
# Arden Syntax Implementation Guide
# Release 3

**Contributors**
Robert A. Jenders, MD, MS; Charles Drew University & University of California, Los Angeles
Peter Haug, MD; Intermountain Healthcare & University of Utah
Klaus-Peter Adlassnig, PhD, MS; Medexter Healthcare & Medical University of Vienna

**Project Sponsor**
HL7 Arden Syntax Work Group

**Co-Chairs**
Robert A. Jenders, MD, MS; Charles Drew University & University of California, Los Angeles
Peter Haug MD; Intermountain Healthcare & University of Utah

**HL7 Project #1479**
**August 2019**

1. Purpose ..... 3
2. Notes and Disclaimer ..... 5
3. Arden Syntax: Context and History ..... 6
4. Syntax Description ..... 8
4.1. Fundamentals ..... 8
4.2. Language Concepts ..... 10
4.2.1. Data Types ..... 10
4.2.2. Statements ..... 10
4.2.3. Expressions ..... 11
4.2.4. Operators ..... 12
5. Basic Tasks (by Example) ..... 14
5.1. Sort a List of Objects ..... 14
5.2. Convert String to DateTime ..... 15
5.3. Calculate the Current Age in Years from a Given Birthday ..... 16
5.4. MLM-to-MLM Interaction ..... 16
6. Programming / Engineering Use Cases ..... 20
6.1. Guidelines ..... 20
7. System-Level Engineering Use Cases ..... 24
7.1. Standards-Based Stack for Connecting Arden-Syntax-Based Applications to an EHR 24
7.2. Integrating CDS in PDMS with Minimal Effort ..... 26
7.3. Integrating CDS in a Commercially Available PDMS Using Proprietary Interfaces ..... 27
8. Clinical Use Cases ..... 29
8.1. Drug–Disease Interaction ..... 29
8.2. Body Mass Index ..... 31
8.3. Abnormal Test Result Detection ..... 32
8.4. Detection of Possible Patient Deterioration ..... 34
8.5. Least Squares Linear Regression ..... 39
8.6. Calculate Creatinine Clearance and Glomerular Filtration Rate ..... 42
8.7. Hepatitis A Immunization ..... 46
9. Standard Data Models and Data Access ..... 51
9.1. Data Models and Their Role in CDS and Arden Syntax ..... 51
9.2. Role of Terminologies ..... 52
9.3. Current Arden Syntax Data Model ..... 53
9.4. FHIR as a Standard Data Model ..... 53
10. Preview of Arden 3.0: The Curly-Braces Challenge and Other Standards ..... 55
11. Representation of Business Processes by Combining Arden and Workflow Languages 59
11.1. A Brief Introduction to OMG Standards for Business Process Management ..... 60
11.2. Potential Relationship of OMG Standards to Arden Syntax ..... 61
11.3. Examples ..... 61
12. F.A.Q. ..... 64
13. References ..... 72

# 1. Purpose

The Arden Syntax for Medical Logic Systems is a structured, executable formalism for the explicit representation of scientific, clinical, administrative, and other knowledge used in clinical decision support systems. As such, it functions as a programming language for such systems, allowing knowledge authors and clinical domain experts to implement knowledge-based interventions such as alerts, informational notices, and reminders, order sets, turnaround forms and the like in order to realize their quality improvement, clinical, administrative, and public health objectives. Expressed in a way that resembles English-language syntax, the Arden Syntax also facilitates validation of knowledge bases by domain experts. In light of this utility, a number of vendors of clinical information systems and decision support systems have incorporated Arden Syntax into their products, leading to its adoption and use at numerous sites worldwide.

Despite the relative ease of use and functionality of the Arden Syntax, both novel and experienced users have questions regarding how best to use the syntax to address particular clinical applications. Examples include generation of alerts related to drug–disease interactions, the implementation of multi-part clinical guidelines, immunization decision support and others. In addition, users and potential users of Arden Syntax may need to know how to use Arden Syntax to accomplish basic engineering tasks such as object manipulation, list sorting, and the like.

The purpose of this implementation guide is to help answer these questions by providing, in addition to a summary of the Arden Syntax itself, ideas and examples regarding how Arden Syntax may be used in these different situations. This guide is not intended to be exhaustive in this regard, but it is meant to provide guidance on how to use the Arden Syntax to solve real-world challenges related to the implementation of clinical decision support. Further, while the summary of the Arden Syntax features presented herein provides the important highlights of this key standard, readers are directed to the actual Arden Syntax specifications for a complete definition of the language.

This implementation guide was composed when Arden Syntax v2.10 was the latest approved version of the standard and v3.0 was under development. While the examples and ideas featured here include elements of the syntax that are new to these versions and may not be present in earlier versions, substantial parts of the implementation guide also leverage the backward compatibility of Arden Syntax, allowing users of an earlier version also to make use of this implementation guide.

Finally, the reader should be aware that, while the authors have been diligent in providing useful, accurate content derived from real-world solutions already implemented in clinical decision support systems, no guarantee of accuracy or effectiveness is made regarding the examples and other information presented in this implementation guide. Any user or implementer of Arden Syntax assumes all liability regarding the use of any material contained in this guide.

The authors of this implementation guide hope that you find it a useful addition to other Health Level Seven publications related to clinical decision support (CDS) in ways that allow you to make

best use of the powerful and rich standard for representing CDS knowledge that is the Arden Syntax.

# 2. Notes and Disclaimer

This implementation guide is not normative. Knowledge of the Arden Syntax standard is a prerequisite for reading this document. For a detailed description of this standard, we would like to refer to the Arden Syntax specification available on the Health Level Seven (HL7) International's website.

# 3. Arden Syntax: Context and History

Computer-based clinical decision support (CDS) has been shown to improve the quality of health care treatment and the performance of health care professionals. CDS involves delivering knowledge to decision-makers in clinical settings in order to improve the quality of decisions and the outcomes to which they lead. CDS sometimes is described in terms of the Five Rights: Delivering the right knowledge to the right person at the right time in the workflow in the right format via the right channel [1]. Some channels are bidirectional to allow user response to be collected and documented.

In order to provide computer-based CDS, the knowledge to be delivered must be represented in digital form. In this light, CDS can be divided into two broad classes: Services that facilitate delivery of knowledge and explicit, computable representations of the knowledge itself that can be shared via transfer and reuse. In the case of a knowledge delivery service, standards facilitate communication between electronic health record (EHR) systems and other clinical software and knowledge sources, allowing connection of systems and sources from multiple vendors without having to negotiate and implement ad hoc methods for each connection. In the case of explicit knowledge encoding, standards facilitate sharing of knowledge by minimizing the changes necessary for the knowledge to be executed or used in different information systems.

The HL7 Infobutton standard is an example of a knowledge delivery service standard. It facilitates queries from users of EHR systems in the context of particular care activities and particular patients, providing knowledge from knowledge sources that is pertinent to these contexts. By contrast, examples of explicit knowledge encoding include the HL7 GELLO (a loose acronym for Guideline Execution Language Object-oriented), Order Set, and Arden Syntax standards.

A knowledge representation formalism constitutes one part of an overall CDS system. Units of knowledge encoded using the formalism are stored in the knowledge base (KB), independent of but linked to the inference engine or event monitor that executes units of the KB in combination with patient data to produce tailored, context-specific knowledge-based interventions that then can be delivered to the appropriate recipient such as a clinician, patient, or administrator.

A prominent example of a knowledge formalism for encoding units of knowledge in the KB is the Arden Syntax for Medical Logic Systems. This is a computable language for encoding medical knowledge. It was previously adopted as a standard by the American Society for Testing and Materials (ASTM) as document E 1460, under subcommittee E31.15 Health Knowledge Representation. Adopted in 1992, it became Arden Syntax version 1.0.

Beginning in 1998, sponsorship of this standard was moved to HL7 International. Maintenance and further development of the standard is now overseen by the HL7's Arden Syntax Work Group. The Arden Syntax version 2.0 was formally adopted by HL7 and the American National Standards Institute (ANSI) in August 1999. Since then the standard has evolved, including the addition of new features and functionalities responding to the needs of users and vendors. Presently, the

standard's latest version—version 2.10—was adopted by HL7 and certified by ANSI in May 2014. Recent descriptions on the origins and evolution of the Arden Syntax can be found in [2, 3].

Arden Syntax uses medical logic modules (MLMs) as units of knowledge representation. Each of these MLMs contains sufficient knowledge to make a single medical decision. MLMs have been used to generate clinical reminders and alerts, interpretations, diagnoses and therapeutic advice, screening for clinical research, quality assurance functions, and administrative support. Using a computer program called an event monitor, MLMs run automatically, generating advice where and when it is needed.

This implementation guide describes the key features of the Arden Syntax and how it may be used in a variety of scenarios to deliver CDS.

# 4. Syntax Description

## 4.1. Fundamentals

Medical knowledge in Arden Syntax is as stated above arranged within MLMs, each of which contains sufficient knowledge to make a single decision. These MLMs are well organized and structured into categories and slots with specific content:

```
maintenance:
  title:       ;;
  mlmname:     ;;
  arden:       ;;
  version:     ;;
  institution: ;;
  author:      ;;
  specialist:  ;;
  date:        ;;
  validation:  ;;
library:
  purpose:     ;;
  explanation: ;;
  keywords:    ;;
  citations:   ;;
  links:       ;;
knowledge:
  type:        ;;
  data:        ;;
  priority:    ;;
  evoke:       ;;
  logic:       ;;
  action:      ;;
  urgency:     ;;
resources:
  default:     ;;
  language:    ;;
end:
```

The slots constituting an MLM are grouped into four categories: maintenance, library, knowledge, and resources. Each category starts with its name, followed directly by a colon (e.g., maintenance:). Both the four categories and the set of slots within each category have to appear in the correct order (see Figure 1).

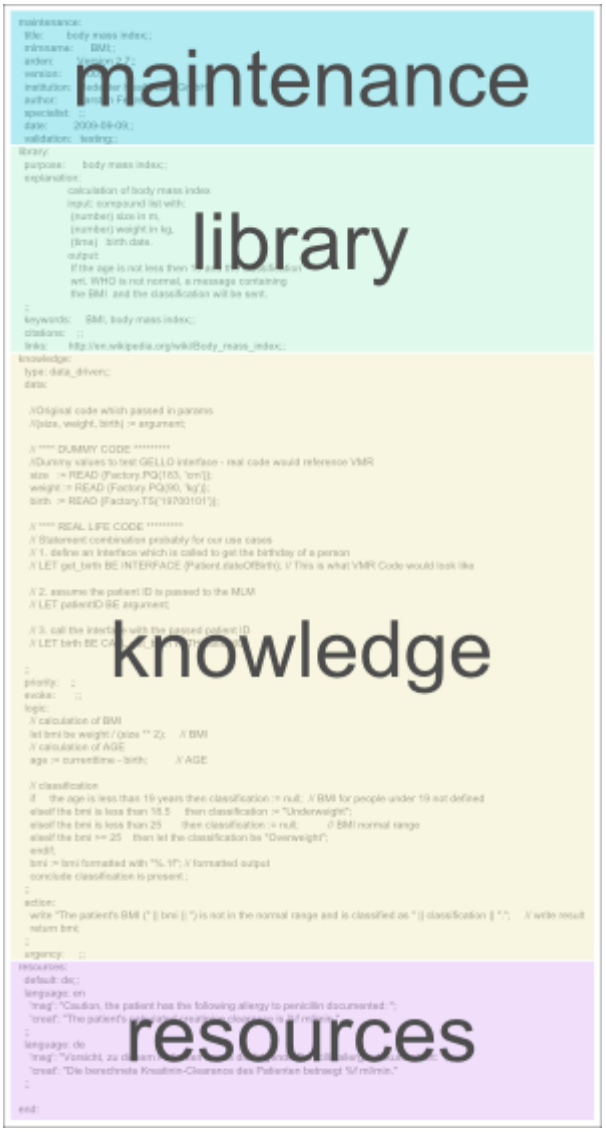



Figure 1

The maintenance category contains information unrelated to the MLM's health knowledge and is used for MLM knowledge base maintenance and change control. The library category provides health personnel with explanatory information as well as links to relevant health literature related to the MLM's health knowledge. The knowledge category actually defines the MLM's action, data access, and logic. The resources category specifies localized textual resources that can be used within the knowledge category.

An MLM is identified by using the following three pieces of information: name, institution, version.

## 4.2. Language Concepts

### 4.2.1. Data Types

The basic function of an MLM is to retrieve patient data, manipulate the data, reach a decision, and possibly perform an action. The data necessary may come from various sources, for example via a direct query to the patient database, from a constant in the MLM, or resultant from an operation on other data. Available data types within the Arden Syntax are: Null, Boolean, Number, Time, Duration, String, List, Object, Truth Value, and Fuzzy Sets. Every data item consists of a value part, a primary time part (e.g., time of data retrieval), and its applicability. The primary time is the most clinically relevant time, which differs depending of the type of the data and the circumstances of their acquisition. In some cases there will be several candidate primary times depending on the clinical context.

| Category | Example Type of Data | Candidate Primary Time |
|---|---|---|
| Volatile Observations (simple) | Blood glucose level | Time sample obtained from patient |
| Volatile Observations (simple) | 24h urinary protein excretion | Time point of end collection period |
| Volatile Observations (complex) | Aggregated test result, such as creatinine clearance | The primary time of the most recent constituent |
| Volatile Observations (complex) | Score calculation based on multiple test results (e.g., Charlson Index) | The primary time of the most recent constituent |
| Non-Volatile Observations | Blood type | |

### 4.2.2. Statements

Slots in Arden Syntax are structured, that is, composed of a set of statements. Each statement specifies a logical constraint or an action to be performed. In general, statements are carried out sequentially in the order that they are listed. All statements except for the last statement in a slot must end with a semicolon (;). Possible statements are:

- Read statement: reads data from external resources
- Event statement: assigns an institution-specific event definition to a variable
- Message statement: assigns an institution-specific message (e.g., an alert) to a variable
- Destination statement: assigns an institution-specific destination to a variable (e.g., email, pager, printer, etc.)
- Interface statement: assigns an institution-specific foreign-function interface definition to a variable
- Assignment statement: places the value of an expression into a variable
- Write statement: sends a text or coded message to a destination
- Include statement: indicates that an external MLM may be consulted for object, MLM, event, interface variable, and resource definitions
- Conclude statement: ends execution in the logic slot

- Argument statement: accesses passed arguments
- Return statement: returns a result to the calling instance
- Loops: while- and for-loops
- If-then-else: permits branching/conditional execution depending on the value of an expression
- Object statement: assigns object declaration to a variable
- Call statement (MLM, event, interface): permits an MLM to call other MLMs, events, or external interfaces
- Trigger: evokes a slot statement that defines how an MLM may be triggered

### 4.2.3. Expressions

Statements are composed of reserved words, special symbols, and expressions. Expressions may contain any of the following:

- Constant: data value that is explicitly represented
- Variable: a placeholder for a data value or special constructs (e.g., an event, MLM, message, or destination) and represents this value in any subsequent expressions. An assignment statement is used to assign a value to a variable.
- Operator and arguments (e.g., a mathematical calculation)

Special case: curly brace expressions

As explained in the F.A.Q. section of this document, institution-specific references to the local clinical data repository are embedded in so-called “curly braces” {}. This convention permits MLMs to be shared between institutions, since ideally only the codes/directives within the curly braces need to be revised; the logic in the remaining slots of the MLMs may remain unchanged. Here are some examples of the contents of curly braces from several implementations:

Event calls:

```
bmiEvent := EVENT {bmiEvent};

pathology_upload := EVENT {'32506','32688'};

med_order_event := EVENT {A Med Order Entered:Antithrombotic Meds};
```

Read Statements:

```
aminoglycoside_order := READ LAST
     {'evoking','dam'="PDQORD1",'auxstr'="0013",
      'constraints'="C****",'status_value'="A",
      'display_header'="R",'display_comp'="V"; ; '23946'};


(ordName,ordFreq,ordStatus,ordPriority,cpFName,cpLName,
cpTitle,cpSeq,ordSeq)
       := READ LAST {An Active Order: Antithrombotic Therapy Exclusion};


last_alert := READ LAST (
     {'dam'="PDQDEC1",'display_header'="TX",'display_comp'="";
      'mlmself','mlm RF_AND_AMINOGLYCOSIDE'}
```

```
     WHERE IT OCCURRED WITHIN THE PAST 3 MONTHs);

previous_alert := READ LAST {Previous Alerts:Missing Antithrombotic Therapy for Stroke Patient}
                       WHERE IT OCCURRED AFTER time_between_alerts AGO;
```

Destination:

```
email_dest := DESTINATION {'email', 'name'="sidelir@cucis.cis.columbia.edu"};

carlene_email := DESTINATION {Email:Carlene Dobber Email};
```

Some vendors have implemented tools to support the creation and maintenance of curly brace expressions. For example, this “Statement Builder” permits users to create READ statements from a variety of options:

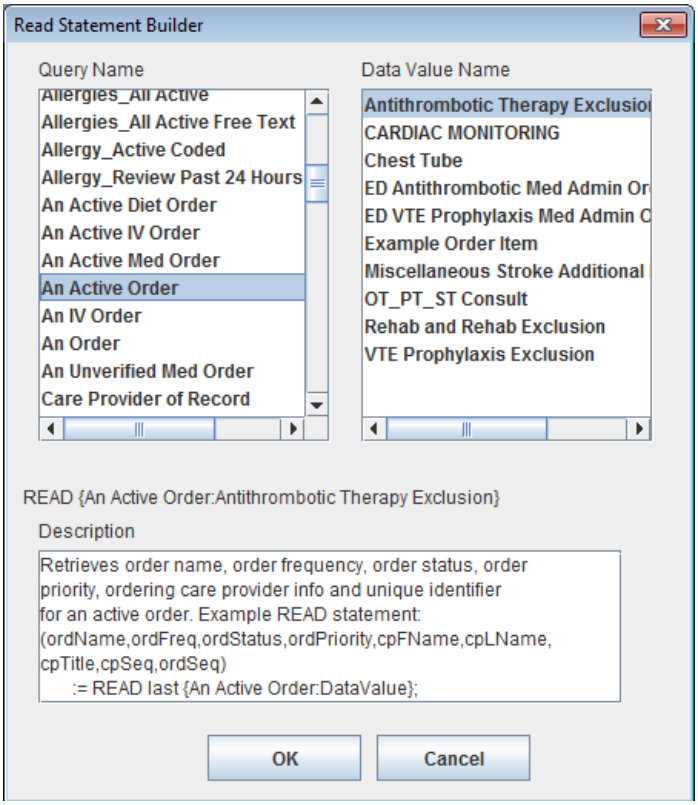



Figure 2

### 4.2.4. Operators

Operators are used in expressions to manipulate data. They accept one or more arguments (data values) and they produce a result (a new data value). There are the following kinds of operators:

- List operators
- Logical operators
- Comparison operators
- String operators
- Arithmetic operators (including trigonometric and logarithmic functions)

- Temporal operators
- Aggregation operators
- Time operators
- Object operators

# 5. Basic Tasks (by Example)

## 5.1. Sort a List of Objects

```
maintenance:
 title:       sample_sort_objects;;
 mlmname:     sample_sort_objects;;
 arden:       Version 2.9;;
 version:     1.0;;
 institution: Medexter Healthcare;;
 author:      ;;
 specialist:  ;;
 date:        2016-12-03;;
 validation:  testing;;
library:
 purpose:     ;;
 explanation: ;;
 keywords:    ;;
 citations:   ;;
 links:       ;;
knowledge:
 type:            data_driven;;
 data:
   guidResult := OBJECT [      // result object: result of guideline query
       compliance,             // the compliance of the patient to a guideline
       eOfIntervention,        // the "ease of intervention"
       text];                  // the text
 ;;
 priority:    ;;
 evoke:       ;;
 logic:

   result_list := ();

   resGuid1 := NEW guidResult WITH FALSE, 12, "The patient is not in compliance with the
guideline.";
   result_list := result_list, resGuid1;

   resGuid2 := NEW guidResult WITH TRUE, 70, "The patient is in compliance with the guideline.";
   result_list := result_list, resGuid2;

   resGuid3 := NEW guidResult WITH TRUE, 23, "The patient is in compliance with the guideline.";
   result_list := result_list, resGuid3;

   /* sorting the result */
   sorted_list := ();
   FOR en IN result_list DO
     inserted := FALSE;
     n := 1;
     list_size := COUNT sorted_list;

     WHILE inserted = false DO
       IF list_size = 0 THEN
         sorted_list := sorted_list, en;
         inserted := TRUE;

       ELSEIF n <= list_size THEN
         el := sorted_list[n];
         IF ((el.compliance = FALSE) AND (en.compliance = FALSE)) AND (el.eOfIntervention >
en.eOfIntervention) THEN
```

```
          sorted_list := sorted_list[1 SEQTO (n-1)], en, sorted_list[n SEQTO list_size];
          inserted := TRUE;

        ELSEIF (el.compliance = TRUE) AND (en.compliance = FALSE) THEN
          sorted_list := sorted_list[1 SEQTO (n-1)], en, sorted_list[n SEQTO list_size];
          inserted := TRUE;

        ELSEIF ((el.compliance = TRUE) AND (en.compliance = TRUE)) AND (el.eOfIntervention >
en.eOfIntervention) THEN
          sorted_list := sorted_list[1 SEQTO (n-1)], en, sorted_list[n SEQTO list_size];
          inserted := TRUE;
        ENDIF;

      ELSE
        sorted_list := sorted_list, en;
        inserted := TRUE;
      ENDIF;
      n := n +1;
    ENDDO;
  ENDDO;
 CONCLUDE TRUE;
 ;;
 action:
   RETURN sorted_list;
 ;;
 urgency:     ;;
end:
```

## 5.2. Convert String to DateTime

In former versions of the Arden Syntax, the knowledge contained in the following MLM was necessary to allow the conversion of a string into a DateTime:

```
maintenance:
 title:       convert_stringdate_to_datetime;;
 mlmname:     convert_stringdate_to_datetime;;
 arden:       Version 2.5;;
 version:     1.0;;
 institution: Medexter Healthcare;;
 author:      ;;
 specialist:  ;;
 date:        2016-12-03;;
 validation:  testing;;
library:
 purpose:     ;;
 explanation: ;;
 keywords:    ;;
 citations:   ;;
 links:       ;;
knowledge:
 type:            data_driven;;
 data:
       DOB_str := ARGUMENT;
 ;;
 priority:    ;;
 evoke:       ;;
 logic:
       /* for example DOB_str := "1973-11-02T12:34:56"; */

       year_str := SUBSTRING 4 CHARACTERS FROM DOB_str;
```

```
       month_str := SUBSTRING 2 CHARACTERS STARTING AT 6 FROM DOB_str;
       day_str := SUBSTRING 2 CHARACTERS STARTING AT 9 FROM DOB_str;
       hour_str := SUBSTRING 2 CHARACTERS STARTING AT 12 FROM DOB_str;
       minute_str := SUBSTRING 2 CHARACTERS STARTING AT 15 FROM DOB_str;
       second_str := SUBSTRING 2 CHARACTERS STARTING AT 18 FROM DOB_str;

       startDate := 1800-01-01T00:00:00;

       DOB := startDate + (year_str AS NUMBER) YEARS - 1800 YEARS;
       DOB := DOB + (month_str AS NUMBER) MONTHS - 1 MONTH;
       DOB := DOB + (day_str AS NUMBER) DAYS - 1 DAY;
       DOB := DOB + (hour_str AS NUMBER) HOURS;
       DOB := DOB + (minute_str AS NUMBER) MINUTES;
       DOB := DOB + (second_str AS NUMBER) SECONDS;

       CONCLUDE TRUE;
 ;;
 action:
       RETURN DOB;
 ;;
 urgency:     ;;
end:
```

As of Arden Syntax version 2.8, this whole conversion can be simply done by using the following statement:

```
       DOB := "1973-11-02T12:34:56" AS TIME
```

## 5.3. Calculate the Current Age in Years from a Given Birthday

Given any date of birth, the corresponding age in years can be calculated using the following expression:

```
Age_duration := currenttime - DOB;
Age_yrs := Age_duration/1 YEAR;
```

## 5.4. MLM-to-MLM Interaction

The following examples illustrate possible interactions between MLMs, such as importing definitions from and calling other MLMs.

This MLM solely contains an object definition:

```
maintenance:
 title:       sample_object_definition;;
 mlmname:     sample_object_definition;;
 arden:       Version 2.9;;
 version:     1.0;;
 institution: Medexter Healthcare;;
 author:      ;;
 specialist:  ;;
 date:        2016-12-03;;
 validation:  testing;;
library:
 purpose:     ;;
 explanation: ;;
 keywords:    ;;
```

```
 citations:   ;;
 links:       ;;
knowledge:
 type:            data_driven;;
 data:
   guidResult := OBJECT [      // result object: result of guideline query
       compliance,             // the compliance of the patient to this guideline
       eOfIntervention,        // the "ease of intervention"
       text];                  // the text
 ;;
 priority:    ;;
 evoke:       ;;
 logic:       ;;
 action:      ;;
 urgency:     ;;
end:
```

By using the following two lines in the data slot of an MLM, it is possible to import and use the object definition of the MLM shown above within another MLM. This procedure allows, for example, large knowledge bases to share common definitions across all involved MLMs:

```
  // MLM
   mlmImport       := MLM 'showcase_spike_definition'  FROM INSTITUTION "Medexter Healthcare";

   // include
   INCLUDE mlmImport;
```

In large knowledge bases, it is often necessary to be able to use particular calculations or parts of rules frequently. For this purpose, reusable parts of rules can be stored in separate MLMs that are accessible for all MLMs of the knowledge base. Additionally, this increases the readability and maintainability of the knowledge base and allows the reuse of outsourced parts in new knowledge bases.

Example for calling other MLMs:

This MLM may be called by other MLMs to standardize the output of patient information in email or SMS (Simple Message System or “text”) messages.

```
/*  Only call this MLM when you know that the calling rule is going to CONCLUDE TRUE
This rule should always be moved into either the Research or Testing folders. */

MAINTENANCE:
  title:       Create Patient HIPAA Headers and Confid Msg ;;
  mlmname:     Create_Patient_HIPAA_Headers_and_Confid_Msg ;;
  arden:       Version 2.5;;
  version:     1.00;;
  institution: MPT;;
  author:      MCK;;
  specialist:  ;;
  date: 2006-01-15;;
  validation:  Testing;;

LIBRARY:
  purpose:     Use a single rule to create a standard patient header, a HIPAA header,
         and a confidentiality message for notification messages. ;;
  explanation:
        This subroutine is called by a main rule to create a standard patient header.
```

```
       The full header is to be used only internally at the site in order to be HIPAA compliant.
       Otherwise, use the HIPAA Header.  Can also include a Confidentiality Message as a footer
in printer or email messages.
       ;;
  keywords:    Patient Header; called rule, subroutine;;
  citations:   ;;
  links:       ;;

KNOWLEDGE:
  type:  data-driven;;

  data:
       /* All READs are done in this subroutine so it is independent of the Calling rule.  */

       (patTypeGroup, patService, patClass, patType) := READ last {Patient Type Information};

       (patFacilityID, patDeptID, patRoomID, patBedID):= READ last {Patient Location};

       (patSex, patLastName, patFirstName, patAcct, patMedRec, patMiddleName)
              := READ last {Person Info};
       ;;

  evoke:       // No trigger event - this MLM is 'triggered' by the rule that calls it.
       ;;

  logic:
       if (patMiddleName is not present) then
              patient_name := patLastName || ", " || patFirstName;
       else
              patient_name := patLastName || ", " || patFirstName || " " || patMiddleName;
       endif;

       patient_header := NL ||  "Patient: " || patient_name || NL
                            ||  "Facility:  " || patFacilityID || NL
                            ||  "Unit: " || patDeptID || NL
                            ||  "Location:  " || patRoomID || NL
                            ||  "Acct Number:  " || patAcct || NL || NL ;

       hipaa_header := "RE: Patient in this Dept "  || patDeptID || ", Room " || patRoomID ||
       "; Acct # " || patAcct ||  NL ;

       confidential_msg := NL || NL || NL || "Confidentiality Notice: This message is for "
       || "the sole use of the intended recipient(s)and may contain confidential and "
       || "privileged information.  Any unauthorized review, use, disclosure or distribution "
       || "is prohibited.";

       CONCLUDE TRUE;
       ;;
  action:      /* Returns a formatted patient header to the main rule in a single
       variable as well as a hipaa header and confidential message */
       RETURN (patient_header, hipaa_header, confidential_msg);
              ;;
  urgency:     49;;
  END:
```

## An example call statement that calls the preceding MLM:

```
// Define the rule in the data slot:
get_patient_header := 'Create_Patient_HIPAA_Headers_and_Confid_Msg';

// Example call statement for rule:
```

```
(patient_header, hipaa_header, confidential_msg) := CALL get_patient_header ;
```

# 6. Programming / Engineering Use Cases

## 6.1. Guidelines

The following MLM is part of a larger knowledge base implementing a guideline for borreliosis detection and support [4]. The calling instance sends a set of patient parameters to this MLM. The MLM checks, if relevant parameters for a decision are in the list. If at some point during the decision process a parameter necessary for the decision is missing, the MLM will create a so called ‘question object’ that is sent back to the calling instance. The calling instance can then use this question object to ask the corresponding medical staff for the specific parameter of the patient. With the additional parameter (additional to the original list) the MLM is called again. If no further parameters are needed for the final decision, the MLM will return a result or a recommendation based on the guideline.

```
maintenance:
 title: borreliosis_erythema_migrans_treatment;;
 mlmname: borreliosis_erythema_migrans_treatment;;
 arden: Version 2.9;;
 version: 1.08;;
 institution: Medexter Healthcare;;
 author: ;;
 specialist: ;;
 date: 2016-12-03;;
 validation: testing;;
library:
 purpose: borreliosis support;;
 explanation:
   managing all borreliosis EM treatment
 ;;
 keywords: borreliosis, decision support;;
 citations: ;;
 links: ;;
knowledge:
 type: data_driven;;
 data:
   parameterlist := ARGUMENT;
   result_obj := OBJECT[result, next_questions];
   question_obj := OBJECT[q_type, q_label, q_info];
 ;;
 priority: ;;
 evoke: ;;
 logic:
   resultobj := NEW result_obj;
   resultobj.result := ();
   resultobj.next_questions := ();

   age := -1;
   gender := -1;

   pregnant := -1;
   breastfeeding := -1;
   allergy_amocicilline_doxycycline := -1;
   contraindication_amocicilline_doxycycline := -1;

   FOR parameter IN parameterlist DO
       IF parameter.key EQ "Age" THEN
```

```
            age := parameter.val;
        ELSEIF parameter.key EQ "Gender" THEN
            gender := parameter.val;
        ELSEIF parameter.key EQ "Pregnant" THEN
            pregnant := parameter.val;
        ELSEIF parameter.key EQ "Breastfeeding" THEN
            breastfeeding := parameter.val;
        ELSEIF parameter.key EQ "Allergy" THEN
            allergy_amocicilline_doxycycline := parameter.val;
        ELSEIF parameter.key EQ "Contraindication" THEN
            contraindication_amocicilline_doxycycline := parameter.val;
        ENDIF;
    ENDDO;

   // if female: pregnant? breastfeeding?
    IF gender eq 1 THEN // female
        IF pregnant eq -1 THEN
            question := NEW question_obj;
            question.q_type := "polar_bool";
            question.q_label := "Pregnant";
            question.q_info := "Pregnancy is the fertilization and development of one or more
offspring.";
            resultobj.result := resultobj.result, LOCALIZED 'pregnant';
            resultobj.next_questions := resultobj.next_questions, question;
        ENDIF;
        IF breastfeeding eq -1 THEN
            question := NEW question_obj;
            question.q_type := "polar_bool";
            question.q_label := "Breastfeeding";
            question.q_info := "Breastfeeding is the feeding of an infant or young child with
breast milk directly from female human breasts.";

            resultobj.result := resultobj.result, LOCALIZED 'breastfeeding';
            resultobj.next_questions := resultobj.next_questions, question;
        ENDIF
    ENDIF;

    // check for allergies
    IF allergy_amocicilline_doxycycline eq -1 THEN // then
contraindication_amocicilline_doxycycline also -1
        question := NEW question_obj;
        question.q_type := "polar_fuzzy";
        question.q_label := "Allergy";
        question.q_info := "mocicilline and/or Doxycycline allergy.";
        question2 := NEW question_obj;
        question2.q_type := "polar_fuzzy";
        question2.q_label := "Contraindication";
        question2.q_info := "Amocicilline and/or Doxycycline contraindication.";
        resultobj.result := resultobj.result, LOCALIZED 'allerg_amocicilline_doxycycline',
LOCALIZED 'contra_amocicilline_doxycycline';
        resultobj.next_questions := resultobj.next_questions, question, question2;

    ELSEIF allergy_amocicilline_doxycycline THEN
        IF pregnant THEN
            resultobj.result := resultobj.result, LOCALIZED
'erythema_migrans_alergy_therapy_pregnant';
        ELSE
            IF age < 18 THEN // child
                resultobj.result := resultobj.result, LOCALIZED
'erythema_migrans_alergy_therapy_child';
            ELSE
```

```
              resultobj.result := resultobj.result, LOCALIZED
'erythema_migrans_alergy_therapy_adult';
           ENDIF;
       ENDIF;
       question := NEW question_obj;
       question.q_type := "polar_bool";
       question.q_label := "Executed EM Therapy";
       question.q_info := "Patient took the medication for the adviced duration and dosage.";
       resultobj.next_questions := resultobj.next_questions, question;

   ELSEIF contraindication_amocicilline_doxycycline THEN // same as
allergy_amocicilline_doxycycline - should be solved with or
       IF pregnant THEN
           resultobj.result := resultobj.result, LOCALIZED
'erythema_migrans_alergy_therapy_pregnant';
       ELSE
           IF age < 18 THEN // child
               resultobj.result := resultobj.result, LOCALIZED
'erythema_migrans_alergy_therapy_child';
           ELSE
               resultobj.result := resultobj.result, LOCALIZED
'erythema_migrans_alergy_therapy_adult';
           ENDIF;
       ENDIF;
       question := NEW question_obj;
       question.q_type := "polar_bool";
       question.q_label := "Executed EM Therapy";
       question.q_info := "Patient took the medication for the adviced duration and dosage.";
       resultobj.next_questions := resultobj.next_questions, question;

   ELSE // no allergy or contradiction
       IF pregnant THEN
           resultobj.result := resultobj.result, LOCALIZED 'erythema_migrans_therapy_adult1';
       ELSEIF breastfeeding THEN // same as pregnant
           resultobj.result := resultobj.result, LOCALIZED 'erythema_migrans_therapy_adult1';
       ELSE
           IF age < 18 THEN // child
               resultobj.result := resultobj.result, LOCALIZED 'erythema_migrans_therapy_child1';
               IF age > 9 THEN
                   resultobj.result := resultobj.result, LOCALIZED
'erythema_migrans_therapy_child2';
               ENDIF;
           ELSE
               resultobj.result := resultobj.result, LOCALIZED 'erythema_migrans_therapy_adult1',
LOCALIZED 'erythema_migrans_therapy_adult2';
           ENDIF;
       ENDIF;
       question := NEW question_obj;
       question.q_type := "polar_bool";
       question.q_label := "Executed EM Therapy";
       question.q_info := "Patient took the medication for the adviced duration and dosage.";
       resultobj.next_questions := resultobj.next_questions, question;
   ENDIF;

   CONCLUDE TRUE;
 ;;
 action:
   RETURN resultobj;
 ;;
 urgency: ;;
 resources:
   default: en;;
```

```
   language: en
        'pregnant': "Check if patient is pregnant.";
        'breastfeeding': "Ask patient if she is breastfeeding";
        'allerg_amocicilline_doxycycline': "Check if patient has an Amocicilline or Doxycycline
allergy.";
        'contra_amocicilline_doxycycline': "Check if there are contraindications for Amocicilline
or Doxycycline.";
        'erythema_migrans_alergy_therapy_pregnant': "Choose Azithromycin or Erytromycin and take
Ceftriaxone (under clinical surveillance) into consideration.";
        'erythema_migrans_alergy_therapy_adult': "Choose Cefuroxim (2x500 mg p.o., 14-21 days),
Azithromycin (1x500 mg p.o., 7-10 days) or Clarithromycin (2x500 mg p.o., 14-21 days)";
        'erythema_migrans_alergy_therapy_child': "Choose Cefuroxim (30 mg/kg/day, 14-21 days),
Azithromycin (5-10 mg/kg/day, 5-10 days) or Clarithromycin (2x500 mg p.o., 14-21 days)";
        'erythema_migrans_therapy_adult1': "Choose Amaxicillin (3x500 mg p.o., 14-21 days).";
        'erythema_migrans_therapy_adult2': "Doxycycline (2x100 mg p.o., 10 days) is also an
option.";
        'erythema_migrans_therapy_child1': "For children use Amaxicillin (2-4 mg/kg/day, 14-21
days)";
        'erythema_migrans_therapy_child2': "For children older than 9 years Doxycycline (2-4
mg/kg/day, 10 days) is an option";;
end:
```

# 7. System-Level Engineering Use Cases

## 7.1. Standards-Based Stack for Connecting Arden-Syntax-Based Applications to an EHR

**Objective**

Organizations may encounter the need to connect an Arden-Syntax-based application to an arbitrary electronic health record (EHR) system. As a result, there may be interest in developing applications incorporating standards-based clinical logic representation and accessing EHR data in a schema-independent fashion. Note that “application” as used here may be either user-interactive (e.g., a browser-based application) or non-interactive (e.g., a background monitoring system).

**Solution: elements**

- An authoring environment for development of computable knowledge in a shareable, standards-based representation (HL7 Arden Syntax)
- A free-standing, standards-based inference engine operating on standards-based representation of clinical logic (HL7 Arden Syntax) and using standards-based, schema-independent data references (HL7 GELLO / Virtual Medical Record [vMR])
- A standards-based, schema-independent interface to the target EHR (HL7 GELLO / vMR) either through a mediation layer or a native implementation
- If a mediation layer is used, exposure of a data access service by the target EHR

**Target behavior summary for an interactive application**

- The browser-based user interface uses standard web services (i.e., http/https) to connect to the middle tier CDS application (e.g., clinical reminder package or patient report card).
- The middle tier application invokes a CDS resource (inference service) to conduct the necessary clinical logic evaluation activity and then formats / produces results for return to the user. (Figure 3)
- The CDS resource in turn uses standard web services to access a standards-based data layer to fetch data from the target EHR. (Figure 4)

**Sample implementation**

The approach described here was implemented as part of an innovation project carried out by the Veterans Health Administration using a GELLO/vMR layer developed by Medical Objects (Australia) and an Arden Syntax authoring and execution environment developed by Medexter Healthcare. Figures 3 and 4 reflect that project. The project successfully demonstrated a complete stack using HL7-compliant software and employing GELLO expressions to provide a solution to the “curly braces problem”.

**NOTE:** Mention of these commercial products does not represent endorsements of any particular company or product by U.S. Veterans Health Administration (VHA) or HL7, and appear here only to be concrete about the nature of the architecture presented in this Engineering Use Case.

**Innovation Initiative # 209 –**
Clinical Decision Support
Transaction Diagram

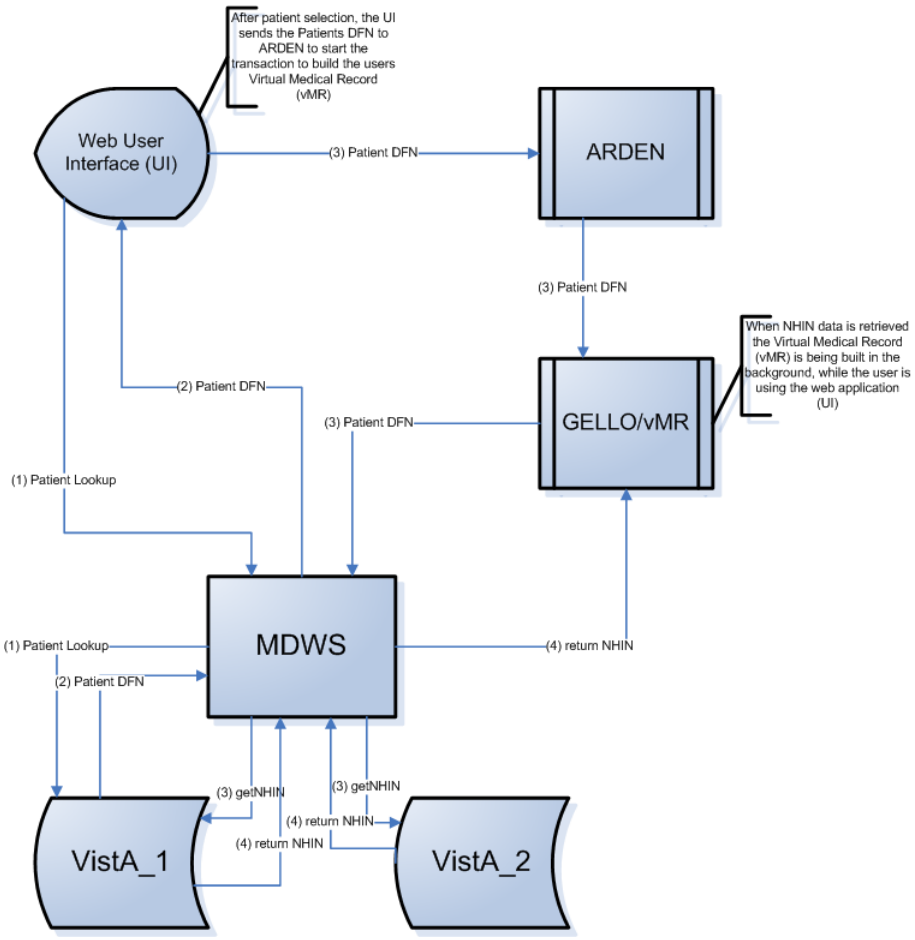



Figure 3: Transaction diagram

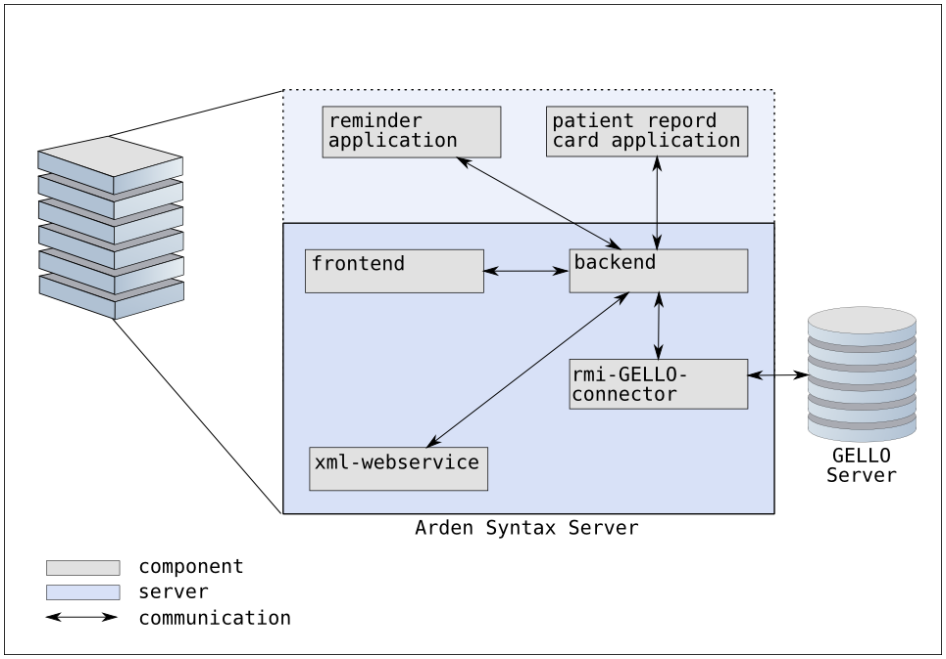



Figure 4: Arden Syntax server internal communication

## 7.2. Integrating CDS in PDMS with Minimal Effort

**Objective**

Integrate an Arden-Syntax-based CDS into an already available patient data management system (PDMS) with minimal effort. Only specific topics should be covered by the CDS (e.g., regular changing score calculations).

**Solution**

- An authoring environment for development of computable knowledge in a shareable, standards-based representation (HL7 Arden Syntax)
- A free-standing, standards-based inference engine operating on standards-based representation of clinical logic (HL7 Arden Syntax)
- An interface to the inferencing engine that can be used from within the already available PDMS
- PDMS capability to use/consume external interfaces/services of the CDS

**Sample implementation**

An Arden Syntax engine equipped with a simple web service interface that is able to receive the identification of an MLM to call and the needed data in simple XML format. Special forms (for each task, e.g., score calculation, parameter check) in the PDMS are defined to collect the necessary data from the user or the internal storage and then sent to the Arden Syntax engine using the web service interface. Returned results from the engine are displayed in the same form.

## 7.3. Integrating CDS in a Commercially Available PDMS Using Proprietary Interfaces

**Objective**

Integrate an Arden-Syntax-based CDS into an already available PDMS or provide an Arden-Syntax-based CDS access to data from an available PDMS.

**Solution**

- An authoring environment for development of computable knowledge in a shareable, standards-based representation (HL7 Arden Syntax)
- A free-standing, standards-based inference engine operating on standards-based representation of clinical logic (HL7 Arden Syntax)
- Implementation of an interface between the data source and the Arden Syntax execution engine
- An interface to the inference engine that can be used from within the already available PDMS
- PDMS capability to use/consume external interfaces/services of the CDS
- Export capability of the PDMS

**Sample implementations**

One implementation of this use case (see Figure 5) was done at the University Hospital of Erlangen, Germany (see [5] for more details). For this implementation, the external interface of

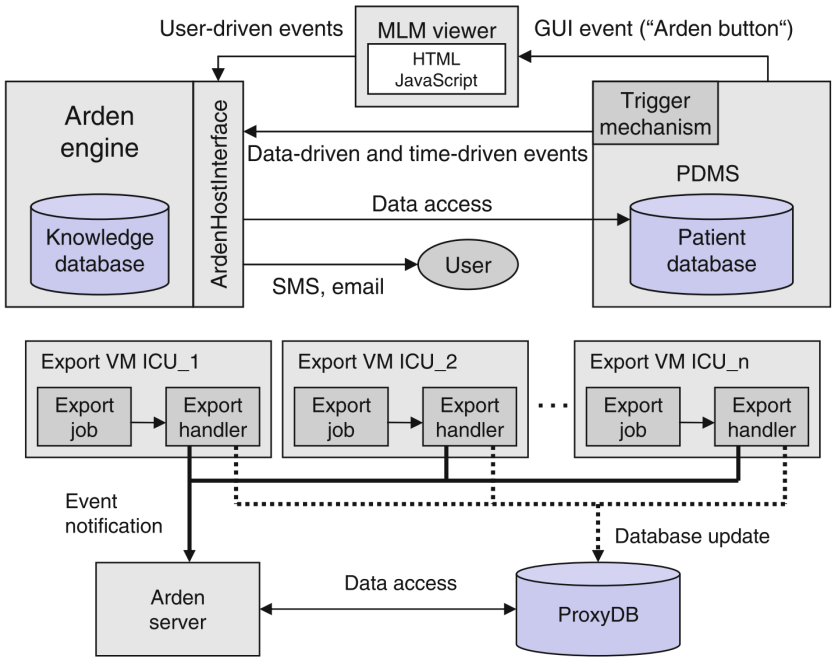



Figure 5

an Arden Syntax execution environment developed by Medexter Healthcare to evaluate curly braces expressions was extended to access data exported from the PDMS.

Another implementation of this use case was done at the University of Colorado Health. At this site, external interfaces of the EPIC hospital information system (HIS) were used to evaluate curly braces expressions. Curly braces expressions are sent through the specifically developed connector to the external interfaces of EPIC. Returning data is converted to execution internal data representation and then provided to the evaluating MLM.

# 8. Clinical Use Cases

## 8.1. Drug–Disease Interaction

Certain medications are contraindicated, must have their doses adjusted or must be monitored in the presence of certain diseases. The presence of at least some diseases may be revealed by laboratory testing. As laboratory test observations have been among the earliest discrete data captured by health information systems, many Arden Syntax MLMs have been written that provide alerts when a drug is ordered in the presence of certain conditions defined by laboratory test observations.

The following MLM demonstrates how to trigger an MLM based on a medication order and to assess then whether a disease (e.g., renal insufficiency) is present based on laboratory testing; alerting the user if this is so and if an alert about this has not been provided recently.

```
maintenance:

 title: Screen for presence of renal insufficiency
          and pharmacy order for an aminoglycoside antibiotic
          (triggered by order for the aminoglycoside) ;;
 filename: AMINOGLYCOSIDE_AND_RF;;
 arden: version 2.9;;
 version: 1.05;;
 institution: The Best Medical Center;;
 author: James Best, MD;;
 specialist: Francine Specialist, MD;;
 date: 2013-02-20;;
 validation: production;;

library:

 purpose:  To determine if a patient has both laboratory evidence
   of renal insufficiency (based on the serum creatinine) and an
   active order for an aminoglycoside antibiotic;;

 explanation:  Aminoglycoside antibiotics, such as gentamicin,
   tobramycin and amikacin, may cause or worsen renal
   insufficiency.  In addition, a typical dose of the antibiotic
   must be adjusted when it is administered to a patient who
   already has renal insufficiency.  This module sends an alert if
   one of this class of antibiotics is ordered for a patient who
   has laboratory evidence of renal insufficiency to help ensure
   that appropriate action (e.g., dosage adjustment) is taken
   if needed. ;;

 keywords: renal failure; aminoglycoside antibiotic;;
 citations: ;;

knowledge:

 type: data-driven;;

 data:
   /* evoke on storage of a pharmacy order */
   storage_of_aminoglycoside_order := event
     {'30343','30345';'30343','30346'};
```

```
   /* read the aminoglycoside order that evoked the MLM */
   aminoglycoside_order := read last
     {'evoking','dam'="PDQORD1",'auxstr'="0013",
      'constraints'="C****",'status_value'="A",
      'display_header'="R",'display_comp'="V"; ; '23946'};

   /* get the last appropriate creatinine */
   raw_creatinines := read last 3 from (
     {'dam'="PDQRES2",'constraints'="C****"; ; '32752'}
     where they occurred within the past 3 months);

   creatinine := last(raw_creatinines where they are number);
   creatinine_boundary := 1.1;  /* upper reference range value */

   /* get the last related alerts */
   last_alert := read last (
     {'dam'="PDQDEC1",'display_header'="TX",'display_comp'="";
      'mlmself','mlm RF_AND_AMINOGLYCOSIDE'}
     where it occurred within the past 3 months);
;;

 evoke:
   /* evoke on storage of a pharmacy order */
   storage_of_aminoglycoside_order;;

 logic:

   if (aminoglycoside_order is null) or
      (creatinine is null) then  /* insufficient data */
     conclude false;
   endif;

   /* avoid redundant alerts */
   if last_alert occurred after 2 weeks before time of
     aminoglycoside_order then
       conclude false;
   endif;

   /* check marker for renal insufficiency */
   if creatinine <= creatinine_boundary then
     conclude false;
   endif;

   /* otherwise creatinine is high and pt is on aminoglycoside */
   conclude true;
;;

 action:

   write "The patient has laboratory evidence of renal " ||
         "insufficiency (serum creatinine of " || creatinine ||
         " mg/dL on " || time of creatinine || ").  Also, an active " ||
         "order for an aminoglycoside antibiotic has been " ||
         "recorded.  Such antibiotics may cause or worsen renal " ||
         "insufficiency, and special dosing may be required.  " ||
         "This may not be applicable to topical preparations.  " ||
         "Appropriate action should be taken as needed.";

;;

end:
```

## 8.2. Body Mass Index

The following MLM allows calculation of the Body Mass Index (BMI):

```
maintenance:
 title:       body mass index;;
 mlmname:     BMI_complex;;
 arden:       Version 2.8;;
 version:     1.00;;
 institution: Medexter Healthcare;;
 author:      ;;
 specialist:  ;;
 date:        2016-12-03;;
 validation:  testing;;
library:
 purpose:     body mass index;;
 explanation: calculation of body mass index
              input: compound list with:
                (number) size in m,
                (number) weight in kg,
                (time)   birth date.
              output: compound list with:
                If the age is not less than 19 and the classification
                wrt.WHO is not normal, a message containing
                the BMI and the classification will be returned.
              The classification follows the definition by the WHO, 2008.
              The interpretation follows the BMI definition. For non-adults
              (age < 19) the definition by Kromeyer-Hauschild is used with the
              3. and the 97. percentile.
 ;;
 keywords:    BMI, body mass index;;
 citations:   ;;
 links:       http://de.wikipedia.org/wiki/Body-Mass-Index;;
knowledge:
 type:            data_driven;;
 data:
   (size, weight, birth) := ARGUMENT;           // input of MLM
   bmiEvent := EVENT {bmiEvent};
 ;;
 priority:    ;;
 evoke:
   bmiEvent;;
 logic:

   // calculation of BMI

   LET bmi BE weight / (size ** 2);             // BMI
   age := CURRENTTIME - birth;                  // age

   // classification wrt. WHO (only for adults)
   IF     age < 19 years THEN classification := NULL;
   ELSEIF bmi < 16       THEN classification := LOCALIZED 'strongunder';
   ELSEIF bmi < 17       THEN classification := LOCALIZED 'modunder';
   ELSEIF bmi < 18,5     THEN classification := LOCALIZED 'slightunder';
   ELSEIF bmi < 25       THEN classification := NULL;
   ELSEIF bmi < 30       THEN classification := LOCALIZED 'obese';
   ELSEIF bmi < 35       THEN classification := LOCALIZED 'obeseI';
   ELSEIF bmi < 40       THEN classification := LOCALIZED 'obeseII';
   ELSE                       classification := LOCALIZED 'obeseIII';
   ENDIF;
```

```
   bmi := bmi FORMATTED WITH LOCALIZED 'msg';  // construct the localized message

   CONCLUDE classification IS PRESENT ;        // if there is a classification, execute the
action slot
 ;;
 action:
   RETURN bmi || classification || ".";         // return result
 ;;
 urgency:     ;;

resources:
 default: de;;
 language: en
   'msg'        : "The patient's BMI %.1f is not in the normal range and is classified as ";
   'strongunder': "severe thinness";
   'modunder'   : "moderate thinness";
   'slightunder': "mild thinness";
   'obese'      : "pre-obese";
   'obeseI'     : "obese class I";
   'obeseII'    : "obese class II";
   'obeseIII'   : "obese class III"
 ;;
 language: de
   'msg'        : "Der BMI %.1f des Patienten ist nicht im normalen Bereich und wird
klassifiziert als ";
   'strongunder': "starkes Untergewicht";
   'modunder'   : "mäßiges Untergewicht";
   'slightunder': "leichtes Untergewicht";
   'obese'      : "Präadipositas (Übergewicht)";
   'obeseI'     : "Adipositas Grad I";
   'obeseII'    : "Adipositas Grad II";
   'obeseIII'   : "Adipositas Grad III"
 ;;
end:
```

## 8.3. Abnormal Test Result Detection

MLMs can be used to alert clinicians to potentially serious test results that may require follow-up. These may be numeric laboratory observations or narrative ones, or coded results associated with a report. The following MLM demonstrates how diagnostic codes associated with the report of the analysis of a Papanicolaou smear can be detected and an alert sent.

```
maintenance:

  title:  Abnormal PAP smears and cervical biopsies.;;
  filename:  pap_monitoring;;
  arden: Version 2.8;;
  version:  1.02;;
  institution:  Best Medical Center;;
  author:  Robert Jones, M.D.;;
  specialist:  Michael Smith, M.D;;
  date:  2014-01-27;;
  validation:  production;;

library:

  purpose:  Screen/follow up patients with abnormal PAP smears and cervical biopsies;;
```

```
  explanation:  Patients who have atypical cells, suspicious cells or malignant cells present in
their PAP smear need a follow up PAP smear or cervical biopsy within 3 months.  Patients who have
abnormal findings on cervical biopsies need follow up within 3 months.;;

  keywords: Papanicolau smear, cervical biopsy, cytology ;;

  citations:  ;;


knowledge:

  type:  data-driven;;

  data:
       /* evoking record  = all pathology reports */
       pathology_upload := EVENT {'32506','32688'};

       /* patient name  */
       name := read last
              {'dam'="GYDAPMP"; "HPBASIC"; "HNAME"};

       /* patient mrn */
       mrn := read last
       {'pcodes'="null mrn "};

       /* patient address - phone number */
       (address, state, city, zip, phone) := read last
       {'dam'="GYDAPMP"; "HADDRESS"; "HADDADDR"; "HSTATE"; "HADDCITY";
              "HZIP"; "HPHONNO" };

       /* get diagnoses from evoking report */
       path_codes := READ
       {'evoking','dam'="GYDATXP"; "PATH"; "PTECH/CLERK"};

       /* get details re: evoking report */
       path_accno := read last
       {'pcodes'="now event_db_key "};

       /* email for research log */
       email_dest := destination
       {'email', 'name'="sidelir@cucis.cis.columbia.edu"};
       ;;

  evoke:  pathology_upload;;

  logic:
       if any(path_codes = "AG9,1")
              then
       Specimen := "Papanicolau Smear";
          Path_DX := "Atypical cells present";
          conclude true;
       elseif any(path_codes = "SG9,1")
              then
       Specimen := "Papanicolau Smear";
          Path_DX := "Suspicious cells present";
          conclude true;
       elseif any(path_codes = "PSG9,1")
       then
       Specimen := "Papanicolau Smear";
          Path_DX := "Positive for malignant cells";
          conclude true;
       elseif any(path_codes = "C19,1")
```

```
            then
        Specimen := "Papanicolau Smear";
           Path_DX := "CIN 1";
           conclude true;
        elseif any(path_codes = "C29,1")
            then
        Specimen := "Papanicolau Smear";
           Path_DX := "CIN 2";
           conclude true;
        elseif any(path_codes = "C39,1")
            then
        Specimen := "Papanicolau Smear";
           Path_DX := "CIN 3";
           conclude true;
        else;
        endif;
        ;;

  action:
        write   "\n            MRN: " || mrn ||
        "\n Patient Name: " || name ||
        "\n     Address: " || address ||
        "\n            " || city || ", " || state || "  " || zip ||
        "\n     Phone: " || phone ||
        "\n" ||
        "\n   Path Acc #: " || path_accno ||
        "\n            Date: " || last(time of path_codes) ||
        "\n     Specimen: " || Specimen ||
        "\n     Path_DX: " || Path_DX
        at email_dest; // the "\n" encodes a line break

        ;;

  urgency:  50;;

end:
```

## 8.4. Detection of Possible Patient Deterioration

The ability of MLMs to quickly identify abnormal test results (Section 8.3 Abnormal Test Result Detection) can be extended to detect patterns of results and vital signs to give an “early warning” that a patient’s condition is deteriorating. This Modified Early Warning Score (MEWS) alert evaluates eight different clinical parameters, calculates an aggregate MEWS score, and if elevated, immediately notifies the clinician and/or “Rapid Response” team.

```
MAINTENANCE:
  title:       MEWS Alert;;
  mlmname:     MEWS_Alert;;
  arden:       Version 2;;
  version:     1.40;;
  institution: InSight;;
  author:      ;;
  specialist:  ;;
  date: 2011-08-04;;
  validation:  Testing;;

LIBRARY:
  purpose:     This rule checks vital signs and identifies patients at imminent risk for
deterioration.;;
```

explanation: This rule was created by a group of InSight HCA SIG members based on Early Warning Score literature (MEWS and ViEWS -- see references below. The rule detects abnormal vital signs, calculates a weighted aggregate "MEWS Score" and notifies caregivers when a patient is at risk of deterioration. The appropriate criteria to use (mews_threshold) is hospital-specific, and is based on the number of vital signs being checked - typical threshold values are 4 to 6. Refer to the articles below, or Google 'Modified Early Warning System' (MEWS) for more information.

Original MEWS criteria (see citation 1)

| | 3 | 2 | 1 | 0 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|---|
| Pulse rate (bpm) | | < 40 | 40-50 | 51-100 | 101-110 | 111-129 | > 129 |
| Respiratory rate | | < 9 | | 9-14 | 15-20 | 21-29 | >= 30 |
| Temperature C | | < 35.1 | 35.1-36 | 36.1—38 | 38.1-38.5 | >38.5 | |
| Systolic BP | < 70 | 71-80 | 81-100 | 101-199 | | >= 200 | |
| Neuro score (AVPU) | | | | Alert | voice | pain | Unresponsive |

Additional ViEWS criteria (see citation 2):

| | | | | | |
|---|---|---|---|---|---|
| O2 Saturation | <=91 | 91.1-93 | 93.1-95 | >95 | |
| Supplemental O2 | | | Room Air | | Any O2 |

;;

keywords: MEWS ;;
citations: 1. Subbe CP, Davies RG, Williams E et al: Effect of introducing the Modified Early Warning score on clinical outcomes, cardio-pulmonary arrest and intensive care utilization in acute medical admissions. Anaesthesia. 2003; 58:775-803.

2. Prythercha DR, Smith GB, Paul E. Schmidt PE, Peter I. Featherstone PI: ViEWS -- Towards a national early warning score for detecting adult inpatient deterioration. Resuscitation. 2010; 81:932-937.

;;
links: ;;

KNOWLEDGE:
type: data_driven;;

data:
```
// Enter the appropriate Event here
triggering_event := EVENT {A Charting Result Entered:Pulse};

// This is the timeframe used for recent results.  Any result older than this will
// be disregarded
result_interval := 4 hours;

// set to null to disable check for previous notifications
time_between_alerts := 4 hours;


// Read mappings for results
(HeartRateText):= READ last {A Text Charting Result:Pulse}
        where it occurred after result_interval AGO;
(RespRateText):= READ last {A Text Charting Result:Respirations}
        where it occurred after result_interval AGO;
(TemperatureText):= READ last {A Text Charting Result:Temperature}
        where it occurred after result_interval AGO;
(BPText):= READ last {A Text Charting Result:Blood Pressure}
        where it occurred after result_interval AGO;
(O2SatText):= READ last {A Text Charting Result:O2Sat}
        where it occurred after result_interval AGO;
//  Indicates that patient is receiving O2 Therapy.  Use EITHER a charted result or an active order (not both)
(O2Admin):= READ last {A Text Charting Result: O2Admin}
        where it occurred after result_interval AGO;
```

```
        // (O2Admin):= READ last {An Active Order:Supplemental O2};

        (LOCText):= READ last {A Text Charting Result:LevelofConciousness}
                where it occurred after result_interval AGO;

        /*Read mappings for patient data.*/
                (patSex, patLastName, patFirstName, patAcct, patMedRec) := READ last {Person
Info};
        (patFacilityID, patDeptID, patRoomID,patBedID) := READ last {Patient Location};
        (patTypeGroup, patService, patClass, patType) := READ last {Patient Type Information};

        // Previous notifications
        previous_notification:= READ last{Previous Notifications:MEWS Alert}
                where it occurred after time_between_alerts AGO;

        //Destinatiions
        admin_email := DESTINATION{Email:Example Email Address};

        //  ************************************
        //Revise these variables as needed
        //  ************************************
        // Designate the specific patient type codes and depts to INCLUDE when processing.  If
        //null, all types/depts will be included
        include_patType :=( "I/P", "IP", "OBS" , "OIB") ;
        include_depts := ("V5W","V6W","V6E","V7E","V7W","V8E","V8W","V9E","V9W");




        // The results you want to check must be set to true
        check_pulse            := true;
        check_respiration      := true;
        check_temp             := true;
        check_SysBP            := true;
        check_map              := false;  // Mean Arterial Pressure
        check_LOC              := false;  // Level of consciousness (AVPU score)
        check_O2sat            := true;
        check_O2_admin         := true;

        // MEWS score threshold.  A score equal to or greater than this will generate an alert.
        //  This is normally 4 to 6.  Set it to zero to force the rule to ALWAYS conclude true.
        mews_threshold := 5;

        // ****************************************
        // Initialized variables -- You should NOT have to change any of these
        mews_score := 0;
        msg := "";
        detail_msg := "";
        mews_msg := "";
        na := "N/A" formatted with  "%12s ";    // displays if result is not available
        NL := "

";

        ;;

  evoke:       triggering_event;
               ;;


 logic:
        // Add your normal validation checks here.  These are examples
```

```
// Check for previous notifications.
IF time_between_alerts is present THEN
        IF previous_notification is present THEN
                CONCLUDE FALSE;
        ENDIF;
ENDIF;

// Check the patient type against the INCLUDE_patType list (if defined).
IF include_patType is present THEN
        IF patType is NOT in include_patType THEN
                CONCLUDE FALSE;
        ENDIF;
ENDIF;

// Check the patient's department against the INCLUDE_depts_list list (if defined).
IF include_depts is present THEN
        IF patDeptID is NOT in include_depts THEN
                CONCLUDE FALSE;
        ENDIF;
ENDIF;


    //  End of validation.  Start checking results now
 IF check_pulse THEN
  //   PULES/HEARTRATE PLUGIN
  // Variables with a "_f " suffix are formatted for output.
  result_name := "Pulse: ";
  result_name_f := result_name formatted with "%-22s";
  result_score := 0;
  abnormal_flag := " ";

  IF HeartRateText is present THEN
        result_text := HeartRateText;
        result_time := time of HeartRateText;
        heartRate:= HeartRateText AS NUMBER;

        IF heartRate < 40 THEN
                result_score := 2 ;
        ELSEIF heartRate <= 50 THEN
                result_score := 1 ;
        ELSEIF heartRate <= 100 THEN
                result_score := 0 ;
        ELSEIF heartRate <= 110 THEN
                result_score := 1 ;
        ELSEIF heartRate <= 129 THEN
                result_score := 2 ;
        ELSEIF heartRate > 129 THEN
                result_score := 3 ;
        ENDIF ;

        IF result_score > 0 THEN
                mews_score := mews_score + result_score;
                abnormal_total := abnormal_total + 1;
                abnormal_flag := "*";
        ENDIF;

        // Build the default, detail, and MEWS messaqes
        msg := msg || result_name || result_text ||", ";

        result_text_f := result_text formatted with "%12s ";
        detail_msg := detail_msg || result_name_f || result_text_f || abnormal_flag
```

```
                || "  " || result_time || NL;

        mews_integer_f := result_score formatted with "%5.f";
        mews_msg := mews_msg || result_name_f || result_text_f || abnormal_flag
                || "  " || result_time || mews_integer_f || NL;

    ELSE
        // The result is not present -- just add N/A to the messages
        msg := msg || result_name || "N/A, ";
        detail_msg := detail_msg || result_name_f || na || NL;
        mews_msg   :=  mews_msg  || result_name_f || na || NL;
    ENDIF;
  //  End of Pules/HeartRate plugin
ENDIF;

IF check_respiration  THEN
  //  RESP RATE PLUGIN
  // Variables with a "_f " suffix are formatted for output.
  result_name := "Respiration: ";
  result_name_f := result_name formatted with "%-22s";
  result_score := 0;
  abnormal_flag := " ";

  IF respRateText is present THEN
        result_text := respRateText;
        result_time := time of respRateText;
        respRate:= respRateText AS NUMBER;

        // Calculate MEWS score
        if respRate >= 30 then
                result_score := 3;
        elseif respRate >= 21 then
                result_score := 2;
        elseif respRate >= 15 then
                result_score := 1;
        elseif respRate >= 9 then
                result_score := 0;
        else
                result_score := 2;
        endif;

        if result_score > 0 then
                mews_score := mews_score + result_score;
                abnormal_total := abnormal_total + 1;
                abnormal_flag := "*";
        endif;

        // Build the default, detail, and MEWS messaqes
        msg := msg || result_name || result_text ||", ";

        result_text_f := result_text formatted with "%12s ";
        detail_msg := detail_msg || result_name_f || result_text_f || abnormal_flag
                || "  " || result_time || NL;

        mews_integer_f := result_score formatted with "%5.f";
        mews_msg := mews_msg || result_name_f || result_text_f || abnormal_flag
                || "  " || result_time || mews_integer_f || NL;

  ELSE
        // The result is not present -- just add N/A to the messages
        msg := msg || result_name || "N/A, ";
        detail_msg := detail_msg || result_name_f || na || NL;
```

```
            mews_msg   :=  mews_msg  || result_name_f || na || NL;
       ENDIF;
       //  End of Resp Rate plugin
      ENDIF;
```

Additional “plugins” for temperature, systolic blood pressure (BP), mean arterial pressure (MAP), level of consciousness, oxygen saturation, and O2 administration have been omitted here for brevity.

```
       // ***********************************************
       // All results have been evaluated.  Here is the standard end of the rule

       IF mews_score >=  mews_threshold THEN

              // Replace this with your normal patient header logic
              patient_header := "Patient:  " || patLastName || ", " || patFirstName || NL
                           ||  "Facility:  " || patFacilityID || NL
                           ||  "Location:  " || patRoomID || NL
                           ||  "Account Number:  " || patAcct || NL || NL ;

              mews_score_f := mews_score formatted with "%2.f";
              msg_summary := "This patient has a MEWS score of " || mews_score_f || " based on
these results:  " || NL;

              mews_report_header := "                                          MEWS" || NL ||
                     "Result                      Value    Chart Time       Score"|| NL ||
                     "---------------           --------    --------------   -----"|| NL;

              mews_report_footer :=  "                                          -----"|| NL ||
                                  " Total                                  "|| mews_score_f;

              msg := msg_summary || msg;
              detailed_mews_msg := patient_header || msg_summary || NL || mews_report_header ||
                            mews_msg || mews_report_footer;

              CONCLUDE TRUE;
       ENDIF;
       ;;

 action:

       WRITE msg;

       WRITE detailed_mews_msg || "^MEWS Alert" AT admin_email;

       ;;

  urgency:     75;;

END:
```

## 8.5. Least Squares Linear Regression

This MLM demonstrates a simple statistical test using the Arden Syntax.

```
MAINTENANCE:
Title:         Least Squares Linear Regression ;;
Filename:      Least_Squares_Linear_Regression ;;
arden:         Version 2.9;;
```

```
Version:       1.02 ;;
Institution:   Medical Center ;;
Author:        Michael A. Jones, Pharm.D. ;;
Specialist: ;;
Date:          2000-07-12 ;;
Validation:    testing ;;

LIBRARY:
Purpose:       Fits a straight line to pairs of coordinates using the method of least squares.
                      ;;

Explanation:   Estimates the slope and y-intercept of the line.  Calculates coefficient of
               determination (r-squared), the coefficient of correlation, and the standard
               error of estimate.

       Input:         Required (In order for the rule to run) :
                      Last two pairs of coordinates input as argument 1 as (x1, x2, ...).
                      Where X's are numbers with a primary time for each X.

       Output:        Returns a list containing:
                      y-intercept, slope, coefficient of determination,
                      coefficient of correlation, and the standard error of estimate.
 ;;

Keywords:   ;;

Citations:
1.     Linear Regression, in Some Common Pascal Programs.
       Osborne/McGraw-Hill, Berkeley, CA 1982.

2.     Kuzma JW: Basic Statistics for the Health Sciences.
       Mayfield Publishing Company, Palo Altom CA 1984.

3.     Press WH, Flannery BP, Teukolsky SA, Vetterling WT: Numerical Recipes in Pascal.
       Cambridge University Press, New York, NY 1996.

4.     draper NR, Smith H: Applied regression Analysis, 3rd ed.
       John Wiley & Sons, Inc. New York, NY 1998.
;;

KNOWLEDGE:
Type:  data_driven;;

Data:
       /* Initialize variables */
       error_occurred := false;
       msg := "";
       y_intercept := 0;
       slope_of_line := 0;
       coefficient_of_determination := 0;
       coefficient_of_correlation := 0;
       standard_error_of_estimate := 0;

       Data_Set := Argument;  /*     Data type: LIST */
              /*
              Data_Set is a list of data pairs in the form of (x1, x2, ...);
              The Data_Set must contain at least two pairs of data.
              The Y datum is the primary time associated with each x datum.
              */

       CALLed_from_another_MLM := EVENT { CALLed from another MLM };
```

```
      ;;

Evoke:  CALLed_from_another_MLM ;;

Logic:

        // Error Trapping
        if (Data_Set is not present) or (Data_Set is not list) then
        msg := "Error in Data_Set (argument 1)! ";
        error_occurred := true;
        conclude true;
        endif;

        if (Data_Set[1] is not present) or (Data_Set[1] is not number) then
        msg := "Error in Data_Set: Element 1 is not a number!";
        error_occurred := true;
        conclude true;
        endif;

        if (Data_Set[2] is not present) or (Data_Set[2] is not number) then
        msg := "Error in Data_Set: Not enough data! ";
        error_occurred := true;
        conclude true;
        endif;

SumX := (now - now)/1 day;
SumY := 0;
SumXsqr := (now - now)/1 day;
SumYsqr := 0;
SumXtimesY := (now - now)/1 day;

i := 1;
while i <= count of Data_Set do
        DataPoint := Data_Set[i];
        SumY := SumY + DataPoint;
        DS_days := (now - time of Data_Set[i])/1 day;
        SumX := SumX + DS_days;
        SumYsqr := SumYsqr + (DataPoint * DataPoint);
        SumXsqr := SumXsqr + (DS_days * DS_days);
        SumXtimesY := SumXtimesY + (DataPoint * DS_days);
        i := i +1;
        /* prevent runaway loop */
        If i > 100 then
                breakloop; // execution of the while-loop will stop immediately
        endif;
enddo;

Num_of_pairs := count of Data_Set;

slope_of_line := (Num_of_pairs * SumXtimesY - SumY * SumX) / (Num_of_pairs * SumXsqr - SumX *
SumX);

y_intercept := (SumY - slope_of_line * SumX) / Num_of_pairs;

X := slope_of_line * (SumXtimesY - SumX * SumY / Num_of_pairs);

coefficient_of_determination := X / (SumYsqr - (SumY * SumY) / Num_of_pairs);

coefficient_of_correlation := sqrt(abs(coefficient_of_determination));

StandErrorNumerator := SumYsqr - SumY * SumY / Num_of_pairs - X;
```

```
standard_error_of_estimate := sqrt(StandErrorNumerator/(Num_of_pairs - 2));

Results_List := (y_intercept, slope_of_line, coefficient_of_determination,
                      coefficient_of_correlation, standard_error_of_estimate);

conclude true;
;;

Action:
if error_occurred then
       write "Rule Least_Squares_Linear_Regression contained an error on : " || now || msg;
       return "error";
else
       return Results_List; // Send results to the calling MLM
endif;

;;
END:
```

## 8.6. Calculate Creatinine Clearance (CrCl) and Glomerular Filtration Rate (GFR)

Example of documentation for long-term of maintenance of MLM, and use of branching logic based on patient characteristics.

```
MAINTENANCE:
  title:              Calculate_CrCl_and_GFR;;
  mlmname:            Calculate_CrCl_and_GFR;;
  arden:              Version 2.8 ;;
  version:            3.03;;
  institution:        University Hospital ;;
  author:             Michael A. Jones, Pharm.D. ;;
  specialist:  ;;
  date:               2008-12-08 ;;
  validation:         Production ;;

LIBRARY:
  purpose:            Calc CrCl or GFR ;;
  explanation:        Uses Cockcroft & Gault method if patient weight is available.
                      If patient weight is not available uses Abbreviated MDRD Study equation.

       Version 2
       4/2/09  maj    X      Modified MDRD to IDMS-Traceable MDRD Study Equation
to account for changes in lab reporting of Serum Creatinine results.
       6/22/09 maj           Update logic, search on 6/22/09 below.
       6/25/09 maj           more updating, search on 6/25/09.
       7/22/09 maj           Height - added conversion for cm to inches.
       7/22/09 maj           Height - excluded Ht < 58 inches for IBW calculation.
       4/01/11 maj    X      Add calc to estimate old SCr value from IDMS-Traceable SCr value.
                             f(x) = (y-b)/m,
                             where x = old SCr value
                                   y = IDMS-Traceable SCr value
                                   m = 0.986925
                                   b = -0.07


       ;;
  keywords:    ;;
  citations:
       1. Devine BJ. Gentamicin therapy. Drug Intell Clin Pharm. 1974; 7:650-5.
```

2. Cockcroft DW, Gault MH. Prediction of creatinine clearance from serum creatinine. Nephron. 1976; 16:31-41.

3. Levy AS, Coresh J, Balk E, et al. National Kidney Foundation practice guidelines for chronic kidney disease: evaluation, classification and stratification. Ann Intern Med. 2003; 139:137-47.

4. Spruill WJ, Wade WE, Cobb HH. Estimating glomerular filtration rate with a modification of diet in renal disease equation: implications for pharmacy. Am J Health-Syst Pharm. 2007; 64:652-60.

5. Moranville MP, Jennings HR. Implications of using modification of diet in renal disease versus Cockcroft?Gault equations for renal dosing adjustments. Am J Health-Syst Pharm. 2009; 66:154-61.

6. Stevens LA, Nolin TD, Richardson MM, et al. Comparison of Drug Dosing Recommendations Based on Measured GFR and Kidney Function Estimating Equations. Am J Kidney Dis xx:xxx. 2009 by the National Kidney Foundation, Inc.

7. Levey AS, Stevens LA, Schmid CH, et al. A New Equation to Estimate Glomerular Filtration Rate. Ann Intern Med. 2009;150:604-612.

8. Jones MA, Golightly LK, Stolpmen N: Use of Recalibrated Serum Creatinine Concentrations for Adjustment of Drug Dosages: Determination of Values Compatible with Conventional Dosing Recommendations. Ann Pharmacother 2011;45:748-756.

```
;;
  links:       ;;

KNOWLEDGE:
  type:  data-driven;;

  data: (age, height, Adm_weight, Last_daily_Wt, Gender, Last_SCr) := ARGUMENT;

        HemoDialysis := READ last {A Charting Result:Hemodialysis Output} ;  // 12/10/08 maj

        Wt_available := FALSE;  // changed both to false on 6/25/09 maj
        Ht_available := FALSE;
        msg := " ";

        /* Variable for new line to use in formatting text. NL: ="<space><enter><enter>";*/
        NL := "

";
        ;;

evoke:  /* triggering_event - Called by another MLM. Subroutines do not have a
                     triggering event */
;;

  logic:

/* estimate Adjusted_SCr value from IDMS-Traceable SCr value. see Ref 8.
        f(x) = (y-b)/m,
        where x = old SCr value
              y = IDMS-Traceable SCr value
              m = 0.986925
              b = -0.07
*/
        m := 0.986925;
        b := -0.07;
        Adjusted_SCr := (Last_SCr - b)/m;
```

```
/* Creatinine Clearance */

        if (HemoDialysis is present) then
                CrCl := "error" ;
                msg := "Patient is on HemoDialysis, estimates of CrCl are not accurate!"  ;
                CONCLUDE TRUE;
        endif;

        if height > 100 then  // convert from cm to inches - added 7/22/09 maj
                height := height / 2.54 ;
        endif;

        // added 6/25/09 maj;
        // added check for height >= 58 inches for IBW calculation 7/22/09 maj
        if ((height is present) and (height is number) and (height >= 58)) then
                Ht_available := TRUE;
        else
                Ht_available := FALSE;
        endif;


        // added 6/25/09 maj
        if ((Last_daily_Wt is present) and (Last_daily_Wt is number))  then
                weight := Last_daily_Wt;
                Wt_available := TRUE;
        elseif ((Adm_weight is present) and (Adm_weight is number)) then
                weight := Adm_weight;
                Wt_available := TRUE;
        else
                Wt_available := FALSE;
        endif;


        if (age >= 18 years) then
                if Ht_available then
                        //  Calc the Ideal Body Weight: Devine method (ref. 1).
                        if (Gender = "M") then
                                IBW := 50.0 + (2.3 * (height - 60));
                                else
                                IBW := 45.5 + (2.3 * (height - 60));
                                endif;

                        // Get the Adjusted Body Weight
                        if Wt_available then
                                if (weight < IBW) then
                                        ABW := weight;
                                else
                                        ABW := IBW;
                                endif;
                        else
                                ABW := IBW;
                                msg := msg || "No Wt available, used IBW of " || IBW || " kg." ||
       NL ;
                        endif;

                elseif ((Last_daily_Wt is present) and (Last_daily_Wt is number)) then
                        ABW := Last_daily_Wt;
                        msg := msg || "Ht not available, used last daily weight of " ||
Last_daily_Wt || " kg " || NL;

                elseif ((Adm_weight is present) and (Adm_weight is number)) then
```

```
                        ABW := Adm_weight;
                        msg := msg || "Ht not available, used Adm weight of " || Adm_weight || "
kg " || NL;
                else
                        msg := msg || "Height & Weight are not available, used CKD-EPI method" ||
NL;
                endif;

                if ((age > 60 years) and (Last_SCr < 1)) then
                        Last_SCr := 1;
                        Adjusted_SCr := 1; // added 4/27/09 1500 maj
                        msg := msg || "Age > 60 yrs, used SCr of " || Last_SCr || " mg/dL for CrCl
calculation." || NL ;
                endif;

                if Ht_available then
                        // calculate the creatinine clearance: Cockcroft & Gault method (ref. 2).
                        CrCl_method := "CrCl_method: Cockcroft & Gault using IBW";
                        if (Gender = "M") then
                                crcl := ((140 - (age/1 year)) * ABW) / (72 * Adjusted_SCr) ;
                        else
                                crcl := 0.85 * ((140 - (age/1 year)) * ABW) / (72 * Adjusted_SCr) ;
                        endif;
                        RenalFunction := crcl; // added 6/22/09 maj

                // added 6/22/09 maj
                elseif Wt_available then
                        // calculate the creatinine clearance: Cockcroft & Gault method (ref. 2).
                        CrCl_method := "CrCl_method: Cockcroft & Gault using actual weight";
                        if (Gender = "M") then
                                crcl := ((140 - (age/1 year)) * ABW) / (72 * Adjusted_SCr) ;
                        else
                                crcl := 0.85 * ((140 - (age/1 year)) * ABW) / (72 * Adjusted_SCr) ;
                        endif;
                        RenalFunction := crcl;
                else
                        /*
                        Calc Abbreviated MDRD Study equation: GFR in mL/min/1.73 m^2
                        There is some controversy about this equation with no weight (ref. 3 & 4),
But it is better than nothing. Will continue to watch the literature for
alternatives. Since wt is not available, assume BSA = 1.73 m^2, then GFR approximates CrCl

                        Added 4/2/09 maj  CHANGED factor value 186 to 175.
                        IDMS-Traceable MDRD Study Equation1
                        NOTE: This equation should be used only with those creatinine methods that
have been recalibrated to be traceable to IDMS. For more information about
recalibration, visit NKDEP's Laboratory Professionals section.

GFR (mL/min/1.73 m2) = 175 x (Scr)-1.154 x (Age)-0.203 x (0.742 if female) x (1.212 if African
American) (conventional units)

The equation does not require weight because the results are reported normalized to 1.73 m2 body
surface area, which is an accepted average adult surface area.
                        */
                        /* turned off 6/22/09 maj
                        CrCl_method := "GFR_method: used Abbreviated MDRD ";

                        crcl := (175 * (Last_SCr ** (-1.154))) * ((age/52 weeks)**(-0.203)) ;

                        IF (Gender = "F") then
                                crcl := crcl * 0.742 ;
                        ENDIF;
```

```
                        */
                        // IF African American then  crcl := crcl * 1.21 - Ethnic group is not
   available.

                        // CKD-EPI method for replaced MDRD method 6/22/09 maj
                        // Can not use African American equations as Ethnic group is not
   available.
CrCl_method := "GFR_method: 2009 CKD-EPI (no adjustment for African Americans at this time)";

                        age := (age/1 year) ;

                        if Gender = "F" then
                                if Last_SCr <= 0.7 then
                                        GFR := (144 * ((Last_SCr/0.7) ** (-0.329))) *
       ((0.993)**(age)) ;
                                elseif Last_SCr > 0.7 then
                                        GFR := (144 * ((Last_SCr/0.7) ** (-1.209))) *
       ((0.993)**(age)) ;
                                endif;
                        endif;

                        if Gender = "M" then
                                if Last_SCr <= 0.9 then
                                        GFR := (141 * ((Last_SCr/0.9) ** (-0.411))) *
       ((0.993)**(age)) ;
                                elseif Last_SCr > 0.9 then
                                        GFR := (141 * ((Last_SCr/0.9) ** (-1.209))) *
       ((0.993)**(age)) ;
                                endif;
                        endif;

                        RenalFunction := GFR;

                ENDIF;
                msg := msg || CrCl_method || NL;

        ENDIF;

        Result_List := (RenalFunction, msg);

        CONCLUDE TRUE;
                ;;
  action:
        RETURN Result_List;
                ;;
  urgency:      50;;
  END:
```

## 8.7. Hepatitis A Immunization

The Centers for Disease Control and Prevention (CDC) routinely uses advisory councils to produce recommendations for immunization. The following MLM captures the recommendations for Hepatitis A immunization from the 2006 recommendations. It is based on Arden Syntax, version 2.8.

```
maintenance:
        title: Hepatitis A Immunization MLM;;
        mlmname: Hep_A_Immunization;;
        arden: Version 2.9;;
```

```
	version: 1.00;;
	institution: Intermountain Health Care;;
	author: Peter Haug, M.D.;;
	specialist: Peter Haug M.D.;;
	date: 2011-09-10;;
	validation: testing;;
library:
	purpose: Suggest initial dose of Hepatitis A vaccine schedule.;;
	explanation: Hepatitis Vaccination Advise under different circumstances;;
	keywords: Hepatitis A; vaccine; immunization;;
	citations: Centers for Disease Control and Prevention. Prevention of Hepatitis A through
active or passive immunization: recommendations of the Advisory Committee on Immunization
Practices (ACIP). MMWR 2006;55(No. RR-7);;
	links: to be added;;
knowledge:
type: data_driven;;
data:
	/* text within curly brackets would be replaced with an institution's own query */

	Let PatientIDRecord BE OBJECT [AccountNum, Name, Birthdate, Sex];
	LET Patient BE Read As PatientIDRecord Latest {select accountnum, name, dateofbirth, sex
from EHR} ;

	LET outpatient_visit BE EVENT {select encounter_type, start_time_stamp, end_time_stamp,
attending, location from INPUT_BUFFER};

	LET Medication BE OBJECT [Drug_Name, Form, Dosage, Route, Schedule];
	LET VAQTA BE Read As Medication {select med, form, dose, route, schedule from EHR where
med = 'VAQTA'} ;
LET HAVRIX BE Read As Medication {select med, form, dose, route, schedule from EHR where med =
'HAVRIX'} ;
LET TWINRIX BE Read As Medication {select med, form, dose, route, schedule from EHR where med =
'TWINRIX'} ;

	Let GenericMessage BE OBJECT [MessageID, MessageCode, Message_Text];
	Let Vaccine_Message BE New GenericMessage;
	;;

evoke: outpatient_visit;;

logic:
	/* Check age */
	if
		Patient_Age Is Null
	then
		Patient_Age := Now - Patient.Birthdate;
	endif;

	if
		Patient_Age is Less Than 12 months
	then
		Conclude False ;
	endif;

	/* Young Patient */
	if
		Patient_Age is Greater Than 12 months and
		Patient_Age is Less Than or Equal 18 years
	then
		Apply_young_patient_rules := true;
	endif;
```

```
	/* Older Patient */
	if
		Patient_Age is Greater Than 18 years
	then
		Apply_older_patient_rules := true;
	endif;

	/* Rules-no previous vaccine. */
	if
		Exist(VAQTA) or Exist(HAVRIX) or Exist(TWINRIX)
	then
		Hepatitis_vaccination:= true;
	endif;

	if
		Apply_young_patient_rules and
		Not Hepatitis_vaccination
	then
		Vaccine_Message.Message_Text := "Give VAQTA vaccine, 25U, IM now and repeat in 6
to 18 months.";
		Conclude true;
	endif;

	if
		Apply_older_patient_rules and
		Not Hepatitis_vaccination
	then
		Vaccine_Message.Message_Text := "Give VAQTA vaccine, 50U, IM now and repeat in 6
to 18 months.";
		Conclude true;
	endif;

	/* Rules-one previous vaccination. */
	if
		All(VAQTA Is Null)
	then
		Count_VAQTA:=0;
	endif;

	if
		Any(VAQTA Is Present)
	then
		VAQTA := VAQTA Where it Is Present;
		Count_VAQTA:=Count(VAQTA);
	endif;

	if
		All(HAVRIX Is Null)
	then
		Count_HAVRIX:=0;
	endif;

	if
		Any(HAVRIX Is Present)
	then
		HAVRIX := HAVRIX Where it Is Present;
		Count_HAVRIX:=Count(HAVRIX);
	endif;

	if
		All(TWINRIX Is Null)
	then
```

```
		Count_TWINRIX:=0;
	endif;

	if
		Any(TWINRIX Is Present)
	then
		TWINRIX := TWINRIX Where it Is Present;
		Count_TWINRIX:=Count(TWINRIX);
	endif;

	if
		Exist Count_VAQTA and
		Exist Count_HAVRIX and
		Exist Count_TWINRIX and
		Count_VAQTA + Count_HAVRIX + Count_TWINRIX = 1
	then
		Needs_second_dose := true;
	endif;

	/* Second dose young people */
	if
		Apply_young_patient_rules and
		Needs_second_dose and
		Exist(VAQTA) and
		Latest VAQTA Occurred Before 6 months Ago
	then
Vaccine_Message.Message_Text := "Give VAQTA vaccine, 25U, IM now. This completes Hepatitis A
vaccinations.";
		Conclude true;
	endif;

	if
		Apply_young_patient_rules and
		Needs_second_dose and
		Exist(HAVRIX) and
		Latest HAVRIX Occurred Before 6 months Ago
	then
		Vaccine_Message.Message_Text := "Give HAVRIX vaccine, 25U, IM now. This completes
Hepatitis A vaccinations.";
		Conclude true;
	endif;

	if
		Apply_young_patient_rules and
		Needs_second_dose and
		Exist(TWINRIX) and
		Latest TWINRIX Occurred Before 6 months Ago
	then
		Vaccine_Message.Message_Text := "Give TWINRIX vaccine, 25U, IM now. This completes
	Hepatitis A vaccinations.";
		Conclude true;
	endif;

	/* Second dose Older people */
	if
		Apply_older_patient_rules and
		Needs_second_dose and
		Exist(VAQTA) and
		Latest VAQTA Occurred Before 6 months Ago
	then
		Vaccine_Message.Message_Text := "Give VAQTA vaccine, 50U, IM now. This completes
Hepatitis A vaccinations.";
```

```
		Conclude true;
	endif;

	if
		Apply_older_patient_rules and
		Needs_second_dose and
		Exist(HAVRIX) and
		Latest HAVRIX Occurred Before 6 months Ago
	then
		Vaccine_Message.Message_Text := "Give HAVRIX vaccine, 50U, IM now. This completes
Hepatitis A vaccinations.";
		Conclude true;
	endif;

	if
		Apply_older_patient_rules and
		Needs_second_dose and
		Exist(TWINRIX) and
		Latest TWINRIX Occurred Before 6 months Ago
	then
		Vaccine_Message.Message_Text := "Give VAQTA vaccine, 50U, IM now. This completes
Hepatitis A vaccinations.";
		Conclude true;
	endif;
	;;
action:
	write Vaccine_Message.Message_Text;;
end:
```

# 9. Standard Data Models and Data Access

Knowledge sharing can involve knowledge reuse and knowledge transfer. In the former, units of knowledge originally rendered in a specific locale for one purpose are employed for another purpose. For example, an Arden Syntax MLM that assesses the presence of chronic kidney disease in a patient may originally have been implemented for use in clinical decision support for drug dosing, but it could later be used to assess complications of diabetes mellitus. Because the reused MLM is being employed in the same clinical setting with the same information technology milieu as the original use, little if anything has to be changed in the MLM to facilitate its new use. By contrast, in knowledge transfer, a unit of knowledge such as an MLM is moved to another location, such as a different health care organization, and used for the same purpose as originally implemented. In this situation, the data used by the clinical decision support system may be different, with different codes, data models and database management systems when compared to the originating location. Consequently, in order to implement an MLM that originated elsewhere, alterations in data mappings would be required in order to ensure correct execution of the MLM. Because a foundational rationale for the creation of the Arden Syntax in the first place was facilitating knowledge sharing, it is important in managing this process to address issues related to data models and vocabularies. Because there was a paucity of standard data models and no consensus on which to use in clinical information systems, the creators of the Arden Syntax did not include a standard data model.

## 9.1. Data Models and Their Role in CDS and Arden Syntax

A data model is an abstract representation of the entities in a domain, their structure and the relationships among them. In a clinical data model, an entity may be an object that represents a class of related data elements and their attributes. For example, a class or entity in many clinical data models is “observation,” with attributes such as the type of observation (e.g., laboratory, physical examination, etc.), its date and time, its value, the identity of the observer and so on. Depending on the data model, an attribute of an entity may itself be an entity with its own attributes, thereby supporting nesting of entities.

The reason that a data model is important in clinical decision support systems is that the system will require access to data in order to apply or process its clinical knowledge. Because a data model may be used as the logical model of how data are stored in a repository that is queried by a decision support system, knowledge of the data model is required to devise the query that is presented by the clinical decision support system (CDSS) to the database management system (DBMS) in order to retrieve the data. Once retrieved, these data are referenced within units of knowledge such as an Arden Syntax MLM within the CDSS. Then, if an MLM is shared to a location that uses a different data model in its DBMS, data mappings and references within the MLM would have to be changed to reflect a new local data environment. Thus, alignment between a local data model and the data model used in an MLM or other unit of clinical knowledge representation may be problematic. The challenge of a lack of alignment would extend both to the queries used to retrieve the data from a DBMS and the references to these data in logical statements once they have been retrieved from the DBMS and placed in the working memory of

the CDSS. While this challenge exists for all clinical knowledge representation formalisms intended for knowledge transfer, in the context of the Arden Syntax it is known as the “curly braces problem” because of the syntactic construct used to enclose site-specific data mappings in an MLM. (See also Chapter 10.)

Use of a standard data model within Arden Syntax MLMs would help ameliorate this challenge. If the standard data model within the MLM aligned with the model used in the local data repository, then data retrieval is direct and easy; the terms in the query would match the objects in the database. If the standard data model still did not align with a local DBMS, a one-time mapping could be implemented between the two that would help automate the process of implementing shared MLMs in a specific data environment, avoiding the need to modify every shared MLM in order to link its data model with the local data model. The result of the use of a standard data model would be a reduction in time and complexity of sharing MLMs, which in turn would promote increased access to computer-based clinical decision support.

Since the time of creation of Arden Syntax v1.0 in 1989, considerable work has been undertaken to define formal data models for use in clinical information systems and even in particular in clinical decision support systems. Widely used data models in this context include the Clinical Data Interchange Standards Consortium (CDISC) Study Data Tabulation Model (SDTM) and the Observable Medical Outcomes Partnership (OMOP) Common Data Model (CDM). Within HL7, the Reference Information Model (RIM) was created to standardize references to data in messages and other standards in the HL7 suite of standards. The Virtual Medical Record (vMR) standard was derived from the RIM specifically to facilitate expression of clinical decision support knowledge and has been used in standards-based projects that showcased CDS, e.g., OpenCDS. However, the vMR is less emphasized with the introduction and growing popularity of the Fast Healthcare Interoperability Resources (FHIR) as a standard data model. In preliminary studies, both the vMR [6] and FHIR [7, 8] have been found sufficient as data models for representing the concepts expressed in a corpus of MLMs. The Clinical Information Modeling Initiative (CIMI) is an ongoing project in HL7 that seeks to identify, evaluate and create tools for data models that can be used in biomedicine and HL7 standards, and work in CIMI will inform use of data models for clinical decision support in the future.

## 9.2. Role of Terminologies

Nevertheless, while a standard data model is important in order to facilitate knowledge sharing, it is not sufficient. A data model defines classes of data, but it does not necessarily identify specific data elements within its classes. This is part of the role of a terminology. A standard terminology will identify the preferred or canonical term for expressing a concept along with a code and other attributes that characterize it. A vocabulary contains one or more terminologies along with the relationships among the concepts therein expressed. A standard data model, then, can be used to identify the class of a data element, while a terminology can be used to identify a specific instance or value of that element.

Just as with data models, there is no general agreement on a single, comprehensive terminology for use in clinical information systems generally or CDSS in particular. Moreover, the use of local or customized terminologies for specific purposes renders the situation even more complex than it would be otherwise. Nevertheless, in a preliminary study, the Systematized Nomenclature of Medicine - Clinical Terms (SNOMED CT) has been found adequate as a reference terminology for standardized data queries for a corpus of MLMs encoded in the Arden Syntax [9].

In addition to providing codes that can be used to denote concepts and data classes unambiguously in MLMs, standard vocabularies also typically identify ISA or class/subclass relationships that can be useful in implementing clinical reasoning in a number of domains. The fact that a concept in a vocabulary is a member of another class can be used in domains such as drug-allergy evaluation and cardiovascular risk assessment. For example, instead of listing in an MLM every single drug that is a member of the class “penicillin” when attempting to match a proposed medication to a patient’s penicillin allergy, an MLM author could use terminology service resources in FHIR to query a terminology service for the codes of all the medications in the penicillin class of antibiotics.

## 9.3. Current Arden Syntax Data Model

Both an advantage and a disadvantage of all the versions of Arden Syntax until the present has been the lack of a standard data model. Indeed, the creators of the Arden Syntax, recognizing the widespread variability in clinical data models and the lack of consensus in this regard, deliberately avoided endorsing a standard data model, instead creating a syntactic construct to map a local data model, such as those expressed in READ statements that query clinical databases, to variables in MLMs: The curly braces. While promoting flexibility, the disadvantage of this approach is that data references need to be updated whenever an MLM is shared with an organization that uses a different data model.

Recognizing the value in providing additional structure would be useful when representing concepts or data in MLMs and given that standard data models typically represent classes of data as objects with attributes, HL7 extended the Arden Syntax to allow for declaration of an OBJECT, which would then represent the structure of a variable. In this case, there is no nesting; only a single level of attributes can be associated with an object, e.g., REFERENCE_RANGE := OBJECT [low, high]; This allows storage of and references to values as REFERENCE_RANGE.low and REFERENCE_RANGE.high. Values of variables in the Arden Syntax can be scalar (a single value or object) or a list of values (much like an array in a third-generation programming language). In addition to lists and objects, native Arden Syntax data types include time, duration, string, Boolean, time of day, day of week and fuzzy types (number, time and duration).

## 9.4. FHIR as a Standard Data Model

The HL7 Fast Healthcare Interoperability Resources standard (https://www.hl7.org/fhir/) classes of data and their attributes represented as structured objects known as resources. FHIR is

increasingly popular as a way to represent data in transmission and storage. Because FHIR resources also offer attributes that allow for terminology binding, thereby combining the classification and representation power of a data model and standard terminologies, in structured objects, FHIR is a reasonable candidate for use as a standard data model within the Arden Syntax.

If it were introduced as the standard data model for the Arden Syntax, FHIR resources and attributes could be used directly in logical expressions in MLMs, likely using a dot notation like objects as they are currently defined in the Arden Syntax standard. For example, when referencing a problem on a patient's problem list in order to confirm an active problem, an MLM author could use the Condition resource to refer to the status (Condition.clinicalStatus) and ascertain whether it was active or resolved. Attributes such as clinicalStatus could be bound to internal HL7 or external standard code sets such as SNOMED CT as allowed in the FHIR standard.

Introduction of a standard data model in the Arden Syntax is the focus of work on v3.0 of the standard. At the time of writing, this work is ongoing in HL7 but also is awaiting further maturity and stabilization of the FHIR standard. However, even before finalization of v3.0, it would be possible to make use of FHIR as a data model in the Arden Syntax by defining variables using the OBJECT statement that correspond to the structure of FHIR resources; using data mappings in the curly braces to define queries to retrieve these data from a clinical data repository; and then using the object dot notation to refer to individual attribute values of these resources in logical statements representing clinical knowledge.

For example, using FHIR as an adjunct in version 2.10 of the Arden Syntax, an MLM could define an object that represents an element on a patient's problem list; read values from the database into that variable; and then use the variable, represented as a FHIR resource, in logical statements:

```
Condition := OBJECT [clinicalStatus, code];  /* FHIR Condition resource */
Condition := READ {select problem from problem_list};
        /* where 'problem' and 'problem_list' refer to a local data model */
IF (Condition.code = "E11.9") AND (Condition.clinicalStatus = "active")
        THEN conclude true;
                /* identifies an active diagnosis of Type 2 diabetes mellitus without…*/
                /* …complications using ICD10 as a standard terminology */
```

Work is continuing on incorporation of a standard data model in the Arden Syntax as part of v3.0.

# 10. Preview of Arden 3.0: The Curly-Braces Challenge and Other Standards

The current version of the Arden Syntax is version 2.10. A key goal for Arden Syntax, version 3.0 is to address the well-known curly-braces problem. Briefly, the creators of the Arden Syntax recognized the challenge and necessity of mapping to local data stores and queries when developing clinical decision support knowledge bases. However, given lack of widespread agreement on the data model, terminologies and query language to do so, they did not standardize such mappings in the Arden Syntax. Instead, they allowed MLM writers to encode such references in curly braces so that they could be easily identified and substituted when MLMs were shared beyond the organization in which they were created originally.

Now, in light of the rapid evolution of the Fast Healthcare Interoperability Resources (FHIR) standards, the Arden Syntax Work Group is conducting exploratory modeling using FHIR resources as a tentative extension to Arden Syntax. This would enhance standardization of data access and eliminate or reduce use of the curly braces.

A goal of this modeling approach will be to change the syntax as little as possible. We will initially attempt to maintain a large degree of backward compatibility to ease the transition for current Arden Syntax users. When, however, it is necessary to alter the Arden Syntax to take full advantage of FHIR, we will make the necessary changes.

Below is a simple sample MLM that illustrates the form this approach is taking. It demonstrates an approach to replacing the customized object definitions found in Arden Syntax, version 2.9. In Arden Syntax 3.0, these would be replaced by objects modeled using FHIR resources. The example is commented to provide some discussion of one FHIR-based modeling approach currently under consideration.

Example: In patients with diabetes, it is good clinical practice to test the hemoglobin A1c values every 6 months. This MLM is triggered upon patient arrival for an encounter. It determines whether the patient has diabetes based on SNOMED codes from the problem list or International Classification of Diseases (ICD 10) diagnoses coded during previous episodes of care. If no hemoglobin A1c is found within the last 6 months, it triggers an alert.

```
maintenance:
        title: Rule 1-HbA1c Timing;;
        mlmname: HbA1c_Timing;;
        arden: Version 2.9;;
        version: 1.00;;
        institution: Intermountain Healthcare;;
        author: Peter Haug (Peter.Haug@imail.org);;
        specialist: Peter Haug (Peter.Haug@imail.org);;
        date: 2016-11-07;;
        validation: testing;;
library:
        purpose: Alert for HbA1c if greater than 6 months.;;
```

```
        explanation: This MLM will send an alert if the patient is a diabetic (diabetes and
problem list or discharge diagnoses) and no HbA1c is recorded within the last 6 months.;;
        keywords: diabetes; HbA1c;;
        citations: to be added.;;
        links: to be added;;
knowledge:
type:  data_driven;;
data:
        LET Diabetic_Diagnoses BE (E10.10, E10.11, E10.21, E10.22, E10.29, E10.311, E10.319,
E10.321, E10.329, E10.331, E10.339, E10.341, E10.349, E10.351, E10.359, E10.36, E10.39, E10.40,
E10.41, E10.42, E10.43, E10.44, E10.49, E10.51, E10.52, E10.59, E10.610, E10.618, E10.620,
E10.621, E10.622, E10.628, E10.630, E10.638, E10.641, E10.649, E10.65, E10.69, E10.8, E10.9,
E11.00, E11.01, E11.21, E11.22, E11.29, E11.311, E11.319, E11.321, E11.329, E11.331, E11.339,
E11.341, E11.349, E11.351, E11.359, E11.36, E11.39, E11.40, E11.41, E11.42, E11.43, E11.44,
E11.49, E11.51, E11.52, E11.59, E11.610, E11.618, E11.620, E11.621, E11.622, E11.628, E11.630,
E11.638, E11.641, E11.649, E11.65, E11.69, E11.8, E11.9, E13.00, E13.01, E13.10, E13.11, E13.21,
E13.22, E13.29, E13.311, E13.319, E13.321, E13.329, E13.331, E13.339, E13.341, E13.349, E13.351,
E13.359, E13.36, E13.39, E13.40, E13.41, E13.42, E13.43, E13.44, E13.49, E13.51, E13.52, E13.59,
E13.610, E13.618, E13.620, E13.621, E13.622, E13.628, E13.630, E13.638, E13.641, E13.649, E13.65,
E13.69, E13.8, E13.9, O24.011, O24.012, O24.013, O24.019, O24.02, O24.03, O24.111, O24.112,
O24.113, O24.119, O24.12, O24.13, O24.311, O24.312, O24.313, O24.319, O24.32, O24.33, O24.811,
O24.812, O24.813, O24.819, O24.82, O24.83);

        /* This list of ICD 10 codes is part of a value set for diabetes. In this case it has
been imported at authoring time. An alternative requiring new syntax would be to reference the
ValueSet FHIR resource. */

        LET Diabetic_Problems BE (190330002, 190331003, 190368000, 190369008, 190372001,
190389009, 190390000, 199223000, 199225007, 199226008, 199227004, 199228009, 199229001,
199230006, 23045005, 237599002, 237618001, 237627000, 28032008, 313435000, 313436004, 314771006,
314772004, 314893005, 314894004, 314902007, 314903002, 314904008, 359642000, 44054006, 46635009,
4783006, 75682002, 76751001, 81531005, 9859006);

        /* This list of SNOMED codes is part of a value set for diabetes. In this case it has
been imported at authoring time. An alternative requiring new syntax would be to reference the
ValueSet FHIR resource.  */

        LET Condition BE OBJECT [clinicalStatus, verificationStatus, category, code, subject ];
        LET Observation BE OBJECT [status, code, subject, effective];
        LET Encounter BE OBJECT [status, subject];

        /* Three FHIR resources are invoked: Condition, Observation, and Encounter. Only those
components used in this MLM are referenced. Since these objects are effectively predefined by the
FHIR resources, an alternate syntax might be: */
        /* Include FHIR (Condition, Observation, Encounter) */

        LET subject BE OBJECT [identifier];
        LET identifier BE OBJECT [value];
        LET code BE OBJECT [coding];
        LET coding BE OBJECT [system, code, display];
        LET effective BE OBJECT [effectiveDateTime];

        /* These components of the 3 FHIR resources are included to maintain consistency with the
current Arden syntax. Since the components of FHIR resources are already defined, and alternate
syntax would have them included by reference. */

        LET Latest_HbA1c BE READ AS Observation LATEST
                WHERE Latest_HbA1c.subject.identifier.value = Current_Patient
                and Latest_HbA1c.Status = "final"
                and Latest_HbA1c.Code.Coding.system = "http://loinc.org"
                and Latest_HbA1c.Coding.code = "55454-3";
```

```
        /* Retrieves the most recent HbA1c using the FHIR Observation resource and the LOINC
coding system. */
        /*

        LET Diabetic_Problem BE READ AS LATEST Condition
                WHERE Diabetic_Problem.subject.identifier.value = Current_Patient
                and Diabetic_Problem.clinicalStatus = "active"
                and Diabetic_Problem.code.coding.system = "http://snomed.info/sct"
                and Diabetic_Problem.code.coding.code IN Diabetic_Problems;

/* Retrieves the Latest Diabetic problems using the FHIR Condition resource and the SNOMED coding
system. */
/* Note that a new reserve word "Current_Patient" is used. In places where a reference to
"patient" is use by FHIR, this specifies the patient whose episode of care we are monitoring. */

        LET Diabetic_Diagnosis BE READ AS LATEST Condition
                WHERE Diabetic_Diagnosis.subject.identifier.value = Current_Patient
                and Diabetic_Diagnosis.verificationStatus = "confirmed"
                and Diabetic_Diagnosis.code.coding.system =
                        "http://www.cdc.gov/nchs/icd/icd10cm.htm"
                and Diabetic_Diagnosis.code.coding.code IN Diabetic_Diagnoses;;

/* Retrieves the Latest Diabetic discharge diagnose using the FHIR Condition resource and the
ICD10 coding system. */

        Let Encounter_Event BE EVENT Encounter_Event
                WHERE Encounter_Event.status = "arrived"
                and Encounter_Event.subject.identifier.value = Current_Patient;

/* An event triggered when the status of the encounter resource is set to "arrived". */
;;
evoke: Encounter_Event;;
logic:
        IF
                Exist(Diabetic_Problem) or
                Exist(Diabetic_Diagnosis)
        THEN
                Diabetes_Present := True;
        ENDIF;
        IF
                Diabetes_Present and exist Latest_HbA1c
                and Latest_HbA1c Occurred not within past 6 months
        THEN
                conclude true;
        ENDIF;

        conclude false; ;;
action:
        WRITE "subject is a diabetic with no HbA1c in last 6 months. Please order one.";;

resources:
        default: en;;
        language: en;;
end:
```

Notes regarding this example:

1. As mentioned above, the list variables that contain the value set for a diabetic problem and the value set for diabetic diagnosis can be replaced by an appropriate ValueSet

resource. In the example above, codes from relevant value sets are simply copied into appropriate list variables. We are contemplating an extension of the syntax that would support direct reference to the value sets. However, we would not anticipate each MLM reading needed value sets at runtime. Instead, at compilation, the MLM would copy the values into an appropriate variable.

2. The distinction between primary time and event time is important in decision logic. In Arden Syntax 3.0, we will designate a default primary time for each resource. In the case of the Observation resource, the primary time would be Observation.effectiveDateTime. This would allow us to maintain the simplicity of statements like:

IF Diabetes_Present and exist Last_HbA1c and Last_HbA1c Occurred not within past 6 months
THEN conclude true;

3. The Arden Syntax **eventtime** will remain the time when the (correct) information was recorded.

The FHIR specification is undergoing rapid change. It has moved from Draft Standard for Trial Use Iteration 1 (DSTU1) to DSTU2 and is expected to transition to Standard for Trial Use Iteration 3 (STU3) in the near future. We are encouraged to think that the final version of the FHIR modeling system will allow us to eliminate a large part of the need for the curly braces. However, as long as decision authors deem it necessary to invoke data sources based on data models local to their institutions, some version of the curly braces will continue to appear in MLMs. Other formalisms, such as the HL7 Virtual Medical Record, have been considered in this regard, but because of the centrality and rising importance of FHIR, it is FHIR that will figure prominently in the work group's effort to address the curly-braces challenge.

Another challenge that is ripe for incorporation in a future version of the Arden Syntax is the fact that sometimes formalisms used to represent computable clinical knowledge do not ideally align with business processes and workflow. This is an important aspect of the Five Rights model of clinical decision support, in which the right information must be delivered to the right person, in the right intervention format, through the right channel and, particularly in this context, at the right time in workflow. Informal analysis with CDS vendors has identified that they and their customers have implemented workarounds when using the Arden Syntax because of its perceived lack of explicit support for workflow modeling. Consequently, the Arden Syntax Work Group will address this challenge as part of the evolution to version 3.0. This may include alignment with standards such as the Business Process Modeling Language (BPML) in order to provide explicit support for workflow integration in the Arden Syntax.

# 11. Representation of Business Processes by Combining Arden and Workflow Languages

The Arden Syntax is a standard designed to support the development, implementation, and exchange of executable representations of clinical decisions. It was developed specifically for the medical domain and has demonstrated its ability to represent the logic behind a variety of relevant decisions. Its greatest strength is when it is tightly integrated into electronic health record (EHR) environments and can effectively consume services (such as FHIR services) represent there.

Recently, other standards from the domain of business process management have been suggested as filling a complimentary need in the medical domain. These standards, which were developed by the Object Management Group (OMG), are principally concerned with workflow and the logic that supports it. They have been proposed as an additional component of the healthcare computing environment. A group has formed (the BPM+ community) to study and extend these standards with the goal of representing in computable form detailed clinical care processes. The OMG standards that are the focus of these efforts are Business Process Model and Notation (BPMN), version 2.02, Decision Model and Notation (DMN), version 1.2, and Case Management Model and Notation (CMMN), version 1.0. These three standards are intended to complement each other in the development of medical workflows of arbitrary complexity.

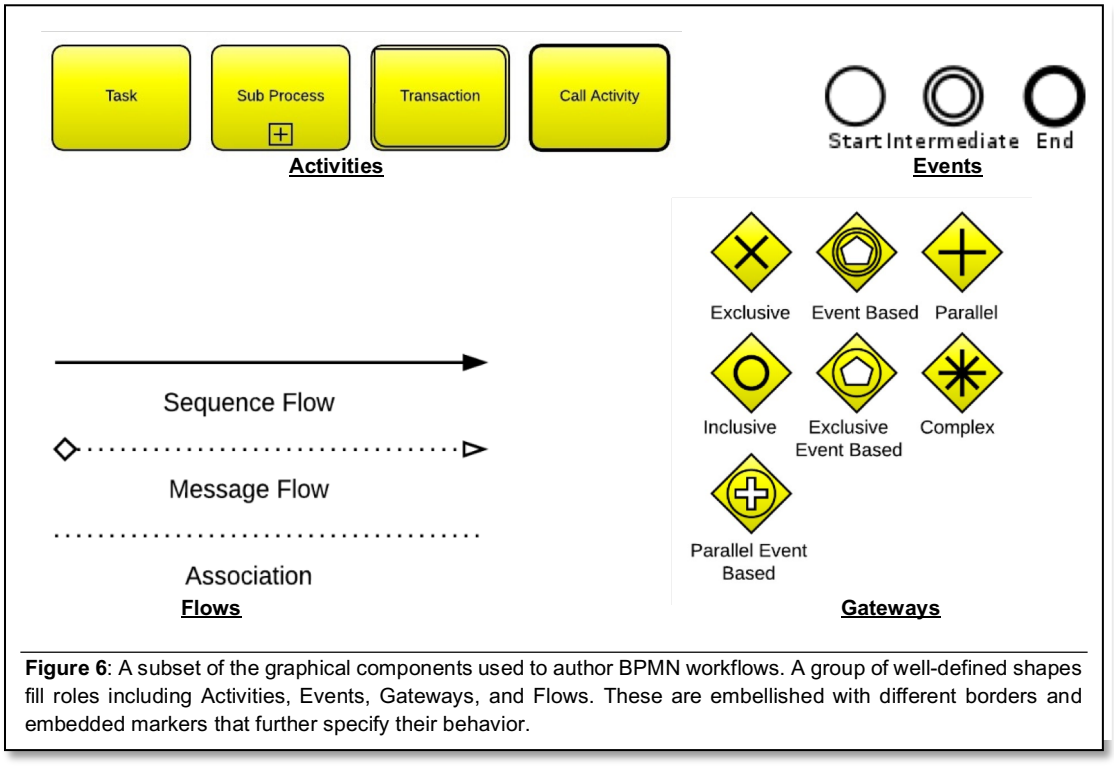



**Figure 6**: A subset of the graphical components used to author BPMN workflows. A group of well-defined shapes fill roles including Activities, Events, Gateways, and Flows. These are embellished with different borders and embedded markers that further specify their behavior.

A characteristic of these three standards that makes them appealing in the business world and that may prove invaluable in a medical environment is a focus on highly standardized graphical authoring environments (Figure 6). Each of these standards carefully details characteristics of an

authoring environment that is built around a constrained graphical approach to depicting its logic. This approach has been shown to ease the transfer of procedural knowledge from domain experts to technical personnel responsible for developing executable process models. In their most advanced form, these graphical models are associated with an executable output that accelerates the implementation of the modeled workflow.

As Arden Syntax 3.0 evolves, a key goal of the workgroup will be to study the relationships among these standards and to demonstrate ways in which the functionality of one may complement another.

## 11.1. A Brief Introduction to OMG Standards for Business Process Management

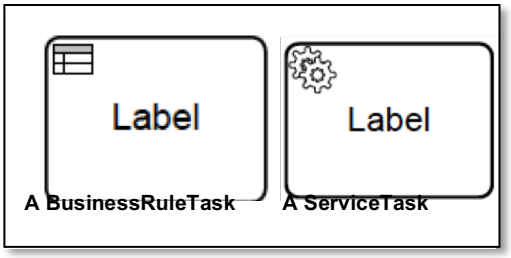
As mentioned above, there are three key standards incorporated into BPM+. These are:

- BPMN: a graphically oriented language specifically for designing workflows. Version 1 was released in 2004 and was focused on the graphical description of business processes. It evolved through two revisions and then was released as version 2.0 in 2011. This was the first version to feature a standardized executable output in addition to the graphical workflow depiction. Since its release dozens of general-purpose implementations of both authoring and execution environments have been created. The standard is currently in version 2.02.
- DMN: an authoring and execution standard for stateless decisions. Version 1.0 was published in September 2015, while the most recent release, version 1.2, was published in January 2019. The focus is to standardize and ease the review of decision logic for content experts and implementers alike. A combination of graphical tools, flexible decision tables, and logical expressions is used to create decision models, easily deployed as services. The standard attempts to support ease of authoring while producing an unambiguous executable decision model. DMN fills a gap in BPMN; decisions are a key determinant of workflow. A typical medical workflow may require dozens of independent clinical decisions.
- CMMN: a graphical language for authoring “Cases”. A Case is an instance of complex decision-making and workflow management that requires explicit oversight by a content expert. CMMN provides an environment in which an expert can bring to bear a variety of resources (often including BPMN and DMN artifacts) to manage a collection of decisions and workflows in which decision-making is too complex to fully capture in computer executable form.

BPM+ technologies have seen broad uptake in industry. They are valuable adjuncts for use in a service-oriented architecture. BPMN, in particular can be seen as a tool for orchestrating services in an environment where care givers and computable processes work together to deliver healthcare processes. A large set of tools is available to build and manage decision systems based on this standard. DMN is comparatively new, but fills an obvious gap in BPMN workflows. CMMN remains more complex but has a clear application to the management of medical cases.

## 11.2. Potential Relationship of OMG Standards to Arden Syntax

The OMG standards are attractive for several reasons. The graphical authoring approach offers an improved way for medical experts to capture and review the detailed logic of medicine. In addition, there is rapid uptake of these standards with dozens of venders offering authoring and execution environment. However, the key advantage may be the ability to represent true clinical workflows. In the Arden Syntax, MLM's are typically created to compute stateless, point-in-time, decisions. Operators are available to execute logic against temporal variables, but any stateful behavior is typically supported within the application environment that uses Arden as its source of medical logic. BPMN offers an opportunity to express real stateful workflows.




Arden, however, offers features that may enhance BPMN for use in medicine. These occur in two places. First, the BPMN graphical syntax includes an activity that represents the activation of a service to compute business logic. This is the BusinessRuleTask, however the specification does not provide specific direction concerning how the services should be implemented. It corresponds to the *logic* slot in the Arden syntax and could be represented by an Arden expression. Second, there is no specific syntax in BPMN for querying clinical data from an EHR. The syntax from the Arden *data* slot could be used to inform a BPMN ServiceTask designed to query data from an EHR and present it within the workflow in a form understandable to clinical users. For Arden, version 3.0 the service would be expected to query against FHIR APIs supplied by the EHR.

The Arden Syntax Working Group is evaluating approaches to integrating functionality found in the Arden Syntax with features provided by modern workflow languages. There appear to be substantial opportunities to this approach.

## 11.3. Examples

Simply translating an MLM into BPMN is straightforward. Data is extracted using the BPMN ServiceTask and logic is executed using the BusinessRuleTask. A simple illustration of this idea is found in Figure 7. It recapitulates the Arden MLM in above, "Rule 1-HbA1c Timing". The evoking process is represented as a signal event emitted as a part of a care process. The data needed by the MLM is supplied by FHIR-based services and the MLM's logic is embedded in a business rules task. The result is returned as a message to the care process that emitted the signal event. Note that, while this captures the "workflow" of the MLM, the details of execution are hidden inside the graphical process representations.

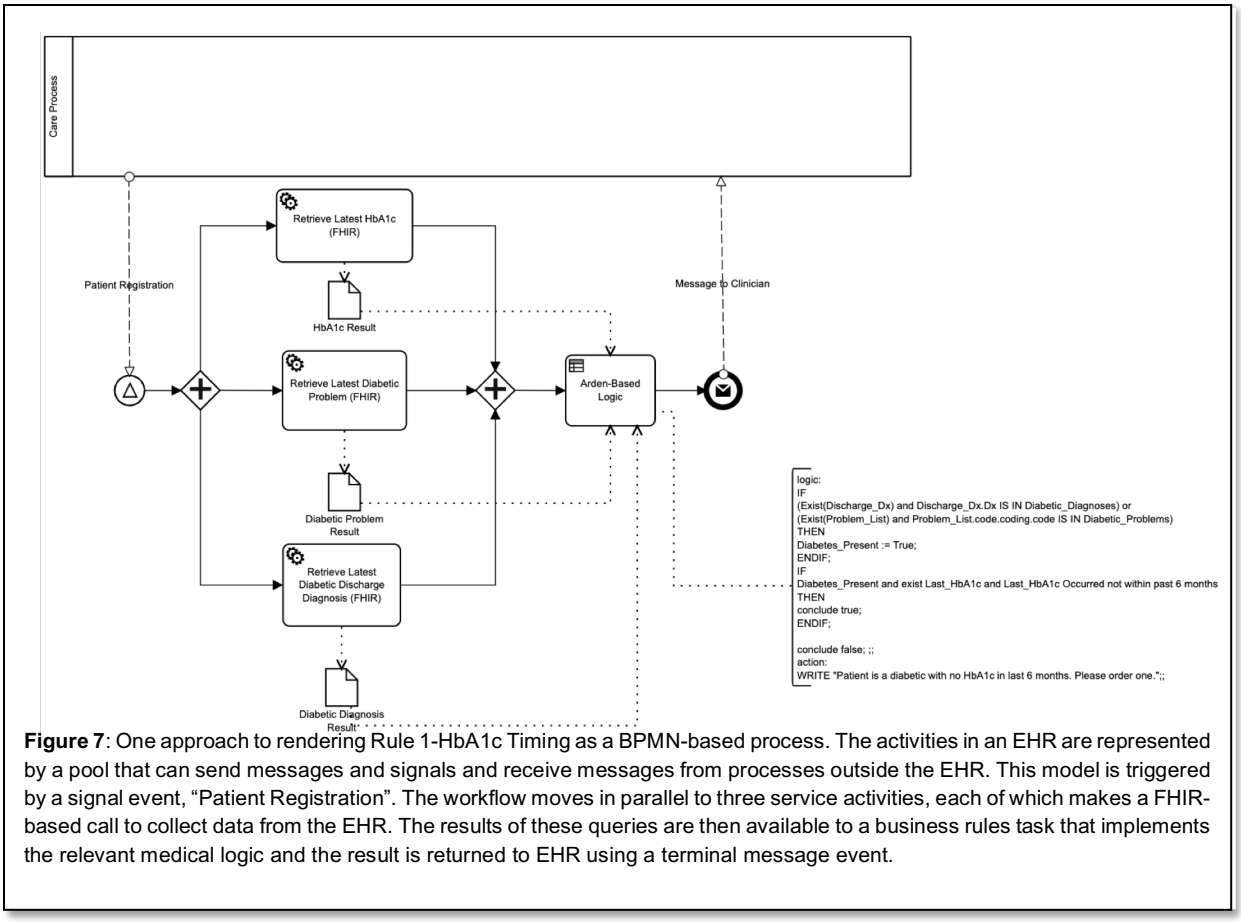



**Figure 7**: One approach to rendering Rule 1-HbA1c Timing as a BPMN-based process. The activities in an EHR are represented by a pool that can send messages and signals and receive messages from processes outside the EHR. This model is triggered by a signal event, "Patient Registration". The workflow moves in parallel to three service activities, each of which makes a FHIR-based call to collect data from the EHR. The results of these queries are then available to a business rules task that implements the relevant medical logic and the result is returned to EHR using a terminal message event.

While this simple example makes little use of the stateful behaviors available to BPMN process models, it is possible to elaborate the model to illustrate a more sophisticated behavior. Figure 8 illustrates a true workflow, extending over time, as the process moves from determining that the needed laboratory result is missing to prompting the clinician for a new order for an HbA1c and then following-up with the patient to plan and document any needed changes in treatment or monitoring.

BPMN workflows can be much more complex than this and can orchestrate the interaction of multiple caregivers in a variety of roles following a variety of clinical pathways. They may prove valuable in scripting key processes to support team-based care in a number of clinical conditions. The working group plans to explore the role of constructs from the Arden Syntax to help manage the semantics of the FHIR-based query services and of the medical logic embedded in the business rule activities. In this way a language originally designed to make clinical decision support rules more accessible to medical experts could be leveraged to standardize and enhance the processes through which care is provided.

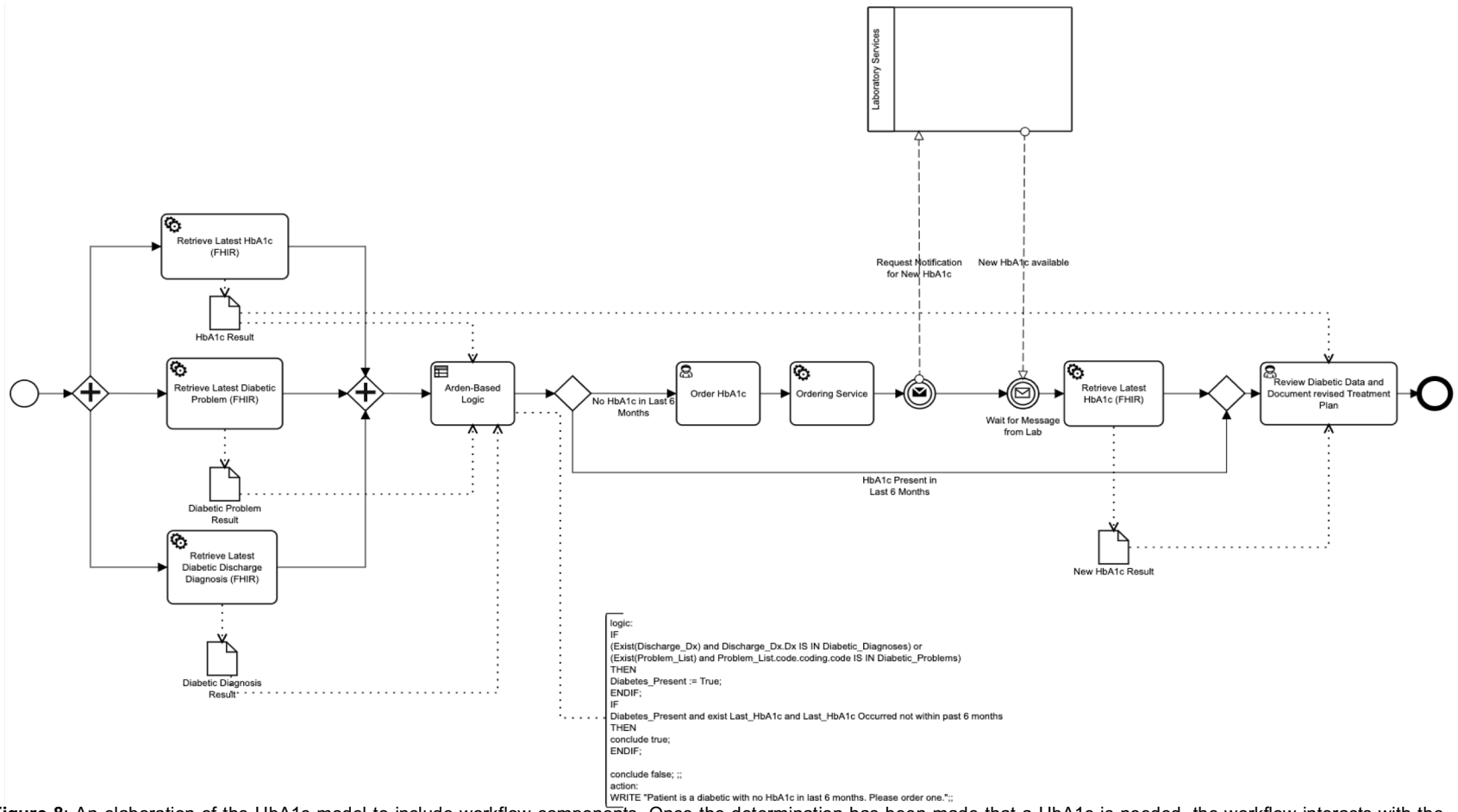



**Figure 8**: An elaboration of the HbA1c model to include workflow components. Once the determination has been made that a HbA1c is needed, the workflow interacts with the clinician to request the order, and ordering service (FHIR) sends the order, a message to the laboratory requests notification when the results are available, and the workflow pauses to wait for a message from the lab indicating the availability of the result. Upon notification, the workflow resumes, a FHIR query retrieves the result, and the clinician can discuss and document treatment options with the patient. The workflow could continue with a plan for significant alterations to treatment and monitoring if appropriate.

# 12. F.A.Q.

**Q: What is the Arden Syntax standard?**

A: The Arden Syntax for Medical Logic Systems is a structured, executable formalism for the explicit representation of scientific, clinical, administrative and other knowledge used in clinical decision support systems. As such, it functions as a programming language for such systems, allowing knowledge authors and clinical domain experts to implement knowledge-based interventions such as alerts, informational notices, and reminders, order sets, turnaround forms and the like in order to realize their quality improvement, clinical, administrative, and public health objectives. Expressed in a way that resembles English-language syntax, the Arden Syntax also facilitates validation of knowledge bases by domain experts. The Arden Syntax standard potentially allows the sharing of computerized health knowledge bases among personnel, information systems, and institutions. Its scope has been limited to those knowledge bases that can be represented as a set of discrete modules. Each module, referred to as a Medical Logic Module (MLM), contains sufficient knowledge to make a single medical decision. Contraindication alerts, management suggestions, data interpretations, treatment protocols, and diagnosis scores are examples of health knowledge representable using MLMs. Additionally, each MLM contains management information to help maintain a knowledge base consisting of MLMs as well as links to other knowledge sources. With this format, MLMs can be created directly by health personnel and can immediately be used by information systems conforming to this specification.

History: The Arden Syntax evolved from alerts and reminder systems at the LDS Hospital in Salt Lake City (i.e., the Health Evaluation through Logical Processing [HELP] system), the Regenstrief Institute in Indianapolis (the Regenstrief Medical Record System [RMRS]), the Columbia University Medical Center in New York (the first Arden Syntax system), and several other academic efforts. The group first met at the Arden Homestead in Harriman, NY, hence the name. The Arden Syntax was born out of the realization that the power of those alerts and reminder systems lies in the knowledge, and that there was a need to make this knowledge portable, shareable, between clinical information systems. Many commercial vendors since adopted the standard and included it in their products (see below).

The first version of the Arden Syntax was administered and issued by the American Society for Testing and Materials (ASTM, see below). Since 1998, the Arden Syntax group is part of the Health Level Seven (HL7) International organization, home of many widely accepted standards in health care informatics. The Arden Syntax Work Group administers the Arden Syntax standard.

**Q: How do I get a copy of the Arden Syntax standard?**

A: Visit the Arden Syntax product page at the HL7 International website.

**Q: How do I join the Arden Syntax Work Group?**

A: The Arden Syntax Work Group is part of the HL7 International organization. For information on HL7 meetings, contact HL7 at:

HL7 International
3300 Washtenaw Avenue
Suite 227
Ann Arbor, MI 48104-4250
phone: (734) 677-7777
fax: (734) 677-6622
email: hq@hl7.org
Web site: hl7.org
http://www.hl7.org/

To join the HL7 Arden Syntax Work Group (ardensyntax@lists.hl7.org) or Clinical Decision Support Work Group (dss@lists.hl7.org) listservs, see instructions on the HL7 website.

Contact persons for information on Arden Syntax are the two Arden Syntax Work Group co-chairs:

Robert Jenders, MD, MS, FACP, FACMI
Center for Biomedical Informatics and Department of Medicine, Charles Drew University
Department of Medicine, University of California, Los Angeles
email: jenders@ucla.edu

and

Peter Haug, MD
University of Utah and Intermountain Healthcare
email: peter.haug@imail.org

**Q: Which vendors of clinical information systems use the Arden Syntax in their clinical decision support applications?**

A: The following vendors have applications available that support the Arden Syntax (as of December 2016):

Agfa Healthcare (http://www.agfahealthcare.com)
Allscripts (http://www.allscripts.com/)
Cerner (http://www.cerner.com/)
McKesson (http://www.mckesson.com/)
Medexter Healthcare (http://www.medexter.com/)

**Q: Which health care organizations have Arden-Syntax-based systems in use?**

A: The list below includes the main installed sites per vendor (in alphabetical order) and is not exhaustive. Please contact the vendor directly for any additional information. This list is dated December 2016.

- Alamance Regional Medical Center, Burlington, NC (Allscripts)
- Chester County Hospital – West Chester, PA (Cerner)
- Covenant Health, Knoxville, TN (McKesson)
- ID Berlin, Germany (Medexter Healthcare)
- JFK Medical Center, Edison, NJ (McKesson)
- McLeod Regional Medical Center, Florence, SC (Cerner)
- Mississippi Baptist Health Systems, Jackson, MS (McKesson)
- Ohio State University, Columbus, OH (Cerner)
- Sarasota Memorial Hospital, Sarasota, FL (Allscripts)
- St. Mary's Hospital, Waterbury, CT (McKesson)
- St. Mary's Medical Center, Knoxville, TN (McKesson)
- St. Vincent's Hospital, Birmingham, AL (McKesson)
- University Hospital of Erlangen, Germany (Medexter Healthcare)
- University of Colorado Hospital Authority, CO (Medexter Healthcare)
- Vienna General Hospital/Medical University of Vienna, Austria (Medexter Healthcare)

Recently, a number of research and teaching institutions in Austria, Germany, Switzerland, and Japan use Arden Syntax in their educational and research projects (Medexter Healthcare).

**Q: Where can I get MLMs for my Arden Syntax decision support system?**

A: Your software or systems vendor may also supply you with a set of MLMs, if Arden Syntax MLMs are part of the company's product line. Medical knowledge content in form of MLMs is actively offered by Medexter Healthcare.

**Q: What implementation tools are available to help me use the Arden Syntax in my systems?**

A: Most commercial applications are developed by vendors for use primarily within their own environment. As of December 2016, only Medexter Healthcare offers Arden Syntax software systems and/or components that allow you to embed them into your own clinical systems or environments, and can even be embedded into other vendor's clinical systems.

**Q: What are the benefits of Arden-Syntax-based CDS systems?**

A: Clinical alerts and reminder systems have proven to be effective in improving the quality and reducing the cost of health care. Many publications demonstrate the effect of these systems in areas such as effective medication use, reduction of adverse drug events, infection control, preventive care and wellness, etc. See below for a list of publications documenting the effect of clinical decision support on quality and cost of health care.

Through the use of Arden Syntax, the knowledge that drives these systems becomes portable. As a result, hospitals and health care providers can incorporate knowledge into their systems that was developed and refined elsewhere, without having to “reinvent the wheel”.

**Selected bibliography on CDS:**


- Shortliffe EH, Sepúlveda MJ. Clinical Decision Support in the Era of Artificial Intelligence. Journal of the American Medical Association 2018;320(21):2199-2200.

- Jenders RA. Advances in Clinical Decision Support: Highlights of Practice and the Literature 2015-2016. IMIA Yearbook of Medical Informatics 2017;26(1):125-132.

- Greenes RA (Ed.) Clinical Decision Support, Second Edition. Oxford: Academic Press, 2014.

- Bright TJ, Wong A, Dhurjati R, Bristow E, Bastian L, Coeytaux RR, Samsa G, Hasselblad V, Williams JW, Musty MD, Wing L, Kendrick AS, Sanders GD, Lobach D. Effect of Clinical Decision-Support Systems: A Systematic Review. Annals of Internal Medicine 2012;157(1):29-43.

- Osheroff JA, Teich JM, Levick D, Saldana L, Velasco FT, Sittig DF, Rogers KM, Jenders RA. Improving Outcomes with Clinical Decision Support: An Implementer’s Guide, Second Edition. Chicago: Healthcare Information and Management Systems Society, 2012.


**Q: Where can I read more about the Arden Syntax?**

A: A number of application and research papers have been published involving the Arden Syntax or systems that use it:

**Selected bibliography on Arden Syntax:**


- Adlassnig K-P, Haug P, Jenders RA. Arden Syntax: Then, Now and in the Future. Artificial Intelligence in Medicine 2018;92:1-6.

- Hripcsak G, Wigertz OB, Clayton PD. Origins of the Arden Syntax. Artificial Intelligence in Medicine 2018;92:7-9.

- Jenders RA, Adlassnig K-P, Fehre K, Haug P. Evolution of the Arden Syntax: Key Technical Issues from the Standards Development Organization Perspective. Artificial Intelligence in Medicine 2018;92:10-14.

- Anand V, Carroll AE, Biondich PG, Dugan TM, Downs SM. Pediatric Decision Support Using Adapted Arden Syntax. Artificial Intelligence in Medicine 2018;92:15-23.

- Schuh C, de Bruin JS, Seeling W. Clinical Decision Support Systems at the Vienna General Hospital Using Arden Syntax: Design, Implementation, and Integration. Artificial Intelligence in Medicine 2018;92:24-33.

- De Bruin JS, Schuh C, Seeling W, Luger E, Gall M, Hütterer E, Kornek G, Ludvik B, Hoppichler F, Schindler K. Assessing the Feasibility of a Mobile Health-Supported Clinical Decision Support System for Nutritional Triage in Oncology Outpatients Using Arden Syntax. Artificial Intelligence in Medicine 2018;92:34-42.

- Castellanos I, Kraus S, Toddenroth D, Prokosch H-U, Bürkle T. Using Arden Syntax Medical Logic Modules to Reduce Overutilization of Laboratory Tests for Detection of Bacterial Infections—Success or Failure? Artificial Intelligence in Medicine 2018;92:43-50.

- Hussain M, Afzal M, Ali T, Ali R, Khan WA, Jamshed A, Lee S, Kang BH, Latif K. Data-Driven Knowledge Acquisition, Validation, and Transformation into HL7 Arden Syntax. Artificial Intelligence in Medicine 2018;92:51-70.

- Seitinger A., Rappelsberger A, Leitich H, Binder M, Adlassnig K-P. Executable Medical Guidelines with Arden Syntax—Applications in Dermatology and Obstetrics. Artificial Intelligence in Medicine 2018;92:71-81.

- Jung CY, Choi J-Y, Jeong SJ, Cho K, Koo YD, Bae JH, Kim S. Transformation of Arden Syntax's Medical Logic Modules into ArdenML for a Business Rules Management System. Artificial Intelligence in Medicine 2018; 92:82-87.

- Kraus S, Drescher C, Sedlmayr M, Castellanos I, Prokosch H-U, Toddenroth D. Using Arden Syntax for the Creation of a Multi-Patient Surveillance Dashboard. Artificial Intelligence in Medicine 2018;92:88-94.

- Kraus S, Enders M, Prokosch H-U, Castellanos I, Lenz R, Sedlmayr M. Accessing Complex Patient Data from Arden Syntax Medical Logic Modules. Artificial Intelligence in Medicine 2018;92:95-102.

- Karadimas H, Ebrahiminia V, Lepage E. User-Defined Functions in the Arden Syntax: An Extension Proposal. Artificial Intelligence in Medicine 2018;92:103-110.

- Kraus S, Castellanos I, Albermann M, Schuettler C, Prokosch H-U, Staudigel M, Toddenroth D (2016) Using Arden Syntax for the Generation of Intelligent Intensive Care Discharge Letters. In Hoerbst A, Hackl WO, de Keizer N, Prokosch H-U, Hercigonja-Szekeres M, de Lusignan S (Eds.) Exploring Complexity in Health: An Interdisciplinary Systems Approach, Proceedings of MIE2016 at HEC2016, Studies in Health Technology and Informatics 228, Amsterdam: IOS Press, 471-475.

- Wolf K-H, Klimek M (2016) A Conformance Test Suite for Arden Syntax Compilers and Interpreters. In Hoerbst A, Hackl WO, de Keizer N, Prokosch H-U, Hercigonja-Szekeres

M, de Lusignan S (Eds.) Exploring Complexity in Health: An Interdisciplinary Systems Approach, Proceedings of MIE2016 at HEC2016, Studies in Health Technology and Informatics 228, Amsterdam: IOS Press, 379-383.

- Kimura E, Ishihara K (2015) Internal Domain-Specific Language Based on Arden Syntax and FHIR. In Sarkar IN, Georgiou A, de Azevedo Marques PM (Eds.) MEDINFO 2015: e-Health-enabled Health, Proceedings of the 15th World Congress on Health and Biomedical Informatics, Studies in Health Technology and Informatics 216, Amsterdam: IOS Press, 955.

- Adlassnig K-P, Fehre K. Service-Oriented Fuzzy-Arden-Syntax-Based Clinical Decision Support. Indian Journal of Medical Informatics 2014;8(2):75-79.

- Gietzelt M, Goltz U, Grunwald D, Lochau M, Marschollek M, Song B, Wolf K-H. ARDEN2BYTECODE: A One-Pass Arden Syntax Compiler for Service-Oriented Decision Support Systems Based on the OSGi Platform. Computer Methods and Programs in Biomedicine 2012;106:114-125.

- Samwald M, Fehre K, De Bruin J, Adlassnig K-P. The Arden Syntax Standard for Clinical Decision Support: Experiences and Directions. Journal of Biomedical Informatics 2012;45(4):711-718.

- Jung CY, Sward KA, Haug PJ. Executing Medical Logic Modules Expressed in ArdenML Using Drools. Journal of the American Medical Informatics Association 2012;19:533-536.

- Vetterlein T, Mandl H, Adlassnig K-P. Fuzzy Arden Syntax: A Fuzzy Programming Language for Medicine. Artificial Intelligence in Medicine 2010;49:1-10.

- Kim S, Haug PJ, Rocha RA, Choi I. Modeling the Arden Syntax for Medical Decisions in XML. International Journal of Medical Informatics 2008;77:650-656.

- Hripcsak G, Ludemann P, Pryor TA, Wigertz OB, Clayton PD. Rationale for the Arden Syntax. Computers and Biomedical Research 1994;27:291-324.

- Hripcsak G. Writing Arden Syntax Medical Logic Modules. Computers in Biology and Medicine 1994;24(5):331-363.

**Q: Why should I use Arden Syntax rather than C, C++, Visual Basic, JAVA, or any other programming language or formalism that already exists? What are the advantages of the Arden Syntax?**

A: The Arden Syntax was specifically developed for health care applications and for embedding MLMs into clinical information systems. It has now been extended to an interoperable CDS technology platform to be connected to a variety of data sources, such as various kinds of clinical

information systems, data repositories, and the Web. Its suitability can be further illustrated by the following features:

- *The target user is a clinician:* The Arden Syntax is now comparable to a full-feature programming language. It does include complex structures common in many other programming languages. However, it kept its original goal that MLMs are meant to be written and used by clinicians with little or no training in programming.

- *Explicit links to data, trigger events, and messages to the target user:* The Arden Syntax was built to be embedded in existing clinical information systems. It clearly defines the hooks to clinical databases as well as how an MLM can be called (evoked) from a trigger event. It now can also communicate with many external systems and data sources.

- *Time functions:* Almost all medical knowledge somehow involves stating the time when an event has occurred. Therefore, Arden Syntax defines that every data element and every event has a data/time stamp that is clinically significant. The Arden Syntax contains many time functions (explicitly defined) to help users handle date and time in MLMs. Arden Syntax defines the notion "duration" (e.g., year, month, week) and shows how the user can check, for example, if an event has occurred in the last two days. In any other language, these definitions would be dependent on the person implementing the MLM, but Arden Syntax defines them explicitly.

**Q: What are the limitations of the Arden Syntax?**

A: A problem that occurs with any form of clinical knowledge representation is the need to interact with a clinical database in order to provide alerts, reminders, and any other CDS functionality. Since database schemata, clinical vocabulary, and data access methods vary widely, any form of encoding of clinical knowledge (such as an MLM) must be adapted to the respective institution in order to use the local clinical repository. This hinders the sharing of knowledge. Due to the fact that Arden Syntax is the only standard for clinical knowledge representation, this problem is associated with Arden Syntax, but it is not unique to it.

The Arden Syntax explicitly isolates references to the local data environment in curly braces "{}" in an MLM, so this is sometimes called the "curly brace problem". Efforts are underway in the HL7 Arden Syntax Work Group to help solve this problem, but this issue cannot be addressed by the Work Group alone; it requires industry-wide standardization of data access/storage.

Another potential limitation of the Arden Syntax is that it does not explicitly define notification mechanisms. Instead, this is left to local implementation and is like database queries expressed by curly brace expressions in an MLM. Explicit notification mechanisms in the Arden Syntax itself may be part of a future edition.

**Q: Some vendors refer to their systems as "compliant" with the Arden Syntax standard. What can I understand by that?**

A: When using an Arden-Syntax-compliant system, the user is able to create, import, customize, or otherwise implement MLMs without the need for vendor or system developer intervention. Additionally, the user can take an MLM from any institution, alter the content of the curly brace expressions, and make other related adjustments; the resultant MLM will be able to compile and execute at the user's institution.

**Q: What are the future plans for the Arden Syntax?**

A: The HL7 Arden Syntax Work Group is currently working on the Arden Syntax version 3. The new version will provide guidance on a standard data model that can be used by the Arden Syntax.

# 13. References


[1] Osheroff JA (Ed.) Improving Medication Use and Outcomes with Clinical Decision Support: A Step-by-Step Guide. Chicago: Health Information and Management Systems Society, 2009.

[2] Hripcsak G, Wigertz OB, Clayton PD. Origins of the Arden Syntax. Artificial Intelligence in Medicine 2018;92:7-9.

[3] Jenders RA, Adlassnig KP, Fehre K, Haug P. Evolution of the Arden Syntax: Key Technical Issues from the Standards Development Organization Perspective. Artificial Intelligence in Medicine 2018;92:10-14.

[4] Seitinger A, Rappelsberger A, Leitich H, Binder M, Adlassnig KP. Executable Medical Guidelines with Arden Syntax—Applications in Dermatology and Obstetrics. Artificial Intelligence in Medicine 2018;92:71-81.

[5] Kraus S, Castellanos I, Toddenroth D, Prokosch H-U, Bürkle T. Integrating Arden-Syntax-Based Clinical Decision Support with Extended Presentation Formats into a Commercial Patient Data Management System. Journal of Clinical Monitoring and Computing 2014;28:465-473.

[6] Jenders RA. Evaluation of the Health Level Seven Virtual Medical Record Standard as a Query Data Model for the Arden Syntax. AMIA Annual Symposium Proceedings 2013;701.

[7] Jenders RA. Evaluation of the Health Level Seven Fast Health Interoperable Resources (FHIR) Standard as a Query Data Model for the Arden Syntax. AMIA Annual Symposium Proceedings 2014;1438.

[8] Jenders RA. Utility of the Fast Healthcare Interoperability Resources (FHIR) Standard for Representation of Non-Query Data Mappings in the Arden Syntax. AMIA Annual Symposium Proceedings 2016;1449.

[9] Jenders RA. Evaluation of SNOMED CT as a Reference Terminology for Standardized Data Queries in the Arden Syntax. Studies in Health Technology and Informatics 2017;245:1326.