*open*EHR Instruction proposal

Authors:{T Beale, S Heard}[1]

Amendment Record

Issue	Details	Who	Completed
0.6	Major rework to explanatory sections; models modified.	T Beale	13 Jun 2005
0.5.2	Minor renaming	T Beale	15 May 2005
0.5.1	Renamed ‘implicit workflow’	T Beale	5 May 2005
0.5	Initial Writing.	S Heard, T Beale	20 Apr 2005

1. Ocean Informatics Australia

1 Introduction

1.1 Background

This proposal deals with instructions, the *open*EHR term for act specifications, and act executions. Instructions correspond to the future or 'prospective' category of information one can find in the EHR, in contrast with past information (observations) and decisional and opinion information (known in *open*EHR as 'evaluations'). All medication orders, drug administrations and recalls are some kind of Instruction, and its correct modelling and use in EHR systems is of vital importance to their use in clinical medicine.

Instructions in healthcare are often complex. Generally this complexity is dealt with by using natural language. Consider the following example of the instructions for a patient with asthma who is having an exacerbation.

> "Take prednisolone 25mg 2 tablets in the morning until your peak flow has been above 350 l/min for 2 days, then take 1 tablet in the morning. I want to review you in one week, you will probably still be taking your medication."

This management regime may be based on a guideline suggesting that if a person's peak flow falls below 50% of their best (maximum recorded value when well) or predicted (median value for height and gender) peak flow - whichever is lower - then give 0.5-1mg/kg/day in a single dose in the morning until their peak flow is above 90% of the target value on two consecutive mornings; subsequently, give half the dose for the same period of time.

Instructions in the health record are explicit statements about what someone should do - these may be personalised expressions of a guideline or evidence-based best practice. Some personalisation can be automated on the basis of weight or age. There are however many aspects which require judgement based on the wishes of the patient, the functioning of their kidneys and liver, their mental state and many other intangible factors, and which are therefore hard if not completely impractical to automate.

The aproach used in this proposal unites ideas from previous attempts at modelling Instruction in *open*EHR with a new analysis of clinical processes and standard states from S Heard, and relies heavily on the concepts of the AgentWork model and the ActiveTFL language described in Müller [2].

1.2 Requirements

Broadly speaking, Instructions can be divided into two groups: those which will be directly processed by humans (e.g. by reading the Instruction information in the EHR and acting on it), and those which will be processed by workflow software. The same clinical activity, such as the prednisolone administration above might equally be completely manual (patient simply reads instructions from medication box and occasionally reports back to GP), fully automated (notifications are created for each administration of drug and peak flow measurement throughout the day), or partially automated.

Requirements for a model of Instructions in *open*EHR include the following.

- All kinds of interventions should be representable using the same model, including medication orders, surgical procedures, and other therapies or invasive investigations. Complex and simple medications should be representable.
- The freedom must exist to model any particular intervention in as much or little detail as required by local circumstances. For example, the order for a laparoscopic cholecystectomy might be represented in one system by a single term simply naming it along with a date. In

another system it may be useful to represent the procedure in the fine-grained detail of anasthesia, incisions etc.

- Clinicians must be able to specify workflows in their own terms, i.e. using terms like “prescribe”, “dispense”, “start administering”, etc.
- Instructions representing diverse clinical workflows must be queryable in a standard way, so that it can be ascertained what instructions are ‘active’, ‘completed’ and so on for a patient.
- It should be possible to provide a coherent view of phases of an instruction, or a series of related instructions, which have been executed in different healthcare provider environments.
- The representation of the specification of an Instruction (statement of what is to happen in the future) must be distinguished from the representation of its execution (data recording what actually did happen, including modifications, events on the way), both for practical reasons of workflow engine operation (i.e. knowing what part of a workflow has already executed) and for the clinical reason that what is executed may not exactly match what was specified.
- Allow the recording of *ad hoc* act executions in the record, i.e. acts for which no Instruction was defined (at least in the EHR in question).
- To correctly link Instructions with the acts and observations which unfold during their execution.
- The model of Instruction and related classes must be archetypable, based on workflows defined by clinicians.

Regarding automation of Instructions, the goal is to enable *open*EHR to support workflow automation without re-inventing it. The EHR is treated as a memory device available to a workflow engine, but does not try to replicate its functionality, nor oblige any automatic workflow processing to be done at all. Additional requirements which apply to instructions which are to be executed by a workflow engine include:

- To support sophisticated triggers, such as “low white blood cells”, “temperature >39.5^{o}C”, which cause Instructions to progress in their cycle.
- To support sophisticated timing specifications for when things should happen, including delays/waits, administration linked to time of day, meals or other patient activities.
- To support notifications in such a way that they can be defined reusably (e.g. by using syntax such as `notify(“some message content”, “receiver”)`, which can be translated locally into appropriate actions such as an email or SMS).
- To allow situations in which a workflow does not execute as defined, e.g. where unplanned exceptions occur.

2 Workflow in Clinical Medicine

2.1 Typology of Processes

From the point of view of patient care and the EHR, all processes of interest are aimed at achieving something for the patient (obviously there are other processes which take place in a hospital for example, such as inventory management of the dispensary, but these do not concern us here). Any given process may therefore have multiple actors (often including the patient), and therefore multiple worklists. However each process will only have a *single patient-related goal*; this is the unifying characteristic of clinical workflows.

In clinical medicine, there appear to be the following common situations, which we can distinguish as being workflows ‘within a single clinical environment’, and processes occurring among participating organisations.

Workflows enacted within a single clinical environment

This kind of workflow is characterised by:

- a natural language definition; optionally a formal, computable definition
- having the scope of one independent clinical process which is meaningful on its own; e.g. a 2 week 3-drug chemotherapy one Instruction, since it has to be delivered in a unit; you can't meaningfully deliver only 2 of the drugs. Even a planned coronary bypass, or a liver transplant would be one Instruction. (Such complex interventions have a number of 'safely executable' pieces; e.g. patient counselling; patient preparation; anasthesia; the actual procedure; post-operative medication etc. Here we mean that the actual coronary bypass operation = 1 instruction; the total set of things that must be done for the patient from start to finish to achieve such surgery would be multiple Instructions).
- being enactable by one or more clinical Actors, e.g. two nurses, or a nurse and the patient;
- being enactable on usually one subject, but not limited to that - e.g. a live kidney transplant has to describe two subjects; amniocentesis also has two subjects.
- possibly being unlimited in time, e.g. a permanent medication instruction for asthma, as described in the previous section.
- corresponding to a procedure which occurs within the one care delivery organisation setting (including patient's home, aged care home etc); this is even if the clinical participants are from different organisations.
- we can also say it corresponds to the interaction of one subject ('case') with one provider organisation

The defining characteristic of this kind of workflow seems to be the last one above: that *it occurs in a single care delivery context*.

The term 'organisational workflow' can be used to describe this kind of workflow.

Examples:

- one chemotherapy regime admninistration
- any basic patient medication administration (including multi-drug)

Processes across organisations

There are business processes, which some people call workflows, which are not about doing something to the patient, but rather about moving the patient to and from different workflow processing contexts where organisational workflows can be enacted.

Such processes are characterised by:

- often not being *defined* as such, but being definable in natural language, and/or formal terms
- having the scope of one business process which delivers an intended clinical outcome; for example, if the intended clinical outcome is "prednisolone administration" (i.e. this is what the doctor and patient actually want to happen), then the business process that achieves it is the prescribe/dispense/administer/repeat/review etc process, which encompasses as actors 2 providers (GP and pharmacy) and the patient. Similarly, if the required clinical intervention is "coronary bypass graft", then the business process is steps such as plan, order, counselling, patient prep, anasthesia, the operation, post-op care events, etc. These are the things that need to be done by the health system as a whole in order for one particular hospital to actually do the operation.
- having an overall “state”, including values like “planned”, “active”, “cancelled” and so on.
- the overall process consists of multiple workflows each enacted in a distinct environment (care organisation, home etc), which are coordinated to form an end-to-end process via communication and synchronisation.

These kinds of processes are therefore about the synchronisation of multiple individual workflows in order to achieve a larger aim, whereas the first type above is clinical process (clinically defined, clinical goals, clinical actions).

The term “inter-organisational business process” can be used to describe this category of process. The defining characteristic of is that it corresponds to the interaction of the patient with the health system as a whole, with the aim of achieving a defined goal.

Examples:

- the typical prescribe / dispense / admin / review process
- a PAP recall - modelled like a repeated medication, with notifications going to the patient every 6-24 months depending on the patient
- lab order / response
- any referral or discharge

2.2 Choreography and Orchestration

In workflow jargon, synchronisation between multiple workflow processes is known by two terms: “choreography” and “orchestration”.

There does not appear to be a global consensus in the workflow community on these terms (see e.g. http://oreillynet.com/pub/wlg/6652). However, convenient and logical definitions are available (http://www.innoq.com/blog/st/2005/02/16/choreography_vs_orchestration.html):

- In orchestration, there’s someone — the conductor — who tells everybody in the orchestra what to do and makes sure they all play in sync.
- In choreography, every dancer follows a pre-defined plan — everyone independently of the others.

Technically this could be differentiated as:

- *orchestration*: there is a central workflow processor whose job it is to perform the signalling and notification to other workflow processors in each organisation - this is a hierarchical, or client/server topology
- *choreography*: there is no central workflow processor, and synchronisation is done by each organisations workflow processor knowing how to communicate to the other relevant workflow processors in other orgs.

In the first, there would be a single, centrally enacted process description, and the effect is really more like a hierarchical workflow. In the second, there may be no single process definition: each organisation may have only its "scriptsheet", and some knowledge of the other organisations/actors with which it needs to communicate. This latter situation appears to correspond most closely with today's clinical business processes, although there is nothing to say that centralised coordination of multi-provider processes will not occur in the future.

Consequently, for the *open*EHR model of `INSTRUCTION`, provision is made for synchronising communication between distinct organisational workflows.

2.3 Where are workflows defined?

Workflow definitions can clearly apply to different populations, from the single individual, to a large category of patient, such as "IDDM patient", to "any patient". How specific they are may determine where the process definitions are recorded - in the EHR or elsewhere.

2.3.1 Case Specificity

Individual Patient workflow

Procedures such as many types of surgery and complex, personalised medications such as anti-coagulation and chemotherapy will usually be specific to the patient, even if the overall model is more or less the same for all patients of that category. Additionally , many simple medications are also specific to the patient at least in dose if not duration, particularly those use for chronic problems such as decongestants, insulin, thyroxin etc.

Kind-of-patient workflow

Many workflows, such as simple surgery and many anti-biotic drug administrations, follow exactly the same process definition for all patients of a particular category. For example, any otherwise healthy adult for whom the diagnosis is "giardiasis" might be given metronidazole 500mg.

Any-patient workflow

Some workflows are completely independent of the patient, such as the one that a GP follows in her surgery:

- accept patient;
- ascertain patient concern;
- perform examination;
- determine assessment;
- create prescription;
- send patient out;
- back to first step.

It is important that this generic process, in which the GP is the enactor processing a queue of patients, is understood as being distinct from the *clinical care process* being defined by the GP for any one of those patients (aimed at resolving severe back pain for example). From the point of view of the EHR, we are only interested in patient-centric care processes that achieve patient goals, not processes whose goals are for other actors. The workflow described above is designed to satisfy the GP's goal of "processing all her patients", and is therefore not of direct interest. From the patient point of view, the passage through the GP's surgery may be represented as simply the "prescribe" step in the larger business process designed to fix the patient's back problem.

2.3.2 Definition and persistence

For patient safety it would appear that any workflow definition must be recorded in the individual patient's EHR, including for processes which are essentially the same for a large category of patients. This is mainly to provide safe, versioned information persistence for such definitions, even if they are routinely stored elsewhere, such as in a workflow service. A number of potential problems exist with relying on workflow services to persist workflow definitions that are to be applied to the patient including the following.

- The definition may be forever opaque to EHR software, and only processable by a particular workflow tool, which may change over time. Not being able to see the definition of a prescription or other intervention in the EHR would not be acceptable to clinicians.
- The definition might be stored in a succession of different formalisms, meaning that even if it were readable once by EHR software, it may not be later on.
- The definition might not be version controlled, so that a reference to a particular process (e.g. "polio vaccination") might resolve to today's version of a treatment (e.g. oral Sabine administration), when in fact the patient had been given an older treatment (the old Salk vaccine).
- Various interventions for the patient over time could easily be formalised in different ways due to different workflow software and formalisms in use in different provider institutions, or even in the same one. Over time, tracking and decoding such definitions could easily become problematic.

In today's clinical environment, most of these problems are not visible, for the simple reason that workflow and decision support are applied to so little of the clinical process, and interventions are not described formally at all, but in narrative text. However in environments where more automation is used, the above problems will quickly manifest.

The conclusion is that the definition of clinical workflows must be formalised in the EHR model of *open*EHR; in other words, *open*EHR must choose or define a standard way to do this (contrary to the approach of the WfMC which leaves workflow formalisms to each vendor). However: they only need to be formalised to the extent of enabling workflow services to create, store and resume processing; the formal representation used inside workflow systems is not directly the business of the EHR model of process definitions.

3 Design Approach

3.1 Overview

The approach used to model Instructions in *open*EHR consists of the following elements:

- models of inter-organisational business processes, including a standard state machine;
- models of organisational workflows;
- an archetypable object model of Instruction, Act, and related concepts, integrated into the *open*EHR EHR Information Model;
- archetypes describing instructions for actual medications and interventions.

Common business processes in the clinical world, such as the chain of activities ‘prescribing’, ‘dispensing’ and ‘administering’, can be described in a semi-formal or formal way. A standard state machine can be defined which describes the progress of any such process in a general, abstract way. Correspondences can be defined between the stages of a business process and the transition events of the standard state machine, enabling the current state of any particular process to be known and recorded, regardless of its details. By this means, a query to the EHR can find out what processes are in states such as ‘active’, ‘suspended’ and so on.

Organisational workflows are defined by clinicians or in software tools to model the fine-grained execution of one or more steps of a business process, usually the ‘administration’ step. They are expressed in natural language if they are to be executed by humans such as the patient or nurse (even ‘take 3 per day with meals until course finished’ is a workflow definition), and optionally in a formal way which is executable by an automatic workflow processor.

This proposal defines subtypes of the *open*EHR `ENTRY` concept, namely `INSTRUCTION`, `ACT`, and `INSTRUCTION_ACT`, along with helper classes, which enable the representation of both the definition and the execution objects of an explicit workflow in the EHR. An instance of `INSTRUCTION` in the EHR indicates an intended intervention on a patient - it is always a description of an explicit workflow. Instances of `INSTRUCTION_ACT` occur in the EHR when anything to do with the actual execution of the instruction occurs; both the business process events (prescribe, dispense, suspend etc) as well as activities described exlicitly by the `INSTRUCTION` itself. These instances are marked appropriately with the current state of the standard state machine (reflecting where in the business process things are), and also the id of the last-executed activity in the `INSTRUCTION` workflow (whose states will almost always be Active). `ACT` instances are used to record ad hoc acts, i.e. acts for which there is no corresponding `INSTRUCTION`.

Two types of archetypes are then written. Archetypes targetted to `INSTRUCTION_ACT` express the mapping of events in business processes to state transitions in the standard state machine, while archetypes of `INSTRUCTION` define workflow processes, such as particular kinds of medication orders, interventions, monitoring and recalls. The architecture is illustrated in FIGURE 1. This assignment of models to archetypes is not initially obvious, however it does make sense as will be shown later on.

3.2 Business Processes

Qualitatively speaking, there are two kinds of workflows with which we must be concerned. The first is the “business process”, i.e. workflows which occur in clinical life, but which often have no formal description, and are not executed by any single agent such as a care team or a workflow engine. Instead they execute by virtue of each step being enacted independently by its own agent, which may

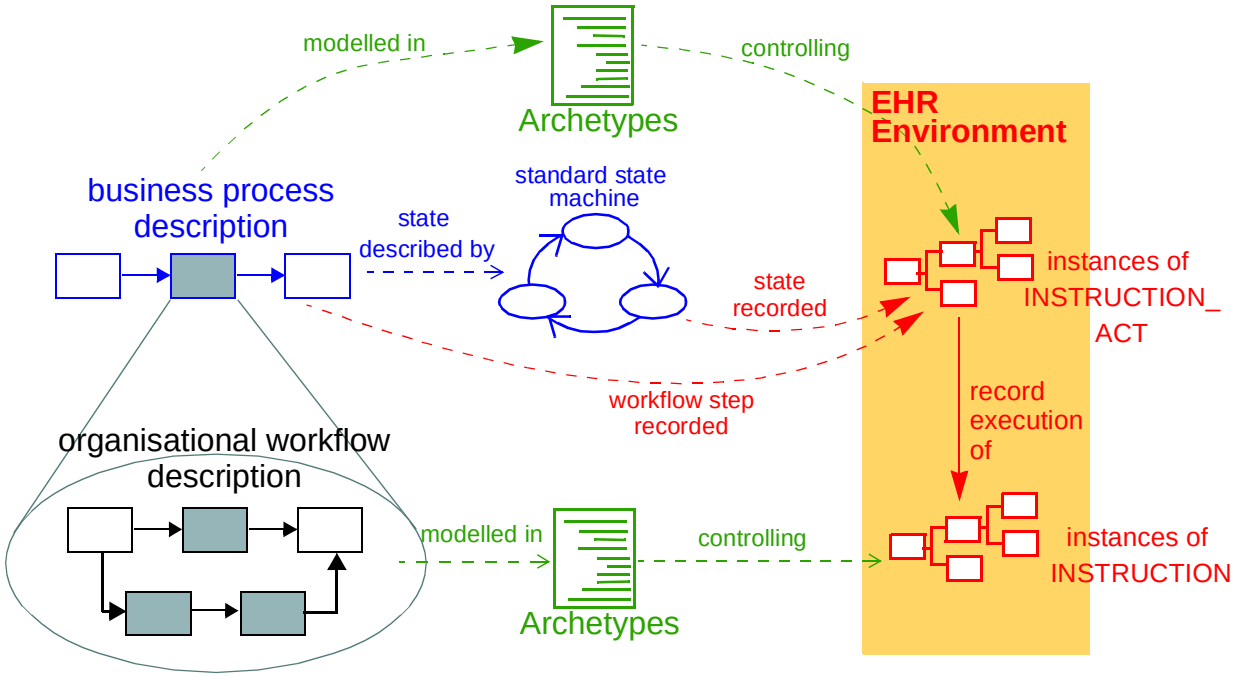


FIGURE 1 Design architecture

be a clinician or a software application. Data flows prompt the next activity to occur. Such workflows are extremely common, and occur every day in hospitals and in community care settings. An example of such a workflow is a standing order for a medication, sometimes called a repeat medication or a long term medication. Such a workflow involves at least the steps of 'prescription' and 'completed'. There is also some kind of repeated step - either in the form of 'dispense' by a pharmacist or 're-issue of the prescription' by the medical practitioner. At some point there is a 'review' of the medication in terms of the continuing indication. In the United Kingdom where the prescription is re-issued without formal review, the workflow can be modelled as shown in FIGURE 2.

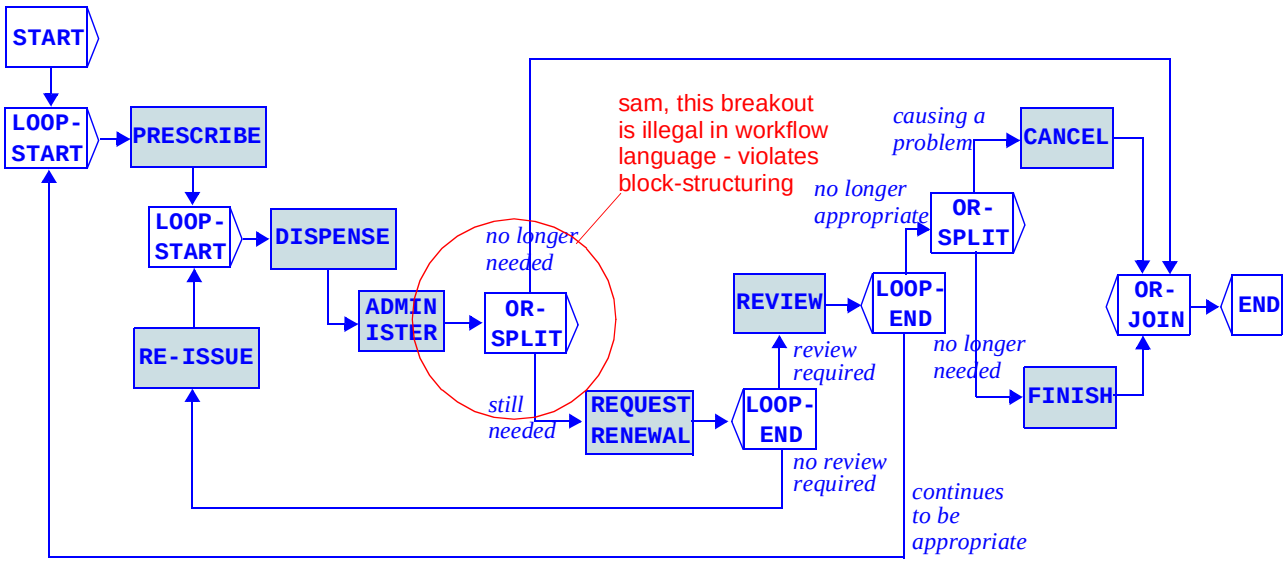


FIGURE 2 UK GP prescribing business process description

This figure is in the AgentWork language defined by Müller. Each step is shown as a grey box, while splits and joins of various paths are shown as white boxes, marked with the type of split or join. Each step corresponds to an action in the real world (which may itself be complex), while decision-making

(including by machines) causes OR-SPLITs, i.e. a separation of alternative paths. Alternative paths are later joined at OR-JOIN points. Repetitive actions are shown between LOOP-START and LOOP-END boxes. AND-SPLIT and AND-JOIN boxes (not shown in this example) delineate parallel sections. The entire figure is shown in blue, to differentiate it from explicit workflows, described below.

To reinforce the classification of this model as a "description of a business process", with no global executor, it is worth considering how it unfolds in most if not all environments where it occurs. Prescription occurs when a doctor creates a list of medications along with instructions for how to take them. This is given, usually as a written or printed form, to the patient, who takes the prescription to the pharmacy. The prescription can be considered as a message from doctor to pharmacist, with the patient acting as the message service. The Dispense step occurs when the pharmacist acts on the message and produces the requested drugs, and a copy of the administration instructions as a printed label. The patient takes the medications home and commences to use them as directed, thus enacting the Administer step. The detail of this latter step is an explicit workflow, since a) it is written down in the form of the patient instructions, and b) the patient (or it may be a nurse or guardian, depending on circumstances) is a conscious executor of the instructions. Other parts of the "outer" workflow, however, such as asking for repeats and having a review are not defined in any coherent document, and exist only as fragments of information in the various agents involved in the overall workflow.

The problem of business processes is that while they are often not defined, clinicians have a common mental model of them, and will always want to know at which point the workflow is: is this particular medication held up at the dispensing stage? Has the patient commenced the drug, or is it postponed for some reason? The value of describing business processes in the EHR environment is mainly to have a description of what goes on in clinical environments, and to connect them with a standard state machine which supports systematic querying. In the future, it may well be that such processes themselves are automated by means of workflow choreography services.

3.3 The Standard State Machine

When a business process is unfolding in a clinical context, EHR users want to know things such as the following:

- what is the current step in the process for the patient?
- for a given patient, what processes are planned, active, suspended, finished and so on.
- for the populations of patients, what is the state of a particular workflow, such as a recall?

The approach chosen here to support such functionality is to define a "standard state machine" whose state transitions can be mapped to the steps of specific business process, such as the one shown in FIGURE 2, enabling it to be used as a descriptive device for recording the state of any business process.

In the standard analysis of state machines [REF], a state transition results when an event occurs, and optionally when a "guard condition" is true at the time. The result of the transition may be an action. This standard transition is usually drawn as shown in FIGURE 3.

FIGURE 3 Standard state transition model

The state machine developed for use with EHR Instructions is illustrated in FIGURE 4. This state machine has been arrived at by long term experience with clinical workflows and act management systems..

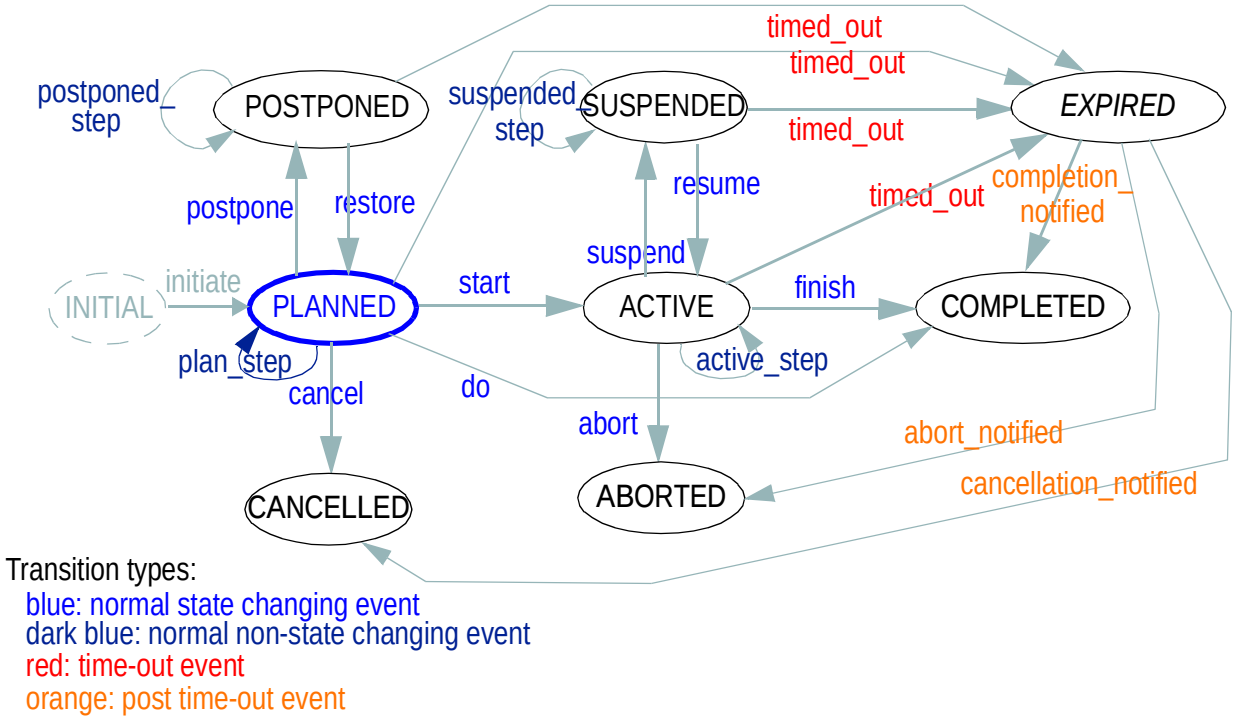


FIGURE 4 *open*EHR Standard State Machine

To Be Determined: AG: thinks we need a transition from initial to active, i.e. with no planning, e.g. patient self-administers non-prescribed aspirin

The states are as follows:

Initial: initial state, prior to planning activity

Planned: the action has been described, but not as yet taken place

Postponed: the action has not taken place and will not without specific conditions being met. Specifically, events and conditions which would normally 'activate' the instruction will be ignored, until it a restore event occurs.

Cancelled: the action was defined, but was cancelled before anything happened; it has not and will not take place.

Active: the action is taking place according to its definition. The entire course of medication or therapy corresponds to this state.

Active_suspended: the action has begun, but will not occur again until specific conditions are met which allow it to resume.

Aborted: the action began but was terminated before normal completion and will not occur again.

Completed: the action began and was completed normally.

Expired: the time during which the action could have been relevant has expired; the action may have completed, been cancelled, or never occurred.

States Cancelled, Aborted and Completed are all terminal states. The Expired state is a pseudo-terminal state, from which transitions are allowed to any of the true terminal states, due to information being received after the fact (such as a patient reporting that they did indeed finish a course of antibiotics). However it is likely to be common in the EHR that Instructions for many simple medications will finish in the Expired state and remain there.

The transitions are self-explanatory for the most part, however a few deserve comment. The *start* and *finish* events correspond to situations when the administration is not instantaneous, as is the situation with most medications. The *execute* event is equivalent to the *finish* event occurring immediately after the *start* event, corresponding to an instantaneous administration, completion of which puts the whole Instruction in the completed state. A single shot vaccination or patient taking a single tablet are typical examples. The states Planned, Postponed, Active, Suspended, each have a xxx_step transition which return the state machine to the same state. Workflow steps which cause no transition are mapped to these events and thus leave the Instruction in the same state. An example is a medication review, which will leave the medication in the Active state.

3.3.1 Mapping Business Processes to the State Machine

Each step of the execution of a business process can be considered to leave it in one of the states of the standard state machine. For example, the Dispense step (in an e-pharmacy environment where such as step was recorded in the EHR) might leave the medication order in the Active state - that is, the computer can assume that the action specified in the instruction is being carried out. However, transition to the Active state at the time of dispensing might not occur in a tightly controlled hospital environment, since the start of administration will be recorded and therefore will act as a signal to advance the state. Clearly this is the actual point (in the real world) when the instruction is truly active.

In general, the clinical workflow for any instruction will differ in the number and names of its steps. The key to connecting them to the state machine model is to map each step to the transition which *will be assumed to have occurred when that workflow step occurs*. For example, the following table shows the mappings for UK GP prescribing with e-pharmacy.

UK GP Workflow Step	State machine transition
START	initiate (initial -> planned)
PRESCRIBE	planned_step (planned -> planned)
DISPENSE	start (planned -> active)
ADMINISTER	active_step (active -> active)
REQUEST RENEWAL	active_step (active -> active)
RE-ISSUE	active_step (active -> active)
REVIEW	active_step (active -> active)
FINISH	finish (active -> completed)
CANCEL	abort (active -> aborted)

3.4 Organisational Workflows

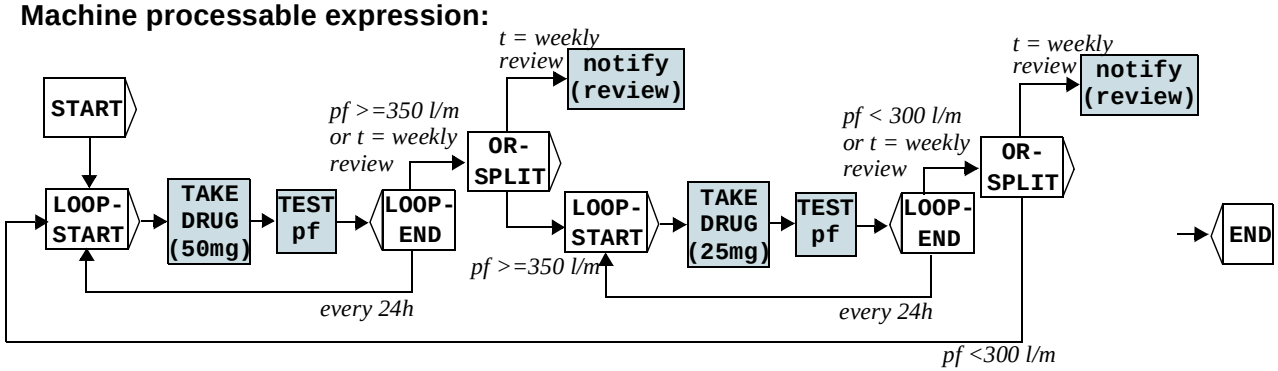
Some steps in a business process such as drug administrations, recalls and monitoring are defined explicitly, and are executed by one or more agents in concert according to a definition. At a minimum, they are defined in natural language, which is used by the executor (usually the patient or a nurse) to enact the instructions. In more sophisticated health computing environments, they are defined in a

machine-processable way, such as in a workflow language. FIGURE 5 illustrates the instructions for taking prednisolone, in both natural language form and as a formal workflow, using the AgentWork language.

Natural language expression:

"Take prednisolone 25mg 2 tablets in the morning until your peak flow has been above 350 l/min for 2 days, then take 1 tablet in the morning. If your peak flow falls below 300l/min, go back to the 50mg dose. I want to review you in one week."

Machine processable expression:



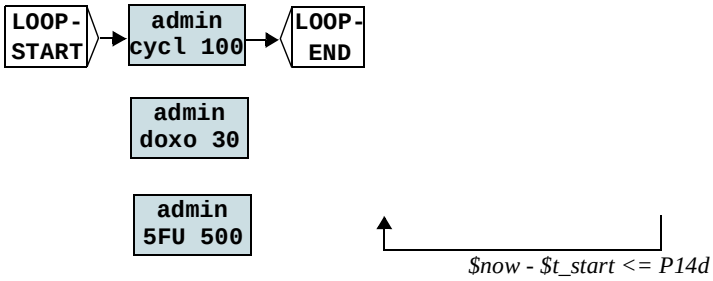
Two possibilities for finishing:
1.we assume that review ends the workflow, and that it might cause a new instance of it to be started
2.we model the review inside this workflow and show the paths back to the loops, or else stopping the medication

FIGURE 5 Prednisolone administration workflow

Organisational workflows not only need to be machine processable, but need to support unexpected events duing execution, such as equipment failures, a patient state which should abort or suspend the workflow and so on. Various research is available on this topic, including:

- The AgentWork language has been designed to model exactly such situations, and provides a disciplined way of moving into the Aborted, Suspended, Completed and Expired states from the Active state in the standard state machine.
- YAWL?

Another example workflow which exhibits some complexity is chemotherapy administration, in which patient state can cause an abort or suspension; similarly unavailability of drugs (some of which have to be mixed to patient-specific concentration) may cause a delay. A workflow for a chemotherapy regimen is shown below.



Regimen Name: CAF
Step 1: Cyclophosphamide 100 mg/m2/day po days 1-14
Step 2: Doxorubicin 30 mg/m2/day iv days 1 and 8
Step 3: 5FU 500 mg/m2/day iv on days 1 and 8 repeat q 4 weeks

FIGURE 6 chemotherapy administration regimen

4 A Model For Instruction

4.1 Technical Requirements

The discussion so far, along with other considerations, leads to the following technical requirements of the model.

To Be Continued: more explanation coming...

- distinct classes for workflow definition and workflow execution
- ability to record occurrences of actions which don't have a workflow definition
- narrative and computable form of workflow expressible
- freedom of hierarchical refinement depth in block structuring (only go down as far as computability requires)
- dynamic modification of workflow supported *a la* AgentWork
- suspension, resumption, abortion, completion and expiry of execution supported
- access to patient-related information from EHR; language of querying
- access to other external data like resource availability etc
- access to environment variables like current date & time
- support for timers / delaying / waiting mechanisms
- support for synchronising communication between cooperating workflows
- handling of patient reviews
- data flows between nodes representable
- support for recording duration of execution of nodes, in order to support dynamic adaptation

4.2 Model Overview

FIGURE 7 shows the `rm.ehr.composition.entry` package. This proposal adds the classes `INSTRUCTION`, `ACT`, and `INSTRUCTION_ACT` as subtypes of `ENTRY`.

`INSTRUCTION` is shown as a direct subtype, thus inheriting existing `ENTRY` properties whose meanings are interpreted as specifying for the future. For example, the *protocol* attribute inherited into `INSTRUCTION` would express the protocol (or method) to be used in execution of the Instruction. Similarly, the *participations* attribute indicates participations (or most likely participation types) which would be required during execution of the Instruction.

The type `ACT` models *ad hoc* actions which do not correspond to previously defined `INSTRUCTIONs`, such as unplanned patient-initiated actions or actions whose prescription was not recorded in the current EHR environment. `INSTRUCTION_ACT` models executions of parts of the workflow defined in a corresponding `INSTRUCTION` object. Note that the instance data for a particular execution could easily be different from the intended action description.

The detailed model of `INSTRUCTION` commits minimally to the structural aspect of formalisms like AgentWork, i.e. the control node types, and the idea that an action node has inputs and outputs. However, as in AgentWork, expressions representing triggers, actions, and notifications are expressed in syntax strings, rather than in any structural manner; these will be archetyped.

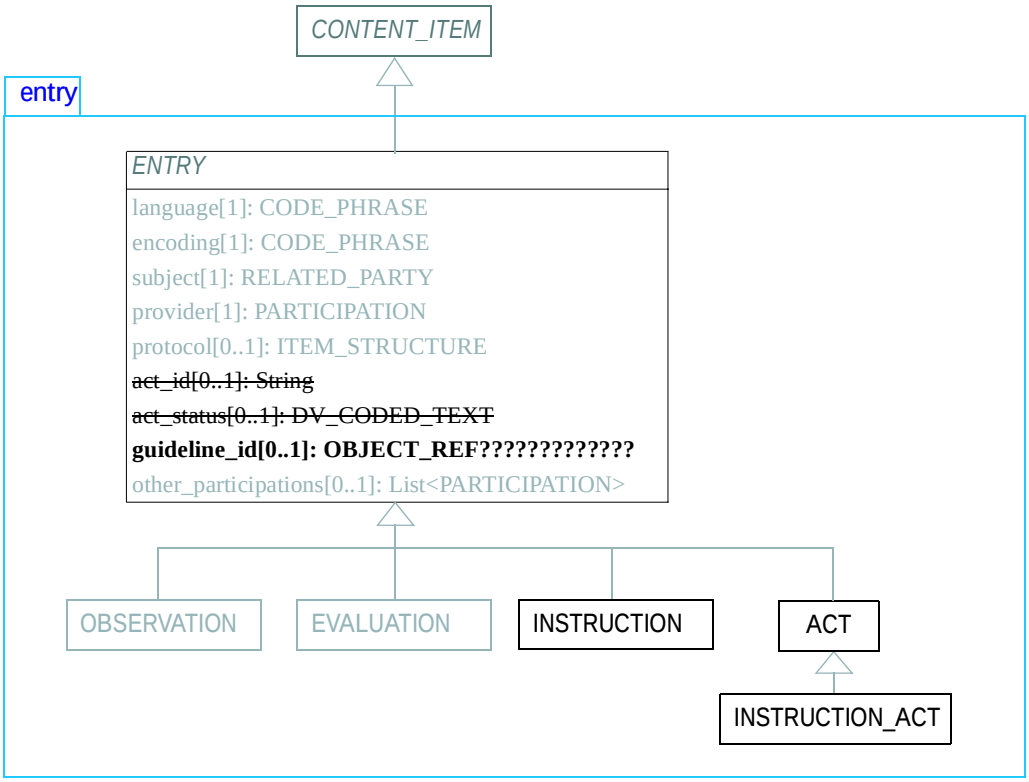


FIGURE 7 rm.ehr.composition.content.entry Package overview

4.3 Instruction Classes

The classes making up the abstraction of `INSTRUCTION` are shown in FIGURE 8. These classes implement the semantics of the XXX language in a 'soft' way, i.e. without directly modelling all of the node types. This is done to a) allow for new kinds of node in the future and b) because it is assumed that in realistic systems, instances of these classes will be built be a workflow editor implementing both visual syntax and underlying semantics of XXX, thus guaranteeing correct structures. It may be reasonable to change this to a higher level of hard-modelling. Note that trigger and notification fields are expressed in a syntax separate from XXX, which avoids hard-modelling of logical expressions used in workflows.

The classes `CONTROL_NODE` and `TRIGGER` allow control nodes to be represented. The `CONTROL_NODE`.*type* attribute is one of the types `START`, `END`, `OR-SPLIT`, `OR-JOIN`, etc. One `TRIGGER` instance is used per outgoing branch of a control node, indicating the event-condition (i.e. trigger) which results in following that branch. Triggers are expressed in a formal language designed for representing archetype-based data references and symbolic EHR queries. The *action* attribute contains a symbolic action, directed at the workflow processor, which might be to write something into the EHR, or it might be to abort or delay the workflow. The *notification* attribute is a similar symbolic expression, where a notification to an external agent is needed, such as to the patient, doctor, or some other receiver. It would be written in a symbolic syntax which can be interpreted by a notification server.

To Be Determined: are action and notification both needed? Have to work through some examples

See below for examples of expressions for trigger, action and notification.

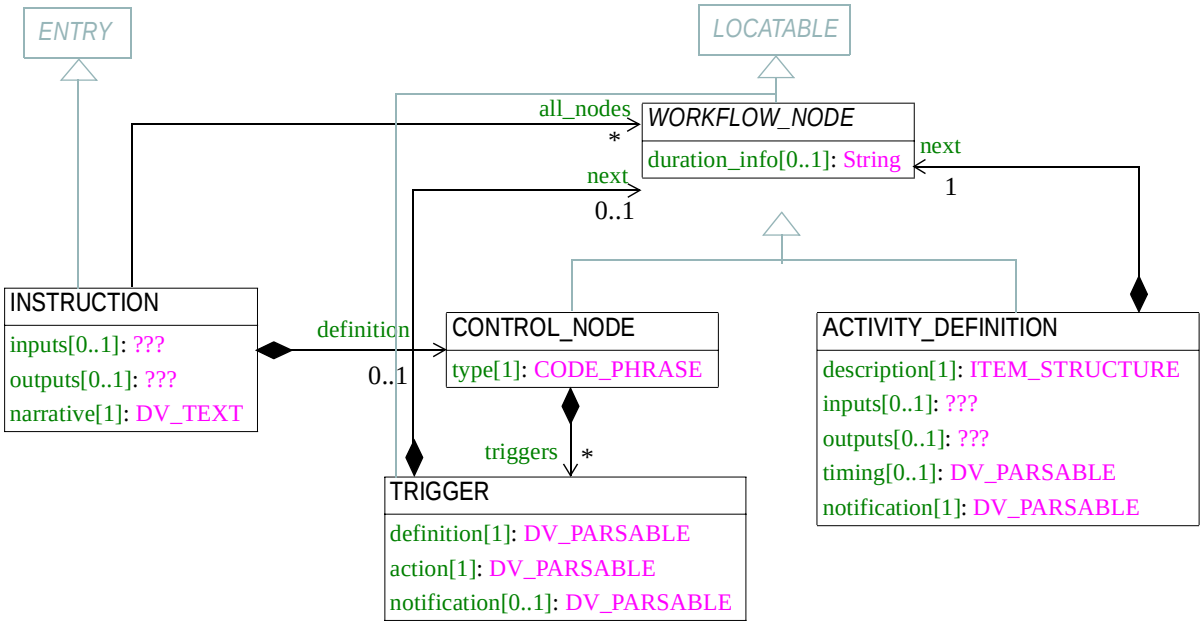


FIGURE 8 Instruction class

4.4 Execution Classes

The detailed model for the two classes modelling act execution is shown in FIGURE 9. An instance of the class `ACT` is essentially an `ENTRY`, with the *description* and *time* representing what action occurred and when. The optional *notification* attribute enables the recording of any notification that *was* made as part of this recording, while the optional *execution_status* can be used to record what state the activity is in - most likely a state similar to those in the standard state machine.

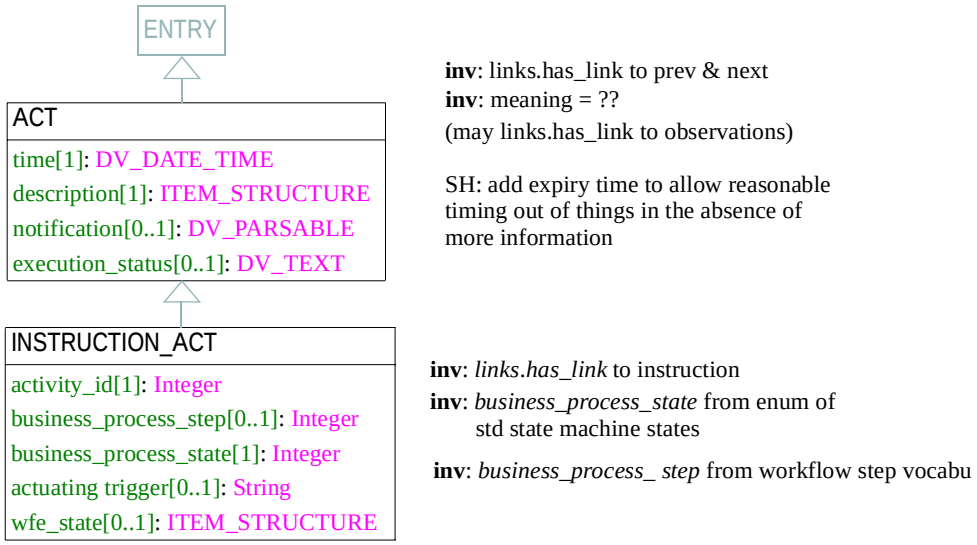


FIGURE 9 Execution classes

The `INSTRUCTION_ACT` class adds the attribute *activity_id* to identify the Instruction Activity to which this act execution corresponds. The attributes *business_process_step* and

business_process_state respectively identify the current step (e.g. “dispense”, “review”) and the current state (e.g. “active”) in the business process. The *actuating_trigger* attribute records the id value of the trigger which was fired from the Activity node to which this Instruction_act corresponds. The *wfe_state* attribute allows the recording of any further state required by the workflow engine; this might well be a single instance of a `DV_ENCAPSULATED` within an `ITEM_SINGLE`, or it may be transparent data expressed as e.g. an `ITEM_TREE`.

4.5 Triggers, Actions and Notifications

Triggers

Trigger expressions are event-condition combinations which cause particular branches at OR-SPLIT and LOOP-END nodes to be taken.

a trigger expression could be something like the following, in ActiveTFL-style syntax:

- WHEN new haematological finding of patient P
  WITH leukocyte count < 1000 #/mm3
  THEN abort (W, P)

In the above, the WHEN-WITH part is a clumsy way of getting at a datum in the EHR. The THEN part is a symbolic instruction to the workflow processor to abort workflow W, for patient P.

In our thinking, the above is something like:

- trigger = ‘$ehr.fetch(“[haematological_finding]/...[leukocyte_count] < 1000”)’
  action = ‘abort(W,P)’

We have not yet worked out the syntax for making EHR queries from within archetypes, but we can suggest the following:

- it probably won’t use paths directly
- instead it will use ‘interface’ symbols from archetypes. These are the symbolic names for paths, which we will define in archetypes (ADL 1.3), in the new ‘declarations’ section. For example, we might allow:

```
let $leukocyte_count = /[haematology
   finding]/data[history]/events[any]/data[structure]/
   items[leukocyte count]/value/value

(in the system, it is really /[at0001]/data[at0099]/...etc)
```

  then, accessing the item in the record can be done with something like:

```
//openEHR-EHR-OBSERVATION.haematology::$leukocyte_count
```

  This syntax will need to be sorted out properly - must work in ADL, and be made Xpath-compliant or mappable - but I think the idea is correct.

In general Müller’s approach in AgentWork wants to use something exactly like archetypes to express constraint patterns on data, as part of condition expressions. However, we have more or less solved this problem in a more general way than he has (actually, his thinking is based on F-logic, which is very general, he just doesn’t have a nesting syntax like ADL to implement it...).

Actions

Actions are instructions to the computing environment, and could include:

- writing something to the EHR
- setting a timer on an EHR, e.g. for a recall
- doing anything to modify the workflow, including abort, suspend, complete, drop_node(), add_node() etc

TBD_1: syntax to be determined, but probably nearly the same as in AgentWork

Notifications

What we end up calling a 'notification' will probably change somewhat, but typical examples should include:

- notification to the patient of recall, e.g. expressed as:

  ```
  notify($patient, "your next PAP test due on $date")
  ```

  where msg is substituted at run time with the right date.
- notify a clinician that a workflow has completed, that some new information is in the EHR etc

4.6 NOTE - on ARCHETYPES

SH: need ACTIVITY archetypes to work as archetypes for ACTs - since want it to work the same way for instructed and non-instructed acts. Act have to record the difference with respect to the relevant Instruction; if no instruction, we need to say everything in the Act; need to share the same archeytpe, e.g. medication_order. Act has to refer back to Activity_def inside Instruction, not just an isolated Activity_definition.

Philosophy must be that all Acts fit within some workflow corresponding to a clinical process; smallest unit of that is Instruction.

5 Distributed Workflow

Notification of act e.g. specialist seeing patient - should cause notification back to author of instruction (usually the GP, or it might be someone in hospital).

A typical multi-provider communication situation:

- Plan referral
- Create referral communication extract (might need consent before it can be sent)
- Referral sent (might be in hand of patient in hand-held situation)
- Referral received
- Appointment arranged
- Patient seen
- Secondary referral
- Secondary referral complete
- Referral maintained
- Referral reviewed
- Communication referred to referrer
- Communication referrer to referred
- Referral closed

6 Examples

This section is designed to validate the models described earlier. A number of common, but often complex, clinical instructions are modelled in archetypes, and the resulting data is shown as instance diagrams of the UML models above.

6.1 GP prescription, no workflow processing

Instruction Archetype

Instruction Instance

Execution Archetype

Execution Instances

6.2 GP prescription including variable dose and conditional triggers

Instruction Archetype

Instruction Instance

Execution Archetype

Execution Instances

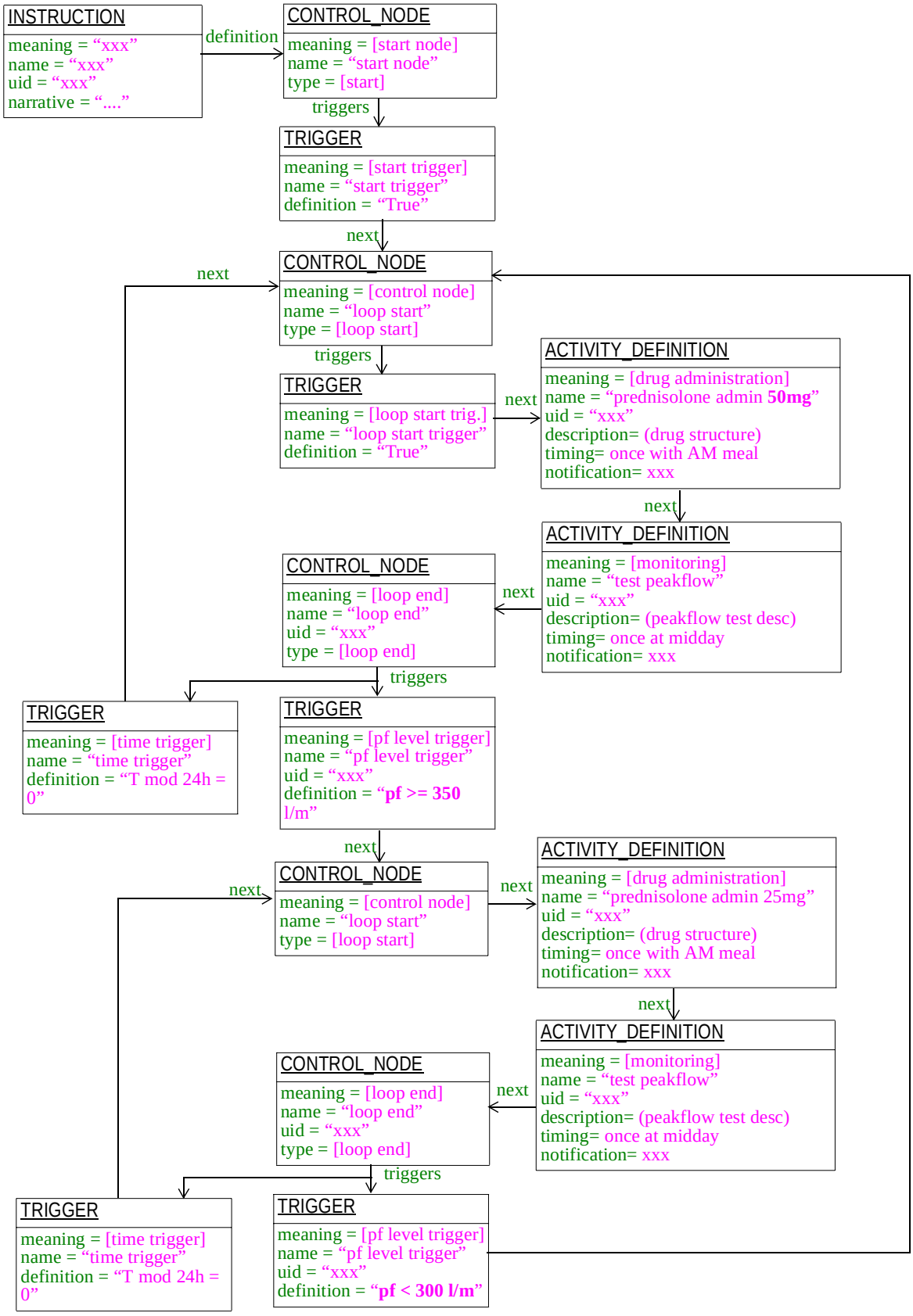


FIGURE 10 Instruction Instance for GP prescription

6.3 Hospital drug admin with e-pharmacy and bedside EHR entry

Instruction Archetype

Instruction Instance

Execution Archetype

Execution Instances

6.4 Multi-drug regimen with workflow-altering triggers

Instruction Archetype

Instruction Instance

Execution Archetype

Execution Instances

6.5 PAP recall

Instruction Archetype

Instruction Instance

Execution Archetype

Execution Instances

6.6 Diagnosis workflow

Instruction Archetype

Instruction Instance

Execution Archetype

Execution Instances

6.7 Pharmacy Order

Instruction Archetype

Instruction Instance

Execution Archetype

Execution Instances

6.8 Surgery Order and Execution

Instruction Archetype

Instruction Instance

Execution Archetype

Execution Instances

7 References

1 Kifer M. *Logical Foundations of Object-Oriented and Frame-Based Languages*. 1995. Available at ftp://ftp.cs.sunysb.edu/pub/TechReports/kifer/flogic.pdf.

2 Müller R. *Event-oriented Dynamic Adaptation of Workflows: Model, Architecture, Implementation.* Available at http://dol.uni-leipzig.de/pub/2002-55. Accessed 25 Mar 2005.