The New Navigators: from Professionals to Patients
R. Baud et al. (Eds.)
IOS Press, 2003

Teaching Arden Syntax using a simulated environment

Harry Karadimas[a], Francois Hemery[a], Christophe Chailloleau[a], Eric Lepage[a]

[a] *Henri Mondor University Hospital, France*

Abstract

The Arden Syntax is a simple syntax that allows physicians to encode medical knowledge in small modules, called medical logic modules (MLM). These MLM are well suited to data-driven alerts and reminders. While the Arden syntax is simple to teach, the students are most of the time limited to the writing of modules and the verification of the module's syntax. Our work proposes an environment that simulates various elements required by MLMs, allowing the students to execute the MLMs they have written immediately. This provides the necessary feedback to understand in depth the inner workings of MLMs.

Keywords
Decision Support Systems, Clinical; Teaching Materials; Hospital Information Systems; Reminder Systems

1. Introduction

The Arden Syntax has now a more than 10 years old history, yet it is still not widely implemented nor widely taught. The lack of implementation can be explained by the high integration cost of open rule-based technology into a clinical information system. However, an effort can be made in the teaching part, by supplying tools that are more adequate. This paper presents an interactive Arden Syntax development environment that allows students to type, compile, run and interact with Medical Logic Modules (MLM), using simulated clinical data, time and events.

2. Materials and methods

2.1. History of the Arden Syntax

The Arden Syntax was conceived in 1989 at a meeting involving several specialists in medical informatics. The objective was to define a common model for sharing medical knowledge [1]. A common model and format for writing rules was obtained, based on external data and events from major medical record implementations existing at that time. In April 1992, the Arden Syntax was approved by the American Society of Testing and Machinery (ASTM), under the number E-1460-92 [2], making it the first standardised syntax available for writing medical decision-support rules. The strengths and limitations of MLMs are now well known. MLMs are adequate to write alerts and reminders. They were not designed to represent complex guidelines and protocols, for which there are other better (although less standardised) approaches, like GLIF [3, 4]. MLMs are made of plain text that is divided into several sections (called slots) for authors purpose, etc... For computer execution, four slots are of particular interest : 1) the `"data"` slot, that describes external data elements. 2) the `"evoke"` slot, that describes how the MLM is started (the term used in the standard is *triggered*). 3) the `"logic"` slot, that contains the actual medical logic in a Pascal-like syntax. 4) the `"action"` slot that contains actions to be executed if the logic slot evaluates to true. An MLM is *data-driven*, which means that addition or modification

of data creates events that trigger MLMs. Triggered MLMs query data from various sources, evaluate it, and execute an action if necessary.

2.2. Constraints for an implementation

In order to facilitate distribution and allow individual use of the environment without side effects, we want our CIS simulator to be autonomous. The environment must be usable without the need to connect to a network. Each student can then experiment with different clinical values the various modifications he makes to his MLMs. We chose initially not to use any external database system. This should illustrate the fact that MLMs were not designed to depend on specific database architecture (relational, object, etc...). The students must also be able to use the environment at home, for further study and assignments. However, the possibility to connect to additional external databases is planned in future versions.

2.3. The Arden2J library used for implementation

The platform presented here uses architectural elements from the Arden/J library, which is part of previously published work [5]. This library has been completely rewritten to take into account the version 2 of the Arden Syntax standard. To reflect this, it is now called Arden2j. The initial concepts have not changed, however. In particular, the external elements declared inside curly braces can use a special syntax to select an appropriate object, which we call a *mapper*, that will handle the declaration appropriately. For this simulation system, we have developed three additional mappers : 1) a mapper to encode literal data elements directly inside the MLM. 2) a mapper to select data from the simulated clinical database. 3) a mapper to declare the events needed by the MLM.

3. Results

3.1. The student environment

The resulting student environment is composed of two independent applications : the authoring environment and the CIS simulation environment. Both are based on a graphical user interface. The environment works on any computer that has the Java development kit v. 1.3.0 (or greater) from Sun Microsystems installed. The authoring environment is used to edit and compile the MLMs. The simulation environment is used to execute compiled MLMs in a simulated CIS.

3.2. Reuse of Arden2j elements

The initial Arden/J environment already had a simulation mode that appeared when the system was launched with the default parameter values [5]. This simulation mode was initially used for teaching, but had several limitations which are now removed. The Arden2j system can now load classes dynamically, and handles different versions of classes. This means that the same MLM has each of its versions loaded separately. MLM instances from different MLM versions run concurrently. New events will always trigger the newest version. The older versions will continue to run until they have finished their execution. For example, if an MLM has to look for some data every five minutes for two hours, it will remain loaded in memory for two hours. The advantage of this approach is that students can add new MLMs, modify MLMs, and load them into the simulation environment without restarting it.

3.3. The MLM authoring environment

The MLM authoring environment has a project manager, which enables the designation of a set of files to avoid that the students lose too much time finding each relevant file. The instructor can then have one project for each concept to illustrate. The design of the authoring environment is based on traditional integrated development environment (IDE) tools. The student can compile the selected file; compilation results (including errors) are displayed under the compiled file. In case of an error, clicking on the error in the message window causes the cursor to move to the line where the error was found. Figure 1 shows a sample editing session. The use of an IDE allows to spend less time with compilation problems than with the command-line compilation approach that we used before. The MLM authoring environment is not made specifically for students; we use it also in our institution for the production system.

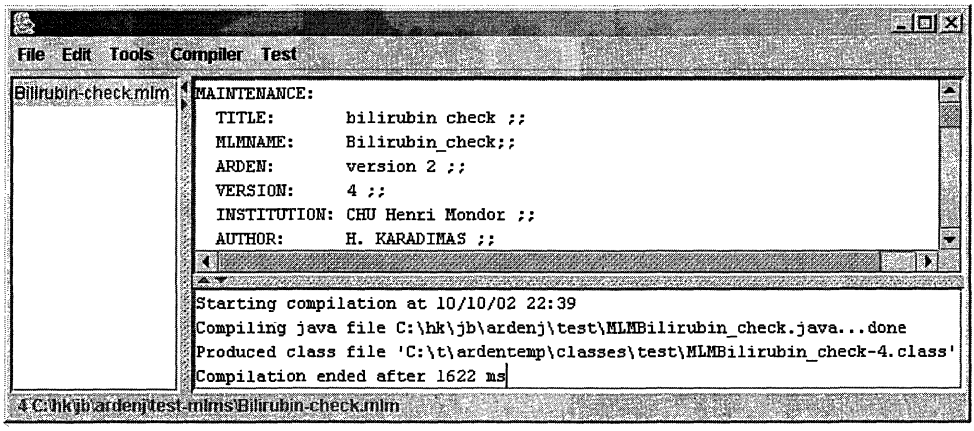

Fig. 1 : the MLM authoring environment

3.4. The MLM execution system

The CIS simulator tool is based on a regular `ArdenSystem` object from the Arden2j library. The Arden System in production at our institution is also based on the same object. This means that students use the same architectural elements as the ones available for production use. The difference is that the elements needed by the student's Arden system are designed specially for a simulation environment. The current time is not provided by the computer's system clock but by a time simulation user interface. This interface can be run in regular discrete increments by pressing the button with the "arrow" icon. The time increment is selected by using check buttons. Time flow can also be simulated step-by-step by pressing the button with the "plus" icon. A particular time can be entered directly in the time field, which is editable. The event monitor used can be forced to throw individual events by choosing them in the event list displayed at the lower left of the window, and pressing the button with the "thunder" icon.

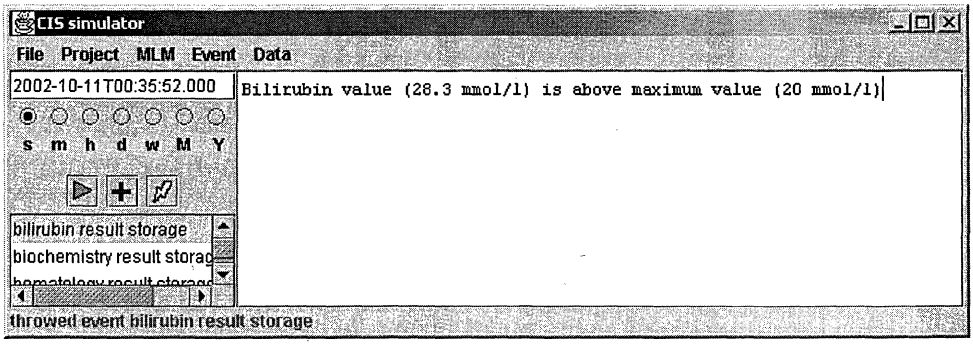

Fig. 2 : the CIS simulation environment

3.5. Clinical data simulation

Clinical data is simulated in a manner specific to the Arden Syntax : all queries in the MLMs return one or more lists of atomic values. The initial clinical data is read in memory from a simple text file that contains named collections of value lists. An example of such a file is given in listing 1. Every value has a primary time (in case of multiple values, the preceding primary time is used). The syntax used for the data is the same as in MLMs except for primary times that are declared inside parentheses to differentiate them from actual values. Comments are allowed, so the data should be read easily by the students.

```
bilirubin result:
(2002-10-12)13.8; (2002-10-13)14.2
potassium result:
(2002-10-12)4; (2002-10-13)3.2
abdominal pain findings:
/* usage : (present, pain_duration) :=
   READ {qaif:student:abdominal pain findings} */
(2002-10-12)true, 2 days;
(2002-10-13)false, null
```

Listing 1 : example of initial clinical data

The student can add any additional data to the in-memory database while the system is running. Each data collection added has a name. When the data is added, a new event is thrown with the collection name and « storage » appended to it. All MLMs that have declared this event in their evoke slot are then triggered. The MLMs can write to the database as well. This allows the illustration of MLM chaining using "intermediate states" [6].

3.6. Use of the teaching platform with Master of Science (MS) students in Medical Informatics

This study environment has been used in the teaching course for Arden syntax systems with the students in medical informatics MS, in the years 2001 and 2002. The course took place in a room where each student has an IBM-PC compatible machine. The direct use of the system generated many questions among the students. The recurrent question concerned the implementation details of a real MLM compliant system, its integration with a clinical information system (CIS), and the differences between this study environment and a CIS-integrated environment.

4. Discussion and perspectives

We have been teaching an introductory course about the Arden Syntax for 4 years before we used the simulation tool. For the last two years, we have been using the simulation tool in its successive versions. The nature and number of the questions asked by the students changed radically when the use of the simulation tool was introduced. The students were more concerned about CIS integration issues than in the previous years. The students also had less questions about basic MLM functions like triggering, data querying, as they discovered all this through the simulation tool. From a more general point of view, MLM execution can also illustrate some shortcomings of the Arden Syntax. For example, compared to more sophisticated automated decision support systems, MLMs lack organised chaining control mechanisms. They also lack abstractions and classifications. If the students have access to other knowledge acquisition tools that separate more distinctly data, knowledge and inference mechanisms, such as Protégé [7] for example, they can evaluate even better the benefits and shortcomings of each approach (we do not however teach Protégé in our courses).
Several additions are planned for a future version. These additions were suggested by the students during the course. We plan to add : 1) a generalised mapping to a relational database (the mapping we have in production can not run outside our CIS). 2) dictionaries of medications and diseases. 3) simulation of a medical record user interface to have a more coherent data entry.

5. Conclusion

We have produced an Arden Syntax development environment that allows, beyond the traditional syntax edition and verification, the execution of MLMs inside a simulated CIS. We submitted this environment to students during their Arden Syntax course. The feedback of the students was positive. The simulation of MLMs allows a better understanding of the mechanisms involved in an MLM decision support implementation. This tool is conceived to teach the Arden Syntax, but it can also be used as an introductory tool for automated decision-support techniques teaching.

6. References

[1] Clayton PD, Pryor TA, Wigertz OB, Hripcsak G. Issues and Structures for Sharing Medical Knowledge among Decision-Making Systems : The 1989 Arden Homestead Retreat. In: Symposium on Computer Applications in Medical Care; 1989; 1989. p. 116-121.

[2] ASTM. Standard specification for defining and sharing modular health knowledge bases (Arden Syntax for Medical Logic Modules). Philadelphia; 1992.

[3] Peleg M, Boxwala AA, Bernstam E, Tu S, Greenes RA, Shortliffe EH. Sharable representation of clinical guidelines in GLIF: relationship to the Arden Syntax. *J Biomed Inform* 2001;34(3):170-81.

[4] Peleg M, Boxwala AA, Ogunyemi O, Zeng Q, Tu S, Lacson R, et al. GLIF3: the evolution of a guideline representation format. *Proc AMIA Symp* 2000:645-9.

[5] Karadimas HC, Chailloleau C, Hemery F, Simonnet J, Lepage E. Arden/J: an architecture for MLM execution on the Java platform. *J Am Med Inform Assoc* 2002;9(4):359-68.

[6] Sherman EH, Hripcsak G, Starren J, Jenders RA, Clayton P. Using intermediate states to improve the ability of the Arden Syntax to implement care plans and reuse knowledge. *Proc Annu Symp Comput Appl Med Care* 1995:238-42.

[7] Li Q, Shilane P, Noy NF, Musen MA. Ontology acquisition from on-line knowledge sources. *Proc AMIA Symp* 2000:497-501.

Address for correspondence
Harry Karadimas, DIH CHU Henri Mondor, 94010 CRETEIL FRANCE
e-mail : harry.karadimas@hmn.ap-hop-paris.fr