Agent-Based Analysis and Support
for Incident Management

Mark Hoogendoorn[1], Catholijn M. Jonker[2], Jan Treur[1], and Marian Verhaegh[3]

[1]Vrije Universiteit Amsterdam, Department of Artificial Intelligence
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
{mhoogen, treur}@cs.vu.nl
[2]Radboud University Nijmegen, Nijmegen Institute of Cognition and Information
Montessorilaan 3, 6525 HR Nijmegen, The Netherlands
C.Jonker@nici.ru.nl
[3]Quartet Consult, Jaap Edenlaan 16, 2807 BR Gouda, The Netherlands
info@quartetconsult.nl

Abstract. This paper presents an agent-based approach for error detection in incident management organizations. The approach consists of several parts. First, a formal approach for the specification and hierarchical verification of both traces and properties. Incomplete traces are enriched by enrichment rules. Furthermore, a classification mechanism is presented for the different properties in incident management that is based on psychological literature. Classification of errors provides insight in the functioning of the agents involved with respect to their roles. This insight enables the provision of dedicated training sessions and allows software support to give appropriate warning messages during incident management.

1 Introduction

The domain of incident management is characterized by sudden events which demand immediate, effective and efficient response. Due to the nature of incident management, those involved in such processes need to be able to cope with stress situations and high work pressure. In addition to that, cooperation between these people is crucial and is not trivial due to the involvement of multiple organizations with different characteristics (e.g. police, health care, fire department). As a result of these difficulties, often errors occur in an incident management process. If such errors are not handled properly, this may have great impact on the successfulness of incident management.

Research within the domain of computer science and artificial intelligence is being performed to see whether automated systems can improve the current state of affairs in incident management (see e.g. [12]). One of the problems is that the information available is incomplete and possibly contradictory and unreliable. As a result, more advanced techniques are needed to enable automated systems to contribute an improvement of the incident management process.

This paper presents an agent-based approach to monitor, analyze and support incident management processes by detecting occurring errors and providing support

to avoid such errors or to limit their consequences. The approach is tailored towards the characteristics of incident management. First of all, the approach includes a method which deals with incomplete information. In addition, a diagnostic method based on refinement within the approach can signal whether certain required properties of the incident management organization are not satisfied, and pinpoint the cause within the organization of this dissatisfaction. The approach is based on the organizational paradigm nowadays in use in agent systems [1,4] which allows the abstraction from individual agents to the level of roles. Such an abstraction is useful as typically specification of the requirements in this domain is done on the level of roles (e.g. the police chief should communicate a strategy for crowd control). In case errors are observed in role behavior, they are classified to have more insight in what kind of errors are often made by a particular agent participating in the organization, in order to propose a tailored training program for this agent. In the future the approach as a whole can be incorporated in cooperating software agents for monitoring and providing feedback in training sessions, and software agents which can even monitor incident management organizations on the fly, giving a signal as soon as errors are detected, and providing support to avoid their occurrence or to limit their consequences.

Section 2 introduces the domain of incident management and, more specifically, the situation in the Netherlands. Thereafter, Section 3 introduces the formal language used to specify traces and behavior. Section 4 presents an approach for handling incomplete information by means of enrichment rules whereas Section 5 presents properties in the form of hierarchies for incident management organizations. Furthermore, Section 6 presents the classification scheme for errors, including specific incident management decision rules. Results of a case study are presented in Section 7 and finally, Section 8 is a discussion.

2 The Domain of Incident Management

In this Section, a brief introduction to the domain of incident management in the Netherlands is given. In the Netherlands four core organizations are present within incident management: (1) the fire department; (2) the police department; (3) health care, and (4) the municipalities involved. The first three parties mentioned each have their own alarm center in which operators are present to handle tasks associated with the specific organization.

A trigger for starting up an incident management organization is typically a call to the national emergency number, which is redirected to the nearest regional alarm center in which all three parties have their own alarm center. The call will be redirected to the most appropriate alarm center of the three parties. In case the operator of that alarm center considers the incident to be severe enough to start up the full incident management organization, he informs the alarm centers of the other organizations as well. Initially, the three alarm centers will send the manpower they think is appropriate for the incident reported. After the manpower has arrived on the scene, each part of the organization in principle acts on its own, each having a different coordinator of actions. In the case of the fire department this is the

commander of the first truck to arrive, for health care it is the paramedic of the first
ambulance and for the police there is no such coordinator as they have a supporting
role. Each of the coordinators are in charge until the dedicated operational leaders of
the organization arrive at the scene. The responsibilities of the organizations are
briefly described as follows: the fire department takes care of the so called "cause and
effect prevention", the health care organization is in charge of providing medical care,
and the police takes care of routing of the various vehicles and crowd control. After
the initial phase without structural coordination, an organization is formed in order to
coordinate all actions of the individual organizations in case this is still necessary.
The fire department is in charge of the operational side of this organization and the
mayor of the municipality is in charge of the policy part. The mayor is responsible for
the formation of the disaster staff for coordinating policy decisions, and is therefore
informed of the situation. The operational coordination structures are formed after
deliberation between the various parties on the scene has resulted in a mutual demand
for such a coordination structure. In case it is decided to form the operational and/or
disaster staff, the operators of the alarm centers start warning the relevant people. For
more details on the full coordination structure, see [8].

3 Modeling Method Used

This section describes the language TTL (for Temporal Trace Language) [6] used for
expressing dynamic properties as well as the expression of traces. Furthermore, the
language meta-TTL is introduced for second-order dynamic properties.

3.1 The Language TTL for Dynamic Properties

In TTL [6], ontologies for states are formalized as sets of symbols in sorted predicate
logic. For any ontology Ont, the ground atoms form the set of *basic state properties*
BSTATPROP(Ont). Basic state properties can be defined by nullary predicates (or
proposition symbols) such as hungry, or by using n-ary predicates (with n>0) like
has_temperature(environment, 7). The *state properties* based on a certain ontology Ont are
formalized by the propositions (using conjunction, negation, disjunction, implication)
made from the basic state properties and constitute the set STATPROP(Ont).

In order to express dynamics in TTL, important concepts are *states*, *time points*,
and *traces*. A *state* S is an indication of which basic state properties are true and
which are false, i.e., a mapping S: BSTATPROP(Ont) → {true, false}. The set of all possible
states for ontology Ont is denoted by STATES(Ont). Moreover, a fixed *time frame* T is
assumed which is linearly ordered. Then, a *trace* γ over a state ontology Ont and time
frame T is a mapping γ: T → STATES(Ont), i.e., a sequence of states γ_t (t ∈ T) in STATES(Ont).
The set of all traces over ontology Ont is denoted by TRACES(Ont).

The set of *dynamic properties* DYNPROP(Ont) is the set of temporal statements that
can be formulated with respect to traces based on the state ontology Ont in the
following manner. Given a trace γ over state ontology Ont, a certain state at time point t
is denoted by state(γ, t). These states can be related to state properties via the formally
defined satisfaction relation, indicated by the infix predicate |=, comparable to the

Holds-predicate in the Situation Calculus. Thus, state(γ, t) |= p denotes that state property p holds in trace γ at time t. Likewise, state(γ, t) |≠ p denotes that state property p does not hold in trace γ at time t. Based on these statements, dynamic properties can be formulated in a formal manner in a sorted predicate logic, using the usual logical connectives such as ¬, ∧, ∨, ⇒, and the quantifiers ∀, ∃ (e.g., over traces, time and state properties). For example, consider the following dynamic property for a pattern concerning belief creation based on observation:

if	at any point in time t1 the agent observes that the situation is a disaster,
then	there exists a time point t2 after t1 such that
	at t2 in the trace the agent believes that the situation is a disaster

This property can be expressed as a dynamic property in TTL form with free variable γ as follows:

$$\forall t:T \ [\ state(\gamma, t) \models observes(itsadisaster) \Rightarrow \exists t' \geq t \ \ state(\gamma, t') \models belief(itsadisaster) \ ]$$

The set DYNPROP(Ont, γ) is the subset of DYNPROP(Ont) consisting of formulae with γ occurring in which is either a constant or a variable without being bound by a quantifier. For a more elaborate explanation of TTL, see [6].

3.2 The Language Meta-TTL for Second-Order Dynamic Properties

The formalizations of the properties sometimes take the form of second-order dynamic properties, i.e., properties that refer to dynamic properties expressed within TTL. Such second-order dynamic properties are expressed in meta-TTL: the meta-language of TTL. The language meta-TTL includes sorts for DYNPROP(Ont) and its subsets as indicated above, which contain TTL-statements (for dynamic properties) as term expressions. Moreover, a predicate holds on these sorts can be used to express that such a TTL formula is true. When no confusion is expected, this predicate can be left out. To express second-order dynamic properties, in a meta-TTL statement, quantifiers over TTL statements can be used.

4 Handling Incompleteness of Information by Enrichment Rules

The trace of occurrences as logged during or reported from an incident management process usually is incomplete and therefore difficult to analyze. To overcome this incompleteness problem, additional assumptions have to be made on events that have occurred but are not explicitly mentioned in the logged trace. Such assumptions are addressed in this section. These extra assumptions enrich the trace with elements that are derived from the information in the trace itself, for example at later time points in case an analysis is performed afterwards. An example is the assumption that if at some time point an estimation of the situation is communicated, then at previous time points the necessary information to make that assessment was received or observed by the communicating role.

Addition of such elements to enrich a trace are based on rules which express that given certain trace elements, an additional element can be assumed. These rules in principle can be of two forms: Strict rules which can always be applied and provide conclusions that are certain, and defeasible rules which are used in case strict rules are insufficient to obtain a trace with a reasonable amount of information. However, it is

not always possible to claim that a rule is a strict rule. Therefore, such rules are considered premises for the whole analysis.

Examples of such rules are presented below. Rule EP1 states that everybody present on the scene is assumed to have an internal judgment about the seriousness of the disaster:

***EP1:* Internal judgment at scene**
if   at time t role R is present at the scene
 and   situation S is the case
 and   S is classified as being a disaster
then   there exists a later point in time t2 < t+d at which R has an internal judgment that this situation is a disaster

Formal:

$\forall$ R:ROLE, t:TIME, S:SITUATION
[state(γ, t) |= physical_position(R, scene) &
state(γ, t) |= current_situation(S) &
state(γ, t) |= disaster(S)]
$\Rightarrow \exists$t2>t & t2 < t+d [state(γ, t2) |= internal_judgment(R, disaster(S))]

Furthermore, in case a role receives a communication that the situation is a disaster and this role does not communicate that he does not believe it being a disaster, then it is assumed that he has the internal judgment that it concerns a disaster:

***EP2:* Internal judgment based on communication**
if   at time t R1 communicates to R2 that the current situation S is a disaster
 and   there exists no time point at which R2 communicates to R1 he thinks the situation is not a disaster
then   at every time point t2 > t R2 interprets the current state of affairs as being a disaster

$\forall$R1,R2:ROLE, P:POSITION, t:TIME, S:SITUATION
[state(γ, t) |= communication_from_to(R1, R2, disaster(S)) &
 $\neg\exists$t'>t [state(γ, t') |= communication_from_to(R2, R1, not(disaster(S)))]]
$\Rightarrow \forall$t2 > t [state(γ, t2) |= internal_judgment(R2, disaster(S))]

5 Property Hierarchies for Incident Management Organizations

This section presents generic properties for incident management organizations in the Netherlands. The properties are presented in property hierarchies, which has as an advantage that diagnosis of properties can be done in a top down fashion. Such a diagnostic process starts by checking highest level property, and in case such a property is not satisfied pinpoints the error by gradually going down the tree to the unsatisfied properties.

5.1 Warning of Relevant Parties

The warning of relevant parties by the operator is a high level property stating that: "the operator should alarm all necessary parties in case it is informed of an incident":

***P1(d):* Warn relevant parties**
if   at time t the operator is informed about an incident type I by a role R1,
 and   for incident type I role R2 should be informed according to the disaster plan

then　　　there exists a time t2 later than t and before t + d at which R2 is informed about the incident type I

∀I:INCIDENT_TYPE, t:TIME, R1, R2:ROLE
[state(γ, t) |= communication_from_to(R1, operator, I) &
 state(γ, t) |= according_to_plan_should_be_involved_in(R2, I)]
⇒ ∃t2 > t & t2 < t + d [state(γ, t2) |= communication_from_to(operator, R2, I)]]

This property can be refined into a number of similar properties restricted to specific categories of roles that should be informed. For diagnosis, at the highest level property P1(d) can be checked, for example with the result that P1(d) is not satisfied which means that not all relevant parties were informed (but without information on which specific categories were not informed). At one level lower, the diagnosis can be refined by checking the refined properties, resulting in an indication of which of the categories of relevant roles were not informed.

5.2 First Arriving Ambulance

Second, the behavior of the first arriving ambulance is addressed. First, a formal definition of the first arriving ambulance is given:

first_arriving_ambulance(γ:TRACE, t:TIME, A:AMBULANCE)
An ambulance is the first arriving ambulance if:
the ambulance arrives at the scene of an incident at time t
and there does not exist a time t' < t at which another ambulance arrived at the scene of the incident

[state(γ, t) |=physical_position(A, scene) & ¬∃t'< t, [∃B:AMBULANCE [state(γ, t') |=physical_position(B, scene)]]

On the highest level, the first arriving ambulance behavior is described by three important aspects: (1) signaling the green alarm light; (2) communicating a situation report, and (3) presence of at least one person belonging to the ambulance until the officer on duty arrives at the scene:

P2: **First arriving ambulance global behavior**
if　　　 at a time t ambulance A is the first to arrive at the scene
 and　　 at time t3 > t the officer on duty arrives at the scene
then　　 for all t2 ≥ t and t2 < t3 at least one person belonging to the ambulance should be present at the ambulance
and　　 for all t4 ≥ t the ambulance is signaling the green alarm light
and　　 there exists a time t5 later than t at which the driver of that ambulance communicates a correct interpretation of the situation to the operator.

∀A:AMBULANCE, t, t2:TIME
[first_arriving_ambulance(γ, t, A) &
state(γ, t2) |= physical_position(officer_on_duty, scene) &
¬∃t'''< t2 [state(γ, t''') |= physical_position(officer_on_duty, scene)]]
⇒
∀t3 < t2
　 [t3 ≥ t ⇒ [∃R:ROLE [state(γ, t3) |= physical_position(R, A)]]]
& ∀t4 > t [state(γ, t4) |= alarm_lights(A, green)]
& ∃t5 > t, X:SITUATION[state(γ, t5) |= communication_from_to(driver, operator, situation_description(X)) &
　　　　　　　　　　situation(X)]]

This property can be related to lower level properties as shown in Figure 1. When trying to diagnose why the highest level property is not satisfied, the properties on the lower level can be checked. In case such a property is not satisfied, and it concerns a leaf property, at least one cause for the non-fulfillment of the high-level property has

been found. Otherwise, go further down the tree to find the cause. In the tree a number of properties are present to enable satisfaction of P2. First of all, the signaling of the green light, as expressed below.

Fig. 1. Property hierarchy for the first arriving ambulance.

P3: First ambulance green light behavior
if   at a time t ambulance A is the first to arrive at the scene
then   for all later points in time t2 the ambulance is signaling the green light.

$\forall$A:AMBULANCE, t:TIME
[first_arriving_ambulance(γ, t, A) $\Rightarrow$ $\forall$t2:TIME > t [state(γ, t2) |= alarm_lights(A, green)]]

Second, the presence of a person belonging to the ambulance for the time until the officer on duty is present:

P4: First arriving ambulance personnel presence
if   at a time t ambulance A is the first to arrive at the scene
 and   at time t3 > t the officer on duty arrives at the scene
then   for all t2 ≥ t and t2 < t3 at least one person belonging to the ambulance should be present at the ambulance

$\forall$A:AMBULANCE, t, t2:TIME
[first_arriving_ambulance(γ, t, A) &
state(γ, t2) |= physical_position(officer_on_duty, scene) &
¬$\exists$t'''< t2 [state(γ, t''') |= physical_position(officer_on_duty, scene)]]
$\Rightarrow$ $\forall$t3 < t2 [t3 ≥ t $\Rightarrow$ [$\exists$R:ROLE [state(γ, t3) |= physical_position(R, A)]]]

Finally, a property expressing the communication of the correct situation to the operator:

P5(d): First arriving ambulance interpretation
if   at a time t ambulance A is the first to arrive at the scene
then   at a later point in time t2 < t + d the driver of that ambulance communicates a correct interpretation of the situation

$\forall$A:AMBULANCE, t:TIME
first_arriving_ambulance(γ, t, A)
$\Rightarrow$ $\exists$X:SITUATION, t2:TIME < t + d & t2>t
    state(γ, t2) |= physical_position(driver, A) &
    state(γ, t2) |= communication_from_to(driver, operator, situation_description(X)) &
    state(γ, t2) |= situation(X)]

Note that parameter d includes the time to interpret the situation plus the time to start communicating that particular interpretation. Testing whether the interpretation was correct can be performed afterwards (e.g., the amount of casualties). The property P5 can be refined again into three lower level properties. First of all, when arriving at the scene, the paramedic should investigate the current state of affairs:

P6(d): Paramedic investigation
if   at a time t ambulance A is the first to arrive at the scene
 and   at time t a paramedic is in the ambulance
then   at a later point in time t2 < t + d the paramedic of that ambulance will start an investigation and not be at the ambulance any more

$\forall$A:AMBULANCE, t:TIME
[first_arriving_ambulance(γ, t, A) &
state(γ, t) |= physical_position(paramedic, A)]]
$\Rightarrow$ $\exists$t2:TIME < t + d & t2 > t
    [state(γ, t2) |= not physical_position(paramedic, A) & state(γ, t2) |= investigating(paramedic)]
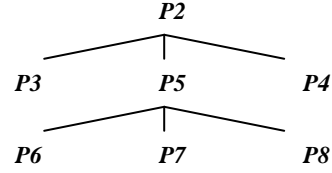
Second, the paramedic will return, communicating the current situation:

***P7(d):* Paramedic communication**
if   at a time t ambulance A is the first to arrive at the scene
 and   at time t the paramedic is in the ambulance
 and   at time t2 the physical position of the paramedic is not inside the ambulance
then   at a later point in time t3 < t2 + d the paramedic of that ambulance will communicate a
            correct interpretation of the situation to the driver

∀A:AMBULANCE, t,t2:TIME
[first_arriving_ambulance(γ, t, A) &
 state(γ, t) |= physical_position(paramedic, A) & t2 > t &
 state(γ, t2) |= not physical_position(paramedic, A) &
 state(γ, t2) |= investigating(paramedic)]
⇒ ∃t3:TIME < t2 + d & t3 > t2, X:SITUATION
   [state(γ, t3) |= physical_position(paramedic, A) &
    state(γ, t3) |= communication_from_to(paramedic, driver, situation_description(X)) &
    state(γ, t3) |= situtation(X)]

Finally, once the driver has received the communication, he will communicate this to the operator:

***P8(d):* Driver communication**
if   at a time t the driver of the first ambulance at the scene receives a situation description
            from the paramedic
then   at a later point in time t2 < t + d the driver of that ambulance communicates a correct
            interpretation of the situation to the operator

∀A:AMBULANCE, t,t2:TIME, X :SITUATION
[first_arriving_ambulance(γ, t, A) &
 state(γ, t2) |= communication_from_to(paramedic, driver, situation_description(X))
⇒ ∃t3:TIME < t2 + d & t2 > t [state(γ, t3) |= communication_from_to(driver, operator, situation_description(X))]

5.3 Disaster Staff Activation

Furthermore, properties have been specified for the formation of the disaster staff and activities following from the disaster staff. On the highest level the correctness of these processes in the disaster staff can be described as follows: In case the operator has the internal judgment that the current situation is a disaster, the operational leader will eventually output actions belonging to a strategy communicated by the disaster staff.

***P9:* Successful disaster staff**
if   at time t the operator judges the current situation as a disaster
then   there exists a later point in time t2 at which the disaster staff communicated a strategy
and   there exists an even later time at which the operational leader communicates an action
            appropriate for the strategy according to the disaster plan.

∀t:TIME
[state(γ, t) |= internal_judgement(operator, disaster)
⇒ ∃t2:TIME > t, S:STRATEGY
   [state(γ, t2) |= communication_from_to(disaster_staff, operational_leader, S) &
    ∃t3:TIME > t2, A:ACTION, R:ROLE
    [state(γ, t3) |= appropriate_action_according_to_plan(S, A) &
     state(γ, t3) |= accompanying_role(A, R)] &
     state(γ, t3) |= communication_from_to(operational_leader, R, perform(A))]
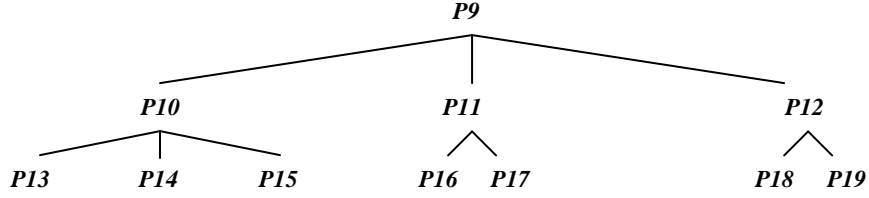
Fig.2. Property hierarchy for the disaster staff activation and functioning.

Such properties can be related to lower-level properties as shown in Figure 2. On the intermediate level, three properties are present. First, the correct initiation of a disaster staff is expressed:

P10: **Correctly activated disaster staff**
if   at time t the operator interprets the current situation being a disaster
then   at a later point in time t2 the disaster staff will be informed (and assumed to be present as a result)

$\forall$t:TIME, R:ROLE
[state(γ, t) |= internal_judgement(operator, disaster) &
state(γ, t) |= part_of(R, disaster_staff)
$\Rightarrow \exists$t2:TIME > t + d [state(γ, t2) |= communication_from_to(operator, R, form_disaster_staff)]]

Thereafter, in case the disaster staff is formed, it should be active, which is characterized by an output in the form of a strategy:

P11: **Active disaster staff**
if   at time t the organizational unit called disaster staff is informed
then   at a later point in time t2 > t the organizational unit outputs a strategy S

$\forall$t:TIME
[state(γ, t2) |= part_of(R, disaster_staff) &
state(γ, t2) |= communication_from_to(operator, R, form_disaster_staff)
$\Rightarrow \exists$S:STRATEGY, t2 > t [state(γ, t2) |= communication_from_to(disaster_staff, operational_leader, S)]

Finally, such a strategy should lead to actions be taken by the operational leader:

P12: **Active operational leader**
if   at time t the operational leader is informed of a strategy S to be applied
then   at a later point in time t2 > t the operational leader will command the appropriate actions according to the disaster plan to the roles.

$\forall$t:TIME, S:STRATEGY, A:ACTION, R:ROLE
[state(γ, t2) |= communication_from_to(disaster_staff, operational_leader, S) &
state(γ, t) |= appropriate_action_according_to_plan (S, A) &
state(γ, t) |= accompanying_role(A, R)
$\Rightarrow \exists$t2:TIME > t state(γ, t2) |= communication_from_to(operational_leader, R, perform(A))]

Each of these intermediate properties can again be split up to properties for individual roles within the organization. In order to obtain property P10 a number of properties need to hold. First of all, the mayor should be warned by the operator:

P13(d): **Warn mayor**
if   at time t the operator interprets the current situation being a disaster
then   at a later point in time t2 > t and t2 < t +d the operator communicates the occurrence of a disaster to the mayor.

$\forall$t:TIME
[state(γ, t) |= internal_judgement(operator, disaster) &
$\Rightarrow \exists$t2:TIME > t & t2 < t + d [state(γ, t2) |= communication_from_to(operator, mayor, disaster)]

Thereafter, the mayor should decide to form the disaster staff:

P14: **Form disaster staff**

if   at time point t the mayor interprets the current state of affairs as being a disaster
then   at a later point in time t2 > t the mayor forms the organizational unit called disaster staff

∀t:TIME
[state(γ, t) |= internal_judgement(mayor, disaster) &
¬∃t'< t [state(γ, t') |= internal_judgement(mayor, disaster)] &
⇒ ∃t2 > t [state(γ, t2) |= communication_from_to(mayor, operator, form_disaster_staff)]

Finally, in case the mayor communicates the decision to form the disaster staff, the operator should warn the appropriate parties:

P15(d): **Warn rest disaster staff**

if   at time t the operator receives the request of the mayor to form the disaster staff
 and   role R is part of the disaster staff
then   at a later point in time t2 > t and t2 < t +d the operator communicates to role R that the disaster staff is being formed.

∀t:TIME, R:ROLE
state(γ, t) |= communication_from_to(mayor, operator, form_disaster_staff) &
state(γ, t) |= part_of(R, disaster_staff)
⇒ ∃t2:TIME > t + d [state(γ, t2) |= communication_from_to(operator, R, form_disaster_staff)]]

Regarding the intermediate property P11 the following properties need to hold for satisfaction of the intermediate property. First, after the mayor has decided to form the disaster staff he will eventually request advice from his disaster staff.

P16: **Start deliberation**

if   at time t the mayor decides to form the disaster staff
then   at a later point in time t2 > t the mayor starts a deliberation within the disaster staff by requesting advice

∀t:TIME
[state(γ, t) |= communication_from_to(mayor, operator, form_disaster)
⇒ ∃t2:TIME > t [state(γ, t2) |= communication_from_to(mayor, disaster_staff, request_advice)]]

After such advice is received, he should choose the appropriate strategy:

P17: **Choose strategy**

if   at time t starts a deliberation within the disaster staff by requesting advice
then   at a later point in time t2 the mayor communicates a strategy to the operational leader

∀t:TIME
[state(γ, t) |= communication_from_to(mayor, disaster_staff, request_advice)
⇒ ∃S:STRATEGY, t2:TIME > t state(γ, t2) |= communication_from_to(mayor, operational_leader, S)]

Finally, the intermediate property P12 is refined to two other properties. First, the operational leader should discuss the strategy with his operational team:

P18: **Choose action**

if   at time t the mayor communicates a strategy S to the operational leader
then   at a later point in time t2 > t the operational leader requests his operational team for advice how to implement S

∀t:TIME, S:STRATEGY
[state(γ, t) |= communication_from_to(mayor, operational_leader, S)
⇒
∃ t2:TIME > t state(γ, t2) |= communication_from_to(operational_leader, operational_team, request_advice(S))]

Finally, the operational leader communicates actions to be performed, based on the advices obtained in the discussion.

P19: **Communicate action**

if   at time t the operational leader request his operational team for advice how to implement S
then   at a later point in time t2 the operational leader will communicate actions appropriate for strategy S according to the disaster plan

∀t:TIME, S:STRATEGY, A:ACTION, R:ROLE
[state(γ, t) |= communication_from_to(operational_leader, operational_team, request_advice(S)) &
 state(γ, t) |= appropriate_action_according_to_plan (S, A) &
 state(γ, t) |= accompanying_role(A, R)
 ⇒ ∃ t2:TIME > t state(γ, t2) |= communication_from_to(operational_leader, R, perform(A))]

5.4 Ambulance Routing

Finally, properties are specified regarding ambulance routing. The police should act as follows:

P20: Route plan includes all wounded nests
if   at time t there are n wounded nests
 and   at a later time point t2 > t the police communicates details concerning the route to be taken by the ambulances to cpa (the central ambulance post)
then   this communication should contain such a route description that ambulances will be sent to all wounded nests.

∀W:WOUNDED_NEST, R:ROUTE_PLAN, t:TIME
[state(γ, t) |= physical_position(W, scene) &
 state(γ, t) |= communication_from_to(police, cpa, R)]
 ⇒ state(γ, t) |= route_passes_wounded_nest(R, W)

An alternative property not following standard procedure expresses that the routing is done based explicitly on victim locations:

P21: Send ambulance to all wounded on the scene
if   at time t there is a wounded person at a position P
then   at a later time point t2 an ambulance will be sent to position P
 and   at an even later time point t3 that ambulance will be at position P

∀W:WOUNDED, P:POSITION, A:AMBULANCE, t:TIME
[state(γ, t) |= physical_position(W, P) &
 ⇒ ∃t2 > t [state(γ, t2) |= communication_from_to(operator, A, goto(P))] &
    ∃t3 > t2 [state(γ, t3) |= physical_position(A, P)]

This high-level property can be decomposed into three other properties. First of all, a wounded person will result in a communication to the operator of the physical position of this wounded person:

P22: Communicate wounded location
if   at time t there is a wounded person at a position P
then   at a later time point t2 this position will be communicated to the operator

∀W:WOUNDED, P:POSITION, t:TIME
[state(γ, t) |= physical_position(W, P) &
 ⇒ ∃R:ROLE, t2 > t [state(γ, t2) |= communication_from_to(R, operator, physical_position(W, P))]]

For every communication received by the operator, he eventually communicates the location to an ambulance:

P23: Send ambulance to wounded
if   at time t a wounded person is communicated to be at a position P
then   at a later time point t2 an ambulance will be sent to position P

∀W:WOUNDED, P:POSITION, R:ROLE, t:TIME
[state(γ, t) |= communication_from_to(R, operator, physical_position(W, P))

$\Rightarrow \exists t2 > t,$ A:AMBULANCE [state(γ, t2) |= communication_from_to(operator, A, goto(P))]]

Finally, once the ambulance gets this communication it will arrive at the location at a later point in time:

P24: **Ambulance arrives at wounded**
if   at time point t an ambulance is sent to position P
then   at a later time point t2 that ambulance will be at position P

$\forall$P:POSITION, A:AMBULANCE, t:TIME
[state(γ, t2) |= communication_from_to(operator, A, goto(P))
$\Rightarrow \exists t2 > t$ [state(γ, t3) |= physical_position(A, P)]

6 Human Error Types

This Section presents a classification scheme for the properties in incident management. Such a classification can help to determine the dedicated training needed. The human error classification presented by James Reason [9] is therefore adopted, who introduces a General Error Modeling approach which identifies three basic error types: (1) skill based slips; (2) rule based mistakes, and (3) knowledge based mistakes. This classification scheme is also used in [2] in which incident management is investigated. Rule based, and knowledge based errors come into play after the individual has become conscious of a problem, which is not the case for skill based slips. In that sense, skill based errors generally precede detection of the problem whereas rule based and skill based mistakes arise during subsequent attempts to find a solution to the problem. Skill based and rule based level error occur when humans use stored knowledge structures whereas knowledge based errors occur when such knowledge structures have been exhausted. Errors are much more likely to occur at the knowledge based level.

For the properties specified for incident management the following classification scheme is used. Skill based properties are those properties that are part of the very basic training of incident management workers. For example, how to start the water pump on a fire truck. A property is classified as a rule based property in case an incident management plan literally includes the property. Finally, a property is called a knowledge based property in case an incident management plan states that a decision needs to be taken, but does not specify how to come to this solution. Using this classification scheme, none of the properties from Section 5 are routine based, whereas properties P1, P3, P5, P6, P8, P13, P14, P15, P16, P19, P22, and P24 are rule based properties. Finally, properties P7, P17, P18, P20, and P23 are knowledge base properties. Note that only the leaf properties are categorized as these are the properties that define the individual role behavior within the organization.

In order to identify which types of error the different participants in the incident management organization are making, the following formula expressed in meta-TTL is used:

Type Error $\equiv$
$\forall \gamma$:TRACE, t_1, t_2:TIME, A:AGENT, R:ROLE, P:DYNPROP, Q:DYNPROPEXPR, S:SITUATION,
X:PROPERTY_TYPE
[holds_in_period(has_role(A, R), γ, t_1, t_2) &
 holds_in_period(S, γ, t_1, t_2) &
 holds_in_period(relevant_for(P, R, S), γ, t_1, t_2) &

```
holds_in_period(type_for(P, R, X), γ, t₁, t₂) &
holds_in_period(has_specification(P, Q(R, γ, c₁, c₂)), γ, t₁, t₂) &
¬holds(Q(R, γ, t₁, t₂))]
⇒ makes_error_of_type(A, R, P, X, γ, t₁, t₂)
```

This expresses that if an agent A is allocated to a particular role R in a particular period between t_1 and t_2 in trace γ, and a situation S occurs in that same period in which property P is relevant for role R whereby the type of property P for role R is of type X (where X is either skill based, rule based or knowledge based), and the property has a specification which does not hold in the fragment of this trace, then an error of type X is made concerning property P by role R played by agent A.

7 Case Study

As a means to validate the approach presented above, a disaster which has been thoroughly investigated in the Netherlands is taken as a case study. The disaster concerns a bar fire which occurred in Volendam, the Netherlands, at New Years Night of the year 2001. The logs of the disaster have been thoroughly described in [7] and have been formalized using the approach presented in Section 3. Thereafter, the trace enrichment rules from Section 4 have been applied. A part of the resulting trace is shown in Figure 3, which uses the same ontology as used for the formalization of the properties in Section 5. On the left side of the Figure, the atoms are shown that occur during the incident management whereas the right side shows a timeline where a dark box indicates an atom being true at that time point and a gray box indicates the atom being false. The trace is used to verify whether the properties as specified in Section 5 indeed hold for the Volendam disaster. The following properties were shown not to hold: P2, P4, P5, P7, P8, P9, P10, P14, and P20. In other words, in the Volendam case study the first ambulance did not comply to the global desired behavior because the information was not communicated properly, and because there exist time points at which nobody was present at the ambulance. Furthermore, the disaster staff was not activated properly because the mayor did not communicate that the disaster staff should be formed, and finally the ambulance routing of the police was incorrect, but luckily the direct routing of the health care services was satisfied. These results exactly comply to the conclusions in the disaster report [7] which resulted from a thorough investigation of a committee specialized in incident management.

8 Discussion

This paper presents an agent-based approach which can be used for error detection in incident management organizations. The approach consists of several parts. First, a formal approach for the specification of both traces and properties that can be verified against these traces is presented. In domains like incident management, traces might be incomplete. Therefore, enrichment rules for these traces are identified to cope with this incompleteness. Furthermore, the properties that ought to be verified against these traces can be specified in a hierarchical fashion: in case the highest level property is
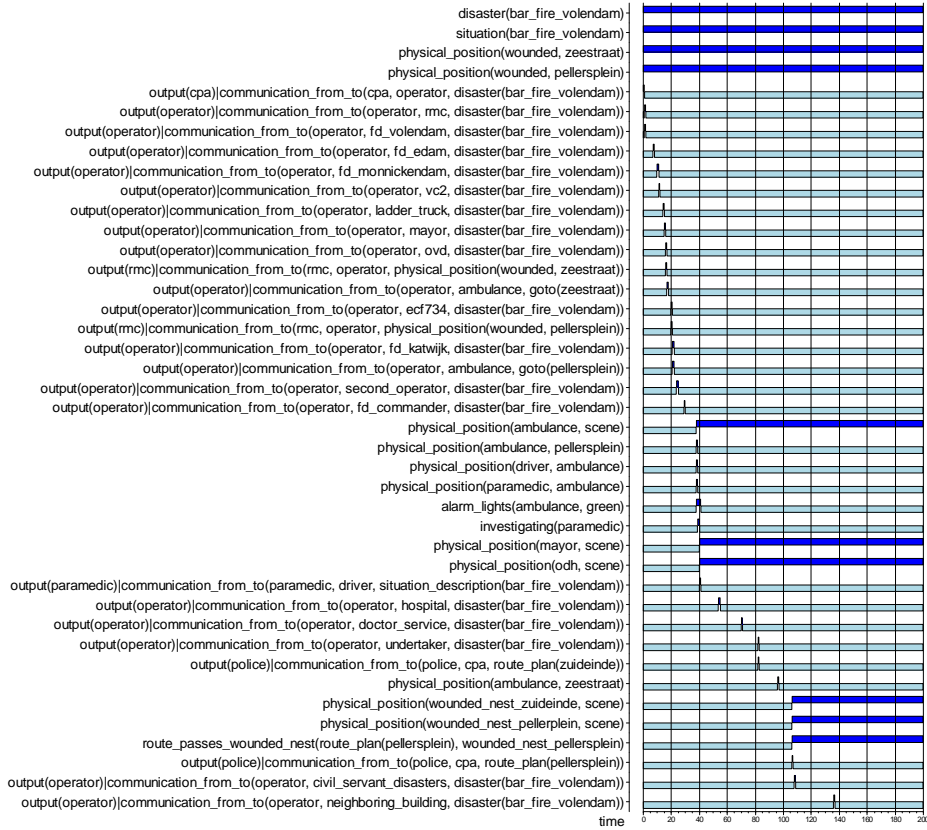
Fig.3. Partial trace of the Volendam case study

not satisfied, the cause of this dissatisfaction can be determined by looking at the properties one level deeper in the tree, which continues until a leaf property is found which is not satisfied. Finally, a classification mechanism is presented for the different properties based on psychological literature. In case an error is observed such a classification immediately gives insight in the functioning of a particular agent playing a role, which enables performing dedicated training sessions or giving appropriate warning messages.

In the future, the approach presented can be incorporated in personal agents of people involved in incident management. Such agents automatically log all incoming and outgoing information in the form of traces and have knowledge on the property the particular role the agent is playing is required to fulfill. In case properties are observed not to be satisfied, a reminder or warning can for instance be given to the person. Such agents can be useful for training sessions, as it can be observed what kind of mistakes a person typically makes, but could possibly even be used during actual incident management.

In the field of information agents, support systems have also been developed for incident management (see e.g. [11]). In such systems however, the agents do not check whether errors are made, but simply provide people with information to make

sure they are aware of their tasks. This does however not offer a mechanism to detect errors and avoid a chain of unwanted events. Approaches for e.g. detection of protocols (see e.g. [10]), also called overhearing, have been introduced. These approaches are however more focused on recognizing patterns, not on detection of errors.

Error detection itself is another related research field. In [3] behavioral properties for a parallel computing system can be specified, and can be checked on the fly. The properties are however specified as simple sequences of states, whereas the TTL language as used in this paper has the ability to express timing parameters between these states, often a necessity in incident management. In [5] properties for error detection are specified by means of a finite state machine which again does not allow for time parameter specification.

Acknowledgements

References

1. Boissier, O., Dignum, V., Matson, E., Sichman, J. (eds.), Proc. of the 1st Workshop From Organizations to Organization Oriented Programming in MAS (OOOP), 2005.
2. Duin, M.J. van, Learning from Disasters (in Dutch), PhD Thesis, Leiden, 1992.
3. Fromentin, E., Raynal, M., Garg, V.K., and Tomlinson, A., On the Fly Testing of Regular Patterns in Distributed Computations, Information Processing Letters 54:267-274, 1995.
4. Giorgini, P., Müller, J., Odell, J. (eds.), Agent-Oriented Software Engineering IV, LNCS, vol. 2935, Springer-Verlag, Berlin, 2004.
5. Jard, C., Jeron, T., Jourdan, G.V., and Rampon J.X. "A general approach to trace-checking in distributed computing systems", In Proc. IEEE International Conference on Distributed Ccomputing Systems, pp. 386-404, Poznan, Poland, June, 1994.
6. Jonker, C.M., Treur, J. Compositional verification of multi-agent systems: a formal analysis of pro-activeness and reactiveness. International. Journal of Cooperative Information Systems, vol. 11, 2002, pp. 51-92.
7. Ministry of the Interior, Investigation Bar Fire New Years Night 2001 (in Dutch), SDU Publishers, The Hague, 2001.
8. Municipality of Amsterdam, Disaster Plan (in Dutch), 2003.
9. Reason, J., Human Error, Cambridge University Press, 1990.
10. Rossi, S., Busetta, P, Towards Monitoring of Group Interactions and Social Roles via Overhearing, In: Klusch, M., Ossowski, S., Kashyap, V., and Unland, R. (eds), Cooperative Information Agents VIII, LNAI 3191, Spinger-Verlag, pp. 47-61, 2004.
11. Storms, P.A.A., Combined Systems: A System of Systems Architecture, In: Proceedings of ISCRAM 2004, pp. 139-144, May 2004, Brussels.
12. Walle, B. van, and Carle, B. (eds.), Proceedings of ISCRAM 2005, 2005.