent |
| Memory | 64 Mo | 256 Mo |
| Disk space | 10 Mo | 10 Mo |
| Additional | XWindows Server | |
| **Solaris** | | |
| OS | Solaris 2.5.1 or later | Solaris 2.5.1 or later |
| Processor | SPARC 500 Mhz | SPARC 600 MHz |
| Memory | 64 Mo | 256 Mo |

| | | |
|---|---|---|
| Disk Space | 10 Mo | 10 Mo |
| Additional | XWindows Server | |

## Software requirements

To install GrafcetVIEW, you must previously install LabVIEW® on your system. The version 1.2 of GrafcetVIEW is available for each version of LabVIEW® since the version 9.

## Installation de GrafcetVIEW

1. Under Windows, insert installation CD-ROM into the CD-ROM driver.
2. Select the menu **Start » Run** and enter the command line *x:\setup*.exe (where x is the letter of the driver into which you put the CD-ROM)
3. During the installation process, the program displays dialog boxes. Click on the button Next to continue the installation, on the button **Previous** to return to the previous step or on the button **Cancel** to stop the installation procedure.
4. Once the installation is finished, the program displays a dialog box which indicates that all is unrolled without problem. Click on the button **Finish**.

# Chapter 2. The GRAFCET

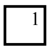
The GRAFCET (Functional Graph of Control Step Transition) is a formal model being used to specify and also to control reactive systems of the type “all or nothing” (Boolean inputs and outputs). An automated system of production (ASP) breaks up into two parts: the control part and the operative part. The operative part includes the process having to be controlled as well as the operator. The control part is intended to process the data coming from the operative part to control it.

The following presentation does not claim to be a definition of the grafcet but just an introduction. Moreover, one will reveal there only the concepts of GRAFCET which were implemented in GrafcetVIEW.

## Construction rules of a grafcet

A grafcet is graph which is composed of *steps* and *transitions*, connected between them by *connections* or *directed arcs*.

### Steps



A *Step* is represented by a square to which a unique number is associated.

A *Step* can be *initial* (represented by a double square).

A *Step* is either *active* or *inactive*.

The whole of active *steps* (called *situation*) entirely defines the state of the system. We specify for each *step*, the actions to be executed. These *actions* are executed only when the corresponding *step* is active. We can associate a condition to those actions, the *action* is then executed only if the *step* is active and the *condition* is performed.

## Transitions

A *transition* is represented by a horizontal line.

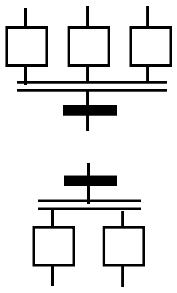
A *transition* represents a possibility of change of the comportment of the system. This change of comportment (the passage from one step to the following) corresponds to the crossing of a *transition*.

A *transition* is *validated* when all previous steps are active. The logical proposal which conditions the *transition* calls the *receptivity*.

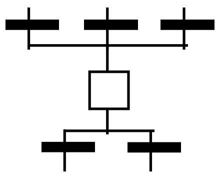
## Directed arcs

Each *directed arcs* links a step to a transition or a transition to a step: there is always strict alternation: step-transition. When this connection is connection is directed upwards, we show the direction by an arrow upwards.

## Convergences and divergences



When the directed arcs start from several steps (known as *steps downstream*) and arrive on the same transition or when directed start from the same transition and arrive on several steps (known as *steps upstream*) then these regrouping are represented by two horizontal parallel lines respectively called convergence and divergence "in *and*".



When separation is in the direction from several transitions to a common step (respectively from a step to several transitions), we name them convergences (respectively divergence) "in *or*". Their representation is done by dividing the directed connections.

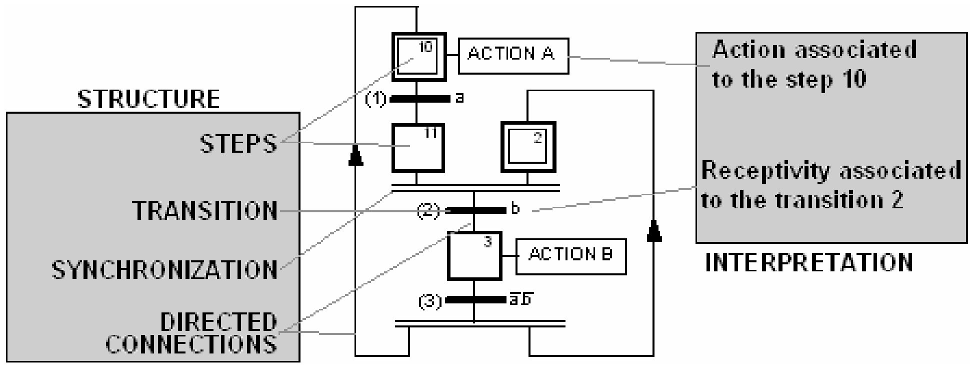


**Figure 1 : Structure of a Grafcet and interpretation**

## Rules of evolution of a grafcet

The evolution of a grafcet is subject to five rules:

**Rule 1:** **Initial situation**

The initial situation of a grafcet characterizes the initial behaviour of the control part with respect to the operative part, to the operator and/or to the external elements. It corresponds to the active steps at the beginning of operation: those steps are the *initial steps*.

**Rule 2:** **Crossing a transition**

A transition is validated when all immediately preceding steps connected to this transition are active. The *crossing* of the transition occurs:

- when the transition is validated,
- **AND** when the receptivity associated with this transition is true.

**Rule 3:** **Evolution of activated steps**

The *crossing* of a transition involves **simultaneously** the activation of all immediately following steps and the inactivation of all immediately preceding steps.

**Rule 4:** **Simultaneous evolution**

Several simultaneously crossable transitions are simultaneously crossed.

**Rule 5:** **Simultaneous activation and inactivation**

If during operation same step is simultaneously activated and inactivated, it remains active.

✍ **M*oreover, two evolution modes are generally accepted: evolution without searching for the stability or evolution with searching for the stability. Grafcet implements the last one.***

*Stability:* For a value of the vector of inputs of the system leading to a given situation, this situation reached will be known as *stable* if after crossing of all the crossable transitions, a new situation can be obtained only on occurrence of an external event.

The outputs associated with the steps belonging to a non-stable situation are not emitted. For a given stable situation, the associated outputs which logical conditions are true are emitted with the value true, the others are emitted with the value false.

✍ ***During an evolution with search for stability, a new value of the vector of the inputs is considered only when a stable situation is reached. Consequently, a completely unstable situation (return to a same situation during the same evolution) involves a looping without end.***

# Chapter 3. Using GrafcetVIEW

This part describes the how to create, how to execute and how to debug a grafcet with GrafcetVIEW.

⚠ ***The use of GrafcetVIEW requires a preliminary knowledge of the LabVIEW programming language. It is pointed out here that the term VI is the abbreviation of Virtual Instrument: it indicates a LabVIEW program.***

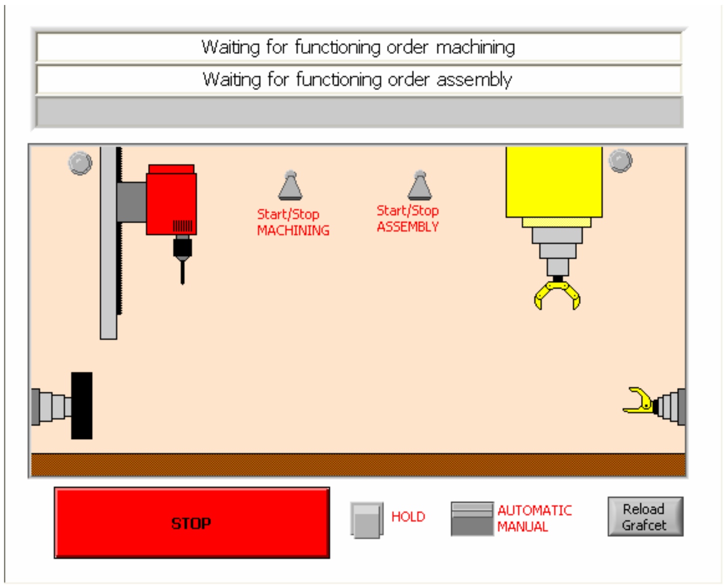
## Principle of GrafcetVIEW

The creation of a process control application with the library GrafcetVIEW is decomposed in 3 phases:

1. Definition of the inputs/outputs of the control part of the system. Those inputs/outputs can be of two types:
   - Software, by the means of the graphical interface (to or from the operator)
   - Material, by the means of physical devices (to or from the process), such as digital input/output devices, serial connections, network, …
2. Edition of the grafcet specifying the comportment of the control part.
3. Integration of the two previous phases into an execution engine which allows executing the application.

We chose to illustrate each point on a test application, following the development of the final application step by step. The selected application is a workshop made up of a manufacture part and an assembly part. The manufacture part receives blanks which it machines. These parts are then deposited in a place of storage. Then, either on order of an operator, or in an automatic way (when the part is detected), an arm manipulator recovers the part to subject it to a machine of assembly where an assembly is carried out. The finished part is then evacuated.

Other examples of use of GrafcetVIEW are included in the library *exemples.llb* under the directory *.../LabVIEW/grafcet Exmpls*.


Waiting for functioning order machining

Waiting for functioning order assembly

**Figure 2: Synoptic of the process**

The grafcet specifying the behaviour of our order part is as follows:

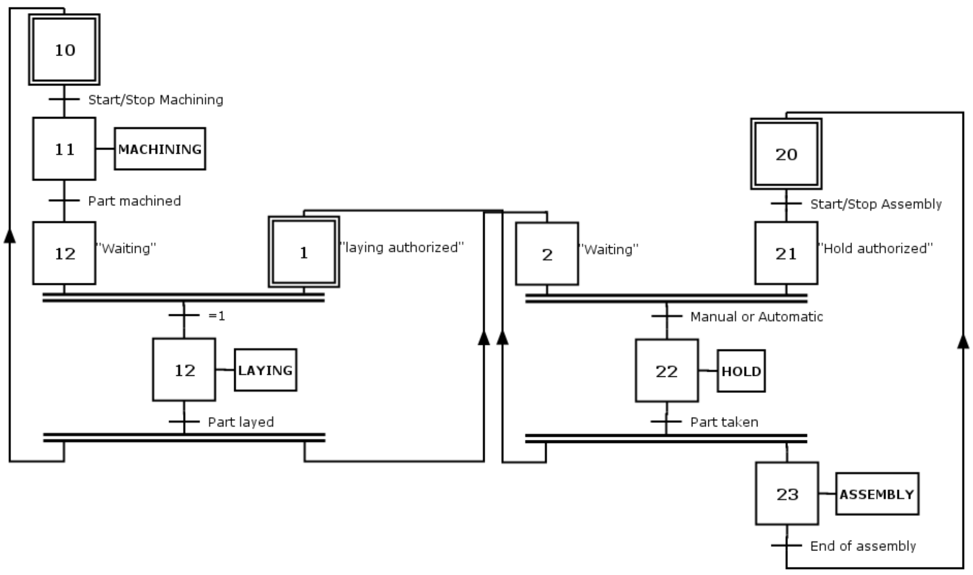


**Figure 3: Grafcet of the control part**

For this system, we defined the inputs of the control part with respect to the operative part, as follows:

| Designation | Reference name | Source | |
|---|---|---|---|
| | | Operator | Device |
| Start/Stop Machining | E0 | ✓ | |
| Start/Stop Assembly | E1 | ✓ | |
| Manual/Automatic | E2 | ✓ | |
| Hold | E3 | ✓ | |
| Machined part | E4 | | ✓ |
| Part laid | E5 | | ✓ |
| Part taken | E6 | | ✓ |
| End of assembly | E7 | | ✓ |

In the same way, one defined the outputs of the control part with respect to the operative part, as follows:

| Designation | Reference name | Source | |
|---|---|---|---|
| | | Operator | Device |
| MACHINING | S0 | ✓ | ✓ |
| WAITING 1 | S1 | ✓ | |
| WAITING 2 | S2 | ✓ | |
| LAYING | S3 | ✓ | ✓ |

| | | | |
|---|---|---|---|
| HOLD | S4 | ✓ | ✓ |
| ASSEMBLY | S5 | ✓ | ✓ |

## Definition of the inputs/outputs

A grafcet specifies the behaviour of a control part, compared with an operative part. The interaction between the two parts is done thanks to the inputs/outputs.

The control part receives information from type “All or Nothing” (AoN) coming from the operative part. This information constitutes the inputs of the control part. The control part (which behaviour is specified by a grafcet) works out a whole of signals (also of AoN type) intended for the operative part, this whole of signals constitutes the outputs of the control part.

Thus, it is necessary to define these sets of boolean variables, also called respectively vector of inputs and vector of outputs. These definitions will be made in an instance of the template VI ' **GrafcetVIEW singleloop.vit**' (in library *APPLI.LLB*); it will constitute the base of our final application.

When you open the VI ‘**GrafcetVIEW singleloop .vit**’, a copy of this VI is automatically created.

## Inputs

An input must be a boolean variable, it can come from two sources: the operator interface (the front panel of one VI) or of outside (for example of an acquisition device, a serial connection, or of a network... it is the interface proceeded). In the case of an acquisition device, it is often necessary to convert a numerical or alphanumeric value into boolean values.

**Build Array**

In practice, to define the vector of inputs, it is just enough to assemble each one of its components using the function ' **Build Array** '.

The order of the components is significant, because it makes it possible to index the inputs (the notation ***Ei*** will indicate the ith component of the vector of inputs).

***Example***

The inputs ‘**Start/Stop production line**’, ‘**Start/Stop assembly line**’, ‘**Manual/Automatic**’ and ‘**Hold**’ are inputs coming from the graphic interface (operator). The inputs coming from port *0* of the

DIO-96 card (acquisition card having 96 digital channels divided into 12 ports of 8 lines) are material inputs (coming from outside). The VI ‘**Read digital port**’ produces an **U32** integer; we convert it into a vector of boolean by using the function ‘**Number to Boolean array**’. Finally, we only keep four first components of this vector, by using the function ‘**Subset array**’.

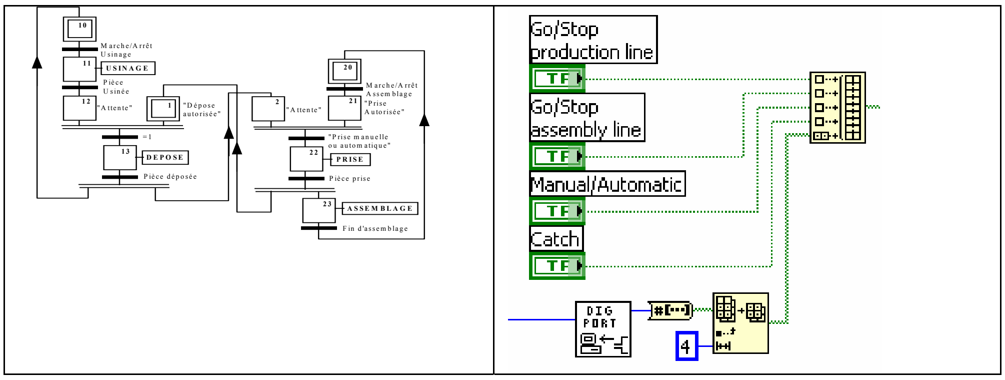
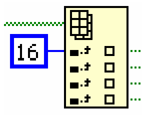


**Figure 4. Inputs**

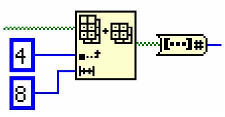
## Outputs

An output is a boolean variable, it can be redirected to two different destinations: to the operator interface (to a boolean indicator) or to the outside (acquisition card, serial connection, network,).



**Index Array**

We can read different components of a 1D vector of boolean by using the function ‘**Index array**’ from the menu ‘**Array & Cluster**’.



We can also use the function ‘**Array Subset**’ from the menu ‘**Array & Cluster**’ to read a part of a vector, for example, to be converted to an integer by using the function ’**Boolean Array to number**’ to be write on a port of an acquisition card.

The order in which we recover these components has an importance because it is in this order which we will make reference to the outputs in the grafcet (the notation $\mathbf{S_i}$ will indicate the i$^{th}$ component of the vector of outputs).

***Example:*** The first six outputs (‘**Machining**’, ‘**Waiting1**’, ‘**Waiting2**’, ‘**Laying**’, ‘**Hold**’, ‘**Assembly**’) are software outputs (intended to inform the operator on the order given to the operative part). Only the outputs 0, 3, 4 and 5 are material outputs (they are the order given to the operative part). They are formatted ('**boolean array to number** ') and are directed to port 1 of the Dio-96 card.

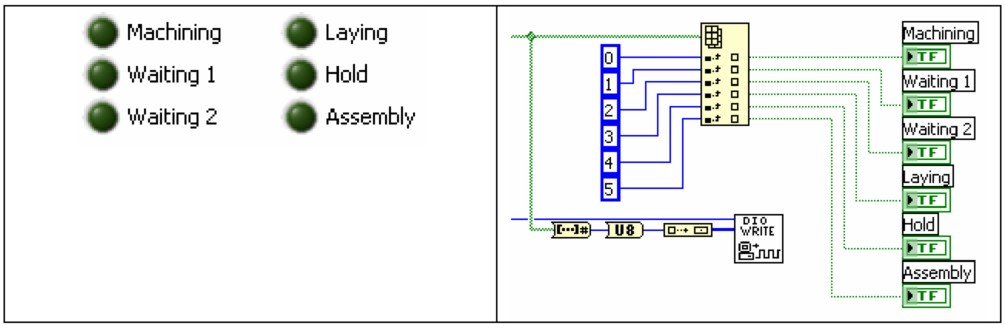
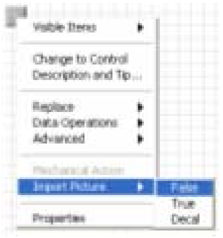


**Figure 5. Outputs**

## Customize inputs/outputs



LabVIEW allows customizing controls and indicators, by using the controls editor (cf. chapter 22 of the handbook of LabVIEW reference).

Thanks to this mechanism, it becomes possible to have a very realistic representation of the controlled process. It is enough to import the images representative of the two states to an exit.

***Example*** The Boolean ‘**Hold**’ has a representation an arm at rest in the false state and an arm taking a part in the true state. Thus the slackened and supported representation become respectively: arm at rest and arm taking a part.

## Edition of a grafcet

The edition of a new grafcet specifying the operation of the control part of the system which we wish to control is done in new VI (**File»New VI**).

## The menu « Functions GrafcetVIEW »

We lay out in the diagram, the VIs constituting the grafcet using the menu *GrafcetVIEW* (appeared in the menu **function** thanks the library **G7VIEW**.**LLB** of the folder **G7VIEW**.**LIB** located under the LabVIEW folder).

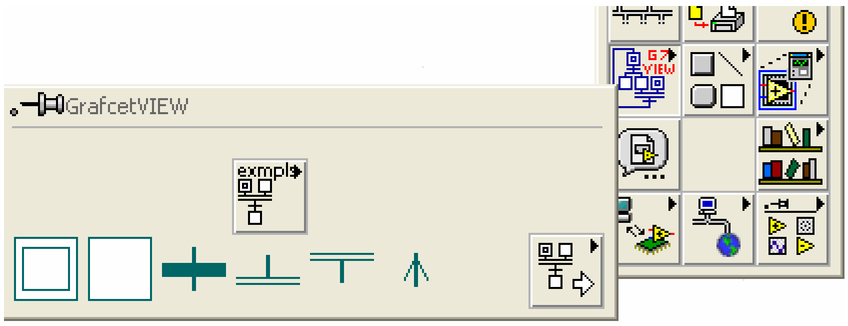
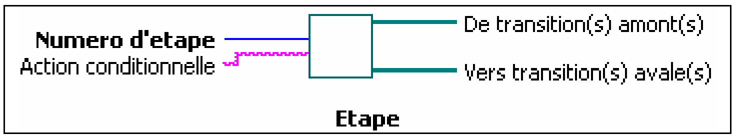


**Figure 6. Menu GrafcefVIEW**

## Steps

### Step

This VI allows representing the steps of the GRAFCET, it must be numbered (thanks to a numerical constant of integer type 32 bits), and you can associate with it a conditional action (thanks to an alphanumeric constant of *string* type).



**I32** **Step number** specifies the number of the step. The number is essential and must be single.

**abc** **Conditional action** specifies the conditional action associated with the steps. Its syntax and its semantics are clarified here after.

**From transitions upstream** is connected to the transitions upstream (which activate the step).

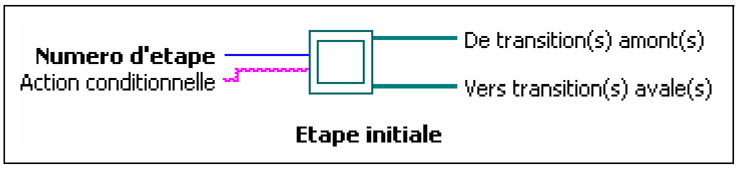
**To transitions downstream** is connected to the transitions downstream (which are validated by the step).

***This VI is implicitly directed from the top to the down.***

It is possible to place the numerical constant numbering the steps on the VI '**step'** (the number is then seen in the step).

## Initial step

This VI allows representing the initial steps of the GRAFCET. It must be numbered (thanks to a numerical constant of the type *Unsigned Integer 32 bits*). You can also associate with it a conditional action (thanks to an alphanumeric constant of *string* type).



**Step number** specifies the number of the step. The number is essential and must be single.

**Conditional action** specifies the conditional action associated with the step. Its syntax and its semantics are clarified here after.

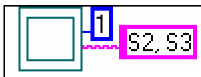
**From transitions upstream** is connected to the transitions upstream (which activate the step).

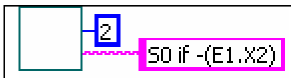
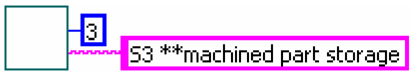
**To transitions downstream** is connected to the transitions downstream (which are validated by the step).

***This VI is implicitly directed from the top to the bottom.***

## Syntax of the conditional actions

A conditional action breaks up into two parts, the action and the condition, separated by the reserved word "if". It is also possible to associate comments to each step of the GRAFCET.

| | |
|---|---|
|  | The *action* is made up of one or several outputs, separate then by commas (,). If the action is empty then, no action is associated with the step. |

| | |
|---|---|
|  | The *condition* can exist only if the associated action is none empty. It is a logical expression being expressed using the operators "." (operator **and**),"+"(operator **or**) and "-"(operator **not**), of the operands "$\mathbf{E_i}$", "**Xi**" and "**ti/Xj/tk**", as well as brackets "**(**"and"**)**". |
|  | The *comments* are located at the end of the conditional action. They are defined by the operator "**". When the step is active, the comment is returned on the output *comments* of VI **G7-Calculation Easy.vi**. |

The notation "**MSi**" indicates the memorizing of the output "**Si**" to the true state, the notation "**DSi**" indicates the passage of the output "**Si**" to the false state (Set and Reset).

The notation « **ISi** » indicates that the grafcet should generate a pulse on the output "**Si**"

## Semantics of the conditional actions

An action associated with a step is revalued only in the stable situations of the grafcet considered (evolution with search for stability).

When the step carrying the conditional action is active then each associated output takes the true value as long as the condition is true and that the step remains active (not memorized action).

If several steps activate the same output then the value of this output is the value of disjunction between the outputs of the various steps.

**Ei** has the value of the i$^{th}$ component of the vector of inputs.

**Xi** is the variable of step associated with the step **i**, it is true if step **i** is active.

**t1/Xj/t2** is a temporization associated with the step variable **j**, where **t1** and **t2** are literal integer values. This temporization takes true value **t1** ms after the activation of the step **j** until **t2** ms after the desactivation of the step **j**.

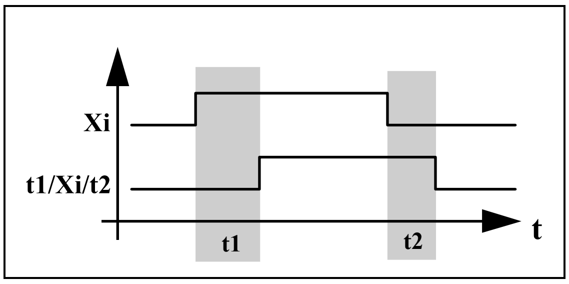
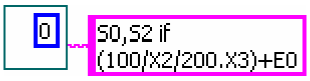


**Figure 7. Conditional action : temporization**

***Example :***



The outputs named **S0** and **S2** (index 0 and 2 in the vector of outputs) take the value true when:

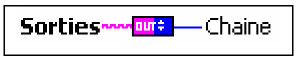
- The situation reached is stable
- *And* the step 0 is active
- *And if* **(**the step 2 is active since more 100ms or inactive since less 200ms, and step 3 is active) *or* (the E0 input is true).

## Design of the conditional actions

GrafcetVIEW also places at the disposal a series of tools allowing generating the conditional actions associated with the various steps of the grafcet. These tools are accessible from the function pallet **GrafcetVIEW**.

## Conversion of the names of the outputs

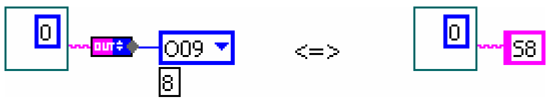
This VI allows generating the name of outputs **Si** from their index **i**.



I32 **String** specifies the index **i** of the output **Si** to use

abc **Output** name of the output associated with the parameter **String**

***Example*** :

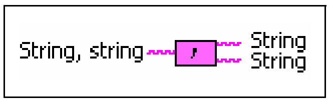


The use of his VI makes it possible to generate the name of an output in the form "**Si**" using an enumeration strictly typified, containing the list of the outputs of the system.

***The value associated with an output in the enumerated type must correspond to index "i" of this one.***

## Concatenation of outputs

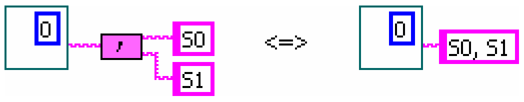
This VI allows to concatenate 2 outputs **Si** and **Sj**.



**String** specifies the name of the outputs to be concatenated

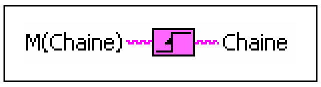
**String, string** contains the name of the concatenated outputs.

***Example:***



## Forcing the activation of an output

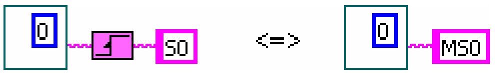
This VI allows forcing the memorizing of the output **Si** to the value true.



**String** specifies the name of the output which we want to force the memorizing of the state to true.

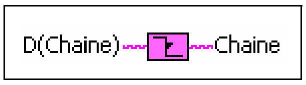
**M(String)** forcing of the memorizing of the state of the output to true.

***Example :***



## Forcing the desactivation of an output

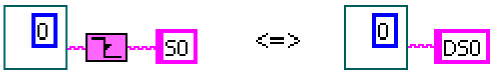
This VI allows forcing the memorizing of the output **Si** to the value false.



**String** specifies the name of the output which we want to force the memorizing of the state to false.

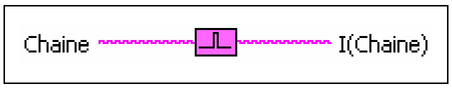
**D(String)** forcing of the memorizing of the state of the output to false.

***Example***:



## Generating Pulse

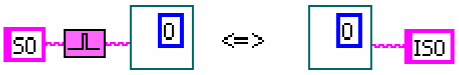
This VI allows generating a pulse on the output "**Si**".



**String** specifies the name of the output on which we wish to generate a pulse.

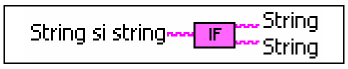
**I(String)** generates a pulse on the output **Si**.

***Example:***



## Conditioning the action

This VI allows defining the condition to which the action could be carried out.

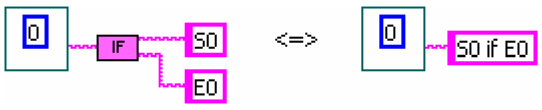


**String** specifies the action to be carried out.

**String** specifies the condition to which the action could be carried out

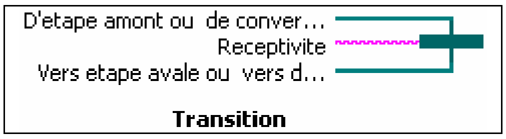
**String if string** conditional action

***Example***:



## The transition

This VI makes it possible to represent a *transition* of the grafcet. It is possible to associate *receptivity* to it (thanks to an alphanumeric constant of *string* type).



**From step upstream or convergence 'in and'** is connected to the step upstream which validates the transition or to a convergence *'in and* ', or of nothing in the case of a transition source.

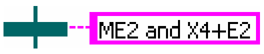
**To step downstream or divergence 'and'** is connected to the step downstream (which are activated by the transition) or to a divergence *'in and* ', or of nothing in the case of a transition well.

**Receptivity** specifies the receptivity associated with the transition. Its syntax and its semantics are clarified here after.

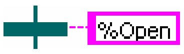
⚠ ***This VI is implicitly directed from the top to the down.***

### Syntax of a receptivity

In GrafcetVIEW, a **receptivity** breaks up into two parts, the event and the condition, separated by the reserved word *and*.



**The event** is either a front going up (noted "**M**"), or a front going down (noted "**D**") of the variables "**Ei**", "**Xi**", "**t1/Xj/t2**". The event can be always occurrent, in this case it is noted **e** (or not noted). Thus an event is written "**MEi**", "**DEi**", "**MXi**", "**DXi**", "**Mt1/Xj/t2**", "**Dt1/Xj/t2**", "**e**" or anything (equivalent to **e**).

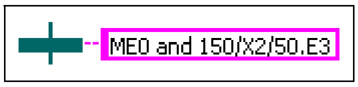


**The event** can also be an order (noted "%order"), sent to the grafcet by the *Order* input of the VI **G7-Calculation Easy.vi**

⚠ ***in the case of an order, the event can not be followed by a condition***.

## Semantics of the receptivities

A transition will be crossed if it is validated (All its *steps upstream* are active), if the associated event arrived (always in the case of **e**) and if the associated condition is true.



This transition will be crossed when:

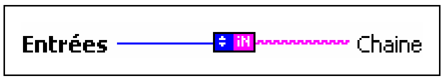
- All its *steps upstream* are active
- *and* the **E0** input passes from the false state to the true state (front going up of **E0**)
- *and* (step 2 is active since at least 150ms or inactive since less 50ms) *and* (the **E3** input is true).

## Tools of design of the receptivities

Just like for the conditional actions, GrafcetVIEW places at the disposal a series of tools making it possible to generate the receptivities associated with the various transitions from the grafcet. These tools are accessible from the function pallet **GrafcetVIEW**.

## Conversion of the names of the inputs

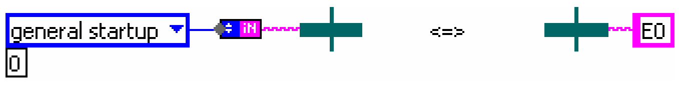
This VI allows generating the name of inputs **Ei** using their index **i**.



**Inputs** specifies the index **i** of the input **Ei** to be used.

**String** name of the input with index **i**.

***Example*** **:**

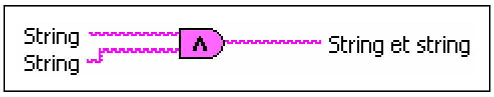


The use of his VI makes it possible to generate the name of an input in the form "**Ei**" using an enumeration strictly typified, containing the list of the inputs of the system.

***The value associated with an input in the enumerated type must correspond to index "i" of this one.***

## Operator 'AND'

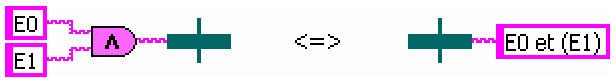
This VI carries out a logical '**AND'** between the two inputs **Ei** and **Ej**.



**String** operands of the operator **AND**

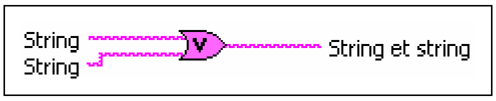
**String and string** results of logical **AND** of the two **String** inputs

***Example:***



## Operator 'OR'

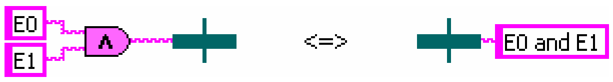
This VI carries out a logical '**OR'** between the two inputs **Ei** and **Ej**.



**String** operands of the operator **OR**

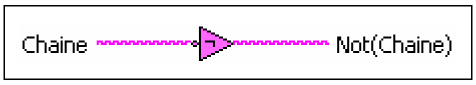
**String or string** results of logical **OR** of the two **String** inputs.

***Example:***



## Operator 'NOT'

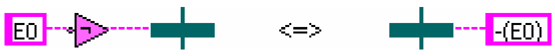
This VI carries out a logical '**NOT**' of the input **Ei**.



**String** operands of the operator **NOT**

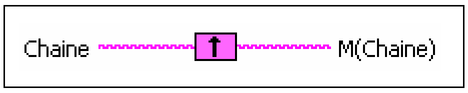
**String** results of logical **NOT** of the input **String**.

***Example***:



## Operator 'Front Going Up'

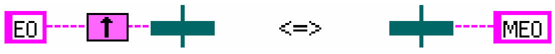
This VI detects the **fronts going up** on the input **Ei**.



**String** operand of the operator **Front Going Up**.

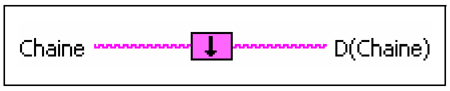
**String** detects fronts going up on the input **String**.

***Example:***



## Operator 'Front Going Down'

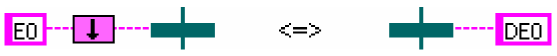
This VI detects the **fronts going down** on the input **Ei**.



**String** operand of the operator **Front Going Down**.

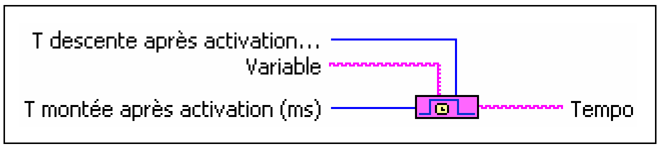
**String** detects fronts going down on the input **String**

***Example:***



## Operator 'Temporization'

This VI generates a temporization **t1/Xi/t2** associated with the step variable **i** (see chapter **Semantics of the conditional actions**).


T descente après activation...
Variable
T montée après activation (ms)
Tempo

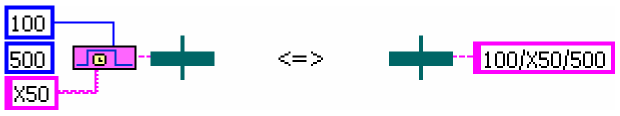
**T descent after activation (ms)** indicates the time passed between the activation of the step **Variable** and the activation of the receptivity.

**Variable** name of the step with which temporization is associated

**T gone up after activation (ms)** indicates the time passed between the desactivation of the Step **Variable** and the desactivation of the receptivity.

**Tempo** indicates the state of the temporization associated with the step **Variable**.

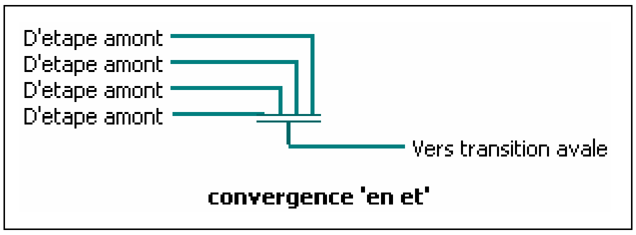
***Example:***



## Convergences and divergences

### Convergence ' in and'

This VI makes it possible to represent convergence '**in and**' of the GRAFCET. We can connect to it up to four steps upstream. It is possible to put several specimens of these VI in cascade if we wish to connect more than four steps.



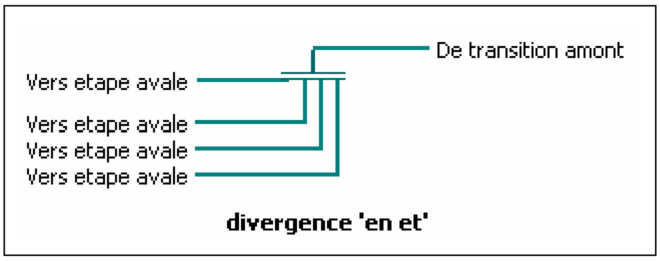
**From step upstream** is connected to a step upstream (which validates the transition).

**To Transition Downstream** is connected to the transition downstream (which is validated by the stages).

⚠ ***This VI is implicitly directed from the top to the bottom.***

## Divergence 'in and'

This VI allows representing the divergence '**in and**' of the GRAFCET. We can connect to it up to four steps downstream. It is possible to put several specimens of these VI in cascade if we wish to connect more than four steps.



**To step downstream** is connected to a step downstream (which is activated by the transition).

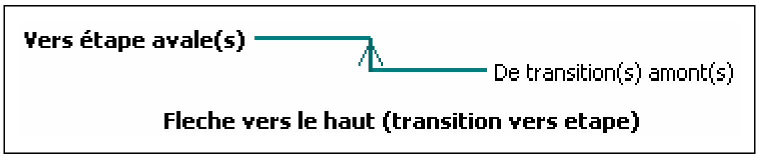
**From transition upstream** is connected to the transition upstream (which activates the steps).

⚠ ***This VI is implicitly directed from the top to the down.***

Convergences and divergences '**in or**' do not have a representation by VI (they are done directly thanks to wiring by the LabVIEW wiring tool).

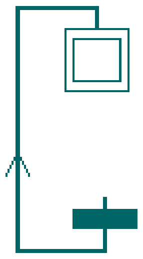
## Arrows upwards

This VI allows displaying rising arrows on the arcs. The VIs *step*, *initial step*, *transition*, *convergence* and *divergence* '**in and**' being implicitly directed from the top to the bottom. The use of this VI is not obligatory, but makes it possible to make appear in an explicit way the implicit orientation of the rising arcs.


Vers étape avale(s)
De transition(s) amont(s)
Fleche vers le haut (transition vers etape)

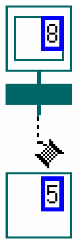
**To step upstream** is connected to a step upstream (activated by the transition) or to a **divergence** '**in and** '.

**From transition downstream** is connected to the transitions downstream (which activate the step).

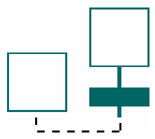


⚠ ***The low connection of this VI must go to the low connection of a transition and high connection must come from the high connection of a step or a divergence ' in and '.***

## Directed arcs



Once the objects (*steps*, *transitions*, etc...) placed in the diagram, it does not remain any more that to connect them between them using the tool winds: it is this link which constitutes the directed arcs between the various entities. The GrafcetVIEW entities being implicitly directed from the top to the bottom, the arcs are thus directed from the entity upstream to the entity downstream. In order to return clarifies the rising arcs, we can insert one VI '**arrow upwards**' (this VI will receive obligatorily its input from a step or a divergence '**in and**' and will emit its outputs to one or more transitions).



⚠ ***GrafcetVIEW does not authorize to place entities other than the VI 'arrow upwards' on a link going up (under penalty of having a broken bond).***

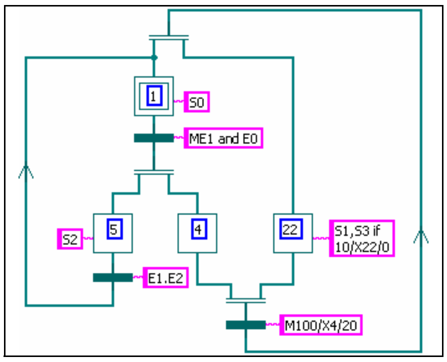
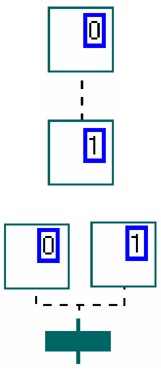


**Figure 8. Example of a grafcet**

## Syntactic analysis of grafcets

### Graphic syntactic analysis



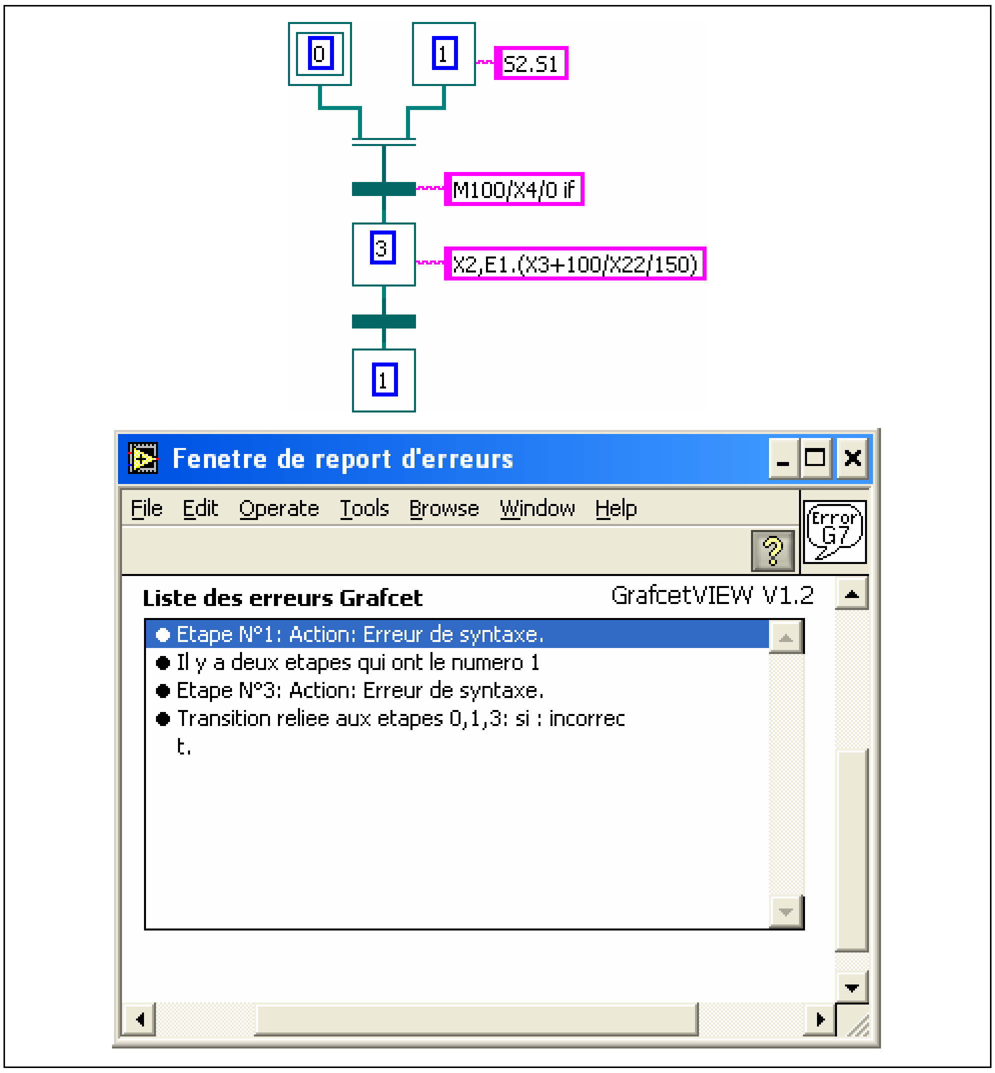
GrafcetVIEW benefits from the properties of the graphic editor of LabVIEW, thus thanks to the typing of the arcs between entities, the correction of the graphic syntax of a grafcet is immediate: if an arc is into dotted then one of the rules of constructions of the GRAFCET is not respected (not respect of alternation step-transition, several steps are connected directly to the same transition, etc...)

The appearance of the broken arrow (VI no achievable) indicates a syntactic error of the grafcet in the course of edition.

### Analyze conditional actions and receptivities

If the graphic syntax of the grafcet is correct, it also should be checked that several steps do not have the same number, that there is no transition not connected to a step, that the conditional actions and the receptivities are correct.

This analysis is realized by executing the VI which contains the published grafcet. If an error occurs in the grafcet, a VI 'window of error report' displays the list of errors.


0
1
S2.S1
M100/X4/0 if
3
X2,E1.(X3+100/X22/150)
1
Fenetre de report d'erreurs
File Edit Operate Tools Browse Window Help
Error G7

Liste des erreurs Grafcet GrafcetVIEW V1.2

- Etape N°1: Action: Erreur de syntaxe.
- Il y a deux etapes qui ont le numero 1
- Etape N°3: Action: Erreur de syntaxe.
- Transition reliee aux etapes 0,1,3: si : incorrec
  t.

**Figure 9. Error dialog box**

## Creating sub-grafcets

One of the properties of the LabVIEW language is the possibility of encapsulating a diagram to be able to re-use this portion of code or quite simply to abstract a functionality from a portion of a diagram (to make more clearly the diagram principal).

GrafcetVIEW proposes the same mechanism: it is possible to encapsulate a portion of a grafcet, to treat on a hierarchical basis the final grafcet. This mechanism is similar to the *macro-step* mechanism, although more general because the encapsulated grafcet can have several inputs, several outputs, of different types (steps or transition). However only the *macro-step* is recognized by standard CEI 848.

Thus, it will be necessary to define the inputs/outputs of the sub-grafcet which we wish to create. To do it, we have controls and indicators provided in the menu **Controls** GrafcetVIEW. These objects are connected upstream to the entities specified in their name (for example the control '**From transition upstream(s).ctl**' will make it possible to connect a transition upstream of the sub-grafcet.

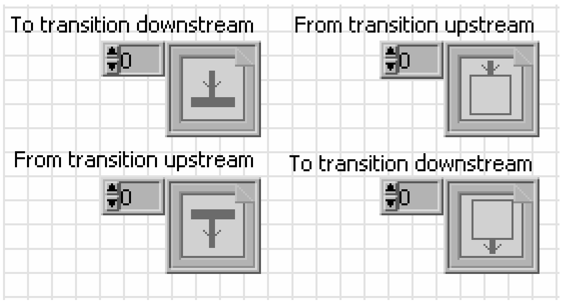


**Figure 10. GrafcetVIEW controls**

Then you have to edit the sub-grafcet (similar to any edition of normal grafcet). In the following example, we chose to abstract the function from drilling of a milling machine. Then, we will represent it sub-grafcet by only one step **'to drill** '.

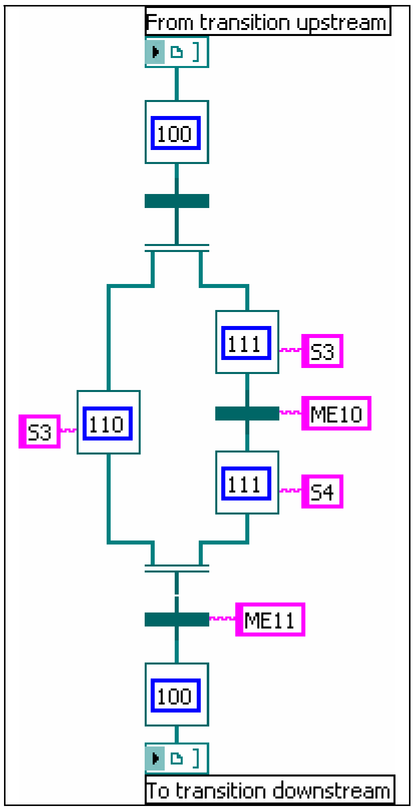
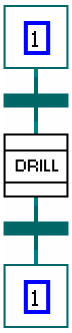
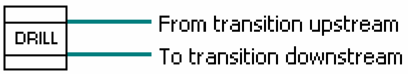


**Figure 11. Sub-GRAFCET**



Once the sub-grafcet created, it any more but does not remain to connect its controls/indicators on the terminations of the associated icon. It is then enough to point out it under-VI inside the principal grafcet.



In the GRAFCET specifying the behaviour of the control part of our workshop, the integration of it sub-grafcet (which here is a *macro-step*) gives the following result:

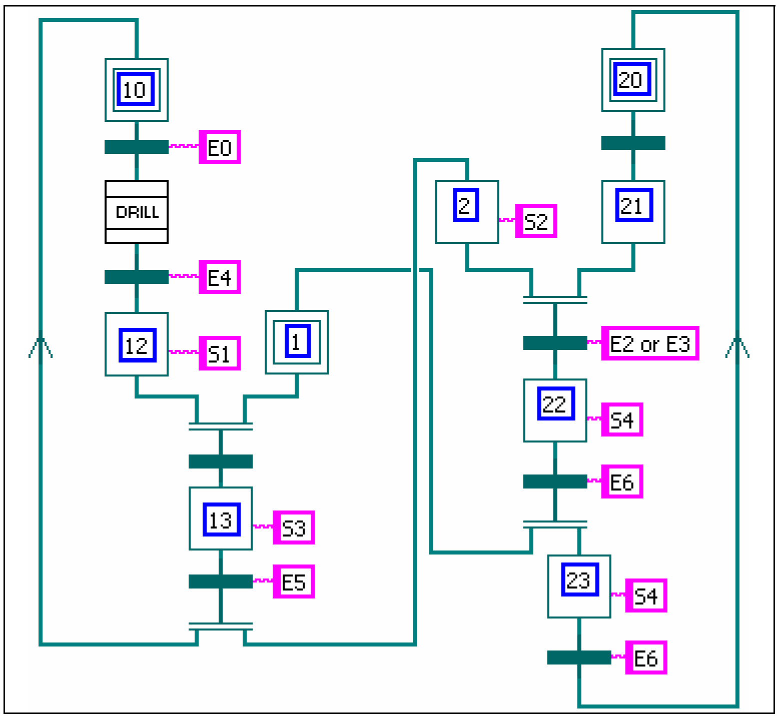


**Figure 12. Main Grafcet**

⚠ ***If we wish to re-use several times the same sub-grafcet, it is necessary to put the classification of the steps like parameter of the sub- grafcet (if not the principal grafcet will see several steps with the same number of step). In the same way, if the conditional actions associated at the steps are different (or receptivities associated with the transitions) from an instance of the same sub-grafcet with another, it is necessary to pass these conditional actions or receptivities like parameters.***

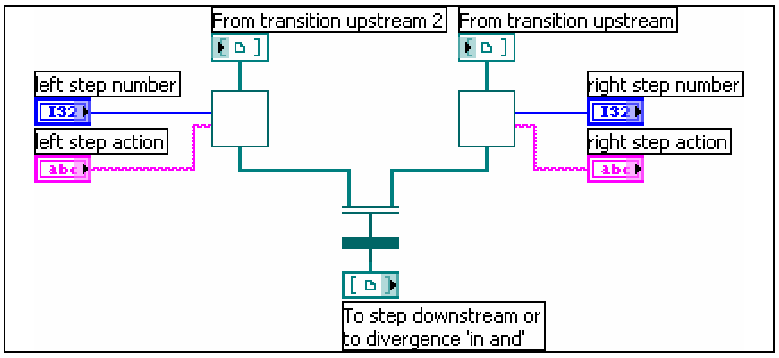


**Figure 13. Several calls to the same sub-grafcet**

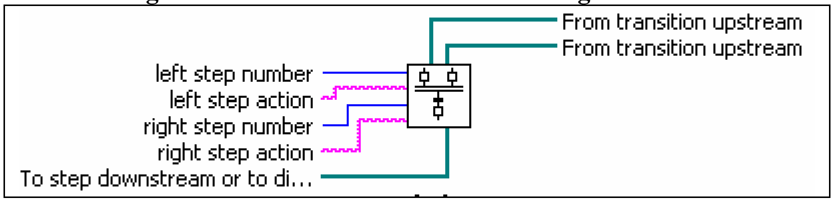


**Figure 14. Creating a sub-grafcet**

## Grafcets no related

Nothing prohibits the creation of not related grafcets (without directed arcs to connect them). In particular small separated grafcets can possibly be used as memorizing... They can quite naturally be referred by step variable.

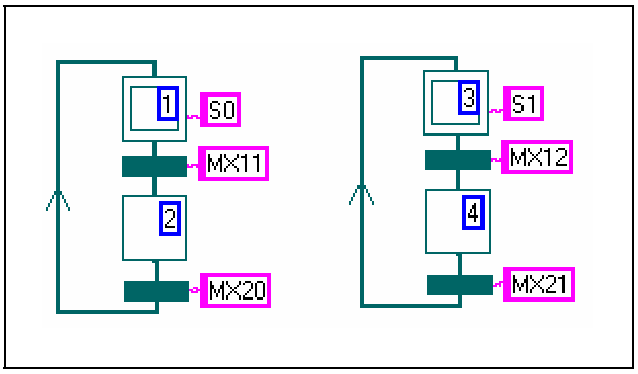


**Figure 15. Example of not related grafcets**

## Creation of the application

When the inputs/outputs of our control part were defined and that the grafcet specifying its behaviour was published, it remains to integrate these various elements in the final VI, one which will make live to our application.

At the time of the edition of the inputs/outputs, we have duplicated the VI '**GrafcetVIEW Singleloop**'. It is in this copy that we will create the final application. In the diagram of this VI, in addition to the inputs/outputs, we find the following structures:

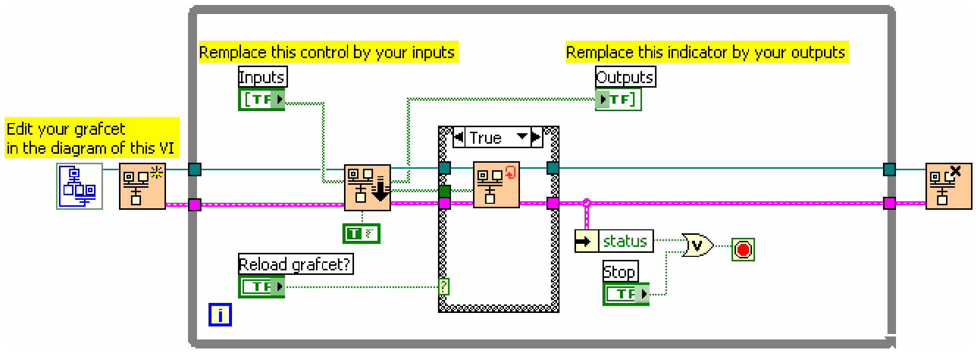

Remplace this control by your inputs
Remplace this indicator by your outputs
Inputs
Outputs
Edit your grafcet
in the diagram of this VI
True
status
Stop
Reload grafcet?

**Figure 16. Skeleton of the final application**

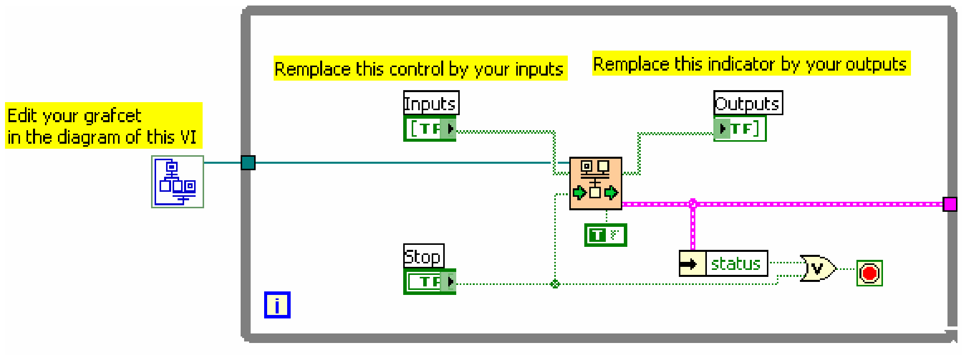

Remplace this control by your inputs
Remplace this indicator by your outputs
Edit your grafcet
in the diagram of this VI
Inputs
Outputs
Stop
status

**Figure 17. Skeleton of the final application (Easy)**

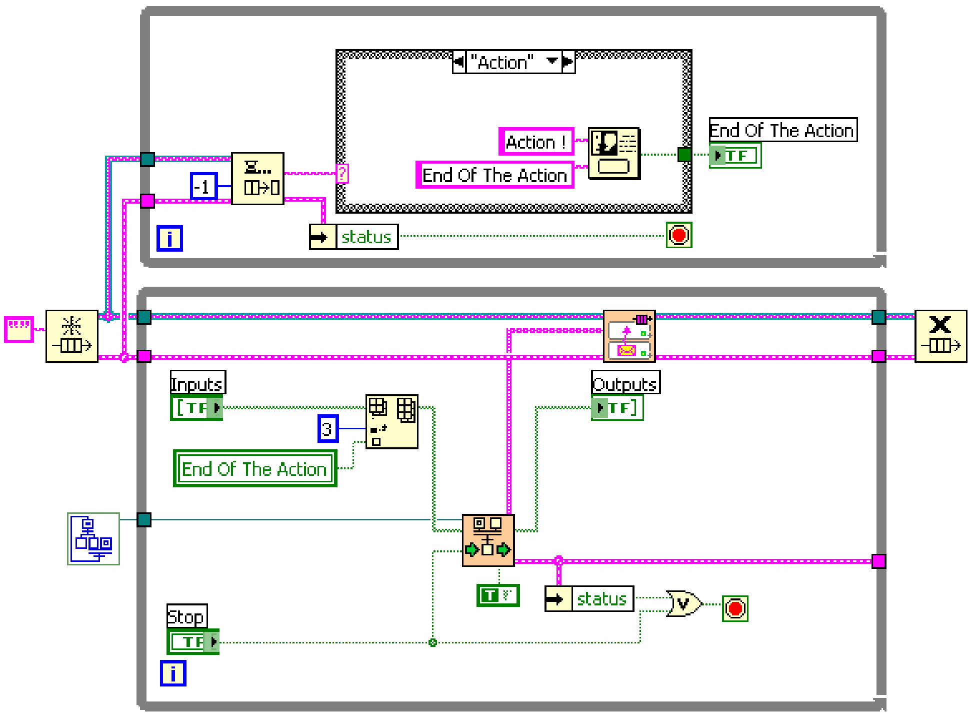


**Figure 18. Skeleton of the final application, hybrid**

This copy of the VI will be used as groundwork: we will complete each structure to construct the final application.

## Choice of the execution model

The user has the choice between two GrafcetVIEW models: Single loop or hybrid Single loop.

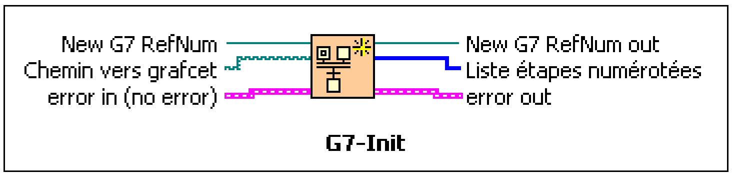
### GrafcetVIEW Single loop

In the case of Single loop (see figure 17 and figure 18), the inputs and the outputs belong to the same iterative structure. In this case, a new evaluation of the inputs is possible only when the outputs were calculated.

This model of execution is appropriate in the majority of applications.

## Initialization of GrafcetVIEW

This VI loads and initializes the published grafcet. This VI makes it possible to dynamically load the grafcet to be executed, passing to it the path of the corresponding VI.



**New G7 RefNum**. Reference to the grafcet to be executed. If no reference is passed to the VI, this one dynamically loads the grafcet which the path is passed in parameter.

**Grafcet Path**. Path to the VI containing the grafcet to be loaded dynamically and to be executed.

**Error in (no error)**. Indicates if an error occurred before the execution of the VI. If an error occurred before the execution of this VI, it is not carried out.

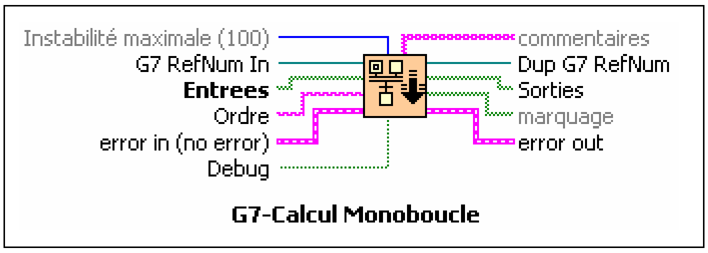
**New G7 RefNum out**. Reference to the grafcet to be run.

**List of numbered steps**. Returns the list of the steps numbers which constitute the grafcet.

**Error out**. If the cluster **error in** indicates an error, **error out** defers the same information. If an error occurs during the execution of the VI, **error out** defers this error.

## Execution Engine of GrafcetVIEW

This VI makes it possible to execute the grafcet which the reference is passed in parameter. It determines the state of the outputs of the system controlled using the state of the inputs read.



**G7 refnum in**. Reference the grafcet to be run.

**Inputs**. Vector of the inputs of the controlled system. The **i**$^{\mathbf{th}}$ component of this vector corresponds to notation **Ei** in the published grafcet.

**Order**. Order to be sent to the grafcet. The orders sent in this manner correspond to the receptivities "%Order".

**Maximum instability**. Maximum calculation of evolution of the grafcet for a given variation of the vector of the inputs of the system. If this value is exceeded, the grafcet is considered unstable and an error is generated.

**Error in (no error)**. Indicates if an error occurred before the execution of the VI. If an error occurred before the execution of this VI, it is not carried out.

**Dup G7 refnum**. Reference the grafcet to be run.

**Outputs**. Vector of the outputs of the controlled system. The **i**$^{\mathbf{th}}$ component of this vector corresponds to the notation **Si** in the published grafcet.

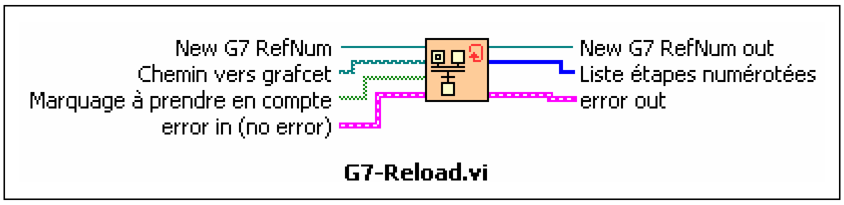
**Marking**. State Vector of the published grafcet. This vector is ordered by order of increasing number of step.

**Comments**. Vector containing the list of the comments associated with the active steps of the grafcet (noted "** *Comment*" in the published grafcet).

**Error out**. If the cluster **error in** indicates an error, **error out** defers the same information. If an error occurs during the execution of the VI, **error out** defers this error.

## Reloading of the grafcet

This VI reloads the grafcet which the reference id passed in argument. That allows taking in account the modifications of the grafcet carried out in a dynamic way.



**New G7 RefNum**. Reference to the grafcet to be reloaded. If no reference is passed to the VI, this one dynamically loads the grafcet which the path is passed in parameter.

**Grafcet Path** (Optional). Path to the VI containing the grafcet to be loaded dynamically and to run.

**Marking to be taken in account.** State vector of the published grafcet. This vector contains the state in which the grafcet was before reloaded.

**Error in (no error)**. Indicates if an error occurred before the execution of the VI. If an error occurred before the execution of this VI, it is not carried out.

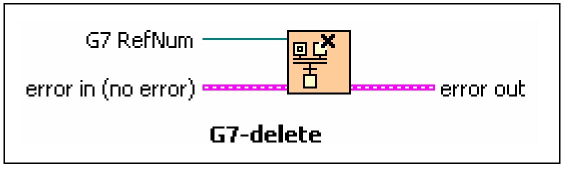
**New G7 RefNum out**. Reference to the grafcet to be run

**List of numbered steps**. Returns the list of the numbers of the steps which constitute the grafcet.

**Error out**. If the cluster **error in** indicates an error, **error out** defers the same information. If an error occurs during the execution of the VI, **error out** defers this error.

## Shutdown of the execution of GrafcetVIEW

This VI stops the execution of the grafcet and closes the reference to this one.



**G7 refnum in**. Reference to the grafcet to be stopped.

**Error in (no error)**. Indicates if an error occurred before the execution of the VI.

**Error out**. If the cluster **error in** indicates an error, **error out** defers the same information. If an error occurs during the execution of the VI, **error out** defers this error.

## Construction of the application

The integration of the inputs/outputs of our application to the groundwork 'GrafcetVIEW Single loop' gives:

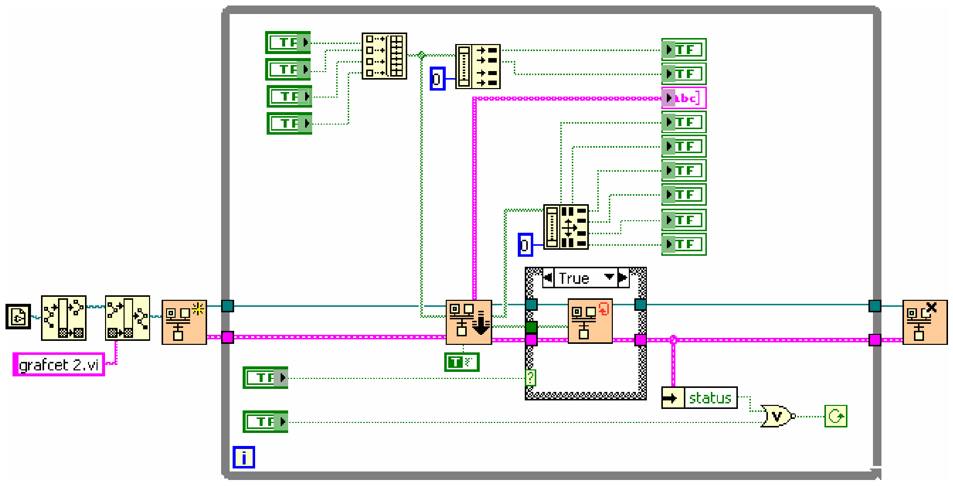
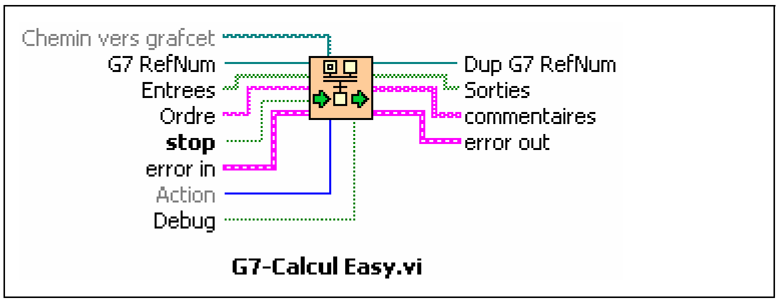


**Figure 19. Example of a single loop application**

## Use of the Engine of execution "high level"

This VI allows running the grafcet which the reference is passed in parameter, without considerer the initialization and stop steps. It calculates the state of the outputs of the system controlled using the state of the inputs read.



**G7 RefNum**. Reference to the grafcet to be run. If no reference is passed to the VI, this one dynamically loads the grafcet which the path is passed in parameter.

**Grafcet Path**. Path to the VI containing the grafcet to be loaded dynamically and to be executed.

**Inputs**. Vector of the inputs of the controlled system. The **i**th component of this vector corresponds to notation **Ei** in the published grafcet.

**Order**. Order to be sent to the grafcet. The orders sent in this manner correspond to the receptivities "%Order".

**Stop** asks the stop of the execution of the grafcet which the reference is passed in parameter.

**Action** allows modifying the behaviour of the grafcet.

| | |
|---|---|
| **Init** | Loads the grafcet in memory and initializes it |
| **Execute** (default) | Executes the grafcet which the reference is passed in argument. At the time of the first call to the VI, the grafcet is automatically loaded in memory. |
| **Reload** | Reload and reinitialize the grafcet. |
| **Reload - keep in progress marking** | Reloads the grafcet in its preceding state |
| **Stop** | Stop the execution of the grafcet |

**Error in (no error)**. Indicates if an error occurred before the execution of the VI. If an error occurred before the execution of this VI, it is not carried out.

**Dup G7 RefNum**. Reference to the grafcet to be run.

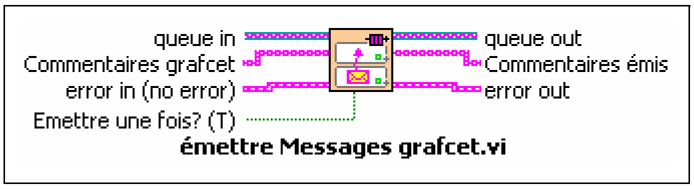
**Outputs**. Vector of the outputs of the controlled system. The **i**$^{th}$ component of this vector corresponds to the notation **Si** in the published grafcet.

**Comments**. Vector containing the list of the comments associated with the active steps of the grafcet (noted "** *Comment*" in the published grafcet).

**Error out**. If the cluster **error in** indicates an error, **error out** defers the same information. If an error occurs during the execution of the VI, **error out** defers this error.

## GrafcetVIEW Single loop hybrid

GrafcetVIEW places at the disposal one VI making it possible to transmit orders to a loop being executed in parallel of the grafcet (figure 19).



**queue in**. Reference to the queue used to send orders to the parallel loop.

**Comments**. Orders coming from the grafcet to be sent to the parallel loop.

**To emit once? (T)**. Indicates if the VI sends the order one or more times.

**Error in (no error)** indicates if an error occurred before the execution of the VI. If an error occurred before the execution of this VI, it is not carried out.

**queue out**. Reference to the queue used to send orders to the parallel loop.

**Emitted comments**. Orders coming from the grafcet to be sent to the parallel loop.

**Error out**. If the cluster **error in** indicates an error, **error out** defers the same information. If an error occurs during the execution of the VI, **error out** defers this error.

## Integration of the published grafcet

It finally remains to integrate the grafcet published previously in the diagram. We will incorporate the VI containing the specification grafcet of our system into the application (see figure 20).

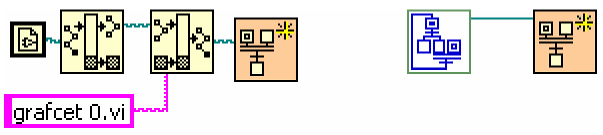


**Figure 20. Integration of the GRAFCET**

## Tools of development of the grafcets

Once the grafcet was created and if there is not error of syntax (arrow of white execution), you can execute the application containing the published grafcet. During its execution, GrafcetVIEW posts a dialog box placing at the disposal a certain number of tools making it possible to control the evolution of the published grafcet.

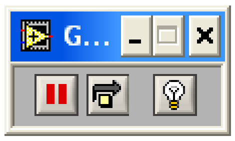


**Figure 21 : Execution pallet of GrafcetVIEW**

### Button Pause

Click on the **Pause** button to suspend the execution of the published grafcet. The grafcet is then fixed in the state in which it was before the setting in pause of the execution. Click once again on the **Pause** button to continue the normal execution of the grafcet.

When the execution of the published grafcet is in pause, GrafcetVIEW posts a new button making it possible to execute the grafcet step by step mode.

## Step by Step mode

Click on the button **Step by Step mode** to pass to the next state and to return in the pause mode.

⚠ The execution in **step by step mode** respects rules of evolution of the grafcet (see Rules of evolution of a grafcet).

The **step by step mode** button is visible only when the execution of the published grafcet is in pause.

## Highlighting execution

Click on the **Highlighting execution** to view all activated step at a given moment. The activated steps are surrounded with a red square. Click again on the button to return in a normal execution mode.

Execution highlighting greatly reduces the speed at which the VI runs.

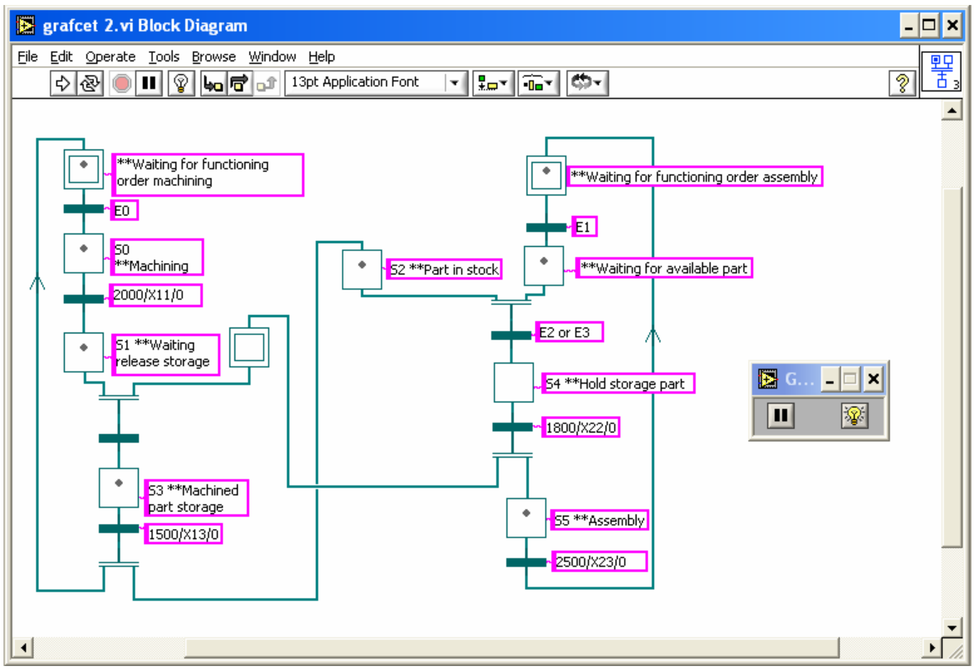

grafcet 2.vi Block Diagram
File Edit Operate Tools Browse Window Help
13pt Application Font
**Waiting for functioning order machining
E0
S0 **Machining
2000/X11/0
S1 **Waiting release storage
S3 **Machined part storage
1500/X13/0
S2 **Part in stock
**Waiting for functioning order assembly
E1
**Waiting for available part
E2 or E3
S4 **Hold storage part
1800/X22/0
S5 **Assembly
2500/X23/0
G...

**Figure 22 : Highlighting execution mode example**

## Example

Throughout this handbook, we illustrated the various phases (definitions of the inputs/outputs, edition of the grafcet) for a concrete example. It is about the control of a production line (simplified) using an acquisition card with 96 numerical inputs/outputs (PC-DIO 96 by Instruments® National).

This application, with a simulation of the operative part, is available in bookshop 'EXEMPLES.LLB' under the name 'EXEMPLE 2'.

✍ We will find, thanks to this example, a manner of testing our application directly. In EXEMPLE2, times of machining, assembly take and hold are simulated by temporizations.

The graphic interface of our application was made to represent as well as possible the controlled system (personalization of the Boolean indicators manufacture, deposits, waiting 1 and 2, take and assembly.

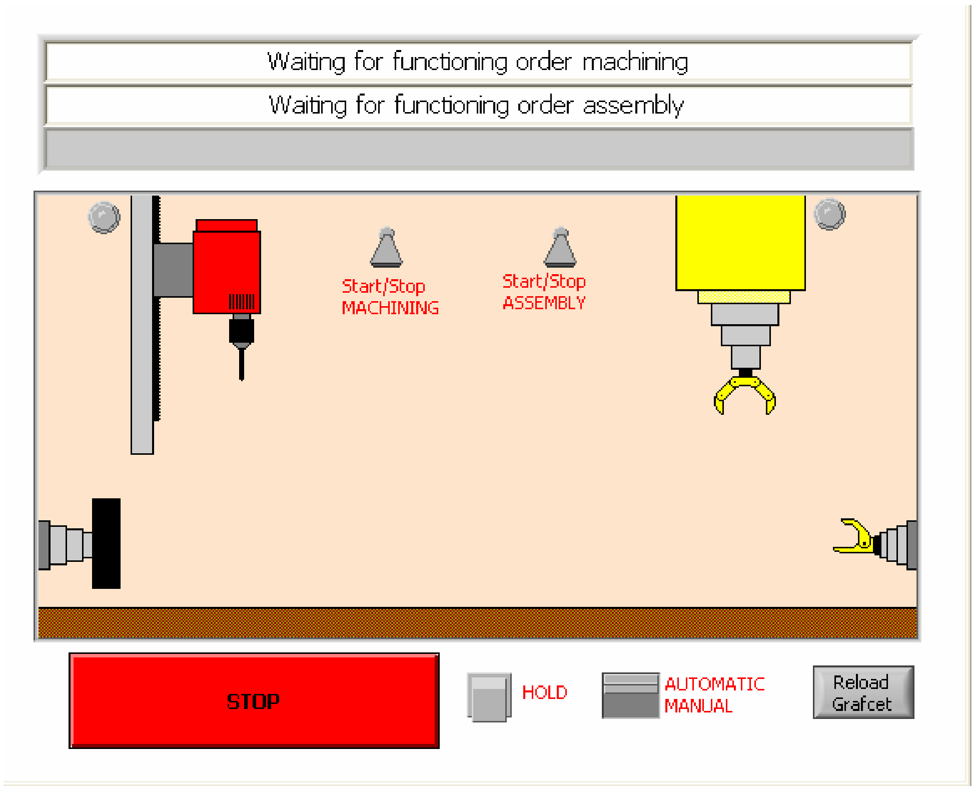

Waiting for functioning order machining
Waiting for functioning order assembly

**Figure 23. Synoptic of the process - Interface operator**

The grafcet specifying the application is as follows:

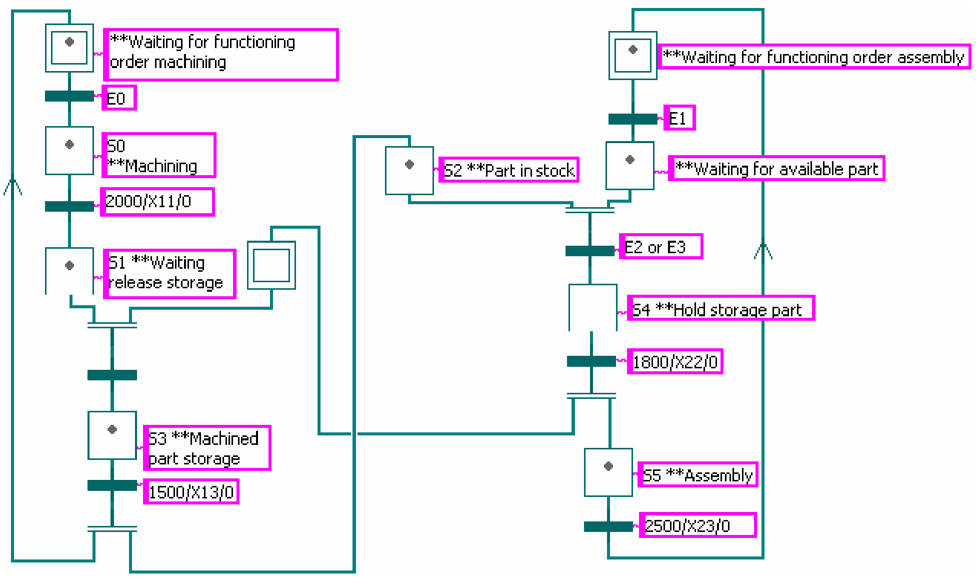

**Waiting for functioning order machining
E0
S0 **Machining
2000/X11/0
S1 **Waiting release storage
S3 **Machined part storage
1500/X13/0
S2 **Part in stock
**Waiting for functioning order assembly
E1
**Waiting for available part
E2 or E3
S4 **Hold storage part
1800/X22/0
S5 **Assembly
2500/X23/0

**Figure 24. GRAFCET specifying the process.**

# Appendix

# Contacts with customers

This appendix contains two forms. The first one will allow you, once that you will have returned it, to obtain useful information which you will use to solve possible technical problems. By filling it before contacting us, you will allow us to act quickly and well. By filling the second form, you will help us to know your impressions on the documentation provided with the TecAtlant products.

**TecAtlant**
Kervarail N°11
56470 Saint Philibert
FRANCE
Phone : + 33 2 97 55 10 62
E-Mail: info@tecatlant.fr

## Technical support form

Make a photocopy of this form and put it up to date each time you carry out physical or software changes. It will be used to you as reference for your current configuration. By filling this form, most precisely possible, before contacting TecAtlant, you will obtain more precise answers of our engineers of application.

If you use other TecAtlant software, want to include the forms of configuration which are in the corresponding user's manuals. Add additional pages if necessary.

Name: ........................................................................................................................

Company: ..................................................................................................................

Address: ...................................................................................................................

Phone number: ..........................................................................................................

Telefax: .....................................................................................................................

Data-processing platform: .......... Model: ............... Processor: ...............................

Operating System: .....................................................................................................

Speed: ................ MHz RAM: ............. Mo Displaying adapter: ..................................

Mouse: yes/no Other adapters installed: ...........................

Hard disk capacity: Mb............... Trademark: ..........................................................

Instruments used: ......................................................................................................

TecAtlant software: ................................................ Version: .........................................

Configuration: ...........................................................................................................

Encountered problem: ...............................................................................................

Error messages met: ..................................................................................................

Following steps start the problem: ...........................................................................

## Documentation form

TecAtlant invites you to bring your comments on the documentation provided with our products. This information will enable us to guarantee products of quality adapted to your needs.

Title: **GrafcetVIEW v.1.22**, **Reference Manual**

Edition Date: **july 2009**

Please bring your comments on exhaustiveness, the clearness and the organization of this handbook: ..........................................................................................................................

..........................................................................................................................

..........................................................................................................................

If you met errors, want to note the corresponding numbers of page and to describe their nature: ..........................................................................................................................

..........................................................................................................................

..........................................................................................................................

We thank you for your collaboration.

Name: ..........................................................................................................................

Function: ..........................................................................................................................

Company: ..........................................................................................................................

Address: ..........................................................................................................................

Phone number: ..........................................................................................................................

Send your mail to: ..........................................................................................................................

..........................................................................................................................

**TecAtlant**
Kervarail N°11
56470 Saint Philibert
FRANCE
Phone : + 33 2 97 55 10 62
E-Mail: info@tecatlant.fr

## References bibliographical

## Reference books

*Comprendre, maîtriser et appliquer le GRAFCET*, M. Blanchard, CEPADUES-EDITIONS, Collection NABLA.

*LE GRAFCET*, N. Bouteille, P. Brard, G. Colombari, N. Cotaina, D. Richet, CEPADUES-EDITIONS.

*Du GRAFCET aux réseaux de Petri*, R. David, H. Alla, HERMES, Traité des Nouvelles Technologies, Série Automatique.

## Standards

Preparation of function charts for control systems. International Standard, CEI/IEC 848, December 1988, CEI - 3 rue Varembé Genève - Suisse.