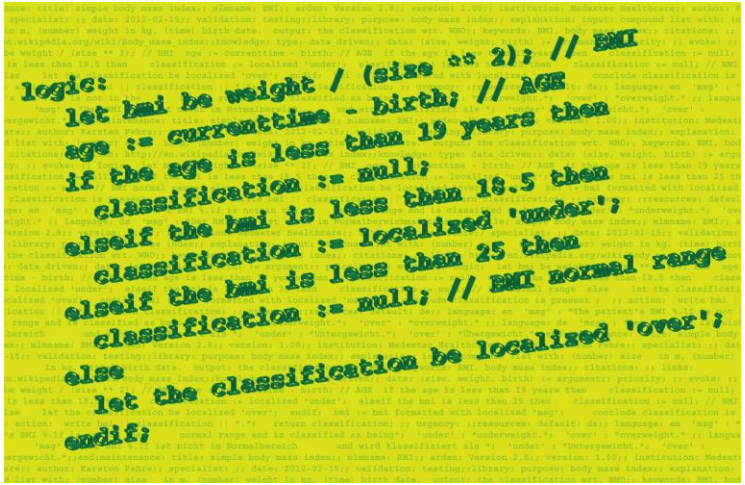
Arden Syntax advanced

Educational material, part 4

Medexter Healthcare
Borschkegasse 7/5
A-1090 Vienna

www.medexter.com

www.meduniwien.ac.at/kpa (academic)



Better care, patient safety, and quality assurance by Medexter, Vienna, Austria

Arden Syntax objects

- **Purpose**
  - Allow for the logical grouping of data elements.
  - May contain multiple named attributes, each of which may contain any valid Arden type (including lists or objects).
  - Allows for complex data structures to be manipulated by an MLM (e.g., lists within lists) which would otherwise not be possible.
- **Definition**
  - **Declaration**: An object is declared by the `OBJECT` keyword.

```
MedicationDose := OBJECT [Medication, Dose, Status];
```

  - **Instantiation**: Objects are created and with the `NEW` keyword.

```
medDose1 := NEW MedicationDose; // empty object
medDose2 := NEW MedicationDose WITH "Ampicillin", "500mg", "Active";
medDose3 := NEW MedicationDose WITH "Amoxicillin", "500mg", "Active";
```

Arden Syntax objects (continued)

- **Reading values**
  - **Access**: By using the dot (.) operator, object fields can be accessed.

```
"Ampicillin" := medDose2.Medication;

("Ampicillin", "Amoxicillin") := (medDose2, medDose3).Medication;
```

  - **Read As**: The `READ AS` statement queries an external data source (e.g., a patient database) and returns a single list of objects. The object type is a compulsory part of the statement, and should have been declared previously.

```
medDoses := READ AS MedicationDose
{
  "SELECT med, dosage, status FROM client"
};
```

Arden Syntax objects (continued)

- **Modifying values**
  - **Attribute assignment**: allows for the assignment to individual attributes of an object.

```
medication := NEW MedicationDose;
medication.Dose := "500mg";
medication.Status := "Active";
```

  - **Enhanced assignment:** Any expression that ends with a **dot** operation or **element** operation may be placed on the left hand side of an assignment.

```
medList[n].Dose := "300mg";
```

- **Object explication**
  - **Extract attribute names**: Returns a list containing the attribute names of the object argument.

```
"Medication", "Dose", "Status" := EXTRACT ATTRIBUTE NAMES medication;
```

MEDEXTER Healthcare

Arden Syntax objects – Example

```
    type: data_driven;;
    data:
        //Declare new object and its structure
        patientObj := OBJECT [
            Temperature,
            HeartRate,
            RespRate,
            PaCO2,
            WBcellCount,
            ImmatureBand
            ];

        //Receive list of values (eg. IDE Test tool, REST, SOAP)
        allValues := ARGUMENT;

        sirsPat := NEW patientObj WITH allValues[1], allValues[2], allValues[3],
            allValues[4], allValues[5], allValues[6];
        ;;
```

MEDEXTER Healthcare

Arden Syntax objects – Example

```
44  logic:
45      //Start - Checking SIRS criteria
46      counter := 0;
47
48      IF sirsPat.Temperature IS GREATER THAN 38
49          OR sirsPat.Temperature IS LESS THAN 36 THEN
50          counter:= counter + 1;
51      ENDIF;
52
53      IF sirsPat.HeartRate IS GREATER THAN 90 THEN
54          counter:= counter + 1;
55      ENDIF;
56
57      IF sirsPat.RespRate IS GREATER THAN 20
58          OR sirsPat.PaCO2 IS LESS THAN 32 THEN
59          counter:= counter + 1;
60      ENDIF;
61
62      IF sirsPat.WBcellCount  IS GREATER THAN 12000 OR
63          sirsPat.WBcellCount IS LESS THAN 4000
64          OR sirsPat.ImmatureBand IS GREATER THAN 10 THEN
65          counter:= counter + 1;
66      ENDIF;
67
68      IF counter IS GREATER THAN OR EQUAL 2 THEN
69          notification:= LOCALIZED 'SIRS';
70          CONCLUDE TRUE;
71      ENDIF;
72       //End - Checking SIRS criteria
73      ;;
74  action:
75      RETURN (sirsPat, notification);
76      ;;
```

Curly braces expressions

- **Purpose**
  - Signify institution-specific definitions and mappings
  - Allow for data and function access outside the Arden Syntax
  - Enable interaction with the host system
- **Syntax**
  - **Read statement:** Reads data from the host system. It is used to isolate those parts of a database query that are specific to an institution from those parts that are universal.

```
bodyTemp := READ {SELECT temp FROM patResults WHERE patID = 'Jeroen'};
```

  - **Event statement:** assigns an institution-specific event definition to a variable. This variable is used in the evoke slot, as part of the call statement to call other MLMs.

```
incBodyTempEvent := EVENT {increased body temperature};
```

Curly braces expressions – Read Example

```
20    type: data_driven;;
21    data:
22        testID := ARGUMENT;
23        (Temperature, HeartRate, RespRate, PaCO2, WBcellCount, ImmatureBand) :=
24            READ {SELECT temperature, heartrate, resprate, paco2, wbcellcount,
25                immatureband FROM sirsvalues WHERE IDPatient = testID};
26        ;;
```

- This **assignment** statement assigns the result of the read statement (using mapping clause "`SELECT … FROM … WHERE IDPatient = testID`") to a list of variables.
- `IDPatient` is a variable that contains the patient ID currently in use and is substituted before execution of the mapping clause
- The content of the **curly brace expressions** must be evaluated by the host system and its syntax is not part of the Arden Syntax

MEDEXTER Healthcare

Practical Part II

ArdenSuite Server Connectors

- **Database Connector**
  - Used to allow MLMs to query data from a database and process the returned data inside MLMs.

```
patient := READ {SELECT * FROM Patient WHERE patID = testID };
```

- **FHIR Connector**
  - Used to allow MLMs to query data from a FHIR server and process the returned data inside MLMs.

```
patient := READ {fhir:Patient/testID};
```

- **OpenEHR Connector**
  - Used to allow MLMs to query data from an OpenEHR server and process the returned data inside MLMs.

```
patient := READ {openehr:query/?aql=select patient/data[at0001|history|]...};
```

ArdenSuite APIs

- **RESTful web service interface**
  - The ARDENSUITE Server provides a REST application programming interface.
- **SOAP web service interface**
  - The ARDENSUITE Server provides a SOAP web service interface described by a Web Service Description Language (WSDL) file.
- **CDS Hooks web service interface**
  - The ARDENSUITE provides a CDS Hooks API that was programed according to CDS Hooks specification version 1.0.