The MADE Reference Information Model for Interoperable Pervasive Telemedicine Systems

N. L. S. Fung[1]; V. M. Jones[1]; H. J. Hermens[1,2]

[1] Biomedical Signals and Systems Group, University of Twente,
7500 AE Enschede, The Netherlands

[2] Roessingh Research and Development, 7500 AH Enschede, The Netherlands

Correspondence to:
N. L. S. Fung
Biomedical Signals and Systems Group
University of Twente
P. O. Box 217
7500 AE Enschede
The Netherlands
E-mail: l.s.n.fung@utwente.nl
Phone: +31 (0)53 489 5625

Summary

Objectives: The main objective is to develop and validate a reference information model (RIM) to support semantic interoperability of pervasive telemedicine systems. The RIM is one component within a larger, computer-interpretable "MADE language" developed by the authors in the context of the MobiGuide project. To validate our RIM, we applied it to a clinical guideline for patients with gestational diabetes mellitus (GDM).

Methods: The RIM is derived from a generic data flow model of disease management which comprises a network of four types of concurrent processes: Monitoring (M), Analysis (A), Decision (D) and Effectuation (E). This resulting MADE RIM, which was specified using the formal Vienna Development Method (VDM), includes six main, high-level data types representing measurements, observations, abstractions, action plans, action instructions and control instructions.

Results: The authors applied the MADE RIM to the complete GDM guideline and derived from it a domain information model (DIM) comprising 61 archetypes, specifically 1 measurement, 8 observation, 10 abstraction, 18 action plan, 3 action instruction and 21 control instruction archetypes. It was observed that there are six generic patterns for transforming different guideline elements into MADE archetypes, although a direct mapping does not exist in some cases. Most notable examples are notifications to the patient and/or clinician as well as decision conditions which pertain to specific stages in the therapy.

Conclusions: The results provide evidence that the MADE RIM is suitable for modelling clinical data in the design of pervasive telemedicine systems. Together with the other components of the MADE language, the MADE RIM supports development of pervasive telemedicine systems that are interoperable and independent of particular clinical applications.

Key Words

Telemedicine, pervasive healthcare, reference information model, semantic interoperability

1 Introduction

To realise their full potential, telemedicine systems should be interoperable, particularly those for pervasive healthcare. This is made necessary due not only to their diversity and highly distributed nature, but also to the dynamic and competitive nature of the wearable and mobile technology market. Indeed, interoperability is also a recognised problem for traditional clinical information systems, and as a result, various standards (e.g. SNOMED CT and DICOM) have been proposed to address different aspects of interoperability. Of particular interest are those which provide a standardised, conceptual model for automated interpretation of clinical data, the main ones of which include openEHR and HL7 v3.

In general, these standards distinguish between a reference information model (RIM), which captures the generic data types required in a clinical statement, and the domain information model (DIM), which is a specialisation of the RIM and specifies the semantic constraints of a particular application, e.g. in the form of archetypes. In HL7 v3 the notion of logical domain analysis models (DAMs) was also introduced for generating DIMs from a generic, domain-independent RIM. One such DAM, for Virtual Medical Records (vMRs), was developed to support clinical decision support and is comparable to the openEHR RIM.

As part of the MobiGuide project (http://www.mobiguide-project.eu/), which aimed to provide pervasive and personalised clinical decision support to patients, the HL7 vMR model (first release) was adopted to support interoperability between the back-end servers of hospitals and the wearable healthcare technologies of patients. It was observed however [1, 2] that the HL7 vMR model was unable to capture all the necessary data requirements, and the solution of extending the vMR data types and relaxing their definitions was proposed and demonstrated. However, to best support the exchange of data in the special case of wearable and mobile healthcare technologies, the authors believe a more appropriate alternative might be to develop a new RIM based specifically on the requirements of pervasive healthcare systems.

In particular, unlike clinical information systems for which openEHR and HL7 are intended, pervasive healthcare systems are required to:

- Provide clinical support to patients in a free-living setting.
- Handle data streams (e.g. from sensors).
- Account for the possibility of delayed and/or out-of-order arrival of data.
- Process low-level sensor data which may not have an immediate clinical interpretation.

Therefore, as part of our research arising from the MobiGuide project, we developed a new, alternative RIM for pervasive healthcare systems which forms part of our larger, computer-interpretable MADE language for modelling and specifying their required clinical functionality. To develop this language (including the RIM), we adopt a combination of model-driven and formal methods approaches, specifically by:

1) Basing the language on a conceptual framework that we previously developed to model the disease management process.

2) Specifying the language formally using the Vienna Development Method (VDM) [3, 4].

The conceptual framework allows us to abstract away from application-specific details, whilst the formal VDM specification provides opportunity for unambiguous mathematical description of the language that enables rigorous analysis, validation and verification using the Overture Tool (http://www.overturetool.org).

In Section 3, we present an overview of the underlying conceptual framework and a detailed description of the resulting RIM. Although the RIM was in practice specified using VDM (with the support of the Overture Tool), the MADE RIM is presented here using the equivalent mathematical notation in Table 1 as well as the standard notation in predicate calculus and set theory; details of the original VDM specification can be found in Appendix A in the supplementary material. Likewise, the same notation is used in place of VDM in Section 4, where we summarise the results of validating our RIM by applying it to a clinical guideline [5] for patients with gestational diabetes mellitus (GDM). The resulting VDM specification of the complete GDM DIM is detailed in Appendix B in the supplementary material. Finally, the discussion and conclusions are found in Sections 5 and 6.

Table 1 Mathematical notation and its corresponding VDM syntax for specifying the MADE RIM and the GDM DIM, excluding the standard notation in predicate calculus and set theory.

Notation	VDM Syntax	Description
TypeName	`TypeName`	The name of a type, i.e. a collection of elements, in the RIM and DIM.
$\mathrm{Type1} = \mathrm{Type2}$	`Type1 = Type2`	Type 1 is the same (i.e. share the same elements) as Type 2.
$T = \mathit{Element}_1 \mid \mathit{Element}_2$	T = **<Element1>** | **<Element2>**	Type T is an enumerated type, with only Element 1 and Element 2 belonging to that type.
$T = T_1 \mid T_2$	T = T1 | T2	Type T is the union of (i.e. the same as) T_1 and T_2.
$T = T_1 \times T_2 \times \ldots \times T_N$	`T = T1 * T2 * … * TN`	Each element of T comprises N components; the first is an element of type T_1, the second T_2, etc.
$T = \wp(T_1)$	T = **set of** T1	Each element of T is a set containing elements of T_1.
$\pi_n(t)$	`t.#n`	The nth component of element t (which itself is also an element of a certain type).
$T_1 <: T_2$ such that $\forall\, t : T_1.\ e$	Type1 = Type2 **inv** t == e	T_1 is a sub-type of T_2, i.e. the elements belonging to T_1 are a subset of those belonging to T_2 subject to Boolean expression e.

2 Objectives

The main objective is to develop a RIM to support the exchange of clinical data in pervasive telemedicine systems. The RIM is validated by applying it to a GDM guideline, deriving from it the appropriate DIM.

3 Methods

3.1 Model of Pervasive Telemedicine Systems

Together with colleagues from MobiGuide, we previously developed and demonstrated a conceptual framework, the MADE framework [6, 7], for modelling the disease management process with the aim of supporting distribution of clinical function in telemedicine systems. Conceptually, the MADE framework does not prescribe which clinical functions must be performed automatically or manually, but it is nonetheless designed to facilitate application on telemedicine systems, which are data-driven and distributed by nature. In particular, the MADE framework captures the data flow in disease management like those of [8, 9], with the data items being operated on by a network of four types of concurrent processes, namely:

- Monitoring (M), the process of making observations about the patient.
- Analysis (A), the process of making assessments about the state of the patient.
- Decision (D), the process of deciding on the appropriate therapeutic plan.
- Effectuation (E), the process of performing the decided plan.

The conceptual model was concretised as shown in Figure 1 to identify the data types relevant for pervasive telemedicine systems. As is the case for the conceptual model, no assumption is made in Figure 1 about any particular data capture mechanism, including whether such data items would be automatically generated or manually inserted by the user. Thus although many sensors embody a Monitoring process in practice, the term sensor is used in this context to simply refer to a device that converts a physical stimuli into an appropriate signal for proceeding by a Monitoring process.

Similar to openEHR and HL7 vMR, our model also captures the notions of observations about the patient and instructions to be executed. However, because the target context of use is a free-living setting, concepts pertaining specifically to patient-clinician encounters, e.g. appointments and supply orders, are outside the scope of the MADE model. Furthermore, two types of instructions are distinguished in accordance with the intrinsic differences between internal "reasoning" and external "actuating" processes. In fact, assessments and therapeutic decision-making are also considered to be different in nature, reflecting the different types of knowledge required, namely declarative and procedural knowledge respectively [10].

3.2 MADE Data Types

As implied in Figure 1, there are six main data types in the MADE RIM, representing measurements, observations, abstractions, action plans, action instructions and control instructions. These high-level data types are in turn constructed from a combination of 10 primitive data types, namely Id, Boolean, Nominal, Enumerated, Count, Proportion, Dimensioned, DateTime, Duration and Schedule, all of which adopt conventional semantics. For example, the data type Dimensioned represents quantities that are characterised by a real-number magnitude (e.g. 4.5) and a unit (e.g. mmol/L).

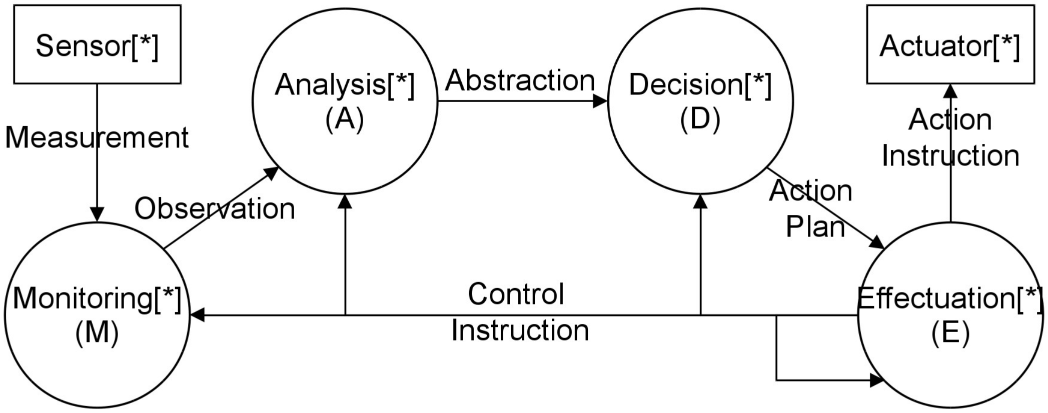

Figure 1 Data flow diagram of the MADE model for pervasive telemedicine systems ([*] denotes multiplicity).

3.2.1 Measurements

One of the six high-level MADE data types represents measurements, which are data items that result from the detection of physical stimuli in the environment and do not have an immediate clinical interpretation. Examples include accelerometer, magnetometer and goniometer data from which measures of the patient's motor function can be derived [11]. In the MADE model, all such measurements are quantified using a dimensioned number and are associated with a valid date-time, which indicates when the measurement holds, i.e. the instant of measurement. Furthermore, as with other MADE data types, all measurements include an identifier to distinguish between different sensed physical stimuli as well as a transaction date-time which, as opposed to a valid date-time, indicates when the data item is known to the system. Thus if a patient's blood glucose level is measured at 10:00 but entered into the system two hours later at 12:00, then the measurement's valid and transaction time would be 10:00 and 12:00 respectively.

Therefore, using the notation presented in Table 1, the measurement data type can be specified as follows:

$$
\begin{aligned}
&\mathrm{Measurement} = \mathrm{MeasurementType} \times \mathrm{TransactionDateTime} \times \mathrm{ValidDateTime} \times \\
&\qquad\qquad\quad \mathrm{Dimensioned}, \text{ where} \\
&\mathrm{MeasurementType} = \mathrm{Id}, \\
&\mathrm{TransactionDateTime} = \mathrm{DateTime} \text{ and} \\
&\mathrm{ValidDateTime} = \mathrm{DateTime}
\end{aligned}
$$

3.2.2 Observations

The result of processing measurements by a Monitoring process is an observation, which is a low-level fact about the environment that has a direct clinical interpretation but, as shown in Figure 1, still requires further analysis for clinical decisions to be made. In the MADE model, two types of observations are distinguished:

$$\mathrm{Observation} = \mathrm{ObservedProperty} \mid \mathrm{ObservedEvent}$$

Observed properties are observations about the physical objects in the environment, e.g. blood glucose level. Like measurements, observed properties are only valid at a specific time point, thus they also comprise a single valid date-time stamp. However, they need not be quantified using dimensioned values. Blood type, for example, may be categorised into A, B, AB or O, whilst urinary ketone levels may be ranked into --, -, +/-, + or ++. Thus the value of different observed properties may be of differing types as specified below:

$$
\begin{aligned}
&\mathrm{ObservedProperty} = \mathrm{PropertyType} \times \mathrm{TransactionDateTime} \times \mathrm{ValidDateTime} \times \\
&\qquad\qquad\qquad\quad \mathrm{PropertyValue}, \text{ where} \\
&\mathrm{PropertyType} = \mathrm{Id} \text{ and}
\end{aligned}
$$

$$\text{PropertyValue} = \text{Boolean} \mid \text{Nominal} \mid \text{Enumerated} \mid \text{Ordinal} \mid \text{Count} \mid \text{Proportion} \mid \text{Dimensioned}$$

The second type of observation (viz. observed event) relates to events which occur in the environment, such as the patient following exercise therapy from 7:00 to 8:00. Unlike observed properties, these observations can only be characterised by a Boolean value indicating whether the corresponding event occurred or not. Thus the properties of events, e.g. exercise intensity, are captured in the MADE model by the observed properties of the objects involved. Furthermore, as exemplified by the exercise example, observations relating to events are associated with a pair of date-time stamps marking the start and end time of an event; in the simple case of instantaneous events, these two date-time stamps are equal:

$$\text{ObservedEvent} = \text{EventType} \times \text{TransactionDateTime} \times \text{ValidDateTimeRange} \times \text{Boolean}, \text{ where}$$

$$\text{EventType} = \text{Id and}$$

$$\text{ValidDateTimeRange} = \text{DateTime} \times \text{DateTime such that}$$

$$\forall\, v : \text{ValidDateTimeRange}.\ \pi_1(v) \leq \pi_2(v)$$

3.2.3 Abstractions

Observations are transformed by Analysis processes into higher-level, more meaningful abstractions by means of a combination of the following two procedures:

1) Mapping individual values of observed properties onto more abstract values, such as categorising a blood glucose observation into hypoglycaemia, normoglycaemia or hyperglycaemia.

2) Combining multiple observations together such that the final result is either a summary statistic or an abstract value that is valid over an extended time period, such as detecting multiple hyperglycaemic episodes in the past week.

Therefore, abstractions are modelled to also contain a valid date time range and an abstraction value which can be of any numerical or categorical data type:

$$\mathrm{Abstraction} = \mathrm{AbstractionType} \times \mathrm{TransactionDateTime} \times \mathrm{ValidDateTimeRange} \times \mathrm{AbstractionValue}, \text{ where}$$

$$\mathrm{AbstractionType} = \mathrm{Id} \text{ and}$$

$$\mathrm{AbstractionValue} = \mathrm{PropertyValue}$$

3.2.4 Action Instructions

To specify the action plans of a Decision process, it is useful to first specify the outputs of an Effectuation process (viz. action and control instructions) since they form the main constituents of an action plan. In the MADE model, two different types of actions (and therefore action instructions) are distinguished:

$$\mathrm{ActionInstruction} = \mathrm{HomogeneousAction} \mid \mathrm{CulminatingAction}$$

Firstly, actions are referred to as homogeneous if they do not have a clear end-point and can be divided into sub-parts that retain the same overall properties, for example a continuous activity such as running on a treadmill. Therefore, the corresponding instructions are characterised by a starting date-time stamp, a rate at which the action should be performed (e.g. running at 7 kph) as well as a duration (e.g. for 20 minutes):

$$\mathrm{HomogeneousAction} = \mathrm{ActionType} \times \mathrm{TransactionDateTime} \times \mathrm{StartDateTime} \times \mathrm{Rate} \times \mathrm{Duration}, \text{ where}$$

$$\mathrm{ActionType} = \mathrm{Id},$$

$$\mathrm{StartDateTime} = \mathrm{DateTime} \text{ and}$$

$$\mathrm{Rate} = \mathrm{Dimensioned}$$

Unlike homogeneous actions, culminating actions have a clear end-point which must be achieved, such as administering 30 units of basal insulin in the evening. Although instructions for such actions also have a target start date-time, they exhibit a well-defined target goal state instead of a duration and rate:

$\text{CulminatingAction} = \text{ActionType} \times \text{TransactionDateTime} \times \text{StartDateTime} \times \text{GoalState},$
where
$\text{GoalState} = \text{PropertyValue}$

3.2.5 Control Instructions

Control instructions are used to determine when a process should execute. In the MADE model, all processes are assumed to be pre-existent and thus can only be re-scheduled and/or paused or resumed by a control instruction. Furthermore, to allow correct temporal ordering of multiple control instructions irrespective of any potential transaction delays, these control instructions are also characterised by a valid date-time stamp to indicate when they should be effected:

$\text{ControlInstruction} = \text{TargetProcess} \times \text{TransactionDateTime} \times \text{ValidDateTime} \times$
$(\text{Schedule} \mid \mathit{NULL}) \times (\text{Status} \mid \mathit{NULL})$, where
$\text{TargetProcess} = \text{Id},$
$\text{Schedule} = (\text{RepeatPattern} \times \text{RepeatInterval}) \mid \mathit{Alway} \ \mid \mathit{Never},$
$\text{RepeatPattern} = \wp(\text{DateTime}),$
$\text{RepeatInterval} = \text{Duration} \mid \mathit{Never}$ and
$\text{Status} = \mathit{Paused} \mid \mathit{Running}$ such that
$\forall\, c : \text{ControlInstruction}\ \neg(\pi_4(c) = \mathit{NULL} \wedge \pi_5(c) = \mathit{NULL})$

As specified above, control instructions may contain a special *NULL* element in place of a new schedule or a new status for the target process, which indicates that the target process should continue to operate under its existing schedule or status as appropriate. However, since control instructions would serve no function if both their schedule and status are absent (i.e. if both are *NULL*), the extra condition on control instructions ensures that each must contain a new schedule or a new status for the target process or both.

Furthermore, in the MADE model, all processes run continually unless instructed otherwise by a control instruction. Thus a schedule in the MADE RIM comprises a set of starting date-time stamps (i.e. the repeat pattern) as well as a duration (i.e. the repeat interval) which indicates when to repeat (if ever). Sentinel values *Always* and *Never* indicate that the target process should, respectively, run at each time stamp and never run.

3.2.6 Action Plans

As noted by [12], disease management processes may involve complex workflows comprising sequential, parallel, iterative and/or cyclical activities. However, as implied in Section 3.1, we model pervasive telemedicine systems to operate on cyclic parallel plans, which, through the use of the appropriate schedules, can also be transformed into sequential and non-cyclical plans. Therefore, without loss of functionality, an action plan is modelled as containing a set of scheduled control and action instructions, each specifying when a different concurrent MADE process or physical action should be executed:

$$\begin{aligned}\mathrm{ActionPlan} = {} & \mathrm{PlanType} \times \mathrm{TransactionDateTime} \times \mathrm{ValidDateTime} \times \\ & \wp(\mathrm{ScheduledControl} \mid \mathrm{ScheduledHomogeneousAction} \mid \\ & \mathrm{ScheduledCulminatingAction}) \text{ where}\end{aligned}$$

$\mathrm{PlanType} = \mathrm{Id}$,

$\mathrm{ScheduledControl} = \mathrm{TargetProcess} \times (\mathrm{Schedule} \mid \mathrm{NULL}) \times (\mathrm{Status} \mid \mathrm{NULL})$,

$\mathrm{ScheduledHomogeneousAction} = \mathrm{ActionType} \times \mathrm{Schedule} \times \mathrm{Rate} \times \mathrm{Duration}$ and

$\mathrm{ScheduledCulminatingAction} = \mathrm{ActionType} \times \mathrm{Schedule} \times \mathrm{GoalState}$ such that

$\forall\, p : \mathrm{ActionPlan}\ (\forall\, i, j \in \pi_4(p)\ (\pi_1(i) = \pi_1(j) \Rightarrow i = j))$ and

$\forall\, k : \mathrm{ScheduledControl}\ \neg(\pi_2(k) = \mathit{NULL} \wedge \pi_3(k) = \mathit{NULL})$

The first condition on an action plan ensures that each of its scheduled control and action instruction refers to a different target process or action type. In other words, there cannot be more than one instance of the same type of action or control instruction in the action plan, which prevents the occurrence of internal inconsistencies such as scheduling the same process simultaneously to always and never execute. Furthermore, like that for control instructions, the second condition ensures that scheduled control instructions must contain a new schedule or a new status or both.

4 Results

The MADE RIM presented in Section 3 was formalised using VDM with the Overture Tool, the results of which can be found in Appendix A in the supplementary material. Furthermore, the MADE RIM was applied to a clinical guideline for GDM (self) management [5] which was developed as part of the MobiGuide project and is represented as a collection of semi-formal workflows comprising 51 different action points and 33 decision points. Based on this semi-formal representation, we derived a MADE GDM DIM containing 61 archetypes in total, specifically 1 measurement, 8 observation, 10 abstraction, 18 action plan, 3 action instruction and 21 control instruction archetypes. In this section, we present a summary of our experience of deriving the MADE archetypes, using as an illustration a simplified but exemplary case from the guideline relating to the detection of ketonuria, i.e. the presence of urinary ketones (Figure 2). The VDM specification of the GDM DIM is detailed in Appendix B.

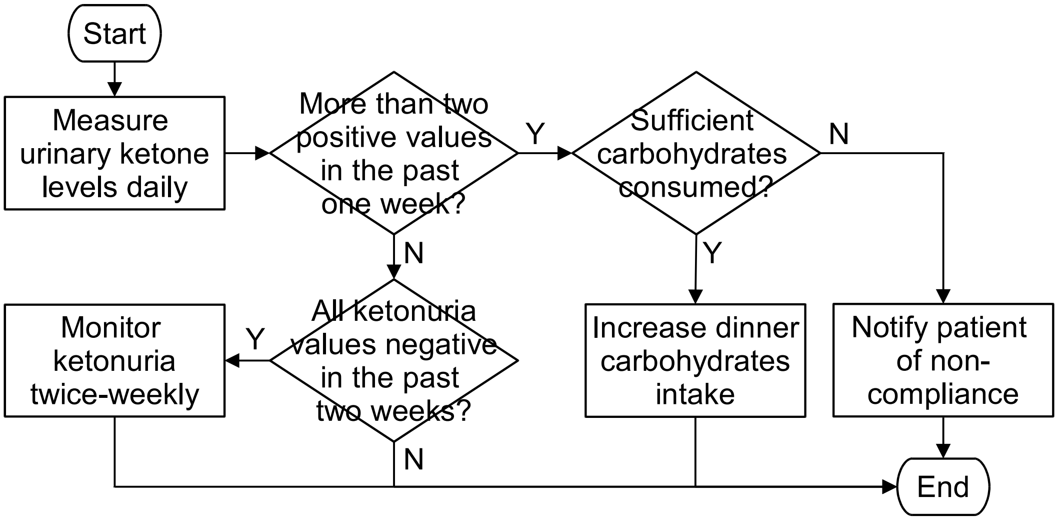
Start
Measure urinary ketone levels daily
More than two positive values in the past one week?
Y
N
Sufficient carbohydrates consumed?
N
Y
All ketonuria values negative in the past two weeks?
Y
N
Monitor ketonuria twice-weekly
Increase dinner carbohydrates intake
Notify patient of non-compliance
End

Figure 2 Simplified extract of a workflow from the GDM guideline [5].

From our experience, there are six generic patterns for deriving the MADE archetypes from a clinical guideline (in the form of a workflow). Firstly, we observed that for each unique measurement action in the guideline, a corresponding observation archetype could be derived. For example, the guideline grades urinary ketone levels into --, -, +/-, + and ++, thus we can derive the following archetype from the action to measure urinary ketone levels in Figure 2:

UrinaryKetone <: ObservedProperty, such that

$\forall\ k$: UrinaryKetone $(\pi_1(k) = \mathit{Urinary\ Ketone} \wedge \pi_4(k) \in \{-⍰, -, +/-, +, ++\})$

Secondly, from each condition in a decision point we can derive either a new abstraction archetype (with two possible values) or, if multiple decision points pertain to the same observation, an additional value in the archetype's value space. Thus from the simplified example in Figure 2, the following two abstraction archetypes were derived:

Ketonuria <: Abstraction, such that

$\forall\ k$: Ketonuria $(\pi_1(k) = \mathit{Ketonuria} \wedge \pi_4(k) \in \{\mathit{Positive}, \mathit{Negative}, \mathit{Average}\})$

CarbohydratesSufficiency <: Abstraction, such that

$\forall\ c$: CarbohydratesSufficiency $(\pi_1(c) = \mathit{Carbohydrates\ Sufficiency} \wedge$

$\pi_4(c) \in \{\mathit{Sufficient}, \mathit{Insufficient}\})$

Similarly, we observed that the actions and decision points thereafter which follow another decision point represent the instructions and action plan resulting from that particular decision. For example, the decision in Figure 2 to monitor ketonuria twice per week can be represented as the following sub-type of an action plan, in which the existential quantifier ∃ is used to specify that each instance of this action plan must contain a scheduled instruction i that:

1) Is of type ScheduledControl (i : ScheduledControl) for controlling the ketonuria monitoring process, and

2) Contains a schedule comprising a repeat pattern and a repeat interval ($\pi_2(i)$: RepeatPattern × RepeatInterval), with the former containing 2 elements and the latter set to weekly (thus enforcing the target process to run twice each week).

KtnTwiceWeeklyPlan <: ActionPlan such that

$\forall\ m$: KtnTwiceWeeklyPlan $(\pi_1(m) = \mathit{Ketonuria\ Twice\ Weekly\ Plan} \wedge$

$$\exists\ i \in \pi_4(m).\ i : \mathrm{ScheduledControl} \wedge \pi_1(i) = \mathit{Monitor\ Ketonuria} \wedge\ \pi_2(i) : \mathrm{RepeatPattern} \times \mathrm{RepeatInterval} \wedge |\ \pi_1(\pi_2(i))| = 2 \wedge \pi_2(\pi_2(i)) = \mathit{Weekly} \wedge \pi_3(i) = \mathit{Running}))$$

However, not all elements in the workflow can be transformed directly to MADE archetypes. In particular, we modelled notifications, which account for 15 out of the 51 action points in the guideline, implicitly by modelling the triggering abstractions and/or action plans; examples are $\mathrm{Ketonuria}$ and $\mathrm{CarbohydratesSufficiency}$ for the non-compliance notification.

Similarly, decision conditions which relate to a specific therapeutic step are captured implicitly by specifying the appropriate control instructions. For example, although not detailed in Figure 2, the patient's carbohydrates intake during dinner should only be increased once, thus we include an extra control instruction that can be used to disable the second and subsequent decisions to change (specifically increase) the carbohydrates intake:

$\mathrm{DinnerWorkflowControl} <: \mathrm{ControlInstruction\ such\ that}$
$\forall\ i : \mathrm{DinnerWorkflowControl}\ (\pi_1(i) = \mathit{Change\ Dinner\ Workflow})$

Finally, unlike the other five high-level MADE data types, measurement archetypes do not have an equivalent in clinical guidelines. This can be expected as measurements do not have a direct clinical interpretation by definition and are therefore derived from technical instead of medical knowledge. For the MobiGuide project, we observed that only one GDM measurement archetype was needed, namely for accelerometer data to compute physical activity levels; all other observations are either directly produced by the sensors (e.g. blood glucose levels) or manually inserted into the system (e.g. urinary ketone levels).

Due to the interactions between different workflows in the GDM guideline, it should also be noted that the required archetypes are occasionally more complex than suggested by the simplified example. Decisions in one workflow may, for example, have a ripple effect on other workflows, whilst tightly coupled clinical concepts may be conveniently merged into a single concept, which was found to be the case for the ketonuria and carbohydrates sufficiency abstraction in the example above. Nevertheless, despite these complexities, a complete MADE DIM was successfully derived for the GDM guideline. Furthermore, like the RIM and the rest of the mathematical model of the MADE language, the GDM DIM was successfully formalised using VDM and verified using the Overture Tool. An example fragment of a VDM specification, which corresponds to the simplified action plan above to monitor ketonuria twice per week is:

```
KtnTwiceWeeklyPlan = ActionPlan
inv plan ==
  plan.#1 = mk_token("Monitor Urinary Ketones Twice Weekly Plan") and
  exists i in set plan.#4 &
    if not is_(i, ScheduledControl) then false
    elseif not is_(i.#2, RepeatPattern * RepeatInterval) then false
    else i.#1 = mk_token("Monitor Ketonuria") and card i.#2.#1 = 2 and
         i.#2.#2 = WEEKLY and i.#3 = <Running>;
```

5 Discussion

In general, employing a RIM allows the designed system to be less dependent on details specific to particular clinical applications and, consequently, be easier to maintain. However, this implies that the RIM should be able to capture all possible data requirements for any possible clinical application, which by its nature, is impossible to validate in advance or in the general case. Nevertheless, to maximise the potential applicability of the MADE RIM, we base the RIM on a conceptual model that not only captures the generic disease management process but is also adapted for pervasive telemedicine systems.

The resulting MADE RIM was formalised using VDM and was successfully tested in this study by deriving a MADE DIM from a complete guideline for GDM patients. In the future, it will be validated further using other clinical applications, but notwithstanding its potential wide applicability, there are three other factors in particular as discussed below which ought to be considered to maximise the overall appropriateness of the MADE RIM.

5.1 Derivation of a MADE DIM

Our experience indicates that deriving the appropriate MADE archetypes from clinical knowledge is not guaranteed to be a trivial task. For example, as exemplified by the GDM guideline, clinical knowledge is often expressed in terms of control flow to reflect the nature of the clinical decision-making process, but due to the data-driven nature of pervasive healthcare systems, it must be converted to a data flow representation for the MADE RIM. As a result, certain guideline elements can only be modelled implicitly, such as decisions which are conditional on specific therapeutic steps.

Furthermore, we observed that a significant proportion of actions in the GDM guideline involve notifications to the patient and/or clinician to perform a task manually, which are currently not modelled explicitly in the MADE RIM. One solution would be to extend the MADE data flow model, but we could argue that these notifications are not clinically relevant per se but merely serve as indications of an underlying, clinically relevant situation, such as the detection of positive ketonuria. Thus by excluding the notifications from the MADE model, we retain a clear separation between the clinical concerns of the RIM (e.g. which clinical circumstances require attention) and the technical concerns of the system design (e.g. the content and modality of any necessary notification).

5.2 Quality Awareness in the MADE RIM

One potential future improvement to the MADE RIM is to extend it with quality-of-data (QoD) awareness properties. Due to the uncontrolled nature of the daily living environment, clinical data may suffer from noise and many other sources of error, including measurement errors due to movement artefacts or from incorrectly positioned sensors. Therefore, to ensure the best clinical support for the patient, it is advantageous to take such potential inaccuracies into account when processing data items, which points to the inclusion of an appropriate QoD model in the MADE RIM. An example in the AI domain is the use of quantified probabilities and utility functions, but in the clinical domain, a multi-dimensional measurement of QoD may be more appropriate, such as that by [13] which encompasses accuracy, dependability, cost, timeliness and quality-of-evidence.

5.3 Interoperability with Clinical Information Systems

Since the MADE RIM is designed specifically for pervasive telemedicine systems, it does not aim to replace the function of existing standards in clinical information systems, including openEHR and HL7. As a result, in order to store the MADE data items in such systems, which may be necessary due to legal and ethical reasons for example, a systematic procedure must also be developed for translating between the MADE RIM and standards in use.

A comparison between the specifications of the different models reveals that the MADE data types can be mapped onto the existing openEHR RIM and HL7 vMR DAM as shown in Table 2. However, it has been shown [1, 2] that the openEHR and HL7 data types are in any case in need of extension to capture all the necessary data requirements in pervasive healthcare systems. For example, [1, 2] demonstrated that transaction times should be added to vMR clinical statements; this holds also for openEHR classes.

Table 2 The correspondence between the data types in the MADE RIM and those in the openEHR RIM and HL7 vMR DAM.

MADE RIM	openEHR RIM	HL7 vMR DAM
Measurement	Observation	Observation Result
Observation		
Abstraction	Evaluation	
Action Plan	Instruction	Observation Order, Procedure Order and/or Substance Administration Order
Action Instruction	Activity	
Control Instruction		

6 Conclusions

As part of research arising from the MobiGuide project, we developed the MADE RIM for pervasive telemedicine systems comprising six main data types: Measurement, Observation, Abstraction, Action Plan, Action Instruction and Control Instruction. The MADE RIM is based on a generic data flow model of disease management to ensure application independence and has been successfully validated against a clinical guideline. In the future, we plan further validation of the MADE RIM's general applicability in the clinical domain, including application to other guidelines.

Apart from guiding the design of the MADE RIM, the adoption of our data flow model also has implications for the design of telemedicine systems which implement the MADE RIM. Specifically, telemedicine systems are assumed and should therefore be designed to be data-driven and to execute a network of concurrent processes during operation. In fact, the MADE RIM represents a milestone in the development of a more generic and extensive computer-interpretable language for specifying required clinical functionality in terms of MADE processes, the main objective of which is to support the development of a wide range of interoperable and application-independent telemedicine solutions.

In future work, quality-awareness will be incorporated in the MADE RIM to address the consequences of the uncontrolled and noisy nature of the free-living setting and the ethical requirement to provide medically appropriate support in all circumstances. Furthermore, translation between the MADE RIM and existing standards will be investigated further to support interoperability with standards-based clinical information systems. Since clinical knowledge is often expressed in terms of control flow, a systematic methodology will also be developed to ensure the correct and complete derivation of the MADE DIM for any specific application.

7 Acknowledgements

The MobiGuide project (http://www.mobiguide-project.eu/) has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 287811. The authors would also like to acknowledge the valuable collaborations and discussions with the partners in the MobiGuide project.

8 Conflicts of Interest

The authors declare that they do not have conflicts of interest in this study.

9 References

1. González-Ferrer A, Peleg M, Parimbelli E, Shalom E, Marcos C, Klebanov G, et al. Use of the virtual medical record data model for communication among components of a distributed decision-support system. In: 2014 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI); 2014. p. 526–530.

2. Marcos C, González-Ferrer A, Peleg M, Cavero C. Solving the interoperability challenge of a distributed complex patient guidance system: a data integrator based on HL7's Virtual Medical Record standard. Journal of the American Medical Informatics Association. 2015;22:587–599.

3. Fitzgerald JS, Larsen PG, Verhoef M. Vienna Development Method. In: Wiley Encyclopedia of Computer Science and Engineering; 2008. p. 1–11.

4. Larsen PG, Lausdahl K, Battle N, Fitzgerald J, Wolff S, Sahara S, et al. VDM-10 Language Manual; 2016. Retrieved December 28, 2016, from http://www.overturetool.org/documentation/manuals.html

5. Goldstein A, Quaglini S, Sacchi L, Panzarasa S, Napolitano C, Larburu N, et al. Deliverable 4.1: CIGs KB. MobiGuide Project (FP7-287811); 2014. Retrieved December 28, 2016, from http://www.mobiguide-project.eu/downloads/category/10-public-deliverables.

6. Broens T, Bults R, Fung N, Jones V, Larburu Rubio N, Widya I, et al. Deliverable 2.3: Distribution of Decision Support Functionality. MobiGuide Project (FP7-287811); 2012. Retrieved December 28, 2016, from http://www.mobiguide-project.eu/downloads/category/10-public-deliverables.

7. Fung NLS, Widya I, Broens T, Larburu N, Bults R, Shalom E, et al. Application of a Conceptual Framework for the Modelling and Execution of Clinical Guidelines as Networks of Concurrent Processes. Procedia Computer Science. 2014;35:671–680.

8. Carson ER, Cramp DG, Morgan A, Roudsari AV. Clinical Decision Support, Systems Methodology, and Telemedicine: Their Role in the Management of Chronic Disease. IEEE Transactions on Information Technology in Biomedicine. 1998;2:80–88.

9. Lasierra N, Alesanco A, Guillén S, García J. A three stage ontology-driven solution to provide personalized care to chronic patients at home. Journal of Biomedical Informatics. 2013;46:516–529.

10. Shahar Y. The “Human Cli-Knowme” Project: Building a Universal, Formal, Procedural and Declarative Clinical Knowledge Base, for the Automation of Therapy and Research. In: Knowledge Representation for Health-Care. vol. 6924; 2012. p. 1–22.

11. Klaassen B, van Beijnum BJ, Weusthof M, Hofs D, van Meulen F, Luinge H, et al. A System for Monitoring Stroke Patients in a Home Environment. In: International Conference on Health Informatics (HEALTHINF 2014); 2014. p. 125–132.

12. Peleg M, Tu S, Bury J, Ciccarese P, Fox J, Greenes RA, et al. Comparing Computer-interpretable Guideline Models: A Case-study Approach. Journal of the American Medical Informatics Association. 2003;10:52–68.

13. Larburu N, Bults RGA, Van Sinderen MJ, Widya I, Hermens HJ. An Ontology for Telemedicine Systems Resiliency to Technological Context Variations in Pervasive Healthcare. IEEE Journal of Translational Engineering in Health and Medicine. 2015;3:1–10.