# QUALITATIVE DESIGN WITH ENVISIONMENT

**Bernard YANNOU**
(Ecole Centrale Paris / Université Paris VI)
Ecole Centrale Paris - Laboratoire Productique Logistique
Grande Voie des Vignes - 92295 Chatenay-Malabry - France
Fax : *(33-1)-41-13-12-72*, Email : *yannou@cti.ecp.fr*

## Abstract

The earlier stages of a design process consist in enumerating (automatically or not) some design alternatives that may be interesting to be studied further. These design alternatives are characterized by a qualitative architecture; i.e., the schematics of the system is defined but parameters are not valued or are defined by an interval. Moreover, for a dynamic system design, functional relationships between dynamic variables (like a monotonic function) can be imposed. Therefore, the dynamic information can be viewed as one or several sets of incompletely specified *ODEs*. Actually, there is no proper analysis tool of such incomplete data in regard to expected dynamic behaviors which we call *dynamic functional specifications*.

We describe, in this paper, a conceptual design toolbox for dynamic systems called QDES (standing for Qualitative DESign). QDES, taking as input a *total envisionment* of a set of *QDEs* for a current design alternative, we revisited such an envisionment in the form of a *constraint satisfaction problem* formulation. Next, QDES is based on : (1) a new model of a *dynamic functional specification* in a phenomenological way, (2) a *design process history* (with new algorithms) showing mutual constraints between expected behaviors (*functional specifications*) and potential behaviors (*envisionment*) of the current design, (3) features of abstraction and simplification enabling large-scale models to be handled.

## 1. Introduction

One of the most important goals of Qualitative Physics is to be used as a tool for design of physical systems. Such a tool would allow the knowledge of dynamics to be used qualitatively (i.e., not numerically) in the initial stages of the design. Two methodologies can be imagined :

(1) developing a synthesis design tool determining the best system matching with some dynamic *functional specifications*,

(2) developing a general analysis design tool letting the designer qualitatively appreciate the degree of matching between the potential behaviors of a design alternative and the dynamic *functional specifications*. This tool could even eliminate some spurious alternatives of architecture from the pre-design phase, thus saving effort and money.

In practice, being still very far from good automatic design procedures, we are just presently interested in a good analyzing tool of physical systems, a design toolbox that a designer can easily handle and for which results are easily interpreted. Thus, it is sufficient to have a phenomenological model of the system containing the only description of possible behaviors (contrary to a causal model). Therefore, our toolbox QDES accepts as input a *total envisionment* of a system which is a finite state-transition-graph [6,8,9,16,25] representing the dynamic behaviors from all initial states. We consider that design must handle *total envisionments* in order to characterize a mechanism in its totality and to be sure not to forget disastrous behaviors. Any sort of *total envisionment* can be considered. From this point of view, QDES represents a general design methodology.

This methodology consists in :

(1) assuming a system model able to match optimally some dynamic specifications. The assumptions of the designer about the system model concern : (a) a component law (e.g., a resistor model), (b) a process law (e.g., a displacement-strain relation of a beam), (c) the schematics of the system (e.g., the kinematic constraints for a mechanism), (d) a control function,

(2) carrying out a *total envisionment* from a (presently one) set of *QDEs* (Qualitative differential Equations),

(3) appreciating in QDES the matching between the *envisionment* of the current system model and the *dynamic functional specifications*, and eventually testing relaxation of such specifications,

(4) returning to (1) in the case of a bad matching.

This methodology is a framework for a lot of very different design problems. For example, QDES can be a tool to identify a symbolic model of *ODEs* (in fact, a system of *QDEs*) from a series of experimental data. Indeed, it is interesting to have a global idea of the matching between experiments and a proposed model before using heavy data fitting procedures.

We were absolutely conscious of the combinatorial explosion inherent to a *total envisionment*. Two answers were brought to partly overcome this problem. Firstly, we developed a minimal and very efficient envisioner based on a *constraint satisfaction problem* formulation. Secondly, handling large-scale *envisionments* is made easier in letting the designer build a hierarchy of models using considerations of abstraction (*aggregation*) and simplification (*filter*). For

these two reasons, we are convinced that QDES is practically a consistent solution for a class of systems corresponding to : a set of incompletely specified *ODEs*, highly non-linear *ODEs*, problems of reasonable size (not too many variables), sets of sophisticated dynamic *functional specifications*.

The phenomenological feature of our system also exists in the hierarchy of models because this hierarchy is characterized by some type of behavior assumptions. Moreover, we will let the designer appreciate the appropriateness of *functional specifications* on the *envisionment* in a visual manner by the global shapes and properties of the graphs.

Section 2 presents our new efficient total envisioner. Section 3 presents our design model beginning by the problems remaining open in the qualitative design field, then giving the definition of a dynamic *functional specification*, presenting a new algorithm of mutual pruning of an *envisionment* and several *specifications*, and ending by a brief description of the design session. Section 4 evokes related work in the field. We conclude in section 5.

# 2. A new total envisioner

We fundamentally revisited *total envisionment* from a set of *QDEs*. This revisiting is based on the efficient use of a *constraint propagation* technique (*arc-consistency* on integers, see §2.2), an explicit typage of qualitative states into *instants* or/and *intervals* and a new simple and efficient *envisionment* algorithm (see §2.7).

The examples of *QDE* given in this section are only toy examples without any design interest. They are only a support for our explanations.

## 2.1. Our notation

At first, let us define some notation and concepts which constitutes our basic vocabulary.

• [ ] is the Sign function of *Sign Algebra* and $\oplus, \otimes, \ominus, \approx$ are the qualitative operators.

• A *Qualitative Variable* which has a value in the Sign set $\{-,ø,+\}$ is related to a continuously differentiable function of time on $\Re$.

• The *Series of Derivatives* of $x$ denoted $SD(x)$ is a list of qualitative variables which are successive derivatives of $x$. The set of $SD(x)$ values is denoted $V(SD(x))$.

e.g. : $SD(x)=(x,x',x'')$ ; $V(SD(x))=\{(-,ø,+),(ø,-,ø),\ldots\}$

• An *analytic* variable is a variable for which a series expansion exists. Then, analyticity is a property of a series of derivatives. An important consequence is that an analytic variable is constant throughout the simulation or always in evolution over the intervals. This last case can be assimilated into analyticity by a twist of the language. Analyticity can be used to model processes evolving at different time scales [17] and simplify a lot an envisionment.

• A *Qsystem* (*Fig. 1*) is a set of *QDEs* (Qualitative Differential Equations) defined by seven Prolog-like predicates[1] : *Sum, Product, Power, Constant, M+, M-* and *Derivatives* as following :

(1) the linear *QDE* :

$$[x1] \oplus \ldots \oplus [xn] \ominus [y1] \ominus \ldots \ominus [ym] \approx ct$$

is written : $Sum((x1,\ldots,xn),(y1,\ldots,ym),ct)$

with $ct$ standing for a constant value : $-, ø$ or $+$.

(2) the monomial *QDE* :

$$[x1] \otimes \ldots \otimes [xn] \approx ([y] \text{ or } ct)$$

is written : $Product((x1,\ldots,xn),y)$

(3) the power $n$ of a variable $x$ :

$$[x] \otimes \ldots \otimes [x] \approx ([y] \text{ or } ct)$$

is written : $Power(x, n, y)$

(4) a constant variable $x$ over time is written :

$$Constant(x, ct)$$

(5&6) the functional relationships of increasing (resp. decreasing) functions are written :

$$M+(x,y) \text{ (resp. } M-(x,y))$$

(7) the *Derivatives* predicate (one per *Qsystem*) has two arguments : the lists of analytic *SD* (first argument) and the lists of non-analytic *SD* (by default) (see *Fig. 1*).

| | |
|---|---|
| | $Product((x,y), xy)$ |
| | $Product((x,z), xz)$ |
| $X'+a.X-a.Y=0$ | $Sum((x',x),(y),ø)$ |
| $Y'+Y+X.Z-b.X=0$ | $Sum((y',y,xz),(x),ø)$ |
| $Z'+c.Z-X.Y=0$ | $Sum((z',z),(xy),ø)$ |
| $Derivatives(((x,x'),(y,y'),(z,z'),(xy),(xz)),())$ | |

*Fig. 1* The *Qsystem* of the Lorenz attractor representing the dynamics of a fluid in convection (a,b,c are positive constants).

• The set of all the variables of the *Qsystem* is denoted *QS* and the set of all *qualitative states* (consistent valuations of such a *QS*) is denoted *V(QS)*.

e.g. : $QS=((x,x',x''),(y,y'))$; $V(QS)=\{((-ø+),(ø+))\ldots\}$

• We denote the instant set as being $P=\{[INSTANT,y] \mid y \in V(QS)\}$ (P for Point) and the interval set as being $I=\{[INTERVAL,y] \mid y \in V(QS)\}$.

## 2.2. Arc-consistency

A very trivial resolution engine enumerates all qualitative states and constrains transitions (see §2.3, §2.4, §3.2). Our program is written in LE-LISP V15[2] with a constraint propagation library called PECOS[3]. We use constraint propagation on integers, especially the *arc-consistency* algorithm and the *daemon procedure*. A qualitative variable $x$ is represented by a constrained variable : $X \in \{-1,0,1\}$ corresponding to $-, ø, +$. The qualitative multiplication is exactly the constraint of integer multiplication.

The qualitative addition $[x] \oplus [y] \approx [z]$ is constrained by :

• the integer constraint (arc-consistency) :

$$-1 <= X+Y-Z <= 1,$$

---

[1] Our QDE vocabulary is very similar to that of the QSIM program [16,17,18].

[2] LE-LISP is a trademark of INRIA (FRANCE).

[3] PECOS is a trademark of ILOG (FRANCE).

• three daemons which operate during enumeration when some conditions are verified :

**if** $X=0$ **then** constrain( $Y=Z$ )
**if** $Y=0$ **then** constrain( $X=Z$ )
**if** $Z=0$ **then** constrain( $Y=-X$ )

Such simplicity does not prevent it from being very efficient.

## 2.3. An explicit typage of states

In most approaches, the basic simulation algorithm for *total envisionment* consists in :

(1) enumerating all qualitative states (set $V(QS)$) from the qualitative algebraic constraints (all but the *Derivatives* predicates) and from the analyticity property;

(2) for all $y$ elements of $V(QS)$ and considering $y$ as an instant (state of zero duration), applying (by various methods) *transition* constraints (from *Derivatives*) and again qualitative algebraic constraints for a future state. Then, enumerating these future states;

(3) repeating step (2) considering $y$ as an interval of time.

Simulation is conventionally carried out to a deep level of two, leading to a complexity of $Card(\{V(QS) \rightarrow V(QS)\})$. This type of algorithm is not economical because we enumerate future states at steps (2) and (3) which were already enumerated in step (1). This is due to the need to generate causal explanations. Moreover a state is here typed in instant or/and interval iff some transition exists a posteriori. Solving these two problems can be achieved in explicitly typing states in instants or/and intervals and in considering $\{I \rightarrow P\} \cup \{P \rightarrow I\}$ as the potential transition set. This would lead to an improvement of the algorithm efficiency by a factor of at least two in the case of distinct sets $P$ and $I$, but practically of three and more. It can be shown that a general sort algorithm into instants and intervals (*Fig. 2*) taking into account which series of derivatives are *analytic* or not, is trivial. It essentially consists in tests of the presence of a zero value in the lists of analytic and non-analytic series of derivatives.

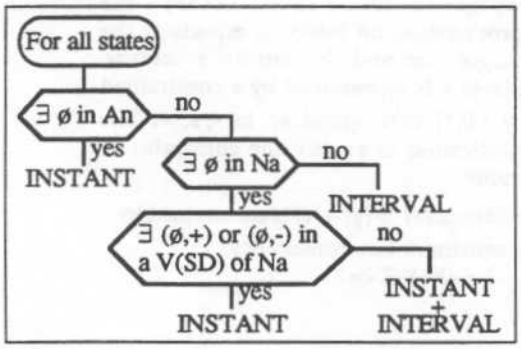


*Fig. 2* General algorithm of sort of states into instants and intervals. *An* is the set of ANalytic *SD*; it is the first parameter of *Derivatives*. *Na* is the set of Non Analytic *SD*; it is the second parameter of *Derivatives*.

## 2.4. Williams' approach of transitions

Williams [24,25] proposes a clear transition automaton of a pair $(f,f')_1$ to $(f,f')_2$ with an explicit instant/interval alternation which we adopt. It can be summarized by three rules :

• *Continuity* of all variables : $-\rightarrow+$ and $+\rightarrow-$ are forbidden transitions.

• *Intermediate Value Theorem* : a value $ø$ of an interval remains $ø$, a value + of an instant remains +, a value - of an instant remains -.

• *Mean Value Theorem* also called *integration law* : $[f_2] \approx [f_1] \oplus [f'_{interval}]$

Being inspired by Kuipers' transition tables [16], it can be shown that this automaton is equivalent to two transition graphs from instants and intervals (*Fig. 3*), which we call *P-Table* and *I-Table* as Kuipers does. Although apparently different, we can show that our tables expressed from Williams' rules can also be extracted from Kuipers' tables because of the same *Mean Value Theorem* and instant/interval alternation. For this reason of compatible semantics, we will say that our algorithm of total envisionment can be an efficient preprocessor to QSIM.

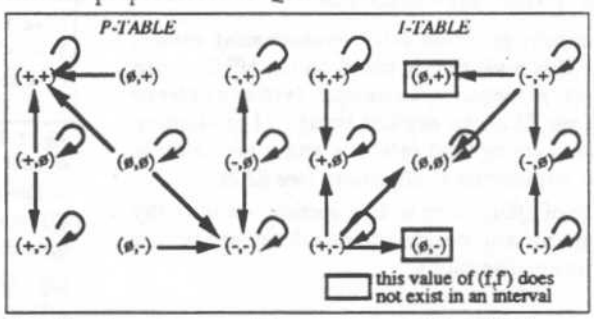


*Fig. 3* Our elementary transition graphs for $(f,f')$ : *P-Table* and *I-Table*

All the transition constraints are expressed in the two tables (*Fig. 3*). We do not have a global examination of state to carry out like de Kleer and Bobrow [5] who use the *Taylor series* instead of the *Mean Value Theorem*. For this reason, all transition constraints between two states can be cleverly implemented by associations of such elementary tables (for each variable and its first derivative). These elementary tables are propagated with the qualitative addition and the daemon procedure previously defined. So, we propagate two transition tables between two adjacent qualitative states which we call the *QS-P-Table* and the *QS-I-Table*.

## 2.5. Higher Order Derivatives (*HOD*)

Symbolically differentiating the *Qsystem* $n$ times (our simulation algorithm is parameterized in $n$, in practice : 0, 1 or 2) enables us to have the n *Higher Order Derivatives* (*HOD's*) expressions (Caution ! This *HOD* definition is somewhat different from Kuipers'). Such additional *QDE* constrain more the problem and consequently eliminate spurious transitions and even spurious intervals when some non-analytic variable exists. Differentiating one time the Lorenz attractor *Qsystem*, new *QDE* appear (to the right of *Fig. 4*, not in

bold) linking the *HODs (x'',y'',z'',xy',xz')* with the initial variables of *QS*. Each predicate of the *QDE* vocabulary has a specific differentiation rule. For example, *M+(x,y)* is differentiated into *Sum((x'),(y'),ø)* and *Constant(x,+)* into *Constant(x',ø)*. Differentiating two times the *Qsystem* only consists in differentiating these new *QDE* obtained at the first differentiation. So, we naturally adopt the "*Sign equality assumption*" of Kuipers [18] for the second differentiation of *M+(x,y)* which gives the debatable : *Sum((x''),(y''),ø)*.

| | |
|---|---|
| *Product( (x,y) , xy )* | *Product( (x',y) , new-a )* |
| | *Product( (x,y') , new-b )* |
| | *Sum( (new-a,new-b), (), xy' )* |
| *Product( (x,z) , xz )* | *Product( (x',z) , new-c )* |
| | *Product( (x,z') , new-d )* |
| | *Sum( (new-c,new-d), (), xz' )* |
| *Sum( (x',x), (y), ø )* | *Sum( (x'',x'), (y'), ø )* |
| *Sum( (y',y,xz), (x), ø )* | *Sum( (y'',y',xz'), (x'), ø )* |
| *Sum( (z',z), (xy), ø )* | *Sum( (z'',z'), (xy'), ø )* |
| *Derivatives( ((x,x',x''),(y,y',y''),(z,z',z''),(xy,xy'),(xz,xz'))* | |
| *, ((new-a,new-b,new-c,new-d))* | *)* |

*Fig. 4* First Differentiation of the Lorenz attractor *Qsystem*. In bold stands the initial *Qsystem*. *(x'',y'',z'',xy',xz')* are the first *HODs*.

Differentiating the *QDE Product( (x',z) , new-c )* leads to the creation of two variables : *new-e* and *new-f* defined by the two *QDEs* : *Product( (x'',z) , new-e )* and *Product( (x',z') , new-f )*. This latter *QDE* will not be created another time during the differentiation of the next *QDE Product( (x,z') , new-d )*. Such monomial recognition (product or power) limit the *QDE* expansion and new variable creations and consequently simplify the enumeration phase. In the same manner, the *Power* predicate, which could seem to be somewhat redundant with the *Product* predicate, has been introduced to be more efficient during the symbolic calculation phase.

Such differentiations take place before the state enumeration phase. But, what is very important to understand is that only the *QS's* variables (in bold) are enumerated, the *HOD* values being deduced or calculated by constraint propagation. Thus, the *HOD* values can be -, ø or + values, but also ? (any of the -, ø or + values), ≠ø, ≠- or ≠+ (*Fig. 5*). Then, each qualitative state is a constrained system between the *HODs* and the new variables created during the differentiations.

| Var. | P 52 | I 53 | P 54 | I 55 | P 56 | I 57 |
|---|---|---|---|---|---|---|
| Analytic | | | | | | |
| x | - + - + | - + - + | - + - ? | - + - ? | - + - ? | - + ? ? |
| y | + - + ? | + - + ? | + - ? ? | + - ? ? | + 0 ? ? | + + ? ? |
| z | - 0 + - | - + ? ? | 0 - + ? | + - + ? | + - + ? | + - ? ? |
| p-xy | - + - | - + - | - + ? | - + ? | - + ? | - ? ? |
| p-xz | + - ? | + - ? | 0 + - | - + - | - + - | - + ? |
| Non-anal. | | | | | | |
| NEW-a-0 | + - | + - | + - | + - | + - | + ? |
| NEW-b-0 | + - | + - | + ? | + ? | 0 ? | - ? |
| NEW-c-0 | - + | - + | 0 - | + - | + - | + ? |
| NEW-d-0 | 0 - | - ? | + - | + - | + - | + ? |
| NEW-e-0 | + | + | 0 | - | - | ? |
| NEW-f-0 | 0 | + | - | - | - | - |
| NEW-g-0 | - | ? | - | - | - | ? |
| NEW-h-0 | - | - | - | - | - | ? |
| NEW-i-0 | - | - | - | - | 0 | + |
| NEW-j-0 | - | - | ? | ? | ? | ? |

*Fig. 5* Example of some qualitative states for the Lorenz attractor *Qsystem*. Only the *QS's* variables : *((x,x'), (y,y'), (z,z'))* are enumerated.

For the Lorenz attractor example, numerous transitions are ruled out with the first and second *HODs* (*Fig. 6*).

No state is ruled out because all the variables were stated as analytic.

| HOD | states | transitions |
|---|---|---|
| 0 | 123 | 418 |
| 1 | 123 | 270 |
| 2 | 123 | 262 |

*Fig. 6* The Lorenz attractor results in function of the number of *HOD* taken into account.

We can note that the *HOD* calculation for a non-linear *Qsystem* is no more complex than that for a linear *Qsystem* because it is not necessary using constraint propagation to qualitatively invert a differentiated *Qsystem* to calculate the *HOD's*. Here, a definite advantage of our approach compared with Kuipers' [18] (only for an envisionment graph determination) is that we are modelling a qualitative variable as an only value whereas Kuipers is modelling a variable as a pair *<qmag,qdir>*. Therefore, Kuipers is compelled to enumerate the *qdir* value of the higher order variables of the series of derivatives. Such a *qdir* corresponds to our first *HOD*. A state (with a unique SD) which has for us an ambiguous (?) first *HOD* value : *V(SD(x))=(-+ | ?)* is translated into three states : *(-+-),(-+ø),(-++)* for Kuipers. This problem of combinatorial complexity (let us imagine several such SD in a state) is identified by Kuipers under the name of *chattering variables*. Fouché, in his PhD dissertation [13], recently completed some serious work on unifying conventional envisionments and behavior generation of QSIM. But he confirms himself that a conventional procedure to have an envisionment in QSIM was to simulate to a depth of two (instead of sorting states into instants and intervals) without landmark values and to proceed almost systematically to a *focus-on-qmag aggregation*. In results, this is exactly the same as our *first-HOD*, but a lot more complex.

## 2.6. Detection of equilibria

Kuipers [16] states :"An interval can only lead to an instant of different value". We show that it is true when this interval does not contain a zero-value in *V(QS)* but false for other intervals. Indeed, we allow the following transitions of *Fig. 7* for *QS=SD(x)=(x,x')* :

*I1(øøø)→P1(øøø)→I2(+++).*

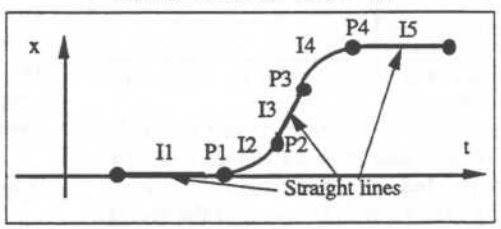


*Fig. 7* Behavior of a variable *x* with stages and slopes.

We interpret it as an unstable equilibrium which evolves from an infinitesimal perturbation, or as a behavior caused by an external controlled variable (let us imagine a robot-arm for which the hand is constrained to follow the curve shown in *Fig. 7*, we impose a *concise history* [26]). This last case is crucial for our future definition of *functional specification* at

§3.1. Such a behavior is completely eliminated by conferring the property of analyticity to the variable.

Until recently, everybody [5,8,16,25] said a quiescent state was an all zero-derivative state and no transition was determined from this state, e.g., $QS=((x,x',x''),(y,y'),(z))$, $y \in V(QS)$ / $y=((+,ø,ø),(-,ø),(-))$ would be a quiescent state. This is immediately contradicted by a counter-example (*Fig. 8*).

On $\Re : x''(t)^2=x(t)$ gives qualitatively :

$$[x''] \otimes [x''] \approx [x]$$

$$Qsystem = \{ Power(x'', 2, x), Derivatives(()), ((x,x',x'')) ) \}$$

A particular solution on $\Re$ is : $x(t) = \frac{t^4}{144}$

Then, envisionment must forecast corresponding states and transitions :

$$I7=(+,-,+) \rightarrow P5=(ø,ø,ø) \rightarrow I9=(+,+,+)$$

Thus we see that *P5* is not a quiescent state because $[\partial^4 x]=+$.

*Fig. 8* Counter-example showing that an all zero-derivative state is not a sufficient definition for quiescent state

An all zero-derivatives state only defines an equilibrium state but it can be an unstable one. It is the case for the example of *Fig. 8* because state *(øøø)* evolves to *(+++)*. We clearly show that this observation can only be *a posteriori* made, i.e., after a complete determination of the graph. The simulation on this example carried out in *Fig. 9* for a non-analytic variable *x* lets one may presume two types of behaviors around the instant *P5*. It is impossible for us to know if we have an immediate evolution : $I7 \rightarrow P5 \rightarrow I9$ (unless we succeed in showing inevitably that $P5=(øøø \mid ø+)$), or an equilibrium during an interval which is able to evolve : $I7 \rightarrow P5 \rightarrow I5 \rightarrow P5 \rightarrow I9$. An element of response can be found by conferring the property of analyticity to the variable : *I5* would not exist and we would have a behavior of immediate evolution.

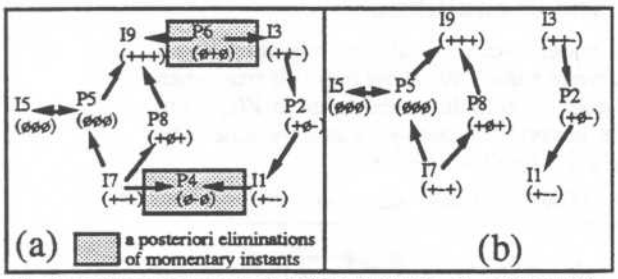


*Fig. 9* Envisionment graph of $[x''] \otimes [x''] \approx [x]$ (*Fig. 8*) without (a) and with (b) simplifications

Other interesting a posteriori deductions can be made on this example. Inconsistent instants (*P4* and *P6*) are revealed on the graph. Indeed, an instant cannot be a source (*P6*=big-bang ?) nor a well (*P4*=end of the world ?) of a graph. After these a posteriori simplifications, we notice that the initial graph is simplified into two non-connected subgraphs (*Fig. 9b*), which reveals a potentially chaotic behavior and is a very important result for a functional interpretation.

A last example can be given on a mechanical problem : the conventional pendulum (*Fig. 10*).

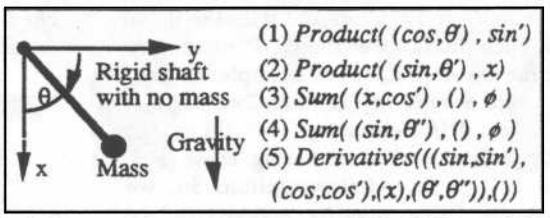


*Fig. 10* The pendulum with no friction : *QDEs* (1,2,3) model the *(sine,cosine)* behavior. *QDE* (4) is Newton's law. All the variables are assumed analytic.

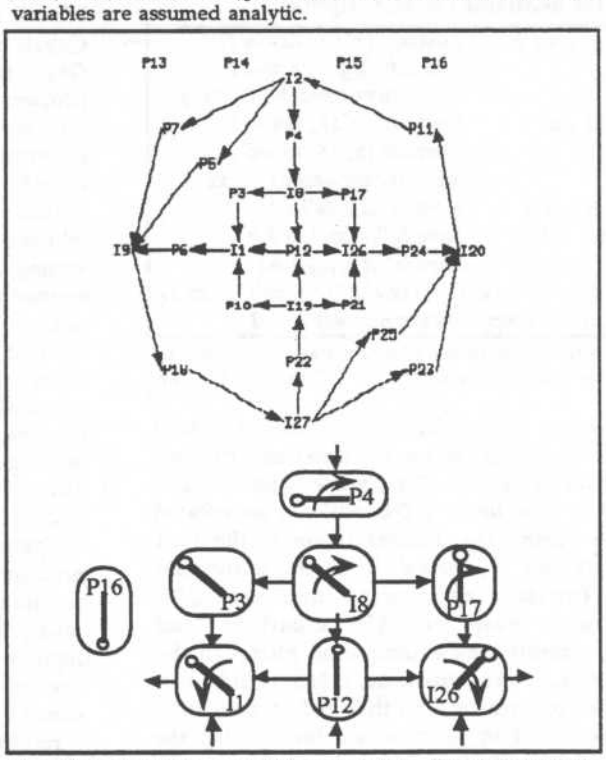


*Fig. 11 Total envisionment* of the pendulum. The three *instants* : *P13*, *P14* and *P15* are inconsistent because they correspond to a quadrant *(sine,cosine)=(ø,ø)*. Arrows in nodes of Fig. b represent $\dot{\theta}$.

There are two equilibrium states : *P12* and *P16*. *P16* is the quiescent state where $\theta=0$. With no friction we can not reach this state of rest unless we are constantly at rest. The instant *P12* is the unstable equilibrium where $\theta=\pi$. The symmetry of the graph of *Fig. 11a* emphasizes well the three cases of the energy amount of the pendulum (*Fig. 11b*) : the pendulum has not enough ($I8 \rightarrow P3$), just enough ($I8 \rightarrow P12$) or enough energy ($I8 \rightarrow P17$) to reach the higher position ($\theta=\pi$). Such a graph is automatically unfolded by a new algorithm we developed for the QDES toolbox.

## 2.7. Total envisionment algorithm

Our algorithm of total envisionment in a constraint propagation environment follows :

(1) Symbolic differentiation of the *Qsystem* to have the *HOD* expressions.

(2) Enumeration of all states (*HOD's* are not enumerated, they are not *chattering variables*).

(3) Sort of states into instants and intervals (*Fig. 2*), taking into account *HOD* values.

(4) Propagation of the two *QS-I-* and *QS-P-Tables* between two states (not yet valuated) by additions of elementary *I-* and *P-Tables* (*Fig. 3*). *HOD's* are taken into account.

(5) Determination of valid transitions :

**For all** $x \in I$
 Propagate( *QS-I-Table*($x \rightarrow ?$) )
  **For all** $y \in P$
   **if** (consistent(Propagate( *QS-I-Table*($x \rightarrow y$) )))
    **then** Transition($x \rightarrow y$)
**For all** $x \in P$
 Propagate( *QS-P-Table*($x \rightarrow ?$) )
  **For all** $y \in I$
   **if** (consistent(Propagate( *QS-P-Table*($x \rightarrow y$) )))
    **then** Transition($x \rightarrow y$)

(6) Elimination of inconsistent momentary instants.

(7) Determination of quiescent states and unstable equilibria.

## 2.8. Temporary conclusion

Here are the key features of our envisioner :

(1) The determination of states and transitions is carried out with a constraint propagation technique of arc-consistency on integers.

(2) The Williams' transition rules are compatible with constraint propagation.

(3) Before the transition phase, all states are enumerated and sorted into instants and/or intervals taking into account which series of derivatives are analytic.

(4) *HODs* are not enumerated, thus we avoid the *chattering variables*.

(5) The transition phase consists in determining : $\{I \rightarrow P\} \cup \{P \rightarrow I\}$.

(6) We clearly show that stable and unstable equilibria can be characterized only a posteriori (i.e., after determination of all the transitions).

(7) We also claim that Kuipers' QSIM algorithm would improve its efficiency greatly if our tool was used as a preprocessor of QSIM. Actually, QSIM does the same work all the time during transitions. Our envisionment graph holds all the local transition information. QSIM qualitative states with landmarks are richer information than ours but they could be indexed in a *hashing table* by the corresponding state of ours.

All these points contribute to a homogeneous, efficient and simple tool of envisionment. This work can be related to the Forbus' "*QP Engine*" [9] which is a total envisioner for the "*QP theory*" [8] using an ATMS. We developed a much more simple envisioner for a *QDE* set. Like Forbus, no conceptual progress is made, but an efficient informatic structure is proposed, the necessary first step to handle large envisionments in the QDES toolbox.

# 3. Design Model

We think that many problems remain open in the qualitative design field. Let us detail some of those that we attempted to solve in QDES :

- We did not find any satisfactory and expressive definition of a *functional specification* in terms of expected behaviors. Up to now, they were restricted to a unique expected behavior [2], or they included other types of specifications like time-scale [17], processes of interest [7]. We would like to treat the problem which is very widespread among physicists of identifying a supposed dynamic model from multiple experimental behaviors (a phenomenological issue).
- How must these *functional specifications* be combined to design a product ?
- What does pruning an envisionment constrained to verify *functional specifications* mean ?
- We know that in real life a *functional specification* can be partly verified; what does "partly" mean ? Is it possible to prune a *functional specification* ?
- We would like to have a tool to test the relaxation of *functional specifications* on the system.

## 3.1. Functional specification

In order to find a satisfactory model for our *functional specifications*, we will place ourselves in the framework of the identification of a dynamic model from multiple experimental behaviors. The result of an experiment is a set of *histories* of some variables [26]. We can for each history qualitatively interpret its shape slicing the curve by time-points of changes of interest. The number of such time-points depends on the degree of derivative we consider; are we interested by a change in derivative or by a change in curvature ? In any case, from several such histories measured by sensors, we can totally order all time-points for an experiment. Then, we can reconstitute one *behavior of segments* [10], a *segment* being a projection of a qualitative state on a subset of variables of the Qsystem (here, the measured variables). For one experiment, we have a total equivalence [26] between a *state-based model* and a *history-based model* with totally ordered time-points (*Fig. 12*).

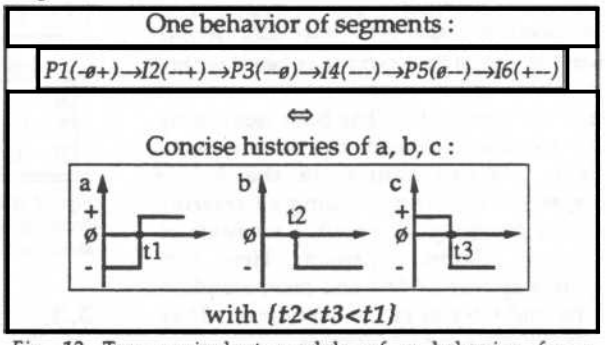


*Fig. 12* Two equivalent models of a behavior for a *Segment=(a,b,c)*

A dynamic model is well characterized and constrained after several experiments. Then, two cases arise in a set of experiments :

- (a) For each measured variable, its history is the same for all experiments. The only difference concerns the ordering between time-points which changes between experiments. Several physical phenomena occur but their relative ordering is not relevant, this is the well-known *occurrence branching* problem [13].
- (b) At least one history of a variable differs between two experiments.

Therefore an exhaustive set of experiments can be viewed as a set of *equivalence classes* for variable histories (conserving the specific time-point orderings in a class). The key idea is here to consider each equivalence class and to express it in a concise and expressive manner avoiding the combinatorial complexity of the *occurrence branching* problem if we were to specify each state-based behavior. Therefore, contrary to a measurement interpretation problem, for our design problem we express such an equivalence class as a set of concise histories for variables (and eventually for their successive derivatives) with *partially ordered time-points* (*Fig. 13*). This is our definition of a *functional specification*. We claim that with the exhaustive set of ordering operators $\{=,\neq,>,>=,<,<=\}$, expressing a *functional specification* with a set (i.e., conjunctions) of ordering relations [1,3,22] and without disjunctive ones, is easy, fast and expressive. Nevertheless, in some very specific cases, disjunctions between ordering relations are necessary. In these cases, we will split the specifications into several conjunctive ones.

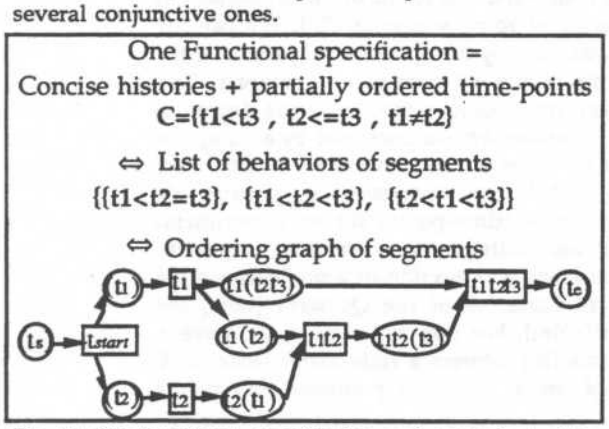


*Fig. 13 One functional specification* and equivalent data (circles stand for *instants* and squares for *intervals*). $t_1(t_2t_3)$ stands for an *instant* at current time $t=t_2=t_3$, $t_1$ being already passed, such data define a unique *segment*.

Once the *functional specification* has been defined by the designer, we are able to generate two other types of equivalent data. We can enumerate the *list of behaviors of segments* and after building an *ordering graph of segments* grouping all possible *segments* of behaviors and their relative transitions. Here, two implicit time-points appear : *tstart* and *tend*, standing for the beginning and the end of the experiment. They are consequently the same for all histories.

## 3.2. Functional specification algorithm

The algorithm interpreting the *functional specifications* as *segments*, behaviors and *ordering graph* (*Fig. 14*) uses our standard tool of arc-consistency on integers and stands in four points :

- We constrain time-points to be integers in a domain from 1 to the number $n$ of time-points (here 3). Again, we constrain time-points with relations of order (arc-consistency on integers) and we enumerate the solutions.
- Next, each enumeration (potential behavior) is transposed in another table switching index (from 1 to $n$) and data (from 1 to $n$). It allows a better idea of the relative orders of time-points to be obtained. Data of this new table is a list of time-points occurring at imaginary time between *tstart* and *tend*. This list may be empty or can contain multiple equal time-points.
- Next, we must filter out some duplicated behaviors. It can be seen in *Fig. 14* that behaviors *B3* and *B5* are identical to behavior *B1*. It is not worth comparing them, it is enough to systematically eliminate behaviors having an empty list inside data. It is clear such an algorithm is far from being optimized (in comparison with [22]), but it has the advantages of simplicity and using our standard tool of arc-consistency technique.
- Results can be now established. For each non-eliminated behavior of the previous table, we move along the time axis from *tstart* to *tend* building instant *segments* and interval *segments*. For that we consider at a current point of the axis the list of the time-points already encountered and for instants a second list of time-point(s) characterizing the current instant. For instance, $t1(t2t3)$ stands for an instant (because of the pair of parentheses) at $t=t2=t3$, $t1$ being already passed. We can notice that such data is a characterization of a *segment* through histories data.

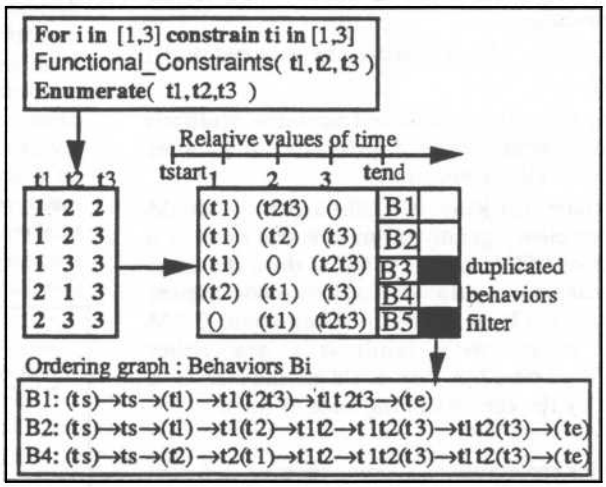


*Fig. 14* Algorithm of determination of *segments*, behaviors and *ordering graph* from the *functional specification* (*ts* stands for *tstart* and *te* for *tend*).

## 3.3. Combining several functional specifications

We have a very interesting property in the equivalence of the *ordering graph of segments* with the list of behaviors of *segments* and consequently with the initial *functional specification*. It means that the number of

paths in the graph between *tstart* and *tend* is equal to the number of behaviors of *segments*. This is not a trivial property because we do not have spurious paths in the graph like in *Fig. 15*. This is due to conjunctions between relations of order.

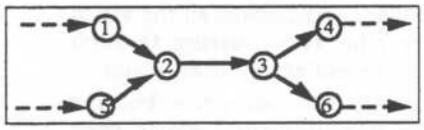


*Fig. 15 (1,2,3,4)* and *(5,2,3,6)* could be consistent paths and *(1,2,3,6)* and *(5,2,3,4)* could be spurious paths

A disjunction would not give this property. The combination of two *functional specifications* to design a unique product is naturally defined by the union of the two lists of behaviors of *segments*. But uniting the two *ordering graphs* would result in a previously evoked problem of disjunction (*Fig. 15*). Although being more expressive and compact than two initial *ordering graphs*, the resulting graph would impoverish the information. It is exactly the same problem as between several generations of behaviors like QSIM does and a resulting envisionment, the latter form contains less information about consistent behaviors. Moreover, such uniting of two *ordering graphs* have no sense when the respective *segments* lie on different variables.

## 3.4. Simultaneous pruning of envisionment and specifications

We are going to explain what pruning an envisionment and a *functional specification* by their mutual constraints means. Let us examine a unique behavior of *segments*. We know through the Forbus' ATMI theory [10] that it is possible to determine all the potential images (*image-paths*) of a behavior of *segments* in the place of paths throughout the envisionment of the system.

Our algorithm of simultaneous pruning follows : for a behavior of *segments*, if at least one *image-path* is found, we will prune the envisionment, pruning all states and transitions not present in any image-path, otherwise it is the behavior of *segments* which will be eliminated (*Fig. 16*). Pruning an envisionment from a set of *functional specifications* simply consists in making a loop on all the behaviors of *segments*. This pruning algorithm (*Fig. 16*) is called intersection and we logically supply the subtraction procedure to have explicit results about the pruned data.

A pruned *functional specification* (one per initial specification) is interpreted without any ambiguity as the part of specifications the system is able to match. Here, we have a model which is very close to real problems.

The pruned envisionment represents the functional configuration space of the system or the vaster part of the envisionment which may be required to match with the feasible part of the *functional specification*. It does not mean that the behavior of the system must be restricted to this pruned envisionment because if any image-path is found, the *functional specification* is already verified. It may only be a good tool to have an idea of non-functional modes of the system by difference of the initial and pruned envisionments. These non-functional modes of behaviors can be the cause of waste, may lead the system to undesirable situations, or can help to simplify or redesign a part of the system.

```
procedure intersection(IN : env , list-specif ;
                       OUT : pruned-env , list-pruned-specif)
  list-St&Tr← (empty-list)
  list-pruned-specif ← (empty-list)
  For each specif in list-specif
    pruned-specif ← (empty-list)
    For each behavior in specif
      list-image-paths ← image-paths(behavior , env )
      if not( is-empty-list( list-image-paths ) ) then
        put-in-list(behavior , pruned-specif)
        put-in-list(States&Transit-of( list-image-paths )
                    , list-St&Tr)
      end if
    end for
    put-in-list(pruned-specif , list-pruned-specif)
  end for
  list-St&Tr←union-of-elements(list-St&Tr)
  pruned-env←env-restricted-to(env , list-St&Tr)
end
```

*Fig. 16* Pruning algorithm for the intersection of the envisionment and a list of specifications. list-St&Tr stands for a "list of States and Transitions" of interest.

But a problem appears in pruning the envisionment from the image-paths. It concerns the previously evoked problem of impoverishment of information passing through an information of behaviors to a sub-envisionment. Here we decided not to propose the information of the envisionment's *image-paths* to the designer as we do for *functional specifications*. The reason lies in the wide extension of such information compared to the generally very small number of simultaneous *functional specifications*. Therefore great care must be taken when interpreting the pruning of an envisionment.

## 3.5. QDES *Q*ualitative *DES*ign Toolbox

We were to supply two major features in our toolbox :

(1) an adequate definition of *functional specification* and pruning procedures and their significations. All this work has been carried out and previously detailed;

(2) two features of great interest allowing the complex world to be understood better [14] :

- reasoning about *abstractions* leading to global inferences and building a hierarchical model with different *grain sizes* of the problem,
- simplifying the problem in making judicious *assumptions*.

We propose to implement abstractions and assumptions in a phenomenological and conventional manner. Making assumptions consists in filtering out some states and transitions and conserving those which have properties of interest : procedure filter( <Properties of Qstates> ). Making abstractions consists in aggregating states to have a higher level of analysis.

*Aggregation* consists in building equivalence classes of the values of a subset of qualitative variables [13]. In fact we obtain a kind of envisionment of *segments* defined by such a subset. The originality of this *aggregation* is to be carried out on all or a part of the states : procedure aggregate(<Properties of Qstates>) on (<Properties of Qstates>). Obviously, the two unary operators on graphs : filter and aggregate both run on the envisionment and the *ordering graphs*.

We naturally include in our QDES design toolbox these two unary operators and the two binary operators : intersection and subtraction between an envisionment graph and a list of *specifications*. We will manage a design history with several history trees, roots being the envisionment graph and the ordering graphs of functional specifications (*Fig. 17*). Each node of the tree is a resulting graph, but typed as envisionment or *ordering graph*. Each path from a root to a leaf node can be viewed as a specific reasoning. Often such reasonings are interdependent because of the binary operations (indirect links in *Fig. 17*).

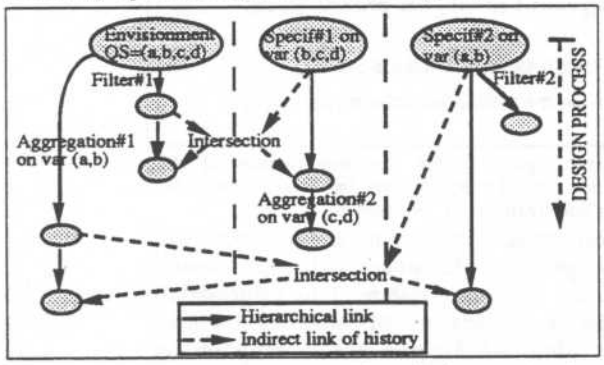


*Fig. 17* Example of a design session with multiple reasonings in parallel

We clearly see that we have a large number of combinations to carry out reasonings and to handle information. But the resulting interpretation is always in the hands of the designer and often the consistency of its design history too.

Nevertheless, some consistency rules can be found to prevent the designer from carrying out inconsistent operations. Let us mention the *grain size consistency rule*. We can consider that an attribute of a tree node is its grain size, i.e., the subset of variables of interest of the graph. An *aggregation* reduces this grain size to the subset of variables given in first argument whatever the states concerned in the second argument (unless no state is consistent). Therefore, a consistency rule during a binary operation consists in verifying that :

**For each** *specif* **in** *list-specif*
  **verify** grain-size(*specif*) ⊂ grain-size(*env*)
**end for**

Otherwise, we would have a useless loss of information.

What we mean by a property of states (<Properties of Qstates> which is argument of intersection and subtraction) remains to be explained. A property is a combination of elementary properties by the *and*, *or*, *not* operators. We can characterize an elementary property by two ways :

- a property intrinsic to states : values, quiescence, unstable equilibrium, analyticity of some variable. This last property used in a filter operation corresponds to a first level time-scale abstraction like Kuipers does [17].
- a property of behavior characterizing all the states in such a behavior : behavior passing through certain states, starting or ending by certain states.

Such properties are surely not exhaustive but the important point is the flexibility we have in such expressions. The scenario of a design session and a language to define an aggregation and a filter were written in a BNF form. For example, we can imagine a designer would know the states able to lead, through a behavior in which $y$ is analytic, to unstable equilibria such that $x$ is positive. This information can be given to him after the following filter operation :

filter( Behavior_Ending( Unstable_Equil._Qstates
  and (x=+) )
  on (Qstates_with_Analytic_Qvars (y)) )

# 4. Related work

Much work was already carried out in this field with different approaches. Sacks [20,21] developed methods to give a qualitative description of a system in a phase space. These methods are surely the best for completely specified planar *ODEs* (or with one parameter [21]), but they are not applicable in the general case : non-planar *ODEs* or partially specified *ODEs* (e.g., *M+* function) like for a design alternative (see abstract and introduction).

Spatial reasoning [12,15] was developed to have a precise idea of kinematics, taking into account limits in the *configuration space* of parameters. But these methods are time consuming, they do not take dynamics into account and are too detailed to give a high level of qualitative interpretation.

Others compare the purpose of a design (one unique expected behavior) to an envisionment of the system [2]. Other slightly different approaches exist [4,7,10,11,17]. They consist in posing the following question : "Assuming several hypotheses on my system, what are their consequences on the behaviors of the system and are they consistent ?" Falkenhainer and Forbus [7] call these hypotheses *modelling assumptions*. These assumptions are propagated in the general model to generate a simplified simulation. These assumptions are even automatically modelled through a tutoring system to answer questions about functional characteristics of the system behavior. Kuipers [17] deals with time-scale assumptions, totally ordering processes with regard to their speed of evolution. In the same manner, Forbus with his ATMI theory [10 and 11,4], tries to relate an envisionment of a system to a partial observation of it interpreting data (sometimes sparse) across time into *histories* and *segments*. The goal is to have global interpretations and in the best case a unique interpretation of the observations [10]. These global interpretations allow the envisionment to be pruned, i.e., to constrain the future behavior [11]. All these approaches consist in

comparing one expected [2] or measured [10] behavior to an envisionment or to assume this behavior and to carry out a simplified simulation [7,17]. Our method consisting in comparing all the expected behaviors to all the potential behaviors is more adapted to give relevant qualitative information in a design process.

## 5. Conclusion

We think we propose a design model which makes the best use of an envisionment for a *QDE* model of a system. We defined a very expressive *functional specification* model and procedures to evaluate the degree of matching between potential behaviors of a design alternative and expected behaviors. Expected behaviors may be realized partly as in real life. Often, it does not cast doubt over the system because specifications may be in part relaxable. In any case, it can be a tool to test the relaxation of specifications. Our design toolbox QDES introduces a rich vocabulary to question the system in a phenomenological way and carry out powerful reasonings. Of course, their interpretations will remain in the hands of the designer. This is done in a visual manner by the global shapes and properties of the graphs. Abstractions and simplifications allow at any time in a reasoning to modify the grain size or simplify the model. Our approach uses a standard tool of constraint propagation technique, and especially arc-consistency on integers. Moreover it is developed on a homogeneous and original theory of envisionment.

## 6. Implementation

The design history part being still under development, we have not yet dealt with large problems. An algorithm of graph unfolding was developed to represent large graphs to the designer in a suitable manner which is essential for interpretation.

## 7. Acknowledgements

This research was supported by Dassault Systèmes. The content of this paper benefited from extensive discussions with Dr. Antoine Missier [1] and he gave me the judicious example *[x'']⊗[x'']=[x]* (*Fig. 8,9*). I also thank Adrian Vasiliu [2] for his graph unfolding algorithm, Prof. Jean-Claude Bocquet [2] and Prof. Louise Travé-Massuyes [1] for their review of this paper and Dr. Alan Swan [2] for reading the English version.

## 8. References

[1] J.F. Allen : "Maintaining Knowledge about Temporal Intervals", *Communications of the ACM*, 1983, 832-843

[2] J. Bradshaw, R.M. Young : "Evaluating the behaviour of the BAe 146 hydraulic system using the Dori system", *ECAI-92*, 739-743

[3] T. Dean, M. Boddy : "Reasoning about Partially Ordered Events", *Artificial Intelligence* 36, 1988, 375-387

[4] D. DeCoste : "Dynamic Across-Time Measurement Interpretation", *Artificial Intelligence* 51, 1991, 273-341

[5] J. de Kleer, D.G. Bobrow : "Qualitative Reasoning with Higher-Order Derivatives", *National Conference on AI*, 1984

[6] J. de Kleer, J.S. Brown : "A Qualitative Physics Based on Confluences", *Artificial Intelligence* 24, 1984, 7-83

[7] B. Falkenhainer, K.D. Forbus : "Setting up Large-Scale Qualitative Models", *AAAI-88*, 301-306

[8] K.D. Forbus : "Qualitative Process Theory", *Artificial Intelligence* 24, 1984, 85-168

[9] K.D. Forbus : "The Qualitative Process Engine", *Technical report of the University of Illinois Department of Computer Science*, 1986

[10] K.D. Forbus : "Interpreting Observations of Physical Systems", *IEEE Transactions on Systems, Man, and Cybernetics* 13, 350-359, May/June 1987

[11] K.D. Forbus : "The Logic of Occurrence", *IJCAI-87*, 409-415

[12] K.D. Forbus, P. Nielsen, B. Faltings : "Qualitative Spatial Reasoning : the Clock Project", *Artificial Intelligence* 51, 1991, 417-471

[13] P. Fouché : *Vers une unification des méthodes de simulation qualitative*, PhD. dissertation, Université Technologique de Compiègne, France, 1992

[14] J. Hobbs : "Granularity", *IJCAI-85*, 432-435

[15] L. Joskowics, E.P. Sacks : "Computational Kinematics", *Artificial Intelligence* 51, 1991, 381-416

[16] B.J. Kuipers : "Qualitative Simulation", *Artificial Intelligence* 29, 1986, 289-338

[17] B.J. Kuipers : "Abstraction by Time-Scale in Qualitative Simulation", *AAAI-87*, 621-625

[18] B.J. Kuipers, C. Chiu, D.T. Dalle Molle, D.R. Throop :"Higher-Order Derivative Constraints in Qualitative Simulation", *Artificial Intelligence* 51, 1991, 343-379

[19] A. Missier : *Structures Mathématiques pour le Calcul Qualitatif, Contribution à la Simulation Qualitative*, PhD. dissertation, LAAS/CNRS, Toulouse, France, 1991

[20] E.P. Sacks : "Piecewise Linear Reasoning", *AAAI-87*, 655-659

[21] E.P. Sacks : "Automatic Analysis of One-Parameter Planar Ordinary Differential Equations by Intelligent Numeric Simulation", *Artificial Intelligence 48*, 1991, 27-56

[22] R. Simmons : "Commonsense Arithmetic Reasoning", *AAAI-86*, 118-124

[23] D.S. Weld, J. de Kleer : *Readings in Qualitative Reasoning about Physical Systems*, Morgan Kaufmann Publishers, 1990

[24] B.C. Williams : "The Use of Continuity in a Qualitative Physics", *National Conference on AI*, 1984

[25] B.C. Williams : "Qualitative Analysis of MOS Circuits", *Artificial Intelligence* 24, 1984, 281-346

[26] B.C. Williams : "Doing Time : Putting Qualitative Reasoning on Firmer Ground", *AAAI-86*, 1984, 105-112

[1] LAAS/CNRS, Toulouse, FRANCE.

[2] ECP, Laboratoire PL, FRANCE.