Simulation of Action in Production Systems

Guerrin, F.

Inra/Cirad, UPR Risque Environnemental et Recyclage, BP 20, 97408 Saint-Denis, Reunion Island, France
E-Mail: guerrin@cirad.fr

Keywords: *Action representation; Activity simulation; Hybrid dynamical system; Temporal logic; Production system.*

EXTENDED ABSTRACT

The attempt to use simulation models as management-support tools puts human decision and action to the fore. Although it is well-known that there is a strong intricacy between decision and action, action representation is here the focus. A formalization to implement in simulation models this concept is proposed and discussed in the light of the 'situated action' paradigm (Suchman 1987), Allen's theory of action and time (Allen 1984), and BRAHMS, a model to simulate people's actual practice (Sierhuis 2001).

This tentative theory originates in modelling and simulation experiences in the field of agricultural production systems. These systems are dealt with at various scales of observation: from livestock enterprises or crop plots to whole-farm systems or groups of farms (Guerrin and Paillat 2003). Within such systems, material (and information) fluxes are issued from processes operated by human agents or natural causes. Two types of fluxes are distinguished: those mainly driven by human agents (workable fluxes) and those mainly driven by natural causes (biophysical fluxes). These fluxes interact through human action that aims at orienting biophysical fluxes by acting on workable fluxes.

The emphasis put on action simulation is justified by how is conceived the use of models in decision-support for managing such systems. Putting aside the prescriptive approach (the model provides the user with the decision) a simulation model is thought of as a reflexive tool aimed at fostering experimentation and apprenticeship by the user on its own practice. What-If? simulation mirroring the interplay of intended actions within the system is deemed useful to support stakeholders' decision-making (Mc Cown 2002). Hence, the model needs not represent the decision cognitive process, but rather, 'what' is being done in fact. The main concern is thus to simulate the actions and their consequences resulting from scenarios described in terms of situations, plans, management rules, constraints, to help the user compare policy trade-offs.

The modelling ontology of action proposed here generalizes and builds upon the features developed within two dynamic simulation models applied to livestock waste management: MAGMA (Guerrin 2001), which simulates the application on crops of manure from various livestock in a one-to-many or many-to-one fashions; APPROZUT (Guerrin 2004; Guerrin and Médoc 2005), which simulates the deliveries of slurry from multiple pig farms to a unique treatment plant in a many-to-one fashion.

An action is represented as a dynamic process by a binary function of time. Action may be singular (occurring once) or cyclic (repeating occurrences over time). The state of an action (0 or 1 values holding on time intervals) is distinguished from the temporal events bounding its occurrences. These are quasi-instantaneous state transitions: 0→1 determining the start dates of actions' occurrences; 1→0 their end dates. They are generated by changes in other processes playing the role of triggering or interrupting conditions. As long as these changes are not detected, action is maintained in its current state. These processes, continuous or discrete, may be a combination of predefined schedules or clocks, external processes accounting for the environment, or other actions. In turn, an action exerts an immediate or delayed effect on target processes (e.g. fluxes controlling stocks) and system performance indicators. This binary formalization of action gives rise to the use of propositional or predicate calculus to reason upon action in a dynamical system framework. The management of actions involves mainly action coordination. It can be achieved by several means namely planning, action composition, and allocation over time of continuous or discrete resources shared by concurrent actions according to their demands and priorities. An advantage of this dynamical system approach is to ease the connection of action models with classical dynamic models accounting for the biophysical processes at work in production systems.

The mathematical functions used to represent these concepts are given and their use is illustrated, for the sake of clarity, on simple toy-example simulations. However, references are made to real issues from livestock effluent management experimented with the MAGMA and APPROZUT models. This modelling ontology of action (still under work) has been implemented in the Vensim simulation software based on systems dynamics.

1. INTRODUCTION

By ‘action’, we mean what is actually performed by any kind of agent (human or artifact). It is used as synonymous with ‘activity’, though this term sometimes denotes composite action. It is distinguished from the concept of ‘task’, understood as a functional specification of action for prescriptive or explanatory purposes. Action is viewed as a process embodied in the biophysical world and thus situated in space, time and society. In ‘situated action’ (Suchman 1987) the emphasis is put on interaction between the agents and their environment. The notions of ‘plan’ and ‘goal’, that are central in the problem-solving approach (find the sequences of actions to reach a predefined target state), are deemed insufficient to simulate action in people’s practice. A plan is rather viewed as a resource for action, not as its sole determinant, and ‘motives’ are substituted for ‘goal’ (Clancey 2002; Suchman 1987). With this in mind, we describe hereafter a tentative modelling ontology of action: How to represent it? (section 2); How to reason about it? (section 3); How to manage it? (section 4). A comparison of this approach with Allen’s theory of action and time (Allen 1984) and BRAHMS, a model to simulate people’s action in practice (Sierhuis 2001) ends the paper (section 5).

2. REPRESENTATION OF ACTION

2.1. General Representation of Action

Any action A is represented as a binary pulse function $s_A(t)$ dependent on a condition $\mathrm{C}_A(t)$:

$$s_A(t) = 1 \text{ if } \mathrm{C}_A(t), \quad 0 \text{ otherwise} \tag{1}$$

$\mathrm{C}_A(t)$ is a time-varying logical proposition (false or true if evaluated to 0 or 1) defined according to variable states or explicit temporal parameters. Hence, an action is a dynamical process described with state values 1 (as long as C_A holds) or 0 (as long as C_A does not hold) over a succession of temporal intervals. Each interval over which $s_A(t) = 1$ defines an occurrence of A (action A ‘holds’). Its bounds are determined by state transitions caused by quasi-instantaneous events denoting a change in the evaluation of $\mathrm{C}_A(t)$ (Figure 1):

- 0 → 1: action occurrence starts at $t = t_A^-$;
- 1 → 0: action occurrence ends at $t = t_A^+$.

As in computer simulation time representation is discrete, by convention, the temporal extent of an action occurrence is the semi-closed interval $[t_A^-, t_A^+)$ with duration $\tau_A = t_A^+ - t_A^-$.

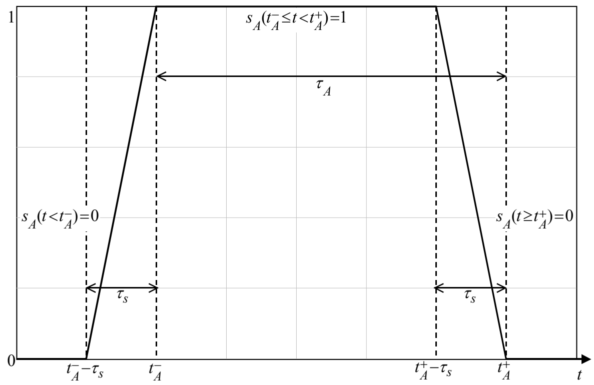

Figure 1. Singular occurrence of action as a temporal interval bounded by its start and end dates.

Hence, the duration of an event is the value τ_s of the simulation time-step that must be carefully chosen with respect to real-time. Actions must also be defined at a grain-size relevant for the model purposes, namely in terms of effects to be accounted for (an action remains ‘primitive’ as long as it is not further decomposed). Transforming an action into effects is made by the product of its state function with a parameter, e.g. a base flow (work time available daily in MAGMA; tanker inflow in APPROZUT), introducing or not a delay.

2.2. Action as a Function of Temporal Bounds

It is assumed here that temporal parameters of action A (start date t_A^-, end date t_A^+, duration τ_A) can be anticipated during the simulation. Determining the dates of the starting (resp. ending) events of action A’s occurrences is done by sampling the time values at which holds a condition $P_A^-(t)$ (resp. $P_A^+(t)$) for triggering (resp. ending) action A:

$$t_A^\pm(t) = \begin{cases} t & \text{if} \quad P_A^\pm(t) \\ t_A^\pm(\max(0, t - \tau_s)) & \text{otherwise} \end{cases} \tag{2}$$

where the superscript $\pm$ denotes the variables relative either to starting (–) or ending (+) an action occurrence and τ_s is the simulation time-step.

$P_A^\pm(t)$ is, like $C_A(t)$, a proposition evaluated to 0 or 1. Note that distinguishing between these propositions allows the conditions to starting, ending or continuing an action to be different. As long as $P_A^\pm(t)$ does not hold, $t_A^\pm$ remains constant (i.e. equal to its value at the preceding time-step). As soon as a triggering event is detected (evaluation of $P_A^\pm(t)$ turns from 0 to 1) a date is worked out that remains the same until a new event occurs. Therefore $t_A^\pm(t)$ describes a stepwise evolution whose value at a time is the date of the previous event, i.e. $\forall t, t_A^\pm(t) \le t$.

This provides the dynamical system some memory for reasoning upon action. Therefore, based on the start and end dates worked out with Eq. (2), the general condition for A is:

$$C_A(t) \equiv t_A^-(t) > t_A^+(t) \qquad (3)$$

that is, A occurs as soon as its starting event is triggered and until its next ending event occurs (see Figure 2).

Using Eq. (2), three cases arise in a dynamical context: (i) the start and end dates are both triggered independently (see example section 2.3); (ii) the start date is triggered and the end date deduced by: $P_A^+(t) \equiv (t = t_A^- + \tau_A)$ (see example below); (iii) the end date is triggered and the start date deduced by: $P_A^-(t) \equiv (t = t_A^+ - \tau_A)$. For the sake of realism, it is necessary to explicitly introduce a delay $\tau_A^\pm \geq 0$ on the processes generating the starting or ending events. In case (i) A's duration results from independent starting and ending conditions, thus the delays τ_A^- and τ_A^+ may be computed independently. In case (ii), as the end date is deduced from the start date, if t_A^- is delayed ($\tau_A^- > 0$) so will be t_A^+. In case (iii), as t_A^- is deduced from $t_A^+ > t_A^-$:

- if $\tau_A^+ \geq \tau_A$, t_A^- can be computed in time for action duration be guaranteed;
- if $0 \leq \tau_A^+ < \tau_A$, t_A^- cannot be computed in time and action duration is reduced to the delay τ_A^+; particularly, when $\tau_A^+ = 0$, then $\tau_A = 0$ and thus, no action occurs.

When determined independently, $P_A^\pm(t)$ may be computed according to: a lookup table encompassing a pre-established schedule (e.g. harvests in MAGMA; personnel's holidays in APPROZUT); a 'clock' variable, allowing repeated actions to be computed according to a periodic function (e.g. the modulo function returning the remainder of t/τ_c with $\tau_c > 0$ the clock's period). Figure 2 shows an example where the triggering condition $P_A^-(t)$ holds when a 10-day period clock equals zero augmented by a delay $\tau_A^- = 0.5$ day, the end date is deduced from the start date and the duration of action is given as a logarithmic function of time.

2.3. Action as Function of External Processes

In contrast with the above example, it is assumed here that neither the start nor the end dates can be anticipated but they both depend upon some external process. In this case, action is driven in a reactive fashion.

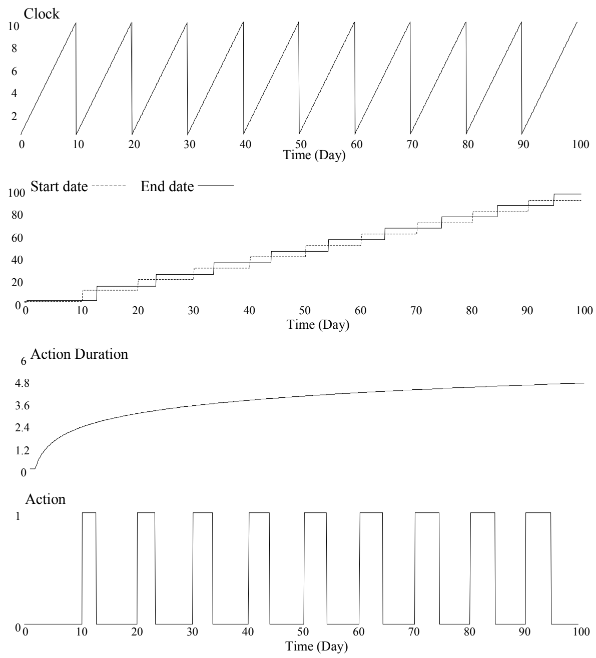

Figure 2. Action triggered by a clock at fixed periods with increasing durations (for $t > 0$).

For example, let $V(t)$ denote a stock level evolution, defined as the integration over time of its rate of change (inflow – outflow). Assume the actual inflow is discontinuous and determined by action A described above (section 2.2) jointly with a base inflow of 10 units/day. Thus: $q_i = 10 \times s_A(t)$. Now we want an action B exerted on the outflow ($q_o = 10 \times s_B(t)$) to control the stock between an upper and a lower threshold $\mathrm{V_{sup}} > \mathrm{V_{inf}}$ using the following policy: B starts as soon as $V(t) \geq \mathrm{V_{sup}}$ and holds until $V(t) \leq \mathrm{V_{inf}}$. Using Eq. (2), and assuming no delay for simplicity ($\tau_A^\pm = 0$), the start and end dates of B are:

$$t_B^-(t) = \begin{cases} t & \text{if } \left(V(t) \geq \mathrm{V_{sup}}\right) \\ t_B^-(\max(0, t - \tau_s)) & \text{otherwise} \end{cases} \qquad (4)$$

$$t_B^+(t) = \begin{cases} t & \text{if } \left(V(t) \leq \mathrm{V_{inf}}\right) \\ t_B^+(\max(0, t - \tau_s)) & \text{otherwise} \end{cases} \qquad (5)$$

The start date (i.e. upper threshold crossing) is known before the end date, whereas the end date remains unknown until the lower threshold is reached, i.e. as long as B occurs. Instead, it is the end date from the previous occurrence of B which is known during that time. Thus, by Eq. (3) the condition for B is: $C_B(t) \equiv t_B^-(t) > t_B^+(t)$. Due to the conditions used (stock above or below thresholds), the $t_B^\pm$s are not unique as in Figure 2. We get instead a new date at each time-step as long as the starting or ending condition holds (Figure 3).

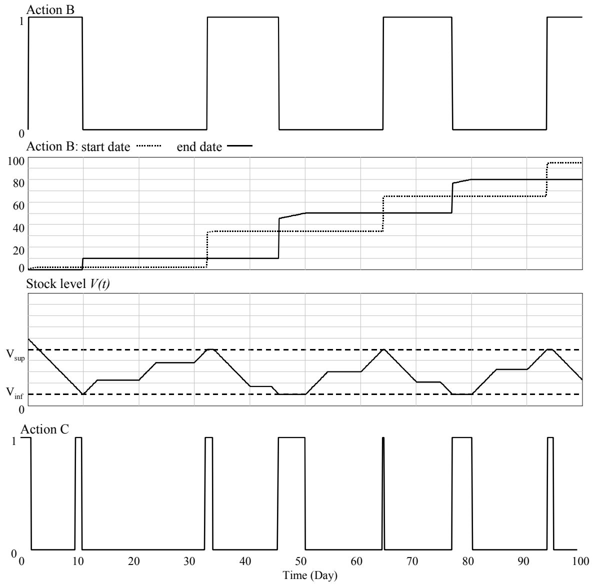

Figure 3. Feedback control of stock by action *B* according to upper and lower thresholds; action *C* is computed independently (see text).

Determining the start and end dates of action occurrences is not always required. Verifying that some condition holds on some variable's evolution may be sufficient. For example, specifying an action *C* that should execute as long as $V(t)$ remains above the upper threshold or below the lower, may be simply computed by stating the condition:

$$C_C(t) \equiv \left(V(t) \geq \mathrm{V_{sup}}\right) \vee \left(V(t) \leq \mathrm{V_{inf}}\right) \qquad (6)$$

The resulting evolution of *C* is given in Figure 3.

It is implicitly assumed in this example that the stock level is the only variable determining action. Taking into account other determinants (e.g., resource availability, temporal constraints, environmental conditions, other actions, etc.) is possible by making more complex the conditions for action (many examples are in MAGMA and APPROZUT).

2.4. Action as a Function of Other Actions

Here is considered the case when an action *E* is to be derived from an action *D* according to some constraints. As constraints, the well-known Allen's temporal binary relations are used (Allen 1984). Among the 13 possible relations, we deal only with the 8 meaningful relations in a dynamic context where *E* cannot start before *D*. Inverse relations could be similarly built in the reverse case (given *E*, derive *D*). Note that Allen's relations are mutually exclusive (not two relations may hold at the same time) and given any two action occurrences, one relation should hold.

Determining afterward *D*'s start and end dates is done with Eq. (2) by detecting when the corresponding events occur:

- Start date is when *D* holds whereas it did not hold at the preceding time-step: $P_D^-(t) \equiv \left(s_D(t)=1\right) \wedge \left(s_D(t-\tau_s)=0\right)$;
- End date is when *D* no more holds whereas it did at the preceding time-step: $P_D^+(t) \equiv \left(s_D(t)=1\right) \wedge \left(s_D(t-\tau_s)=0\right)$.

The duration of *D*'s occurrences can be derived by computing the difference $\tau_D = t_D^+ - t_D^-$ on the temporal intervals when $t_D^+ > t_D^-$, then holding the same value when $t_D^+ \leq t_D^-$ (i.e. while action is occurring and the next t_D^+ is still unknown). Action *E* can then be derived by Eq. (3) after working out its start and end dates from *D*'s start date by Eq. (2). It can also be derived more simply by Eq. (1) subject to condition:

$$C_E(t) \equiv \left(t_D^- + \tau_{D,E}^- \leq t < t_D^- + \tau_{D,E}^- + \tau_E\right) \qquad (7)$$

where t_D^- is the start date of *D*, τ_E is the duration of *E*'s occurrences, and $\tau_{D,E}^- \geq 0$ the delay after which *E* can start with respect to *D*'s start date. The constraints to be satisfied by these parameters for *D* and *E* to comply with Allen's relations are listed in Table 1.

Table 1. Allen's relations and temporal constraints to derive action *E* from *D* (τ_D, τ_E : duration of actions *D* and *E*; $\tau_{D,E}^-$: starting delay of action *E* with respect to *D*'s start date).

Relation	Diagram	Constraint
DURING(*E,D*)	D E	$(0 < \tau_{D,E}^- < \tau_D) \wedge (\tau_E < \tau_D - \tau_{D,E}^-)$
STARTS(*E,D*)	D E	$(\tau_{D,E}^- = 0) \wedge (\tau_E < \tau_D)$
STARTS(*D,E*)	D E	$(\tau_{D,E}^- = 0) \wedge (\tau_E > \tau_D)$
FINISHES(*E,D*)	D E	$(0 < \tau_{D,E}^- < \tau_D) \wedge (\tau_E = \tau_D - \tau_{D,E}^-)$
EQUAL(*D,E*)	D E	$(\tau_{D,E}^- = 0) \wedge (\tau_E = \tau_D)$
BEFORE(*D,E*)	D E	$(\tau_{D,E}^- > \tau_D)$
OVERLAPS(*D,E*)	D E	$(0 < \tau_{D,E}^- < \tau_D) \wedge (\tau_E > \tau_D - \tau_{D,E}^-)$
MEETS(*D,E*)	D E	$(\tau_{D,E}^- = \tau_D)$

3. REASONING ABOUT ACTIONS

3.1. Using Allen's Logic

Given any two actions D and E, the aim here is to find out which Allen's relation hold between them. This may be done by establishing the order between the temporal parameters of actions which must hold for the relation to hold. However, in a dynamical simulation context, any (D,E) relation can only be considered when both D and E hold (i.e. for $t_D^- \le t_E^- \le t$, with t the current time value). Moreover, whether or not this relation holds can be known only from a special time-point; and this fact is kept, by convention, until the next event. For example, DURING(E,D) can be known only once E is completed (at t_E^+) and this fact may be kept until D ends (at t_D^+), that is for $t \in \left[t_E^+, t_D^+\right]$. These temporal orders and extents are given in Table 2.

Table 2. Temporal order on start and end dates for actions (E,D) to verify Allen's relations and temporal extent over which they are kept holding.

Relation	Order	Extent
DURING(E,D)	$t_D^+ < t_D^- < t_E^- < t_E^+$	$\left[t_E^+, t_D^+\right]$
STARTS(E,D)	$t_D^+ < t_E^- = t_D^- < t_E^+$	$\left[t_E^+, t_D^+\right]$
STARTS(D,E)	$t_E^+ < t_E^- = t_D^- < t_D^+$	$\left[t_D^+, t_E^+\right]$
FINISHES(E,D)	$t_D^- < t_E^- < t_D^+ = t_E^+$	$\left[t_E^+, t_D^-\right]$
EQUAL(D,E)	$t_D^- = t_E^- < t_D^+ = t_E^+$	$\left[t_E^+, t_D^-\right]$
BEFORE(D,E)	$t_E^+ < t_D^- < t_D^+ < t_E^-$	$\left[t_E^-, t_E^+\right]$
OVERLAPS(D,E)	$t_E^+ < t_D^- < t_E^- < t_D^+$	$\left[t_D^+, t_E^+\right]$
MEETS(D,E)	$t_E^+ < t_D^- < t_D^+ = t_E^-$	$\left[t_E^-, t_E^+\right]$

3.2. Using Propositional Logic

Using $\langle +, \cdot \rangle$ or $\langle \mathrm{Max}, \mathrm{Min} \rangle$ on $\{0,1\}$, we can define Boolean operations on the state functions of actions $A_i \in A \quad (i = 1,...,n)$:

$$\begin{aligned}
&\forall A_i \in A, \quad s_{\neg A_i}(t) = 1 - s_{A_i}(t) \\
&s_{\wedge A}(t) = \prod_{i=1}^{n} s_{A_i}(t) = \min_i (s_{A_i}(t)) \\
&s_{\vee A}(t) = \sum_{i=1}^{n} s_{A_i}(t) - \sum_{i=1}^{n-1} s_{A_i}(t) \cdot s_{A_{i+1}}(t) = \max_i (s_{A_i}(t)) \\
&s_{\underline{\vee} A}(t) = s_{\vee A}(t) \cdot (1 - s_{\wedge A}(t))
\end{aligned} \tag{8}$$

with the following equivalences:

- Negation $\neg A_i$ holds when A_i does not;
- Conjunction $\wedge A$ holds when all A_i holds;
- Inclusive disjunction $\vee A$ holds when at least one A_i holds;
- Exclusive disjunction $\underline{\vee} A$ holds when exactly one A_i holds.

These operations may also be conveniently derived from a superposition operation ΣA, summing up the A_is and thus taking integer values onto $\left[0, n\right]$:

$$s_{\Sigma A}(t) = \sum_{i=1}^{n} s_{A_i}(t)$$

$$\begin{aligned}
s_{\wedge\neg A}(t) = 1 \quad &\text{iff} \quad s_{\Sigma A}(t) = 0, \quad 0 \text{ otherwise} \\
s_{\wedge A}(t) = 1 \quad &\text{iff} \quad s_{\Sigma A}(t) = n, \quad 0 \text{ otherwise} \\
s_{\vee A}(t) = 1 \quad &\text{iff} \quad s_{\Sigma A}(t) > 0, \quad 0 \text{ otherwise} \\
s_{\underline{\vee} A}(t) = 1 \quad &\text{iff} \quad s_{\Sigma A}(t) = 1, \quad 0 \text{ otherwise}
\end{aligned} \tag{9}$$

Here, negation $\wedge\neg A$ is the conjunction of the negated A_is (it holds when no A_i holds). Superposition can also be used to compare actions either pair-wise (Hamming distance) or altogether:

$$d_A(t) = \begin{cases} 0 & \text{if} \quad \left(s_{\Sigma A}(t) = 0\right) \vee \left(s_{\Sigma A}(t) = n\right) \\ 1 & \text{otherwise} \end{cases} \tag{10}$$

Integrating $d_A(t)$ over time yields the duration for which some actions A_i differ from the others.

4. MANAGEMENT OF ACTIONS

Action management mainly involves action coordination which is crucial because a same agent may perform several activities or separate agents need cooperate. This can be achieved by many ways of which three are demonstrated hereafter: specification and execution of a 'plan' (in the wide sense of explicitly coordinated actions), action composition, and shared-resource allocation. Another means, coordination mediated by an artifact, was illustrated in section 2.3 by the stock example.

4.1. Plan Specification and Execution

Plans may be specified and simulated using Allen's relations described as temporal constraints (Table 1). For example, here is a plan involving 5 cascading actions: P={D, E, F, G, H}. Given D, actions E-H are defined directly or transitively by:

- E: DURING(E,D);
- F: STARTS(E,F) $\wedge$ OVERLAPS(D,F);
- G: FINISHES(G,F) $\wedge$ MEETS(E,G);
- H: BEFORE(G,H).

The plan repeats based on D's clock (Figure 4).

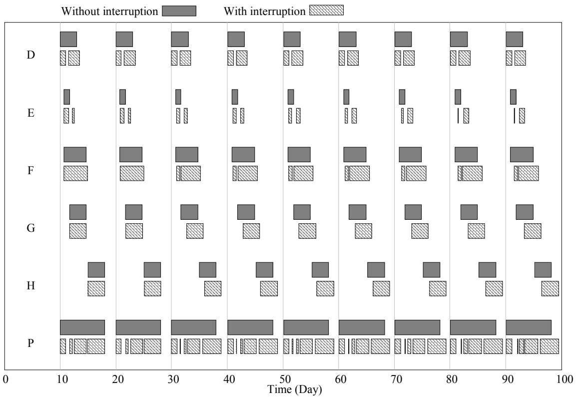

Figure 4. Cyclic execution of a plan $P=\{D, E, F, G, H\}$ based on Allen's relations with or without interruption (represented as a Gantt chart).

Introducing an interruption while D executes (thus splitting each D's planned occurrence in two) impacts the other actions: by splitting occurrences of E and F and by delaying G and H (Figure 4). Due to time shifting, some relations are fully (DURING, BEFORE) or partially (STARTS, MEETS, OVERLAPS) conserved, one is not (FINISHES). In the presence of interruptions, actions may be either cancelled, newly restarted, or resumed, conserving in that latter case their planned duration (as in the example Figure 4). Duration conservation is obtained, for each action, by the means of a local feedback controller defined by a binary function comparing the current action's duration to the planned one. E.g., the controller for action F is:

$$u_F(t) = \begin{cases} 1 \text{ if } \int_0^t \left[s_F(t) - s_F(\max(0, t - \tau_c)) \right] dt < \tau_F \\ 0 \text{ otherwise} \end{cases} \quad (11)$$

with τ_c the clock period of D. The definite integral computes the current duration of F within each plan cycle. F is controlled by working out F's state function $s_F(t)$ by Eq. (1), using as holding condition the conjunction $C_F(t) \wedge (u_F(t) = 1)$, where $C_F(t)$ is the condition for F as if no interruption occurred: F stops when its actual duration equates the planned one τ_F.

4.2. Action Composition

Abstracting the behaviours of separate actions into a single composite action allows groups of actions to be coordinated instead of multiple individual actions. Returning to the example in section 4.1, this can be done by time-slicing the plan into sub-plans with homogenous states. These sub-plans are formalized by conjunctions of a temporal constraint, action states, and action superposition applied to the set $P=\{D, E, F, G, H\}$ (time 't' is omitted in Eq. (12) for simplicity):

$$\begin{aligned} P_1 &\equiv (t_E^- < t_D^-) \wedge (s_D = 1) \wedge (s_{\Sigma A} = 1) \\ P_2 &\equiv (t_E^+ < t_E^-) \wedge (s_D = s_E = s_F = 1) \wedge (s_{\Sigma P} = 3) \\ P_3 &\equiv (t_D^+ < t_E^+) \wedge (s_D = s_F = s_G = 1) \wedge (s_{\Sigma P} = 3) \\ P_4 &\equiv (t_G^- < t_D^+) \wedge (s_F = s_G = 1) \wedge (s_{\Sigma P} = 2) \\ P_5 &\equiv (t_H^- < t_G^+) \wedge (s_{\Sigma P} = 0) \\ P_6 &\equiv (t_H^+ < t_H^-) \wedge (s_H = 1) \wedge (s_{\Sigma P} = 1) \end{aligned} \quad (12)$$

The state of the composite action P is given by the exclusive disjunction of sub-plans P_i:

$$s_P(t) = s_{\veebar\{P_1, P_2, P_3, P_4, P_5, P_6\}}(t) \quad (13)$$

P reflects the complete plan structure, i.e. the right sequence of right actions at the right dates, even with interruptions (Figure 4).

4.3. Shared-Resource Allocation

Managing actions competing for a shared resource at the same time is performed by allocating each action a part of the resource according both to its own demand and priority. This permits cancelling or delaying an action with lower priority or performing in parallel actions with the same priority. The resource may be continuous (as in MAGMA the work time available daily for spreading manure) or discrete (as in APPROZUT the transport capacity of slurry). Priority degrees are numbers (larger the number, higher the priority) assigned to actions according to management practices by an arbitrary scale of constants (as in MAGMA the priority of manure application to crops combining the priorities between the types of manure and the types of crops) or according to dynamic variables (as in APPROZUT the normalised level of slurry stock of pig farms combined with their herd size and distance to the treatment plant).

Given a set of actions $\{A_i\}$, allocation is performed, as described in Guerrin (2001), by computing the height of each action's demand as a stepwise function of the priority scale p:

$$h(i,p) = \begin{cases} d_i / W \text{ if } p_i - W/2 \le p \le p_i + W/2 \\ 0 \quad \text{otherwise} \end{cases} \quad (14)$$

with d_i the demand of action A_i, p_i its priority degree, and W a width parameter to work out its priority range $\left[p_i - W/2, p_i + W/2 \right]$. Then, the amount of resource allocated to action A_i is R_i, given by the surface under the height function comprised between the lower priority bound $p_{\min}$ below which no resource is left and the upper bound of the priority range of action A_i:

$$R_i = \int_{p_{\min}}^{p_i + W/2} h(i,p)\, dp \quad (15)$$

An example of the influence of various organisational choices expressed as priority rules on the workload distribution in a supply chain (pig slurry deliveries from multiple farms) is given in Guerrin and Médoc (2005) using the APPROZUT model.

5. DISCUSSION AND CONCLUSIONS

The modelling ontology presented here complies with many of the requirements deemed necessary by Allen (1984) and Sierhuis (2001) for representing action, although some features are lacking.

Conceived in the frame of dynamical systems, this ontology is firmly grounded in time with a representation that proved powerful for simulating processes and events. These 'occurrences', as in Allen's theory, are basically characterised by the temporal interval over which they hold. Representing complex situations and reasoning about temporally qualified propositions, events, actions and plans is possible either in an absolute or relative sense (i.e. with respect to one action's temporality instead of the base time line). Simulating sequential and parallel actions or events and their interaction is not a problem. 'Hand-off' actions, not directly related to a task (e.g., waiting, talking, resting,…), may be dealt with the same way that any action. Similarly, interrupting and resuming an action, delaying actions and effects, dealing with concurrent actions (using priorities), and assessing consequences on the system's organization, can be addressed in deterministic and random cases (see Guerrin 2001; Guerrin and Médoc 2005). As time moves forward in dynamical systems, if a clear notion of present (current time-step) and, to some extent, of the past (e.g. keeping track of previous events or delaying past events) do exist, the future is not yet apprehended. Integrating anticipatory features is a challenge for which possible solutions are foreseen (e.g. forecasting functions based on previous simulation outputs). An advantage of this dynamical system approach is also to ease the connection of action models with classical dynamic models accounting for the biophysical processes at work in production systems.

Distinguishing between actions and other process properties (e.g. action conditions) is not explicit in the formalization as both are represented by binary functions. Actually, an occurrence (event, process) is an 'action' if it is assumed to be caused by an agent (Allen 1984). This common binary formalization may give rise to some equivalence with propositional or predicate calculus in the framework of dynamical simulation. If forward chaining of If-Then rules is straightforward (as in BRAHMS, actions are based on a condition/action pattern; Sierhuis 2001), the practical interest of more sophisticated logical inferences deserves further attention. If some aspects of multi-tasking (coordinated parallel actions) and hierarchical organization of action can be represented, the concept of 'Agent' is still absent in contrast with BRAHMS where agents are individualized and endowed with communication and reasoning capabilities. Although the notion of goal is also absent (intentionally), further attention should probably be paid to the notions of belief, motive and intention as in BRAHMS and the BDI agent model (Rao and Georgeff 1995) to better reflect the psychological and social aspects of human behaviour (Clancey 2002). However, a cost-benefit analysis of such additional features should be made for decision-support in the context of agricultural production systems.

6. REFERENCES

Allen, J (1984), Towards a general theory of action and time, *Artificial Intelligence*, 23, 123-154.

Clancey, W. (2002), Simulating activities: relating motives, deliberation, and attentive coordination, *Cognitive Systems Research*, 3, 471-499.

Guerrin, F. (2001), Magma: A model to help manage animal wastes at the farm level, *Computers and Electronics in Agriculture*, 33(1), 35-54.

Guerrin, F. (2004), Simulation of stock control policies in a two-stage production system. Application to pig slurry management involving multiple farms, *Computers and Electronics in Agriculture*, 45(1-3), 27-50.

Guerrin, F. and J.-M. Médoc (2005), A simulation approach to evaluate supply policies of a pig slurry treatment plant by multiple farms. Efita-WCCA Joint Conf. on IT, Vila Real, Portugal.

Guerrin, F. and J.-M Paillat (2003), Modelling biomass fluxes and fertility transfers: animal wastes management in the Reunion Island. Modsim 2003, Townsville, Australia.

Mc Cown, R.L. (2002), Changing systems for supporting farmers' decisions: problems, paradigms and prospects, *Agricultural Systems*, 74 (1), 179-220.

Rao, A.S. and M.P. Georgeff (1995), BDI agents: from theory to practice, ICMAS, 1st Int. Conf. on Multi-agent Systems, San Francisco, USA.

Sierhuis, M. (2001), Modeling and simulating work practice, PhD thesis, Social Science and Informatics (SWI), University of Amsterdam, SIKS Dissertation Series N° 2001-10.

Suchman, L.A. (1987), Plans and situated actions: the problem of human-machine communication, Cambridge University Press.