# Méthode du recueil de connaissances expertes

## Contents

Ce chapitre présente la méthode de recueil de connaissances que j'ai mise au point au cours de ma première année de thèse. Un article *soumis* [a] à la revue *Expert Systems with Applications*, rend compte de cette méthode. Une introduction en français résume cet article. Il est ensuite retranscris dans son intégralité en anglais. Une discussion en français prolonge, dans le cadre de la thèse, les réflexions menées dans l'article.

## 4.1 Introduction

Le POD Mildium, même sous sa forme initiale aux spécifications largement incomplètes (voir ann. B), est une ébauche de processus décisionnel. Il s'agissait donc de faire un recueil du processus. Je n'ai pas abordé ce sujet comme un problème de spécification classique en génie logiciel puisqu'il ne s'agissait pas, dans mon cas, de concevoir un système pour *résoudre un problème* mais bien, dans un premier temps de *retranscrire de manière exhaustive* et fidèle, la solution mise au point par les pathologistes (les experts). C'est seulement dans un second temps que l'on se proposait de compléter cette solution par une plus grande richesse de détails.

Notons néanmoins, en ce qui concerne le recueil d'exigences et la spécification formelle pour le génie logiciel, les travaux de Glinz (Glinz, 2002; Glinz et al., 2002) qui définissent un idiome des Statecharts adapté à la représentation des exigences, ainsi que ceux de Svetinovic et al. (2007) qui développent une critique de la méthodologie d'analyse des systèmes par Statechart (mise au point d'un protocole « Vo-ip » dans leur exemple). Ces deux démarches se concentrent sur l'utilisation des Statecharts pour la spécification de systèmes entiers. Ce type d'approche est nettement minoritaire depuis la généralisation de l'approche de la programmation orientée objet et la généralisation d'UML 2.0 [b]. En effet le Statechart qui, dans l'approche fonctionnelle de ses origines, pouvait servir à spécifier l'ensemble du système (Harel, 1987) est dans UML 2.0 utilisé pour décrire le comportement d'une classe. Notre plate-forme de simulation utilisant un langage à objet (C++), le POD est modélisé par un Statechart et une classe UML 2.0 unique. Notre objectif est ici de décrire le plus fidèlement possible le comportement décisionnel des experts avec ce Statechart.

## 4.2 Matériel et méthodes

Matériel et méthodes sont présentés de manière complète en section 4.4.3.1. Le recueil de connaissances a été effectué auprès de quatre Experts.

- les entretiens duraient environ une heure
- le recueil a été effectué sur la base d'un diagramme Statechart.
- du papier calque et un crayon ont été utilisés pour annoter les diagrammes.
- les entretiens étaient enregistrés avec le logiciel Audacity.

## 4.3 Présentation de l'article

L'article se compose de la manière suivante : l'introduction replace d'abord les SED dans le contexte de la modélisation de processus de production agricole « industriels »

a. le 17/08/2008
b. Unified Modelling Language (UML 2.0) voir `http://www.omg.org`

comme l'un des cadres utilisés (notamment via les DEVS[c] de Zeigler et al., 2000). Ensuite sont présentées les motivations présidant à la conception de Mildium, c'est à dire principalement le besoin d'innovation dans le domaine de la protection des cultures en viticulture (voir chap. 1), ainsi que les objectifs de la formalisation : le besoin d'une explicitation de l'intégralité de la connaissance mise en œuvre par les experts pour pouvoir passer à une étape d'expérimentation plus large et mener des simulations dans le cadre d'une ferme virtuelle. L'introduction se termine par le choix du formalisme Statechart, cette section reprenant en condensé les éléments présentés au chapitre 2 à la section 2.6.

Dans un deuxième temps, les enjeux puis les objectifs de l'article sont précisés :

> L'une des difficultés pour la formalisation du POD Mildium réside dans le fait qu'il s'agissait d'un travail d'équipe. Les membres de l'équipe s'accordaient sur les caractéristiques les plus importantes mais il existait de nombreuses différences sur l'interprétation des détails. Au cours des expérimentations de plein champ, la prise de décision s'est effectuée collégialement. Afin d'expliciter l'ensemble de la connaissance du processus, il m'a semblé que j'obtiendrais une richesse de détails plus grande en procédant à l'extraction de connaissances auprès de l'équipe entière, plutôt que de sélectionner l'un d'entre eux et de ne recueillir que sa compréhension du processus. Les membres de l'équipe avaient conçu le POD Mildium ensemble et cela aurait été une perte de ne pas tirer avantage de l'apport de chaque expertise spécialisée.

> Objectifs de l'article :
>
> La problématique peut être exprimée ainsi : après avoir conçu une solution PIC pour le contrôle du pathosystème viticole en utilisant une connaissance experte et avoir identifié les Statecharts comme un formalisme de modélisation pertinent, peut on utiliser ce même formalisme pour extraire la connaissance ? En d'autres termes : est-il possible de procéder à un recueil de connaissances, en utilisant des Statecharts comme médiation entre le « cogniticien »[d] et de multiples experts ?

L'article explore cette question en trois points, (i) en exposant un état de l'art du recueil de connaissances auprès de multiples experts puis, (ii) en présentant la méthode à proprement parler suivie d'une illustration des résultats, enfin (iii) en précisant l'une des méthodes de validation mise en œuvre. Une discussion en forme de retour d'expérience clôt ce travail.

c. Discret EVent System(DEVS)
d. en anglais Knowledge Engineer

## 4.4 Article « expérimentation des Statecharts pour le recueil d'une connaissance agronomique auprès de multiples experts »

**Remarque :** *L'expression POD Mildium est traduite en anglais par : Grapevine powdery and downy Mildew Decision Workflow System (GrapeMilDeWS) .*

Article soumisà *Expert Systems with Applications* le 17/08/2008.

# Experimenting Statecharts for Multiple Experts Knowledge Elicitation in Agriculture [e]

Bertrand Léger[*,$] and Olivier Naud[*]
[*]Cemagref - UMR ITAP - BP 5095 34196 Montpellier Cedex 5
[$]INRA - UMR Santé Végétale -BP 81 33883 Villenave d'Ornon Cedex

**Abstract**
Statecharts were experimented as a mediation tool between multiple experts and a knowledge engineer. After a short survey of knowledge elicitation methods for multiple experts, we present our method for assessing the quality of the elicited model and give critiques on the basis of our case study in vineyards crop protection management.

## 4.4.1 Introduction

Environmental as well as health issues are getting growing consciousness among the public. Designing processes for sustainable agriculture is therefore a major research goal.

Agricultural production processes have been analysed as industrial processes for almost 20 years now (Sebillote and Soler, 1988). In this context, a tradition for agricultural production process simulation has grown, for example (Attonaty et al., 1994; Cros et al., 2001; Bakam et al., 2001). Many paradigms have been used for this purpose including discrete event system specification (DEVS) (Cournut and Dedieu, 2004). If Ziegler's DEVS (Zeigler et al., 2000) is a well known formalism for simulation, DES (for discrete event system) formalisms have also been shown to be suitable for qualitative analysis and control of various systems (Lunze, 2000).

Our research goal is to represent decision making in crop protection. Although most decision support systems in agriculture are rule based (Shaffer and Brodahl, 1998), our approach is process based and expertise based. We call it a decision workflow system (DeWS). We present here a design aid method for modelling expert decision making procedures, with a representation that belongs to the family of DES formalisms.

### 4.4.1.1 Rationales for the case study

The present case study deals with French vineyard. Vine growers consume about 25% of the pesticides used in France while vineyards' total area is 3% of the French farmland. Growers have developed intensive and mostly preventive crop protection techniques, because the high quality vine cultivar are highly susceptible to fungal pathogens and outbreaks are difficult to handle. Although pesticide use reduction efforts have been carried

e. Some materials used in the present article have been presented previously at the IFAC-MCPL2007 conference.

out, further development toward sustainable viticulture requires innovative protection methods. Integrated Pest Management (IPM) is one of those. IPM aims at reducing the amount of inputs while keeping the revenue of farms, through the use of biological control as well as pesticides when necessary (Kogan, 1998). Unfortunately, so far no biological agent has been made available against the main fungal pathogens to satisfy the production targets and the complexity of the grapevine pathosystem (i.e. crop + pathogen + climate). In addition, current lack of detailed epidemiological knowledge does not allow computation of optimal solutions for the design of minimal pesticide strategies. Therefore, pest management procedures need to be designed based on expertise.

An expert based IPM solution was designed by a team of phytopathologist experts from INRA plant health research unit (Bordeaux). We named it "GrapeMilDeWS" (Grapevine powdery and downy Mildews Decision Workflow System). GrapeMilDeWS aims at controlling two of the prevailing vineyard diseases: powdery mildew (*Erysiphe necator*) and downy mildew (*Plasmopara viticola*). It is based on the following hypothesis: knowledge and information can help to substitute numerous and systematic treatments.

GrapeMilDeWS was originally described with a set of guidelines, relying to a large extent on the personal (implicit) knowledge of its expert designers for implementation. It was experimented in the vineyard and demonstrated its efficiency with satisfactory harvest results and reduced number of phytopharmaceutical treatments (Cartolaro et al., 2007).

Formalising this DeWS should allow all necessary hidden knowledge to be made explicit. Modelling had indeed two objectives. First, the formal model would allow others phytopathologist to implement the decision process, thus permitting large scale experimentations. Second, a formal model can be used for computer simulations not only at the process' plot level but also at the farm level, for instance, to check for resources use, workload and induced costs at a larger scale. Finally, the formalisation process should also provide reflexive insight to the designer, allowing them to explore new design alternatives.

#### 4.4.1.2 Choice of formalism

In GrapeMilDeWS, the continuous behaviour of the pathosystem controlled by the crop protection process, are discretised in abstract variables which encode for low, moderate and high disease risk. Thus fed with discrete input, GrapeMilDeWS is modelled as a DES.

GrapeMilDeWS includes the user (or human being) in the control loop. For instance, the disease level are evaluated by workers in the vineyard. This information is aggregated in synthetic discrete values. In reaction to external events, transitions are fired, which generate internal and output events. The output events are decisions for action to be carried out on the system by human operators.

We chose to represent GrapeMilDeWS using a graphical language, i.e. Statechart, introduced in (Harel, 1987). As said by Harel, Statecharts can be described as enriched finite state automata (FSA):

statechart=
state-diagrams + depth + orthogonality + broadcast-communication

- *State-diagrams* are graphical FSA in the sense of both Mealy and Moore machines (Harel and Pnueli, 1985).
- *Depth* is the capacity of hierarchically encapsulating a statechart inside a state; this enable us to avoid confusing details and to get a more abstract view.
- *Orthogonality* is the ability to represent multiple concurrent processes in the same statechart, which is the main feature of statechart responsible for preventing the exponential growth of the state space as the systems complexity increases.
- *Broadcast communication* represents the way events are made available simultaneously to all elements of the statechart. This feature is powerful although it leads to formal difficulties in the implementation which are generally managed through various "flavours" of pseudo-synchronicity of the events (Maggiolo-Schettini et al., 2003). The original version of statechart, was implemented with broadcast communication over the whole system. However, in the UML object oriented version of statecharts, the inter-object communications are point to point (Damm et al., 2003).

We used Telelogic's Rhapsody statecharts (Harel and Kugler, 2004) for our modelling. Rhapsody implements a variant of the UML object semantic of statecharts. As all versions of statecharts are not equivalent, one should refer to (Crane and Dingel, 2007) and for an earlier, exhaustive view of statechart variants to (von der Beeck, 1994).

#### 4.4.1.3 A team work design

One of the difficulties for the formalisation of GrapeMilDeWS was that the design was a team work. The team members agreed upon the most important features, but there were many differences in their interpretation of the details. During the field experiments the decision making was done collegially. In order to make all of the process' knowledge explicit, it seemed that we would get richer details if we elicited the whole team, instead of singling out one of them and capturing his understanding of the process. They had designed GrapeMilDeWS together and it would have been a loss not to capture each specialized expertise's input.

### Objectives of the paper

The question can be stated: once an IPM solution for controlling the vine pathosystem has been designed using expert knowledge and statecharts have been identified as a pertinent modelling formalism; what is the method to elicit the knowledge from the experts? In other words: *can knowledge elicitation be done carried out using Statechart as a mediation model between the knowledge engineer and multiple experts.*

The paper will explore this hypothesis first by reviewing the literature on knowledge elicitation techniques involving multiple experts. The following section will present our elicitation method and its implementation in our case study. An evaluation of the quality of the elicitation is given in section 4.4.4. Before concluding, we discuss our experimental results.

## 4.4.2 Survey of Multiple Experts Elicitation Methods

The expert system community introduced knowledge elicitation of multiple experts as a scientific question. It is still a complex task, we give here a short presentation of

what is at stake. Acquiring knowledge from multiple experts has many pros and some cons. For instance, it may spare the necessity to find a domain "omniscient" expert, who would be replaced by a group of sub-domain specialists. The resulting global expertise would thus be wide and deep (i.e. detailed) and often with reduced access cost to expertise. On the other hand the cons range from schedule difficulties to power plays. If not properly managed, the latter may overcome the advantages (Chopra et al., 2000; Medsker et al., 1995).

When working with multiple experts the knowledge is generally distributed between the individuals according to three different configurations: either (i) the experts work together as a team; or, as in most cases, (ii) they are concurrent on a single "shared" competence domain; last, (iii) the expertise on a domain is spread: each expert holding partial expertise, with potential overlaps or knowledge gaps (Chopra et al., 2000; Medsker et al., 1995).

In case of concurrent expertise, the opinion of experts may diverge. Hameed et al. (2002) present a survey of the various classification grids that were proposed to analyse the discrepancies between experts, explaining how and why experts disagree or do not understand each other. Handling discrepancies between the experts' opinions can be tackled using the following strategies: (i) the *Tsar strategy*, giving the final word to a single expert; (ii) the *satellite strategy* which relies on one expert for most of the elicitation process but at times may turn to satellite experts for deeper insights on specific questions; finally (iii) the consensus is often pursued. The experts are generally asked to build consensus themselves, but some knowledge engineers developed automated consensus building methods (Barrett and Edwards, 1995; Gaines and Shaw, 1993).

Chopra et al. (2000) explain that "it is [...] possible to develop a consensus model of expertise through an iterative process of individual elicitation on a set of elements, assembly of the results and re-elicitation on the new set of elements". When experts are geographically dispersed, the Delphi method offers a solution using an iterative process over questionnaires. In use since the 1960's, this method has been thoroughly tested (Landeta, 2006). With the development of the internet, computer aided solutions (Mendonca et al., 2000) are a second alternative. Yet individual interviews remain the most efficient way of acquiring detailed knowledge, while group sessions are conducted with the goal to foster creativity (Chopra et al., 2000).

Elicitation can be performed using a number of techniques (Cannon-Bowers and Blickensderfer, 1993) and using different ones in the same process is recommended (Barrett and Edwards, 1995). Nowadays, a number of software suites offer many methods and tools (PROTEGE, 2007; PCPACK, 2007). Complete methodologies such as CommonKADS (Schreiber and Akkermans, 2000) or MOKA (Brimble and Sellini, 2000) have been developed over the last decade, and are advocated even outside of the knowledge management community (del Aguila et al., 2001).

Graphical methods have the great advantage to be both a graphic representation, with all the advantages of graphic variables (Bertin, 1973) as well as a mathematical model that

can be worked on with such methods as graph grammars or edge based metrics (Chopra et al., 2000). Interestingly, Statecharts have been tested in the field of psychology to represent people's mood, and turned out to be the panel's favourite elicitation technique (Milton et al., 2006). However using a graph as a media between experts and knowledge engineer is still controversial: Frederiks and van der Weide (2006) advocate for textual intermediation, whereas Ford and Sterman (1998) give a concrete example of a method using graph as the elicitation medium as well as the final result representation. In fact, graphical representations are primarily used to communicate final results (Chopra et al., 2000; Milton et al., 2006).

### 4.4.3 The elicitation method

Our method uses "intermediate knowledge models" in an iterative elicitation process, to ensure the mediation between the phytopathologists, designers of the GrapeMilDeWS, and the Knowledge Engineer (KE). These models are deemed "intermediate" because they allow us to go from an implicit knowledge to a "well formed automata" (i.e. models that may eventually be executed). Indeed, the modelling has two purposes: *transfer of skills and knowledge* and *validation of the expert-based decision process* with tools such as model-checking and simulations.

#### 4.4.3.1 Knowledge elicitation process

Most of the elicitation process is carried out through individual interviews of about one hour. Group sessions may also be setup. The KE prepares the subject of the interview and the documents needed.

A round consists of interviews of each expert and a synthesis is done to close the round after everyone was interviewed once. Rounds are repeated until more interviews do not improve the knowledge and the consensus level is high enough.

We define two kinds of statecharts. Let's call *diagrams* the "intermediate knowledge models" presented earlier. They are qualitative statecharts we used during expert interviews. Let's call *models* the formal statecharts we simulate with Rhapsody. *Diagrams* are informal statecharts oriented towards communication. They have to be understood by both the KE who writes them and the experts who are interviewed. On the other hand, the *models* are the complete computer implementation of the *diagrams* and can be executed in a simulator. The model will react to inputs and produce decisions of the same nature as that of the experts' ones.

Two kinds of diagram coexist at one time: the *admitted diagram* which is the current synthesis built in the previous round and the *individual diagram* which accounts for the work in progress incorporating the modifications from the individual interview of an expert onto the *admitted diagram*. Fig.4.1 summarizes the organization of interviews in rounds, and how the material generated at one round is used as basis for the next.

The changes proposed by the expert during an interview may be an addition, a deletion or a modification of several statechart elements.

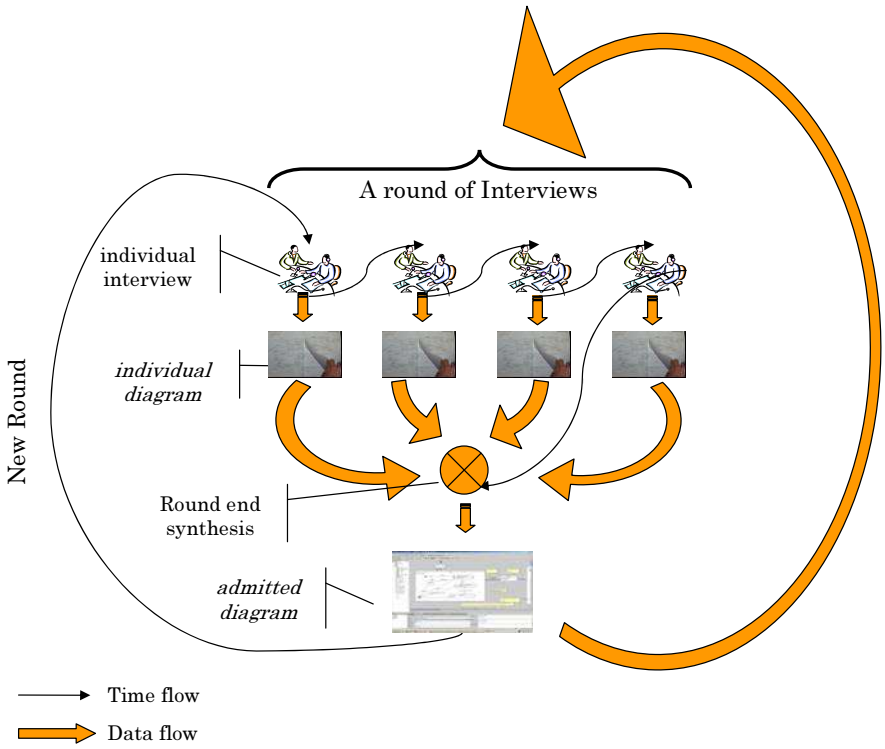


Figure 4.1: Rounds of interviews. The *individual diagrams* output after each interview are synthesised into the admitted diagram. The following round elicits more information on the basis of this *admitted diagram*.

*Interviews* shall pursue two goals. The first goal is to validate the admitted diagram. The idea is that the experts have a common understanding of GrapeMilDeWS. Therefore the level of consensus is supposed high enough so that most of the process would be agreed upon.

The second goal of the interview is to foray deeper into the details of the process. This deeper investigation allows the expert to build up on the settled knowledge from the previous rounds . The interviews are integrally recorded and an individual diagram is edited after each expert's interview.

*Round Synthesis* After each expert has been interviewed, the KE synthesizes the individual diagrams into a new admitted diagram. The synthesis is done mostly with the majority rule but in some cases, the KE uses his judgement to validate information that may not have the majority, but seems most likely to be the underlying truth [f]. Doing so, the KE has to make sure that the experts validate his intuition in the following round. When faced with divergent opinions or ill-defined concepts, the KE has to gather all the experts in a group session.

Synthesis was carried out manually, nevertheless it is theoretically possible to design an automated synthesis algorithm, using graph distance (Bae et al., 2006) and equivalence of the automaton using bisimulation (Massink et al., 2006).

f. we will come back to this in the discussion

*Group sessions* There is no specific procedure defined for group sessions. All the experts are invited. The goal of such sessions is to quickly settle points of discrepancies which could not be synthesized using the interviews information. It can also clarify ill-defined concepts on which individually the experts would not take a stance. After such meetings, individual validation is carried out at the next round.

#### 4.4.3.2 The elicitation experiment

Four experts composed the team. They hold different hierarchical positions in their organization and have had different levels of involvement during the design of GrapeMilDeWS and the experimental phase. Even though the original hypothesis was that the expertise was evenly shared among experts, the possibility of finding some specialization in the knowledge had to be acknowledged. Although all experts have a thorough knowledge of both diseases and of GrapeMilDeWS' principles, their personal involvement on each of these aspects was not equivalent. For instance Mr Pow. is a known powdery mildew expert and was more concerned about this disease. The other experts also tended to align their stance to his when questioned about powdery mildew.

The KE had preliminary discussions with all the experts. Based in these discussions and the documents available, a first informal statechart *diagram* was designed. The entire elicitation process was completed in four rounds of four interviews each. The two first rounds were done within a two week time. The experts were interviewed according to their schedule. That was an issue, for instance Mr Dow. was interviewed twice in a row: last in round 1 and first in round 2. During the last two rounds, scheduling was even harder: at one point, two interviews had to be performed in a day with the risk to bias an interview with the other. The last two rounds were performed in a three week time. The total time for the 16 interviews was 19 hours. Between rounds 1-2 and rounds 3-4, two group meetings were performed with the four experts. They lasted approximately two hours each.

#### 4.4.3.3 Results

Excluding the preliminary diagram, 16 diagrams were designed, one for each interview. For each round, an admitted diagram was designed synthesising the 4 diagrams created during the round. Two formal models were created, one from the admitted diagram after round 2, and the other one after round 4.

*Short description of elicited model for GrapeMilDeWS* The DeWS process breaks down in 7 stages which alternate with 3 evaluations (see fig.4.2). The statechart has been simplified for the publication. An evaluation is an activity state during which GrapeMilDeWS waits for data to be updated. A stage is a global decision state. In each state a substatechart displays the decision logic (see fig.4.3 with `Stage_4` as an example).

Nonetheless, a stage is also a temporal period defined by the phenology or by temporal conditions relative to GrapeMilDeWS' sequence of actions. For example, `Stage_3`'s activation is triggered by the flowering and its exit is triggered by the end of the active period [g] of the third treatment. `Stage_3`'s exit is dependent of the third treatment and of the choice of products, factors which are controlled by GrapeMilDeWS' decisions, i.e.

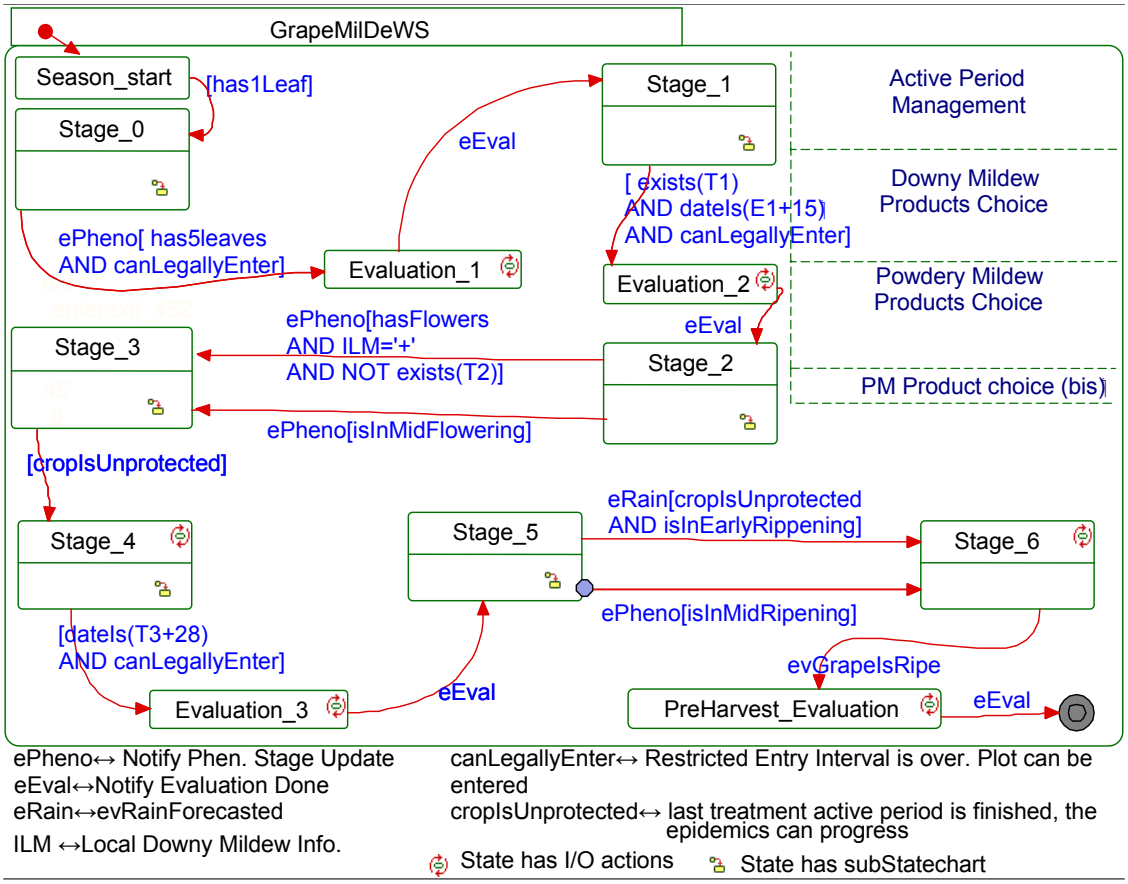


Figure 4.2: Main steps of the GrapeMilDeWS process for a cultural season

endogenous data whereas its entry is triggered exogenously by the phenological stage (see fig.4.2).

The choice of cutting the crop protection season in independent decision stages is novel in IPM and permits fine tuning the decision making strategy at each stage. The treatment decisions in the stages are taken according to three aggregated estimators. Two of them (O and M) are respectively, estimates of the level of powdery and downy mildew updated in the plot during the evaluations. The third estimator, ILM (for local downy mildew information), is provided by the plant health information services. It is updated by this exogenous source of information while the other two depend on GrapeMilDeWS for triggering their update(see fig.4.3).

GrapeMilDeWS models a control system, based on elicited expertise. It manages neither the operational resources needs nor the response of the vineyard. It is designed to simply "throw" requests, which the operational system (i.e. the grower) may fulfil "at its will". In return GrapeMilDeWS receives notifications about execution of requested actions (either evaluations or treatments). This design choice is realistic in viticulture as crop protection holds priority over all other activities. Thus the model is thought to be able to abstract from resource and priority management with little discrepancies.

g. Period after which the efficiency of the product is no longer guaranteed

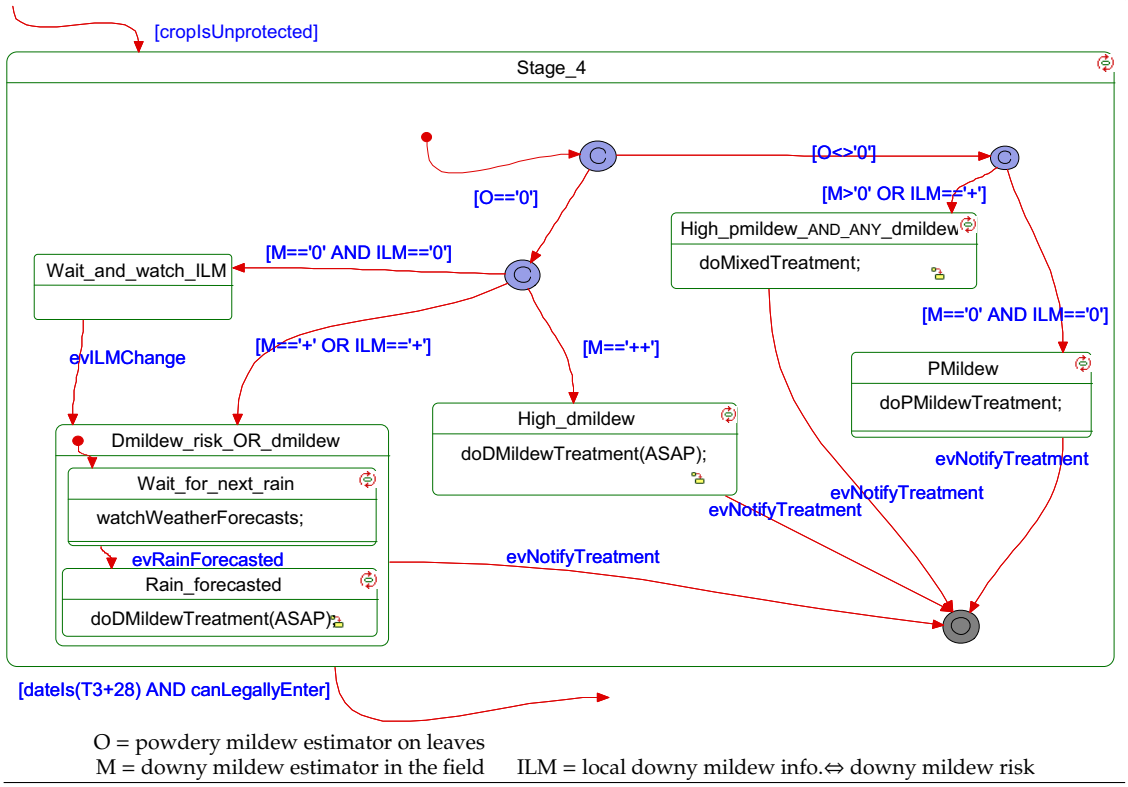


Figure 4.3: Details of Stage_4

### 4.4.4 Assessing the quality of the elicited knowledge

#### 4.4.4.1 A process conformance problem

After the elicitation was completed, the output *model* was marked as a stable version. Assessing the quality of the elicited model was needed to confront the elicited knowledge with the experts' actual practices during the experiments. This may highlight important facts that did not appear during the elicitation process or that the experts may not have been fully aware of.

The GrapeMilDeWS process had been tested for a couple of years prior to the formalisation described here. These tests were thus chosen as reference data to assess the *model* against. Checking the validity of a DeWS is a *process conformance* problem. Rozinat and van der Aalst (2008) tackle such conformance problems in a Petri Net (PN) workflows modelling framework. They play back the logged traces of the process execution into the PN and counting the tokens [h] that need to be added to or removed from the workflow model for it to reproduce the behaviour recorded in the logs. Edition count is thus an estimate of the conformance between the modelled workflow and the instance trace. We have adapted Rozinat and van der Aalst's approach by *replaying* the experiments' decision logs in the GrapeMilDeWS Statechart.

h. Petri Nets states are represented by the configuration of tokens into a graph of places and transitions.

#### 4.4.4.2 Adapting the process conformance problem to GrapeMilDeWS

We shall start by making analogies. Statechart models are systems which react to their environment and communicate through events. Here the experiments decision maker team (i.e. the experts) are assimilated to GrapeMilDeWS (i.e. its statechart representation). During the experiment, they communicated with the experimental vineyard manager using two kinds of messages: treatment requests and evaluation requests which we map onto GrapeMilDeWS' external events interface.

The experimental vineyard manager is seen as part of GrapeMilDeWS' surrounding environment. He was in charge of carrying out the orders from the experts. He sent execution notification messages in return to the experts' requests. These messages are mapped as evaluation and treatment execution notification events.

Because of the organisation of the work between the decision maker team and the vineyard manager, GrapeMilDeWS and its environment are modeled as asynchronous processes. GrapeMilDeWS was designed not to control the timing of its requests' executions.

For simplicity, we solely present here the replay of the main process. The product selection rules and active period monitoring were added for consistency. We abstracted ourselves from GrapeMilDeWS' other concurrent processes and internal communications. In this simplified framework, notifying an evaluation provides immediate update of the variables with no need for extra communications. The replays were done manually following Algorithm 1.

**Data**: Field experiments traces; GrapeMilDeWS Statechart model
**Result**: A chronogram : see Fig.4.1

Map *Experiment traces* onto GrapeMilDeWS' *events* and *variables*;

`/* On a chronogram over the season */`
**foreach** *day* **do** assign the day's events and set the variable values;
**foreach** *day* **do**
  **if** *input or output events exist* **then**
    Estimate *current Stage*;
    when treatments are ordered : select treatment product;
  Generate unrecorded output events;
**foreach** *day* **do** Compute the remaining active period length;
**foreach** *day* **do**
  Deduce and Set substates from Variables, AP, rain forecasts;
  Mark uncertainties;
Check constraint conformance, try modifying states to fulfil conditions, otherwise mark discrepancies;
**foreach** *day where uncertainties and discrepancies exist* **do** Make consistent hypothesis (on Pheno, AP or ILM) to solve uncertainty or discrepancy;
Mark remaining as Errors;

**Algorithm 1**: Replay a field experiment decision traces into main process state space

#### 4.4.4.3 Conformance of the model to the DeWS instances

We have processed the data from 16 experiments (8 plots, 2 years) with above-mentioned algorithms. However, we will illustrate the replay results of the experiment into the statechart with a single example: Table 4.1. This table represents a typical output

| 2006 Lat. M(10-18) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| # | Phenology | APD | APP | M | O | Og | ILM | Date | Transition Trig. Cond./ Output Events | i/o | Current State | Treatment target. |
| 1 | <15 | 0 | 0 | | | | | Mon 01/05 | | | Stage_0 | |
| 2 | [15;20] | 0 | 0 | | | | | Thu 04/05 | **(hyp)evNPSU[PhenStage>=15]** | **>** | Evaluation_1 | |
| 3 | | | | | | | | | evDoEvaluation | **<** | Evaluation_1 | |
| 4 | | | | 0 | + | | 0 | | evNotifyEvaluationDone | **>** | Stage_1.1 | |
| 5 | | 0 | 0 | 0 | + | | + | Tue 09/05 | evILMchange | **>** | Stage_1.2.1 | |
| 6 | [15;62] | 0 | 0 | 0 | + | | + | Tue 16/05 | evRF | **>** | Stage_1.2.2 | |
| 7 | | | | | | | | | evDoMixedTreatment | **<** | Stage_1.2.2 | |
| 8 | | 14 | 14 | 0 | + | | + | Wed 17/05 | evNotifyTreatmentDone | **>** | Stage_1.final_State1 | T1=D&P |
| 9 | | 12 | 12 | 0 | + | | + | Fri 19/05 | **(hyp)[ dateE10+15 ]** | **>** | Evaluation_2 | |
| 10 | | | | | | | | | evDoEvaluation | **<** | Evaluation_2 | |
| 11 | | 9 | 9 | + | + | | + | Mon 22/05 | evNotifyEvaluationDone | **>** | Stage_2.1.1 | |
| 12 | >62 | 2 | 2 | + | + | | + | Mon 29/05 | **(late BBCH>62) evNPSU[PhenStage>=60]** | **>** | Stage_3.1 | |
| 13 | | | | | | | | | evDoMixedTreatment | **<** | Stage_3.1 | |
| 14 | ]62;80] | 0 | 0 | + | + | | + | Wed 31/05 | [timeout(INTERVENTION_DELAY24_48H)] | **>** | Stage_3.1 | |
| 15 | | 14 | 14 | + | + | | + | Thu 01/06 | evNotifyTreatmentDone | **>** | Stage_3.Final_state3 | T3=D&P |
| 16 | | 8 | 8 | + | + | | 0 | Wed 07/06 | evILMchange | **>** | Stage_3.Final_state3 | |
| 17 | | 3 | 3 | + | + | | 0 | Mon 12/06 | **(3d early)[cropIsUnprotected()]** | **>** | Stage_4.1 | |
| 18 | | | | | | | | | evDoMixedTreatment | **<** | Stage_4.1 | |
| 19 | | 14 | 14 | + | + | | 0 | Fri 16/06 | evNotifyTreatmentDone | **>** | Stage_4.Final_State4 | T4=D&P |
| 20 | | 2 | 2 | + | + | | + | Wed 28/06 | evILMchange | **>** | Stage_4.Final_State4 | |
| 21 | | 1 | 1 | + | + | | + | Thu 29/06 | [treatmentDate(T3)+28 ] | **>** | Evaluation_3 | |
| 22 | | | | | | | | | evDoEvaluation | **<** | Evaluation_3 | |
| 23 | | | | + | | 0 | | | evNotifyEvaluationDone | **>** | Stage_5.1.1 | |
| 24 | | | | | | | | | evRF[cropIsUnprotected(dm)] | **>** | Stage_5.1.2 | |
| 25 | | | | | | | | | evDoDMildewTreatment | **<** | Stage_5.1.2 | |
| 26 | | 14 | 0 | + | | 0 | + | Fri 30/06 | evNotifyTreatmentDone | **>** | Stage_5.Final_State5 | T5=D |
| 27 | >80 | 0 | 0 | + | | 0 | + | Tue 18/07 | **(early BBCH<85) evNPSU[PhenStage>=85]** | **>** | Stage_6 | |
| 28 | | | | | | | | | evDoDMildewTreatment | **>** | Stage_6 | |
| 29 | | 10 | | | | | | | evNotifyTreatmentDone | **>** | Stage_6 | T6=D |

*Phenology* is the known Phenology on BBCH scale; *APD* is the Active Period against Downy mildew; *APP* is the Active Period against Powdery mildew;
*M* is the downy mildew assessed on leaves variable; *O* is powdery mildew variable assessed on leaves; *Og* is the powdery mildew assessed on bunches;
*ILM* is the area downy mildew risk estimate variable; *Transition Trig. Cond./ Output Events*: the transition labels (triggering event + guard condition) are followed by the incoming i/o symbol '>' and Output events are followed by the outgoing i/o symbol '<';
*Current State* is the active state name from the main process; *Treatment target* indicates which treatments target which disease.
*evNPSU* stands for evPhenologicalStageUpdate; *evRF* stands for evRainForecasted.

Table 4.1: Projection onto the GrapeMilDeWS state space of decisions taken in 2006 on plot Latresne Merlot

of the replay process. It has been summarized to show only the days where the system's current state evolves. Therefore most of the season's days are hidden although they are defined (e.g. the AP is decremented on a daily basis).

Note, that the phenology was imperfectly known, thus on Tab.4.1 the known phenological data are given as '$> x$' or '$< x$', whereas the uncertain phenology is given as intervals.

Note also, the bold lines which emphasize the marked uncertainties/discrepancies (see algo.1). In parenthesis are the reasons for the line to be marked.

For instance, Tab.4.1-L.2 column *transition trig. cond.* reads: "(hyp) evNPSU >". We knew that the evaluation had been done on that date, so we made the hypothesis that the phenology was 15 (5 leaves unfolded) that same day. This hypothesis permitted to satisfy a constraint without contradicting any fact.

Yet phenology is not the only case where hypothesis had to be made. For instance, the second evaluation (Tab.4.1-L.11) which triggers the entry into `Stage_2`, was actually performed 18 day after the first evaluation (Tab.4.1-L.4). Yet, the formal model requires that `Stage_1` be exited 15 day past the first evaluation. To prevent a conflict between the experiment and the model, we made the hypothesis that Stage_1 had been duly exited and `Evaluation_2` entered on May 19$^{th}$ (Tab.4.1-L.9), making the sequence consistent with the transition guard (see Fig.4.2): *[exists(T1) AND dateis(E1+15) AND canLegallyEnter()]*. We also assumed that the request for the evaluation had been sent (Tab.4.1-L.10), but that for four days, no action had been performed on the plot (keeping in mind that the operational management and the crop protection management are asynchronous processes). Finally it is a sure fact that on the 22$^{nd}$ the answer came back (evNotifyEvaluationDone) and therefore `Stage_2` was entered (Tab.4.1-L.11).

When no hypothesis could be made to have the traces and the model fit, we recorded the discrepancies and identified its nature, as is the case on June 12$^{th}$ when the order for a treatment corresponding to `Stage_4` was issued even though the condition to exit `Stage_3` (i.e. *[cropIsUnprotected()]*) was not yet true (Tab.4.1-L.18): as APD & APP >0, the active periods are not over and the crop is not yet unprotected. We marked the discrepancy and considered the system to be in `Stage_4` since the next action was to request a treatment which was required according to the variable values(see (Tab.4.1-L.17 and `Stage_4`'s statechart on fig.4.3).

With the process roughly presented here, we were able identify repeated discrepancies such as early entry in `Stage_4` or early treatment at `Stage_6` on a particular year. It was also highlighted that some constraints were repeatedly broken. We shall discuss these foundings in the next section.

### 4.4.5 Discussion

The process of elicitation presented in section 4.4.3 is our matured proposal. During the elicitation process, we encountered a number of difficulties that required adjustments.

For instance, due to the size of the model, the expert did not have the time to go through the whole statechart in the one hour interview time. The solution we adopted was to work in parallel both on validation and detail acquisition, but on parts of the model. The KE focused an interview on 2 to 3 Stages for validation and on 1 or 2 Stages

for knowledge acquisition. This was done making sure that an expert was not validating his own input from the previous round.

Barrett and Edwards (1995) warn against the tendency of experts to over rationalise their discourse. For example, a phenological stage was given as the condition for the transition between the first stage and the second evaluation. Yet the real constraint was in fact just a 15 day period between the first and second evaluation. This could be discovered only because the elicitation was pushed into deep details, examining specific input scenarios and because greater familiarity between the experts and the KE had been achieved. When confronted with such case of over rationalisation, we considered, when making a round synthesis, that the KE should not abide by the majority rule too strongly. We chose the information from the expert who gave the true condition and then had the other experts validate that point in the next round.

Admittedly, working directly on statecharts may have slowed the process of "coming to the truth". We had chosen, in the first place, to expose a lot of the implementation details of the formal specification to the experts with the idea that in this way they would be in close control of the formalisation of their expertise. The experts actually did not feel comfortable with the statechart diagrams and we believe it is partly due to too much formalism.

Glinz (2002) proposes a simplified semantics of Statechart for requirement elicitation. Our first idea was that the experts wanted a formal model to clarify their ideas but we found out that too much mathematical formalism was actually dumbing them down. In the latter part of the elicitation process, a more natural language was introduced in the statechart by encapsulating the implementation details. Hence elicitation through statecharts became easier. After changing to a more natural language on the labels of the transitions, we have seen some of the experts actually becoming more agile with the formalism. One was even able to take the pencil and make correction by himself.

In the end, having the admitted diagram as the basis for the model implementation made it very easy to implement and may have counter balanced the extra elicitation time required by the use of too formal statecharts early in the process.

The elicited process was assessed with the replay method presented in section 4.4.4. One of the difficulties in replaying the experiment traces into the statechart, was to build a partial order on the events that take place on the same date. The crop protection was recorded with a daytime granularity although more than one event could take place on the same day. Automating this task with a generic algorithm would have been very difficult if not impossible. However, we did the replay manually and we knew the logical sequence of events ( evaluation request ≻ evaluation notification ≻ treatment request ≻ treatment notification). Therefore we are able to replay the chain of transition on a same day from a stable situation changed by an input to a new one added by some output (see (Tab.4.1-L.22 to 25).

The replay results helped us point out the discrepancies (section 4.4.4.3), such as repeated constraint breaks at various stages and the early treatments at `Stage_4`. These constraint breaks were explained by the fact that the experts resynchronised the various plots. Early treatments at `Stage_4` showed that the experts anticipated certain decisions.

This points out a limit of the Statechart modelling language: it is a reactive system modelling language therefore ill fitted for anticipation (a common practice for operational managers).

Anticipation and synchronisation practices were not mentioned during the elicitation because GrapeMilDeWS was designed as a generic crop protection procedure abstracted from the operational constraints. However, when experimenting, the experts could not free themselves entirely from these operational constraints.

Highlighting these two points shade another light on the DeWS and helped the experts better analyse the process they had designed. It gave them food for thought for improving the future version of GrapeMilDeWS . We believe, that the elicitation method we have presented here is efficient in an incremental design methodology along with field experiments.

### 4.4.6 Conclusion and Future work

The improvement implemented in the latter part of the elicitation is comforting the possibility of using Statechart for process knowledge elicitation. This allowed us to move rapidly from expertise to formal models with little in between transformations. The process can be efficiently run when

- running multiple iterations of expert interviews
- expressing the constraint in natural language
- validating and acquiring details working on the last admitted diagram at each round
- setting up group meeting when the elicitation is vague from all participants; the experts probably need to clarify their concepts

However the lack of simple introspection mechanism in the statechart language, made anticipation difficult to represent. We would recommend the use of one of the many timed automata formalism, e.g. (Alur and Dill, 1994), to the decision and action periods. The non-determinism of these formalisms would be appreciated by the experts who were not comfortable with the strict timing constraints required in standard Statechart. Decision making is a versatile activity, and experts' favourite expression is : "it depends". Non-deterministic timed automata permits to both apply formal verification technologies and yet capture the temporal uncertainties for the decision making.

The next step is to formalise a method for synthesising diagrams. Ways must be found to compare statecharts models and to define equivalences between them. From the practitioner's point of view, finding a robust synthesis method to handle informal elicitation diagrams about temporal processes would also be a useful research.

**We would like to thank the domain experts:** Michel Clerjeau, Philippe Cartolaro, Lionel Delbac, Laurent Delière, for their time and assistance.

## 4.5 Discussion du chapitre

L'objectif principal de recueil et de formalisation de l'expertise a bien été atteint grâce à l'utilisation de statecharts comme médiation entre les experts et le cogniticien.

Au-delà de la démarche présentée dans l'article et des conclusions qui en découlent, on s'interroge sur trois autres aspects : d'abord la représentation de l'incertitude dans le modèle du POD, puis l'objectivation du POD et l'exploration de son espace d'état et enfin on évoque la généricité de l'approche de recueil des connaissances par les Statechart.

### 4.5.1 Mesure d'incertitude

Un des objectifs initiaux du recueil était de modéliser l'incertitude (le protocole prévu initialement est donné en annexe C). Certains des objectifs initiaux ont été abandonnés mais les principaux ont été conservés.

La mesure de l'incertitude a été abandonnée pour plusieurs raisons. J'avais estimé qu'il serait facile de gérer l'ajout et la suppression d'éléments du modèle associé aux déclarations des experts. Toutefois, les experts ne se sont pas appropriés le formalisme Statechart aussi aisément que prévu initialement (section 4.4). *In fine*, les idées développées par les experts ne se traduisaient pas « naturellement » en états, transitions ou conditions. Le passage d'un concept expert à sa retranscription sous forme Statechart est en effet lié à l'interprétation qu'en donne le cogniticien modélisateur et au choix d'organisation du modèle préalablement arrêté. Ainsi une nouvelle information, même très simple pour l'expert, peut amener le modélisateur à revoir profondément la structure du modèle alors que d'autres informations qui, elles, modifient grandement le comportement du processus modélisé, peuvent être retranscrites simplement par la modification d'une condition. Cette question relevait de la relation entre la sémantique experte et la sémantique du modèle, sujet trop vaste pour que j'entreprenne de la formaliser.

J'ai donc procédé empiriquement pour effectuer la synthèse des apports de chaque expert au terme de chaque étape. Je me suis appuyé sur la compréhension du processus acquise au cours des quatre entretiens composant le *round* et les notations reportées sur les diagrammes individuels. Scholz (2001) propose, à partir de ses travaux sur les $\mu-$chart (Scholz, 1998), une méthode incrémentale qui garantit mathématiquement que, d'une étape à l'autre, le modèle gagne en détail et que le comportement n'est pas modifié mais seulement précisé et permet d'aboutir à une spécification complète. Cette méthode structurée peut être utilisée pour aider le modélisateur, mais doit être adaptée au contexte du recueil de connaissances avec de multiples experts. En effet les experts émettent parfois des informations contradictoires ou remettent en cause le consensus précédent (ce que confirme la littérature : voir section 4.4.5). Or la procédure de Scholz n'est pas conçue pour gérer les évolutions d'exigence impliquant des changements de comportement du système.

L'autre argument qui explique l'abandon de la mesure de l'incertitude, est que le POD Mildium étant une solution innovante, on a souhaité produire une forme « canonique » explicite du système afin de la communiquer aisément et de permettre à d'autres chercheurs de s'en saisir. Introduire des mesures d'incertitude aurait rendu le modèle plus complexe à représenter et plus difficile à transmettre.

Cet argument peut également être invoqué à propos du choix de l'utilisation d'un idiome déterministe des Statecharts plutôt que d'une version temporisée du langage. Il semblait plus avantageux *a priori* d'avoir un système très directif comme première approximation puis de le relaxer en introduisant de l'incertitude. Cette hypothèse s'est avérée contraignante dans le recueil de connaissances comme nous l'évoquions dans la section 4.4.5, car elle a forcé les experts à faire des choix arrêtés là où « des degrés de liberté temporels » existaient. On verra les conséquences de ce choix dans le chapitre 6.

### 4.5.2 Objectivation du POD et recherche d'exhaustivité

Initialement, deux types de jeux de rôles étaient prévus pour vérifier la cohérence du recueil et pour explorer plus avant. Le premier type devait proposer des scénarios de condition (par exemple : M='+'et O='++'au stade 18) avec comme objectif de valider la cohérence interne du modèle. Ce premier type de jeu a été abandonné car le contrôle de cohérence a été réalisé par les experts dans la première partie des entretiens. (version du protocole de recueil présentée dans l'article)

Le second type de jeu devait permettre d'explorer des configurations non visitées du pathosystème (c'est à dire des cas d'épidémies annuelles pour lesquelles Mildium n'avait pas été expérimenté). Ce type de jeu s'est avéré difficile à mettre en place faute d'expertise du cogniticien pour construire les scénarios sur lesquels faire jouer les experts concepteurs. Les cas de figures non envisagés ont dès lors été imaginés avec les experts pendant la partie exploratoire des entretiens individuels.

On a procédé de la manière suivante : le modélisateur interrogeait un expert sur une configuration des variables de décision pour laquelle le système lui semblait mal défini. L'expert retranscrivait mentalement les conditions réelles qui pouvaient amener le POD dans cet état, et définissait la ou les sous-procédures à mettre en œuvre dans cette situation. Les experts ont procédé spontanément à cet exercice, leur connaissance du métier leur permettant d'identifier les cas de figure non définis de manière plus rapide que le modélisateur.

Cette approche souligne bien que mon travail se place à la fois dans le cadre du recueil de connaissances (un POD pré-existait à sa modélisation), mais également dans le cadre de l'aide à la conception. En effet le travail de recueil de connaissances a été l'occasion de concevoir des solutions à des cas de figures inédits.

Pour les itérations à venir du POD Mildium, le nombre de partie prenante à la conception pourrait dépasser la dizaine d'experts-expérimentateurs. Le recours au jeu de rôle pourrait s'avérer une procédure intéressante, le second jeu de rôle servirait notamment pour faire émerger les meilleures idées.

### 4.5.3 Eléments de généricité de la méthode de recueil

Une unique expérimentation, avec de multiples experts, a été rapportée dans l'article, mais ce protocole de recueil de connaissances a également été utilisé dans le cadre du projet ADD-Vin, avec un seul expert, afin de modéliser les stratégies annuelles de protection phytosanitaire des quatre viticulteurs du réseau ADD-Vin. Un exemple de ces modèles est présentés en annexe D. Ils ont été recueillis non pas auprès des agriculteurs

eux-mêmes, mais auprès de P. Marandet[i] qui était le truchement entre l'agriculteur et le modélisateur. En effet, la technique de recueil par Statechart étant restée étrangère à certains experts pathologistes, il nous semblait hasardeux de tenter l'expérience avec les viticulteurs.

Le recueil réalisé avec P. Marandet s'est avéré extrêmement rapide : là où seize entretiens avaient été nécessaires pour recueillir le POD Mildium, il n'en a fallu que trois pour recueillir chaque « modèle d'action viticulteur » et ce alors même que ces modèles prennent en compte les stratégies mildiou, oïdium, excoriose, pourriture grise et insectes (avec parfois une stratégie spécifique pour la flavescence dorée). Ces modèles n'ont toutefois qu'une valeur descriptive puisqu'ils n'ont pas fait l'objet d'une validation auprès des viticulteurs dont ils représentent la stratégie phytosanitaire.

Cette seconde expérience m'a permis de tester la validité de la méthode dans d'autres contextes décisionnels. On verra en conclusion (chap. 7) l'importance de cette généricité pour d'autres domaines d'application (génie des procédés alimentaire)

## 4.6 Conclusion du chapitre

La technique mise en œuvre a permis d'expliciter de manière exhaustive le POD Mildium, de le formaliser et a aidé les experts à préciser le comportement attendu pour les cas qu'ils n'avaient pas encore eu l'occasion d'expérimenter. Le chapitre suivant présente le résultat de ce recueil, à savoir le modèle du POD Mildium.

i. Élève ingénieur stagiaire de l'UMR Santé Végétale