A CONCEPTUAL FRAMEWORK FOR EVOLVING INFORMATION SYSTEMS *

E.D. Falkenberg
J.L.H. Oei
H.A. Proper
E.Proper@acm.org

June 23, 2004

PUBLISHED AS:

E.D. Falkenberg, J.L.H. Oei, and H.A. Proper. A Conceptual Framework for Evolving Information Systems. In H.G. Sol and R.L. Crosslin, editors, *Dynamic Modelling of Information Systems II*, pages 353–375. North-Holland, Amsterdam, The Netherlands, EU, 1992. ISBN 0444894055

Keywords: Evolution of Information Systems, Schema Evolution, Temporal Information Systems, Historical Information Systems.

Abstract

In this paper, we will treat evolving information systems, which support the information needs of an evolving organization. Such a system is able to change at the same pace as the organization does, such that there is no need to interrupt the activities of the organization. Such a system supports changes of all time- and application-dependent aspects.

We will present a conceptual framework of an approach to evolving information systems. The architecture of such a system will be explained, where we will make disctinctions between meta model and application model, and between world, action and behavioural aspects of these models.

Furthermore, evolution management of application models will be discussed, whereby update is regarded as recording, correction or forgetting, and whereby state transitions are considered on three different levels, viz. the organizational level, the recording level and the correction level. A formal specification of this approach to evolution management will be provided too.

1 INTRODUCTION

Evolution may be defined as 'gradual development by natural process' or as 'origination of species by development of earlier forms'. Interpreting this definition in the context of

*The investigations were partly supported by the Foundation for Computer Science in the Netherlands (SION) with financial support from the Netherlands Organization for Scientific Research (NWO). Department of Information Systems, University of Nijmegen, Toernooiveld 1, NL-6525 ED Nijmegen, The Netherlands

organizations, evolution means that an organization at a certain point of time is originated in development of its earlier states.

Considering an information system from a systems view on organizations (e.g. [FHL+98]), an information system should allow evolution to the same extent and at the same pace at which an organization system evolves because of changes in the organization domain (universe of discourse, [ISO87]). However, most traditional information systems hardly support any aspects of evolution.

First of all, most traditional systems mainly focus on the static aspects of the organization system. Apart from that, they allow update of the information base only, i.e. the set of facts which obeys a fixed (conceptual) schema with a fixed set of constraints. In other words, update of the conceptual schema including constraints and, furthermore, update of the specification of dynamic aspects - i.e. action and behaviour, if they have been modelled at all - have not yet been supported by these traditional information systems. The evolving information systems at which we aim, however, do support update of all these specifications. A very important requirement for these evolving systems is that no update of any sort should result in the interruption of activities of the evolving organization.

Another limitation of many traditional systems is that they are 'snapshot' systems, i.e. they reflect information valid at a certain point of time, but lack the ability to preserve the history of information. An evolving system, on the contrary, must be 'conservative' ([Fal74]) or 'temporal' ([SA86], [RBL87]) in the sense that it should not forget anything ever fed to the system, unless explicitly asked for. For such systems, the distinction between intrinsic (or valid) and extrinsic (or transaction) time ([SA86], [Bub80]) is important.

When evolving information systems become available, traditional systems might turn out to be 'degenerations' of these evolving information systems.

This paper will present a conceptual framework for a generalized evolving information system. This framework includes fundamental concepts and an architecture for such a system. In this architecture, a distinction is made between a part which is application-independent and time-invariant, and a part which is not. The description of the first part is contained in the so-called meta model, while the latter part is described in what we call the application model.

The meta model of a generalized evolving information system contains a description of the notion of update of the application model, reflecting changes of states (events) in the underlying organization system. The description which we will present in this paper may be characterized as state-transition-oriented and taking place on different levels of abstraction. On the organization level we describe transitions of states of the organization system - these states are modelled in what we call the application model states - and which are caused by events in the organization system. On the next level, the recording level, transitions of histories of these states are considered due to recording of these events. Finally, on a third level, which we will call the correction level, corrections of incorrect recordings are viewed as transitions of recording histories on the recording level. Such a correction will be performed by a roll-back to the latest correctly recorded state. After that, this roll-back should be followed by the recording of all events which happened after the event resulting in the state to which has been rolled back.

In the process of the development of a generalized evolving information system, we distinguish three sub-objectives: the design of a meta model and a language based on it, which can support all aspects of evolution, the implementation of a generalized information system shell which is based on that meta model and language, and a suitable method for the process of designing, building up and maintaining an application model.

Besides positioning evolving information systems with respect to existing (traditional) information systems, the conceptual framework presented in the paper forms the basis for the meta model for such a generalized evolving information system. This meta model determines all conceptual aspects of evolution, i.e. the ability of updating all the constituent parts of the application model, without forgetting any aspect ever fed to the system, unless explicitly asked for. Update, as said before, does not interrupt the activities of the organization system. Note, however, that in this paper we do not treat the evolution of user interfaces, implementation and technical aspects.

2 ARCHITECTURE OF AN EVOLVING INFORMATION SYSTEM

In our systems view on organizations, we consider an information system to be a part of an organization system. This part results from infological abstraction, i.e. abstraction from all physical aspects including representational matters. An organization system is a set of actors performing activities which result in changes of states (events) at specific points of time. (This view is based on the approach taken in [FHL$^+$98].) In this paper an information system is restricted further to actors which are computerized, performing information processing activities. We will call such a computerized actor, an information processor. The information processor may be composed out of several sub-processors, which may also be distributed, but in this paper we do not treat distributed information systems.

The information processor accepts input messages (requests), which, among other things, may reflect changes of a state (events) in the organization domain, triggering the information processor to perform activities. As a result of these activities, the information processor may produce output messages (responses). These output messages are received in turn by the organization domain, which is embedded in the environment of the information system.

We will call the knowledge set of the information processor - all the knowledge needed by the information processor to perform its activities - the processing model. It contains all the necessary knowledge about the grammar and semantics of the language(s) used for formulating messages as well as about the particular application case, such that any specific input message can be checked, processed and transformed into a set of specific output messages which will be considered meaningful from the viewpoint of the particular application.

As mentioned before, the processing model can be divided into a part which is completely application-independent and time-invariant, the meta model, and a part which is not, the application model. As a result, the meta model is provided in a particular information system once and for all, while the application model must be built up and maintained for each new application. The building-up and maintenance of an application model is done by the information processor, which acts on or reacts to events in the organization domain (after receiving input messages) by consulting both the meta model and application model. Thus, unlike the meta model, the application model is not only input, but also output of the activities of the information processor. Besides update of the application model, information can be retrieved from the application model as well. Messages are correspondingly classified into update and retrieval messages. The language for formulating such messages in an information system are based on the meta model of that particular information system. The architecture discussed is depicted in figure 1.

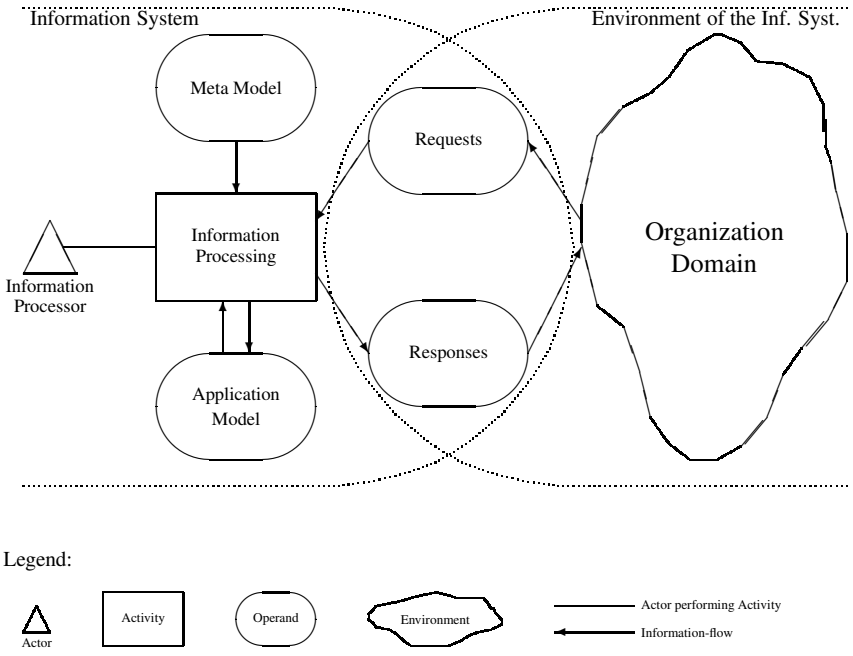

Figure 1: An information system and its environment

In this figure it is represented that the information processor (actor) performs the information processing (activity), having requests, the meta model and the application model as input (-operand), and having responses and the (updated) application model as output (-operand). Requests are generated and responses are consumed by the organization domain (in the environment).

Another distinction will be applied to the processing model. On the one hand, we need a model of that part of the perceived world (organization domain) the interaction between the information system and the environment is about. We call this model the world model. On the other hand, we need rules which determine the actions of the information processor. These rules are specified in what we call the action model.

Furthermore, we will subdivide the action model into a part that specifies activities - we call it the activity model - and a part that describes the (trigger-) relations between events in the organization domain and the activities performed by the information processor in the information system. We will refer to this latter part by the name behaviour model. Our argument for this last subdivision is, that the behaviour model describes *when* activities, under which conditions, and *what* activities should be performed by the information processor, whereas the activity model specifies *how* these activities should be performed.

These distinctions are not only restricted to the application model, they apply to the meta model as well. For example, the world part of a meta model will contain axioms which are true for all possible worlds, while the action part will contain rules which are also always applicable, independently from the application in question. The result of the subdivisions in the processing model is depicted in figure 2.

The three resulting models, world, action and behaviour model, correspond to the three different perspectives for information system design methods as they are discussed in [OHM+88], viz. data- (information-), process- (activity-) and behaviour-oriented. Together, the three resulting models describe all the parts of the organization system which

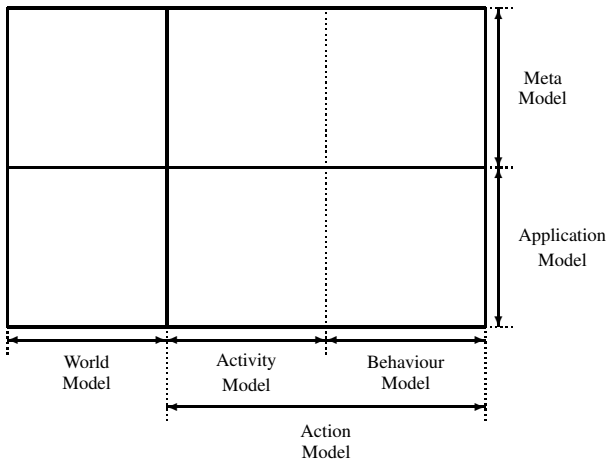

Figure 2: The structure of the Processing Model

are relevant to the information system.

On the basis of this architecture, the distinction between a traditional information system and an evolving information system can be explained more specifically. In a traditional information system, only the facts (instances) obeying a fixed conceptual schema, which constitute only a small part of the application model, can be updated. The intention of an evolving information system, however, is that specifications of the complete world (including the conceptual schema), activities and behaviour, which constitute the complete application model, will all become updatable. And furthermore, that update of the application model can take place without the need to interrupt the operation of the information system within the organization, i.e. without having to stop all the activities of the organization which depend on the information system. Note that we included the demand that an evolving information system should deal with both first-order and second-order dynamics (as discussed in [RV90]).

In order to treat temporal and evolutionary aspects of information systems we need a more suitable notion of update. We consider the traditional triple (insertion, deletion and modification) to be insufficient and, for that reason, in the following section we will replace it by the triple: recording, correction and forgetting (see section 3.1).

3 PRINCIPLES OF THE META MODEL

As stated in the previous section, the processing model can be subdivided in the meta model and the application model. The meta model, which is the application-independent and time-invariant part, contains all the necessary knowledge about the grammar and semantics of the language(s) used for formulating messages. The basis of the meta model is a set of concepts, predicates and axioms, which describe both the specification and manipulation (the evolution management) of the application model, which will be discussed in the following subsections.

3.1 Evolution management of the application model

We require an evolving information system to evolve to the same extent and at the same pace at which the underlying organization system evolves. By the notion of evolution we mean that each state of (the application model of) the information system is originated in changes of its former states. These changes of states in time are called events.

Consequently, whenever there is a change of state in the organization domain (in the environment of the information system) - in other words, an event has taken place in the underlying organization system which should affect the state of the information system - the happening of this event should be communicated to the system by an input message, i.e. an update request. The processing of this update request, which we will call the recording of an event, involves the processing of activities which are triggered by the event and should, finally, result in the appropriate state (-change) of the information system. This state is obtained by performing an update of the application model which contains the description of the underlying organization domain. Apart from this recording of events, correction and forgetting of (a sequence of) past recordings can also be distinguished as kinds of update.

These ideas are reflected in the requirement of the meta model to the effect that it should provide rules and constraints for updating the application model, without forgetting any aspect - including corrections - ever fed to the application model, unless explicitly asked for. We will present a model which describes the different kinds of update causing evolution of the application model of an information system. This evolution of the application model should reflect the evolution of the underlying organization domain. This model, which we will introduce in the sequel of this section, identifies state transitions on three different levels, i.e an organization level, a recording level and a correction level. In section 4 we will formalize the key concepts of this three-level state transition model.

The state description of the application model reflecting a particular state of an organization system at a particular point of time (snapshot) will be termed an application model state. Whenever an event happens at a certain point of time - this point of time is named the intrinsic time of the event - the current state of the organization system changes into a new state. Therefore, if this event is communicated to the system, a new state of the application model should be created.

A sequence of events which have occurred in the organization system, and which are consequently communicated to the system, results in a sequence of application model state transitions, which determines the history of the application model. Therefore, we will name the description of such a sequence of application model states and events an application model history. An example of these application model state transitions, which can be seen as taking place on the organization level, is illustrated in figure 3 (where e_i denotes an event and t_i the intrinsic point of time of such an event).

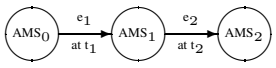

Figure 3: Application Model State (AMS) transitions on the organizational level

We can now describe the effect of recording an event at a certain point of time - this point of time is called the extrinsic time of the event. The recording of an event can be seen as a change from the current application model history, in which the event has not occurred,

into a new application model history, in which it has. We will call such a description of a sequence of application model histories an application model recording history. In figure 4, we have depicted an example of a sequence of application model history transitions, on what we call the recording level. (Recordings are denoted in the format Rec(e_i at t_i) at T_i, where e_i and t_i are defined as before, and T_i is the denotation of the extrinsic point of time.)

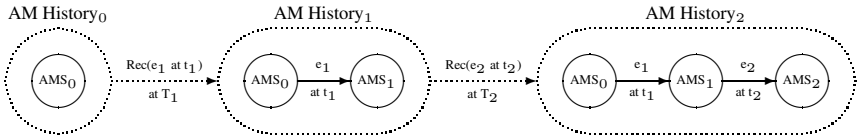

Figure 4: Application Model History (AMH) transitions on the recording level

In the process of recording of events, mistakes can be made: information which is empirically wrong, and which could not be detected by the information processor, might be communicated to the information system, the user might communicate an event to the system which did not happen at all, or the user might have forgotten to communicate to the information system an event which, actually, did occur in the organization system. We call this last situation late recording, which requires a correction too. The reason for that is that the recording of this event - at the proper time - would have caused a state which effects the recording of events occurred at a later point of time.

If a correction of one of these mistakes has to take place, a recording has to be replaced by another one, a recording has to be removed, or a recording has to be inserted, respectively. This correction will be performed by a roll-back to the latest application model state history in which the recording, which has to be replaced, removed or inserted, has not taken place yet. After this roll-back, a sequence of new, hopefully correct, recordings will follow. Some of these recordings may have been performed earlier. This mechanism will lead to a new application model recording history in which the correction - we will speak about correction in all cases that require a roll-back - has taken place. A description of a sequence of application model recording histories describes the complete evolution of the application model. Therefore, we will call such a sequence the application model evolution.

We will now give two examples of an application model recording history transition, which take place on the correction level. The first example deals with the replacement of an event which had been recorded erroneously because of empirically wrong information. In the second example a late recording will be considered.

In the situation of figure 4, suppose that it appears to be necessary to replace the erroneously recorded event e_1 at t_1 by event e_i'. In order to accomplish this, the system has to roll back to the latest correctly recorded history, i.e. AM History$_0$, invalidating the incorrectly stored histories (AM History$_1$ and AM History$_2$). The roll-back should be followed by recording the correct event which happened at t_1, i.e. e_1' and anew recording of all events which has taken place after t_1, i.e. event e_2 which happened at t_2. Note that the recording time of both e_1' and e_2, and the roll back to AM History$_0$ have been performed at T_3. This situation is shown in figure 5.

We may view this correction from a slightly alternative perspective. If we consider the sequence of recordings of events to represent our belief about the 'world', then the correction of a recording implies that the 'world' we previously believed in is no longer valid and a new 'world' stands in its place. Conform this perspective the correction performed in figure 5 is in an alternatively way depicted in figure 6. In this figure the Application Model

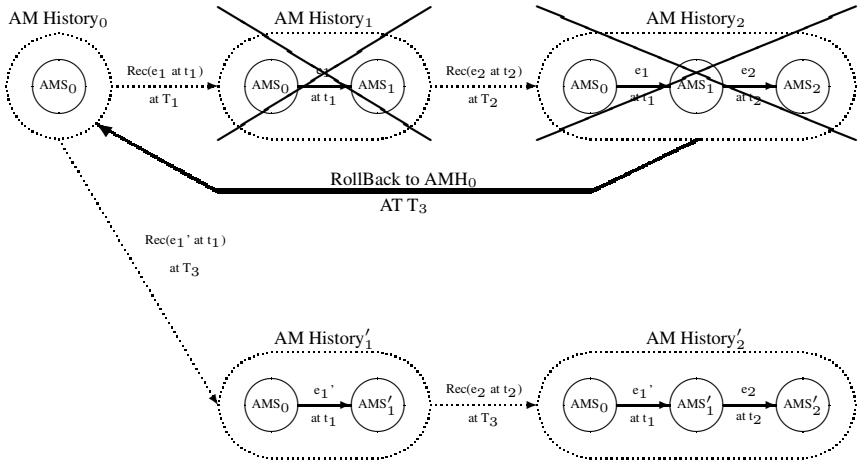

Figure 5: Correction by roll-back: replacement of an erroneously recorded event

Recording Histories represent the 'worlds' and the roll-back is not represented explicitly. It is the description of this sequence of worlds, i.e the sequence of AM Recording Histories to be more precisely - and all information contained therein - which determines the application model evolution. Note that by the way of representation used in figure 6 the three levels of state transitions become clearly visible.

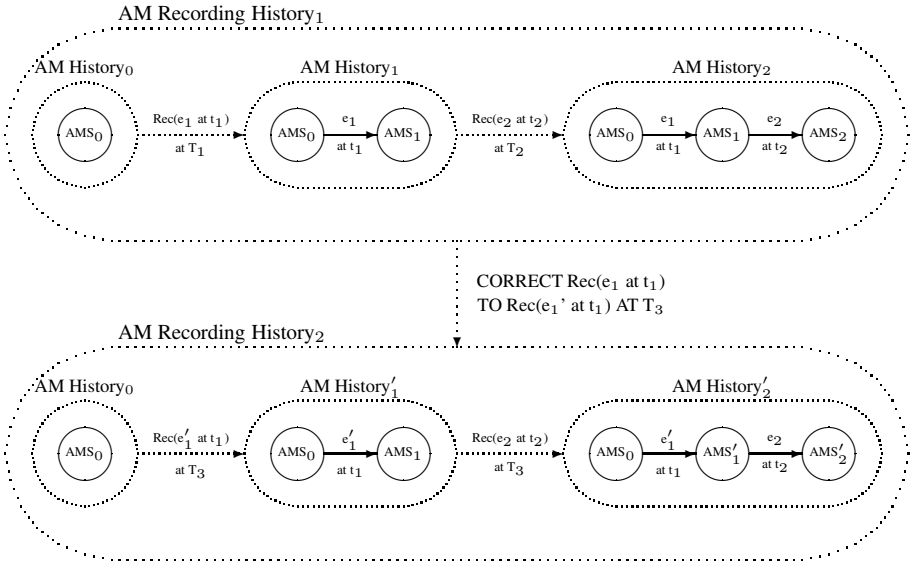

Figure 6: Application Model Recording History (AMRH) transition on the correction level: replacement of an erroneously recorded event

Now we will consider the case of late recordings. In the situation of figure 4, suppose we forgot to record event e_0 which actually happened at intrinsic time t_0. In the same way as in our previous example a roll-back should take place to AM History$_0$, followed by the recording of the forgotten event e_0 and anew recording of events e_1 and e_2. In figure 7 the late recording of an event by means of the roll back operator is depicted.

Note that a late recording of an event may both be seen as an insertion of an event into the

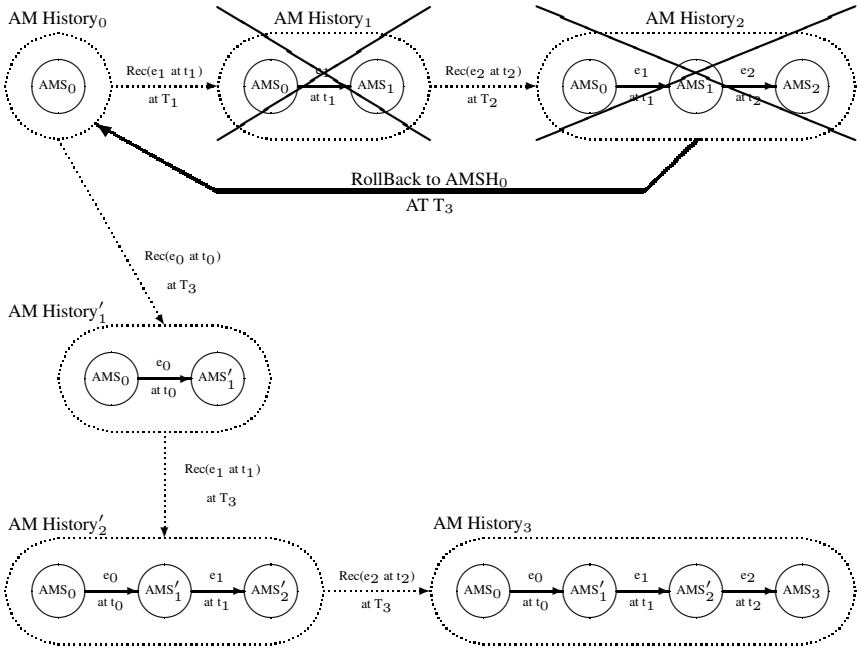

Figure 7: Correction by roll-back: a late recording

sequence of events, and as the correction of 'no event' in 'an event'. Consequently, a late recording can also be viewed from the perspective that our current world is no longer valid. Similar to the two different representations of a correction in the previous example figure 7 has also an alternative representation. In figure 8 it can be seen that the current world gives place to a new world with a new sequence of recordings of events.

The act of forgetting something, finally, may be seen as removing all its trace from the application model so that even the fact that it has been forgotten can not be discovered. So, the forget operator can be thought of as operating on an application model (evolution) as a whole, and resulting in the creation of a new application model (evolution) which contains no trace of the forgotten information.

3.2 Specifying the application model

Besides providing rules and constraints for manipulation (update) of the application model, the meta model should provide a set of concepts and axioms for specifying the application model. In the previous subsection we identified application model states without telling what such a state is constituted of. In this subsection we look at these application model states which should reflect states in the underlying organization domain in more detail.

First of all, building blocks for describing the application model states are required. We will call these building blocks application model elements. Thus an application model state is a set of application model elements. Conform the distinction between a world, activity and behaviour part of the application model, these application model elements will be further specialized.

The application model elements for describing the activity part of the application model - we will call them activity model elements - can be represented conform existing activity

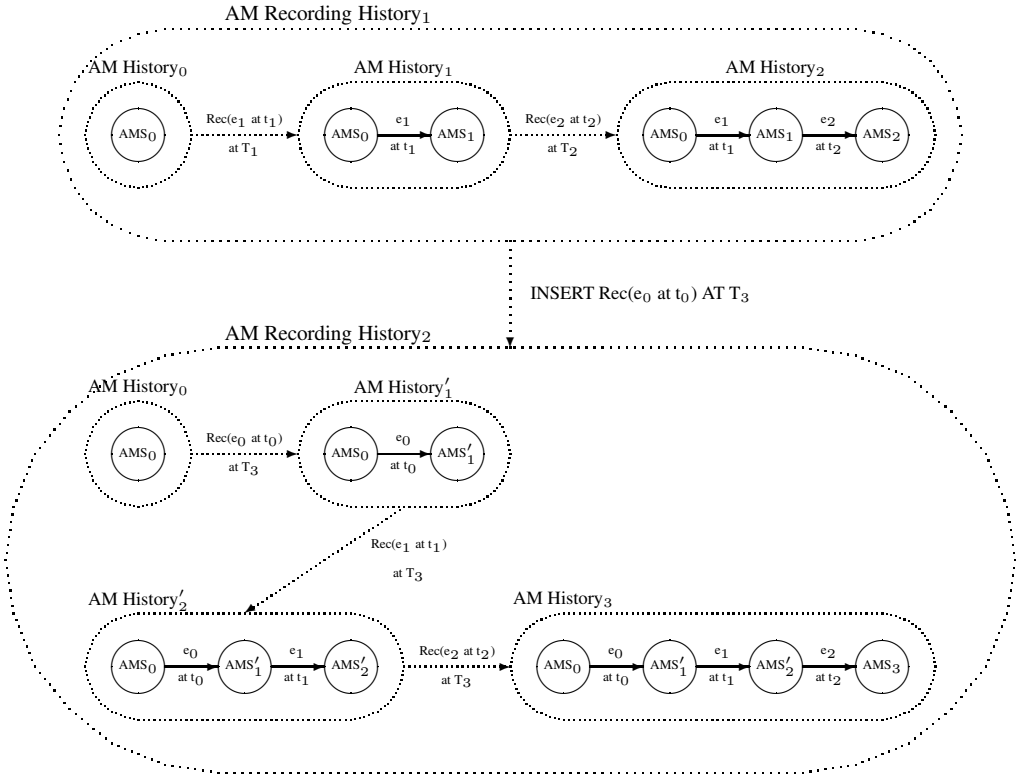

Figure 8: AMRH transition: a late recording

specification techniques. The representation of application model elements for describing the behaviour part - we will call them behaviour model elements - may be based on 'when-if-then' rules ([RR82] and [ALLS88]) which specify (trigger-)relations between events and the activities specified by the activity model elements.

The world part of the application model, finally, is described by application model elements which we call world model elements. These world model elements may be further specialized into elements which specify information base objects, e.g. entities and relationships (e.g. [Che76], [Fal74], [NH89], [OSV82], etc.), and also elements specifying the types of these information base objects. This latter set of elements constitutes the so-called information structure. Moreover, elements can be distinguished which specify constraints on the possible population of this information structure, as well as elements specifying (derivation) rules for deriving information not explicitly stored in the information base ([BHW91]).

An important requirement that involves all application model elements, is that they should be time-stamped. This is a consequence of the fact that, during the evolution of the application model, elements may be created, terminated or changed, without forgetting the former states of these elements. Note that an event causes a new application model state, and that an application model state consists of a set of application model elements. Consequently, an event is modelled by means of a creation, termination or change of an application model element.

A final statement that we will make about these application model elements, involves causal dependencies between the several (specializations of) application model elements. For example, the termination of a world model element causes the termination of all activity model elements which specify operations on this world model element. In general terms, a creation, termination or change of an element may cause another element to be created, terminated or changed.

4 A FORMALISATION OF THE FRAMEWORK

In this section we will give a formalisation of the key concepts as they were informally defined in section 3.1. These concepts will be defined by mainly considering the problem of reflecting events, which took place in the organisation system, in an information system, i.e. the formalisation describes the relationship between events occuring in the organisation system, and events in the information system changing the application model.

The first definitions are concerned with the state spaces of both an organisation system, and an information system which is presumed to reflect the states of the organisation system.

Definition 4.1 *In this definition, we introduce two concepts which will be regarded as basic concepts, in the scope of this paper.*

1. S^{OSS} *is the state space for any possible organisation system. (universe of discourse).*
2. S^{ISS} *is the state space of all information systems for any organisation system.*

For the moment we will presume that the elements of the above sets, i.e. the possible states, are sets of elements (sets of sets!). In this section we are not concerned with the question what kind of elements should be present in these states. In section 3.2, we already discussed

some aspects with respect to the constituent elements of application model states, i.e. the application model elements.

The states of the organisation system have to be mapped onto the states of the information system in some way. The formal relation between the above two sets, is thus given by the following axiom:

Axiom 4.1

There exists a partial homomorphism: $\mathcal{V}_s : S^{OSS} \rightarrow S^{ISS}$*, which reflects the relation between the states in the organization system and the states in the information system.*

We have defined the meta model to be that part of an information system which is both application and time independent. This can be reflected in the following formal definition of a meta model.

Definition 4.2

$\mathcal{MM} \stackrel{\mathrm{d}}{=} \bigcap_{s \in S^{ISS}}(s)$
The meta model is the set of elements which are both time and application independent.

Note that this definition does not state anything about the structure of the meta model nor about the concepts present in the meta model.

Now that we have defined what a meta model is, we can also define the set of application model states (the changing elements):

Definition 4.3

$\mathcal{AMS} \stackrel{\mathrm{d}}{=} \{s - \mathcal{MM} | s \in S^{ISS}\}$
The state space of any possible application model - the allowed set of application model states - consists of those elements of the information system states which are not in the metamodel.

The sets of transitions between the states of the organisation systems, information systems and application model states are defined as:

Definition 4.4

1. $Tr^{OSS} \subseteq (S^{OSS} \times S^{OSS})$ *the set of all possible transitions between states of organisations. Note that not all elements of* $S^{OSS} \times S^{OSS}$ *are proper transitions. Some transitions may be forbidden by a transition oriented constraint.*
2. $Tr^{ISS} \subseteq (S^{ISS} \times S^{ISS})$ *the set of all possible transitions between states of information systems. Note that here too, not all elements of* $S^{ISS} \times S^{ISS}$ *are proper transitions.*

3. $Tr^{AMS} \stackrel{\mathrm{d}}{=} \{\ \langle s_1 - \mathcal{MM}, s_2 - \mathcal{MM} \rangle \mid \langle s_1, s_2 \rangle \in Tr^{ISS} \}$ *the set of all transitions between states of the application models.*
   Note: $Tr^{AMS} \subseteq (\mathcal{AMS} \times \mathcal{AMS})$

We consider an event to be a transition - change of state - occuring at a certain point of time. So when a transition (in Tr^{OSS} or Tr^{ISS}) takes place at a point of time, an event is said to occur. Therefore, the sets of events in the organisation system, as well as their counterparts in the information systems, can be defined as:

Definition 4.5

1. $E^{OSS} \stackrel{\mathrm{d}}{=} (Tr^{OSS} \times \mathcal{T}^{OS})$ *the set of all events which can occur in the organisation system where* $\mathcal{T}^{OS}$ *represents the time-system used in the organisation system.*
2. $E^{ISS} \stackrel{\mathrm{d}}{=} (Tr^{ISS} \times \mathcal{T}^{IS})$ *the set of all events in the information system which reflect the events in the organisation system where* $\mathcal{T}^{IS}$ *represents the time-system used in the information system. This time-system is used for the points of time at which the events in the organisation system are reflected in the information system (extrinsic time).*
3. $E^{AMS} \stackrel{\mathrm{d}}{=} (Tr^{AMS} \times \mathcal{T}^{OS})$ *the set of all events in the application model which reflect the events in the organisation system. Note that the time at which the event occured in the organisation system is reflected directly into the occurence time of the event in the application model (intrinsic time).*

We will presume that there exists an ordering $<_{OS}$ on $\mathcal{T}^{OS}$, and $<_{IS}$ on $\mathcal{T}^{IS}$.

Furthermore, we demand that there can only be one transition at one point of time. This can be reflected in the following axiom:

Axiom 4.2

1. $\langle tr_1, \tau \rangle \in E^{OSS} \Rightarrow \neg\exists_{tr_2}[\langle tr_2, \tau \rangle \in E^{OSS} \wedge tr_1 \neq tr_2]$
2. $\langle tr_1, \tau \rangle \in E^{ISS} \Rightarrow \neg\exists_{tr_2}[\langle tr_2, \tau \rangle \in E^{ISS} \wedge tr_1 \neq tr_2]$
3. $\langle tr_1, \tau \rangle \in E^{AMS} \Rightarrow \neg\exists_{tr_2}[\langle tr_2, \tau \rangle \in E^{AMS} \wedge tr_1 \neq tr_2]$

In the sequel we will need some simple functions which will be used to 'access' the constituent parts of events. These functions are:

Definition 4.6

1. - $\mathsf{FROM} : (E^{OSS} \cup E^{ISS} \cup E^{AMS}) \rightarrow (S^{OSS} \cup S^{ISS} \cup \mathcal{AMS})$
   - $\mathsf{FROM}(\langle \langle s_1, s_2 \rangle, \tau \rangle) \stackrel{\mathrm{d}}{=} s_1$
2. - $\mathsf{TO} : (E^{OSS} \cup E^{ISS} \cup E^{AMS}) \rightarrow (S^{OSS} \cup S^{ISS} \cup \mathcal{AMS})$
   - $\mathsf{TO}(\langle \langle s_1, s_2 \rangle, \tau \rangle) \stackrel{\mathrm{d}}{=} s_2$
3. - $\mathsf{AT} : (E^{OSS} \cup E^{ISS} \cup E^{AMS}) \rightarrow (\mathcal{T}^{OS} \cup \mathcal{T}^{IS} \cup \mathcal{AMS})$

- $\mathsf{AT}(\langle\langle s_1, s_2\rangle, \tau\rangle) \stackrel{\mathrm{d}}{=} \tau$

The time ordering which was presumed to be present in both the organisation and the information system can be inherited by the events by means of the following definition:

Definition 4.7

1. $<_{\mathcal{T}}$ **is-a-predicate-over** $(E^{OSS} \times E^{OSS}) \cup (E^{ISS} \times E^{ISS}) \cup (E^{AMS} \times E^{AMS})$
2. $e_1 <_{\mathcal{T}} e_2 \stackrel{\mathrm{d}}{=} \begin{cases} \mathsf{AT}(e_1) <_{OS} \mathsf{AT}(e_2) & \text{if } e_1, e_2 \in E^{OSS} \\ \mathsf{AT}(e_1) <_{IS} \mathsf{AT}(e_2) & \text{if } e_1, e_2 \in E^{ISS} \\ \mathsf{AT}(e_1) <_{OS} \mathsf{AT}(e_2) & \text{if } e_1, e_2 \in E^{AMS} \end{cases}$

Based on the above defined order, we can define a function returning the first event in a set of events.

Definition 4.8

- $\mathsf{FIRST} : (\wp(E^{OSS}) \cup \wp(E^{ISS}) \cup \wp(E^{AMS})) \rightarrow (E^{OSS} \cup E^{ISS} \cup E^{AMS})$
- $\mathsf{FIRST}(H) \stackrel{\mathrm{d}}{=} e$ *such that* $e \in H \wedge \neg\exists_{e' \in H}[e' <_{\mathcal{T}} e]$

Similar to the fact that there exists a mapping between the states in the organisation system and the information system, there should exist a mapping between the events in the organisation system and the information system. This leads to the following axiom:

Axiom 4.3

There is a mapping $\mathcal{V}_e : E^{OSS} \rightarrow E^{ISS}$ *such that:*

1. $\forall_{s_1, s_2 \in E^{OSS}}[\exists_{t_1, t_2}[\mathcal{V}_e(\langle\langle s_1, s_2\rangle, t_1\rangle) = \langle\langle \mathcal{V}_s(s_1), \mathcal{V}_s(s_2)\rangle, t_2\rangle]]$
   Every event which occured in the organisation system is properly reflected in the information system.
2. $\forall_{e_1, e_2 \in E^{OSS}}[e_1 <_{\mathcal{T}} e_2 \Leftrightarrow \mathcal{V}_e(e_1) <_{\mathcal{T}} \mathcal{V}_e(e_2)]$
   The order in which the events occured in the organisation system is preserved by the mapping.

The actual history of an organisation system, an information system, and the application model respectively is denoted by means of a tuple consisting of an initial state, and a set of events describing the way in which the initial state has evolved. More formally:

Definition 4.9 *The sets of possible histories of organisation systems, information systems and application models, are defined as:*

1. $OSH \subseteq \{\langle \sigma^{OSS}, H^{OSS}\rangle \mid \sigma^{OSS} \in S^{OSS} \wedge H^{OSS} \subseteq E^{OSS}\}$
2. $ISH \subseteq \{\langle \sigma^{ISS}, H^{ISS}\rangle \mid \sigma^{ISS} \in S^{ISS} \wedge H^{ISS} \subseteq E^{ISS}\}$
3. $\mathcal{AMH} \subseteq \{\langle \sigma^{AMS}, H^{AMS}\rangle \mid \sigma^{AMS} \in \mathcal{AMS} \wedge H^{AMS} \subseteq E^{AMS}\}$

In any organisation system the following rather simple property, stating that every first event should originate from the initial state and that of every two successive events the result state of the former event is the initial state of the succeeding event, should hold. This property can be defined formally as:

Property 4.1

Let $\langle \sigma^{OSS}, H^{OSS} \rangle \in$ *OSH be the history of an organisation, then:*

1. $\mathsf{FROM}(\mathsf{FIRST}(H^{OSS})) = \sigma^{OSS}$
   The first event must originate from the initial state.
2. $e_1, e_2 \in H^{OSS} \wedge e_1 <_{\mathcal{T}} e_2 \wedge \neg\exists_{e_3 \in H^{OSS}}[e_1 <_{\mathcal{T}} e_3 <_{\mathcal{T}} e_2] \Rightarrow \mathsf{TO}(e_1) = \mathsf{FROM}(e_2)$
   Every event must originate from the result state of its directly preceeding event, if present.

This property implies the following axiom for the set of events in the information system, and in the application model:

Axiom 4.4

- *Let* $\langle \sigma^{ISS}, H^{ISS} \rangle \in$ *ISH be the history of an organisation, then:*
  1. $\mathsf{FROM}(\mathsf{FIRST}(H^{ISS})) = \sigma^{ISS}$
  2. $e_1, e_2 \in H^{ISS} \wedge e_1 <_{\mathcal{T}} e_2 \wedge \neg\exists_{e_3 \in H^{ISS}}[e_1 <_{\mathcal{T}} e_3 <_{\mathcal{T}} e_2] \Rightarrow \mathsf{TO}(e_1) = \mathsf{FROM}(e_2)$
- *Or when we focus on the application models states. Let* $\langle \sigma^{AMS}, H^{AMS} \rangle \in \mathcal{AMH}$, *then:*
  1. $\mathsf{FROM}(\mathsf{FIRST}(H^{AMS})) = \sigma^{AMS}$
  2. $e_1, e_2 \in H^{AMS} \wedge e_1 <_{\mathcal{T}} e_2 \wedge \neg\exists_{e_3 \in H^{AMS}}[e_1 <_{\mathcal{T}} e_3 <_{\mathcal{T}} e_2] \Rightarrow \mathsf{TO}(e_1) = \mathsf{FROM}(e_2)$

The actual recording of events which occur in the organisation system is (partially) done by humans, and therefore prone to errors. The actual recording function, which in practice will deviate from $\mathcal{V}_e$, is thus defined as:

Definition 4.10

$\mathsf{Rec} : E^{OSS} \rightarrow E^{ISS}$

The mapping (bijection) of events in the organisation system onto events in the information system, as it resulted from the recording performed by the user.

Now we are also in a position to define the extrinsic and intrinsic time of events.

Definition 4.11

1. • $\mathcal{T}^{Intr} : E^{OSS} \rightarrow \mathcal{T}^{OS}$

- $\mathcal{T}^{Intr}(e) \stackrel{\mathrm{d}}{=} \mathsf{AT}(e)$

2. - $\mathcal{T}^{Extr} : E^{OSS} \rightarrow \mathcal{T}^{IS}$
   - $\mathcal{T}^{Extr}(e) \stackrel{\mathrm{d}}{=} \mathsf{AT}(\mathsf{Rec}(e))$

With the above defined recording function, and intrinsic time function, the relation between a history of the organisation system, and a history of application model states can be defined. This relation can be defined formally as:

Axiom 4.5 *Let $\langle \sigma^{OSS}, H^{OSS} \rangle \in$ OSH be an organisation system history, and* Rec *a recording function. Then the application model state history $\langle \sigma^{AMS}, H^{AMS} \rangle \in \mathcal{AMH}$ belonging to the organisation system state history is given by:*

$$\sigma^{AMS} = \mathcal{V}_{ams}(\sigma^{OSS})$$
$$H^{AMS} = \left\{ \left\langle \left\langle \mathcal{V}_{ams}(\mathsf{FROM}(\mathsf{Rec}(e))), \mathcal{V}_{ams}(\mathsf{TO}(\mathsf{Rec}(e))) \right\rangle, \mathcal{T}^{Intr}(e) \right\rangle \middle| e \in H^{OSS} \right\}$$

where $\mathcal{V}_{ams}$ is defined as: $\mathcal{V}_{ams}(s) \stackrel{\mathrm{d}}{=} \mathcal{V}_s(s) - \mathcal{MM}$

Now that we know what a recording is, we can also define the concept of application model recording history more formally. As said before, an application model recording history consists of an initial application model state history, and a sequence of transitions (recordings). An application model recording history reflects the recording function which in itself reflects the believe in the world of the person(s) which perform(s) the recordings, at a certain point of time.

Definition 4.12

1. *$Tr^{AMH} \subseteq (\mathcal{AMH} \times \mathcal{AMH})$ the set of transitions between histories of application model states.*
2. *$E^{AMH} \stackrel{\mathrm{d}}{=} (Tr^{AMH} \times \mathcal{T}^{IS})$ the set of events involving transitions of histories of application model states, i.e. the recordings! The point of time is, of course, the recording time $\mathcal{T}^{Extr}(e)$.*
3. *$\mathcal{AMRH} \subseteq \{\langle \sigma^{AMH}, H^{AMH} \rangle | \sigma^{AMH} \in \mathcal{AMH} \wedge H^{AMH} \subseteq E^{AMH}\}$*
   Note that we presume the definition of FROM, FIRST *and* AT *to be extended to E^{AMH}.*

The axiom which held for a sequence of states, has to hold for a sequence of histories as well. So the following axiom can be formulated:

Axiom 4.6 *Let $\langle \sigma^{AMH}, H^{AMH} \rangle \in \mathcal{AMRH}$ be a recording history, then:*

1. $\mathsf{FROM}(\mathsf{FIRST}(H^{AMH})) = \sigma^{AMH}$
2. $e_1, e_2 \in H^{AMH} \wedge e_1 <_{\mathcal{T}} e_2 \wedge \neg\exists_{e_3 \in H^{AMH}}[e_1 <_{\mathcal{T}} e_3 <_{\mathcal{T}} e_2] \Rightarrow \mathsf{TO}(e_1) = \mathsf{FROM}(e_2)$

For a given recording function and history of the organisation system, the application model recording history resulting from this, can now be related by means of the following axiom:

Axiom 4.7 *Let* Rec *be a recording function and* $\langle \sigma^{OSS}, H^{OSS} \rangle \in$ *OSH an organisation system history. Then the application model recording history* $\langle \sigma^{AMH}, H^{AMH} \rangle \in \mathcal{AMRH}$ *based on them is given by:*

- *The set of events in* H^{OSS} *is recorded by* Rec *in a certain order. Based on this order we can define a sequence of growing sets of events* H_n^{OSS} *based on the extrinsic (recording) time of the events.*

  1. $H_0^{OSS} \stackrel{\mathrm{d}}{=} \emptyset$

     Initially no events have been recorded.

  2. *For* $n \geq 1$*:*

     $$H_n^{OSS} \stackrel{\mathrm{d}}{=} \begin{cases} H^{OSS} & \text{if } H^{OSS} - H_{n-1}^{OSS} = \emptyset \\ H_{n-1}^{OSS} \cup \{e\} & \text{otherwise} \end{cases}$$

     Where: $\mathsf{FIRST}(\{\mathsf{Rec}(f) | f \in H^{OSS} - H_{n-1}^{OSS}\}) = \mathsf{Rec}(e)$

     In words: 'the first recorded event in the set $H^{OSS} - H_{n-1}^{OSS}$*'.*

     The n*-th set of recorded events contains all previously recorded* $(n-1$*-th) events and the first, still to be recorded, event.*

- *Let, for all* n*,* AMH_n *be defined as the application model history based on* Rec *and the organisation system history* $\langle \sigma^{OSS}, H_n^{OSS} \rangle$ *conform axiom 4.5.*

- *The application model recording history itself* $\langle \sigma^{AMH}, H^{AMH} \rangle$ *is now given by:*

  $$\sigma^{AMH} = AMH_0$$
  $$H^{AMH} = \left\{ \langle \langle AMH_i, AMH_{i+1} \rangle, \mathcal{T}^{Extr}(H_{i+1}^{OSS} - H_i^{OSS}) \rangle \,\middle|\, o \leq i < |H^{OSS}| \right\}$$

  Note: $H_{i+1}^{OSS} - H_i^{OSS}$ *contains only one element, so with* $\mathcal{T}^{Extr}(H_{i+1}^{OSS} - H_i^{OSS})$ *we mean the extrinsic time of that single event.*

In an ideal situation, the recording function would obey the axiom (4.3) with respect to $\mathcal{V}_e$. In that case, the actual recording function would be a correct mapping from the events in the organisation system to events in the information system. This requirement can be formalised as:

Requirement 4.1

1. $\forall_{s_1,s_2 \in E^{OSS}} [\exists_{t_1,t_2} [\mathsf{Rec}(\langle \langle s_1, s_2 \rangle, t_1 \rangle) = \langle \langle \mathcal{V}_s(s_1), \mathcal{V}_s(s_2) \rangle, t_2 \rangle]]$
2. $\forall_{e_1,e_2 \in E^{OSS}} [e_1 <_{\mathcal{T}} e_2 \Leftrightarrow \mathsf{Rec}(e_1) <_{\mathcal{T}} \mathsf{Rec}(e_2)]$

A violation of requirement 4.1 now implies:

$$\neg \forall_{s_1,s_2 \in E^{OSS}} [\exists_{t_1,t_2} [\mathsf{Rec}(\langle \langle s_1, s_2 \rangle, t_1 \rangle) = \langle \langle \mathcal{V}_s(s_1), \mathcal{V}_s(s_2) \rangle, t_2 \rangle]] \vee$$
$$\neg \forall_{e_1,e_2 \in E^{OSS}} [e_1 <_{\mathcal{T}} e_2 \Leftrightarrow \mathsf{Rec}(e_1) <_{\mathcal{T}} \mathsf{Rec}(e_2)] \Leftrightarrow$$

$$\exists_{s_1,s_2 \in E^{OSS}} [\neg \exists_{t_1,t_2} [\mathsf{Rec}(\langle \langle s_1, s_2 \rangle, t_1 \rangle) = \langle \langle \mathcal{V}_s(s_1), \mathcal{V}_s(s_2) \rangle, t_2 \rangle]] \vee$$
$$\exists_{e_1,e_2 \in E^{OSS}} [\neg (e_1 <_{\mathcal{T}} e_2 \Leftrightarrow \mathsf{Rec}(e_1) <_{\mathcal{T}} \mathsf{Rec}(e_2))] \Leftrightarrow$$

$$\exists_{s_1,s_2 \in E^{OSS}} [\forall_{t_1,t_2} [\mathsf{Rec}(\langle \langle s_1, s_2 \rangle, t_1 \rangle) \neq \langle \langle \mathcal{V}_s(s_1), \mathcal{V}_s(s_2) \rangle, t_2 \rangle]] \vee$$
$$\exists_{e_1,e_2 \in E^{OSS}} [\neg (e_1 <_{\mathcal{T}} e_2 \Leftrightarrow \mathsf{Rec}(e_1) <_{\mathcal{T}} \mathsf{Rec}(e_2))]$$

From $\exists_{s_1,s_2 \in E^{OSS}}[\forall_{t_1,t_2}[\mathsf{Rec}(\langle\langle s_1, s_2\rangle, t_1\rangle) \neq \langle\langle \mathcal{V}_s(s_1), \mathcal{V}_s(s_2)\rangle, t_2\rangle]]$ follows that there was either a wrong recording of an event, or a recording of an event which actually did not occur in the organisation system at all.

And from $\exists_{e_1,e_2 \in E^{OSS}}[\neg(e_1 <_{\mathcal{T}} e_2 \Leftrightarrow \mathsf{Rec}(e_1) <_{\mathcal{T}} \mathsf{Rec}(e_2))]$ follows that some events were recorded in a wrong order.

Both cases imply that the recording function which the user(s) believed to be true has to be revised.

What does the above mean for the application model recording histories? If the recording function has to be revised, based on axiom 4.7, it should result in a revised application model recording history ($\mathcal{AMRH}$) in which the error is hopefully corrected. This will be done by means of the roll-back mechanism as explained in section 3.1. Thus, after every correction, we end up with a new application model recording history. Since we decided to remember everything, including any correction, an application model's evolution can be described by means of an initial application model recording history, and a sequence of transitions (corrections) between successive application model recording histories.

Definition 4.13

1. $Tr^{AMRH} \subseteq (\mathcal{AMRH} \times \mathcal{AMRH})$ *the set of valid transitions between recording histories.*
2. $E^{AMRH} \stackrel{\mathrm{d}}{=} (Tr^{AMRH} \times \mathcal{T}^{IS})$ *the corrections between application model recording histories. The point of time is the (extrinsic) point of time at which the correction is performed.*
3. $\mathcal{AME} \subseteq \{\langle \sigma^{AMRH}, H^{AMRH}\rangle \,|\sigma^{AMRH} \in \mathcal{AMRH} \wedge H^{AMRH} \subset \mathcal{AMRH}\}$
   Note that we presume the definition of $\mathsf{FROM}, \mathsf{FIRST}$ *and* AT *to be extended to* E^{AMRH}.

It should be clear now that the axiom which held for a sequence of states and a sequence of histories, should hold for a sequence of recording histories as well. So the following axiom can be formulated:

Axiom 4.8 *Let* $\langle \sigma^{AMRH}, H^{AMRH}\rangle \in \mathcal{AME}$ *be the history of an organisation, then:*

1. $\mathsf{FROM}(\mathsf{FIRST}(H^{AMRH})) = \sigma^{AMRH}$
2. $e_1, e_2 \in H^{AMRH} \wedge e_1 <_{\mathcal{T}} e_2 \wedge \neg\exists_{e_3 \in H^{AMRH}}[e_1 <_{\mathcal{T}} e_3 <_{\mathcal{T}} e_2] \Rightarrow \mathsf{TO}(e_1) = \mathsf{FROM}(e_2)$

With this definition of an application model evolution, we conclude the formalization of the framework. We have seen that the definitions of all three levels of the hierarchy ($\mathcal{AMH}, \mathcal{AMRH}, \mathcal{AME}$) indeed have the same construction, something which could already be expected from the discussion in the previous section.

5 CONCLUSION

From a systems point of view on organizations, we required an information system to evolve to the same extent and at the same pace at which the organization evolves, because of changes in the organization domain. In this paper we presented a conceptual framework for such an evolving information system, considering traditional information systems to be degenerations of these evolving systems. In the architecture that we proposed, a distinction of the processing model was made into a meta model and an application model on the basis of application dependency. In accordance with different perspectives, a world model and an action model were distinguished, further subdividing the action model in an activity and a behaviour model.

In order to handle temporal and evolutionary aspects in an evolving information system, we revised the traditional notion of update, resulting in the triple: recording, correction and forgetting. With this notion of update, we required the meta model to provide concepts and axioms which could support the update of all constituent parts of the application model. We saw that update should not forget any aspect ever fed to the system, unless explicitly asked for. The notion of updating the application model was described by introducing a state-transition-oriented model on three levels of abstraction (organization, recording and correction level). This model had been formalized in section 4.

Finally, we discussed specializations of what we called application model elements, being the building blocks for specifying the application model states. Additionally, we stated that dependencies between these elements should be identified and that all application model elements should be time-stamped.

The conceptual framework proposed in this paper positions evolving information systems with respect to traditional systems. The framework is the basis for a meta model of a generalized evolving information system. In this meta-modelling process, further work is necessary. This work involves the completion and formalization of the meta model, and the design of a language for manipulating and specification of application models. Furthermore, an information system shell will be implemented based on that meta model and that language, and a design method for the process of building up and maintaining an application model of an evolving information system are to be developed.

Acknowledgement

We would like to thank Mark M^{c}Loughlin for his valuable contributions during the discussions in the early phases of the EIS (Evolving Information Systems) project.

References

[ALLS88] F. van Assche, P. Layzell, P. Loucopoulos, and G. Speltincx. RUBRIC: A Rule-Based Representation of Information System Constructs. In *ESPRIT '88 Proceedings of the 5th Annual Esprit Conference*, Brussels, Belgium, November 1988. North-Holland.

[BHW91] P. van Bommel, A.H.M. ter Hofstede, and Th.P. van der Weide. Semantics and verification of object-role models. *Information Systems*, 16(5):471–495, October 1991.

[Bub80] J.A. Bubenko. Information Modelling in the Context of System Development. In S.H. Lavington, editor, *Information Processing 80*, pages 395–411. North-Holland/IFIP, Amsterdam, The Netherlands, 1980.

[Che76] P.P. Chen. The entity-relationship model: Towards a unified view of data. *ACM Transactions on Database Systems*, 1(1):9–36, March 1976.

[Fal74] E.D. Falkenberg. Time-Handling in Data Base Management Systems. Report 07/1974, University of Stuttgart, West-Germany, 1974.

[FHL+98] E.D. Falkenberg, W. Hesse, P. Lindgreen, B.E. Nilsson, J.L.H. Oei, C. Rolland, R.K. Stamper, F.J.M. Van Assche, A.A. Verrijn-Stuart, and K. Voss, editors. *A Framework of Information Systems Concepts*. IFIP WG 8.1 Task Group FRISCO, 1998. ISBN 3-901-88201-4

[ISO87] *Information processing systems – Concepts and Terminology for the Conceptual Schema and the Information Base*, 1987. ISO/TR 9007:1987. `http://www.iso.org`

[NH89] G.M. Nijssen and T.A. Halpin. *Conceptual Schema and Relational Database Design: a fact oriented approach*. Prentice-Hall, Sydney, Australia, 1989. ASIN 0131672630

[OHM+88] T.W. Olle, J. Hagelstein, I.G. Macdonald, C. Rolland, H.G. Sol, F.J.M. van Assche, and A.A. Verrijn-Stuart. *Information Systems Methodologies: A Framework for Understanding*. Addison-Wesley, Reading, Massachusetts, USA, 1988. ISBN 0-201-54443-1

[OSV82] T.W. Olle, H.G. Sol, and A.A. Verrijn-Stuart, editors. *Information Systems Design Methodologies: A Comparative Review*. North-Holland/IFIP WG8.1, Amsterdam, The Netherlands, EU, 1982. ISBN 0-444-86407-5

[RBL87] C. Rolland, F. Bodart, and M. Leonard, editors. *Temporal Aspects in Information Systems*. North-Holland/IFIP, Amsterdam, The Netherlands, 1987.

[RR82] C. Rolland and C. Richard. The REMORA Methodology for Information System Design and Management. In T.W. Olle, H.G. Sol, and A.A. Verrijn-Stuart, editors, *Information Systems Design Methodologies: A Comparative Review*, pages 369–426. North-Holland/IFIP WG8.1, Amsterdam, The Netherlands, EU, 1982.

[RV90] G.J. Ramackers and A.A. Verrijn-Stuart. First and Second order Dynamics in Information Systems. In H.G. Sol and K.M. van Hee, editors, *Proceedings of the International Working Conference on Dynamic Modelling of Information Systems*, Noordwijkerhout, The Netherlands, April 1990.

[SA86] R. Snodgrass and I. Ahn. Temporal Databases. *IEEE Computer*, 19(9):35–42, 1986.