ADAM: An object-oriented formalism for describing and representing civil structures

Marco Lazzari, Andrea Spinelli

ISMES SpA
9, Via Pastrengo, 24068 Seriate BG, Italy
phone +39-35-307777, e-mail {mlazzari,aspinelli}@ismes.it

INTRODUCTION

In the field of engineering systems, it is important to correctly integrate graphical, textual and computational resources referring to complex physical structures, such as civil structures or production plants. It is therefore necessary to weave together several information sources in a common interface, keeping alive links with real-time monitoring systems and interpretation systems: moreover, engineering structures such as dams or historical buildings present a wide morphological variability, complicating the task of developing a common user interface.

Therefore, we have experimented and tested an interpretive approach, according to which user interfaces are generated at runtime according to a general framework, in our case, HTML (Hypertext Markup Language). A description of the physical structure is given in the form of an object-oriented model, which is interpreted by the system both at a shallow and at a deep level. At the shallow level, *part-of* relationships are exploited to make it possible to navigate the physical structure: each physical object is represented by a HTML page, with description and attributes, which is generated on-the-fly using the model; the base structure of the page is deduced by inspecting the type of the object, while *the part-of* relationship provides links to subparts of the object. At a deeper level, the model is exploited by a reasoning engine, for instance to evaluate causal relationships among safety-related processes.

The formalism is called ADAM (A DAM Language) and has been developed in the context of the DAMSAFE decision support system for the management of dam safety.

At run time, DAMSAFE interprets ADAM descriptions and generates World-Wide-Web pages baased on ADAM data; at user request, it activates external databases and systems on a local or wide area computer network, through CGI (Common Gateway Interface), as well as a specialised application (writtent in the X-Window-InterViews environment) which provides an interface to a causal network of processes.

Through ADAM descriptions the user informs the system about the components of the dam currently under evaluation (for instance, blocks) and about the data sources to be linked (for instance, the database of monitoring measurements and its access code).

The language has been designed in order to enable users unfamiliar with programming languages to easily set up their own data management environment.

A C++ hierarchy of classes constitutes the ADAM interpreter and allows access to ADAM functionalities from any C++ application.

Through ADAM models the DAMSAFE system is interfaced with several heterogeneous information systems: MIDAS (Management of Information for DAm Safety), providing measurement data, design drawings and numerical safety models; KAL, dealing with data about physical testing; DAMS, concerning legal data about Italian dams; our company's Management Information System, pointing to documents referring about a given dam.

The ADAM module has been deployed since April 1995 for internal use to support a group of engineers and technicians that provides services related to periodical safety assessment of dams.

1. THE CONTEXT: MANAGING STRUCTURAL SAFETY

Significant resources are expended in the field of civil engineering in managing the safety of structures. Data about structural behaviour are collected through tests and visual inspections, while automatic instrumentation and data acquisition systems are used for real time monitoring.

The interpretation of such data is not easy owing to different factors, such as the large amount of data, the uncertainty and incompleteness of information, the need for engineering judgement, knowledge of the particular structure, experience of the behaviour of structures in general and a background of general engineering knowledge in order to interpret the data.

AI concepts and technologies can assist engineers in safety management by providing new software components to the existing information systems such as real time interpretation systems linked to the data acquisition units, qualitative models and reasoning agents supporting the off-line management of information and interpretation, intelligent man/machine interfaces.

During the last six years, the software department of ISMES has worked in the field of the artificial intelligence applications to structural safety. Artificial intelligence provided powerful tools for the design of intelligent modules: causal networks of processes, qualitative modelling, model-based reasoning, hierarchical object-oriented representations were largely used. Moreover, AI techniques, in conjunction with conventional ones, were also used for implementing such representation and reasoning schemes, such as rule-based systems, pattern matching, neural networks.

Our efforts led to the development of several systems which are now operational and help safety managers, engineers and authorities to deal with safety problems of structures or urban nuclei.

This paper aims at showing how intelligent interfaces allowed to improve one of the aforementioned systems, DAMSAFE, transforming a stand-alone prototype into a powerful distributed environment for the management of dam safety.

2. A.I. TOOLS FOR THE MANAGEMENT OF DAM SAFETY

In the last decade several researchers took advantage of artificial intelligence techniques to face some specific problems related to dam safety: Texas Instrument developed the Kelly System[1], an expert system to support data analysis and seepage detection of the Vermilion Dam, an earth dam managed by Southern California Edison; Bruno Franck and Ted Krauthammer, at the University of Minnesota, built a system to assist the users in performing the visual field inspections of small to medium size concrete gravity dams[2]; David Blockley, at the University of Bristol, leads a group that is involved in developing both theoretical research, tools and prototypes to cope with data interpretation[3].

All these system were focused on the conceptual core of the data interpretation process, and each of them was dedicated to a single task, whilst man/machine interaction was considered as a secondary feature and the integration of heterogeneous data sources was not required.

ISMES has been using since 1988 artificial intelligence concepts and tools to support dam safety management, building software products which integrate and upgrade the existing information system related to the data gathered on dams.

In fact, a basic requirement of managing dam safety is the monitoring of the structure in order to collect data which are then interpreted to understand the state of the dam. In Italy, this is done through a two-level organisational structure: a first level identifies and manages possible alarm conditions and, if necessary, calls the second level; the second level manages the available information concerning the dam and evaluates the safety of the structure on a periodical basis or when requested by the first level.

To support the organisation stated above, the pre-existing information system developed by ISMES was comprised of the following sub-systems: a real time automatic monitoring system, a telemetry system and a central data base with associated processing and representation functions.

Furthermore, engineers often require other data to carry out the procedures of dam safety management, such as different types of information (design records, photographs, design drawings, test and monitoring data, qualitative assessments of condition) concerning a dam and different types of models of the dam system (numerical structural models, data models, normative models for behaviour). In fact, in a situation where decisions are required even if the available knowledge is incomplete, it is important to integrate and use in a co-ordinated manner every type of knowledge.

As a consequence, there is a need for a co-operative supporting system, able to help people in managing the complexity of the evaluation. AI can be helpful through providing new ways to model the behaviours of the physical systems (e.g. a qualitative causal net of possible physical processes). This modelling approach is a useful way to integrate different types of knowledge providing a global scenario to understand the physical system behaviour.

To improve the capabilities of the information system in accordance with the above stated requirements, we developed a system, called DAMSAFE, using AI concepts and technologies. DAMSAFE may be interpreted as the evolution of the MIDAS data base management system, a tool developed by ISMES in the early 80's to store data acquired by monitoring systems or manually collected and to allow the subsequent analysis and interpretation of the behaviour of the structure (*off-line* check): DAMSAFE integrates the MIDAS system (data base, graphical and computational tools), adds new types of information (results of visual inspections and tests, design drawings) and provides an additional model of the dam system (a causal net of possible processes occurring in the dam and near environment) and tools to support the interpretation of data and the evaluation of possible scenarios using the causal net.

Two versions of DAMSAFE have been deployed: the first one, delivered in 1992, was a prototype developed on a stand-alone workstation for a specific dam. All the data accessible by the program were resident on the workstation; therefore, this implied that a copy of each interesting data base had to be installed on the delivery machine, which was only acceptable for laboratory prototypes and tests. Moreover, the links of the system to the data bases and the configuration of the system itself were embedded.

The second version, released in 1995, took advantage of recent advances in the field of intelligent interfaces and client/server architectures to set up a safety management environment aimed at:

- ☐ working as a distributed application on a local area network (LAN);
- ☐ integrating several *real* data bases already installed on the LAN;

- ☐ running external computational tools available on the LAN;
- ☐ enabling different users to access these data sources and to share their knowledge;
- ☐ enabling users unfamiliar with programming languages to easily set up their own data management environment (for instance, by adding a new dam, modelling its structure, and creating links to its data sources) and to build their interface to the system.

The aim of the following paragraphs is to highlight the enhancements achieved through the second version of DAMSAFE.

3. INTERPRETIVE INTERFACES

In our experience, the main problem for the deployment of innovative technologies to end-users has been the impossiblity for the knowledge engineer to individually follow the development, that is, configuration, of all the specific cases. For instance, the MIDAS system manages safety-related information for about 200 dams and civil structures; it is therefore necessary to delegate some knowledge engineering task to end-users.

The DAMSAFE models include typically:

- a physical layer, dealing with the physical components of the structure; it describes the part-whole relationships among the components and relates each component with external databases; for instance, a section is related with the measurement instruments in the section, with a photograph of the section, with a design drawing of the section.
- a process description part, which includes a description of the safety-related processes which can occur in the structure; it describes cause-effect relationships among processes. An example of process is 'rotation of dam block'. Processes carry information about their state, as well as some general information.

The system includes a *feature extractor*, which identifies features in the monitoring data (for instance a step or a trend); a *mapper*, which maps features onto processes; an *enforcer*, which applies causal rules to process' states in order to deduce possible hazards from combination of states.

We had to guarantee a lot of flexibility in the knowledge base, so as to allow partial models to be gradually completed according to available time and resources.

To complicate things further, users expressed a strong requirement for a graphical user interface, with should closely adapt to the specific features of each structure.

It was therefore necessary to develop a framework in which the task of describing a new structure did not require the knowledge of a full-fledged programming language; moreover, it was very desirable to build a user interface able to adapt to widely different instances. The interpretive choice, in which user interface are interpreted at run-time, seemed to be necessary; since we had a declarative model of the structure, we undertook the task of derivating the interface directly from the model.

4. ADAM: A TINY LANGUAGE

The first choice in the design of the system was the adoption of the World-Wide-Web platform, which provided a seamless distribution over a LAN and, potentially, over a wide area network, was operating-system independent, and allowed a lot of escapes for external formats.

Our working hypotesis was to specify interfaces directly in HTML, and to use a separate formalism for the models. This approach was rejected for two reasons: HTML was too difficult for end-users, and the model had to be expressed anyway.

We chose to create a tiny language, nicknamed ADAM, which would embody our specific requirements, provide modeling capabilities and allow us to integrate deep-modelling in the application domain with user interface hints and specifications.

We developed a WWW server (using the CERN httpd) with a collection of CGI scripts which read an ADAM specification of the model of choice, generate on-the-fly HTML pages describing the model and perform access to several databases.

In fact, the objects include enough information to generate URLs referring to specific items in the external databases, such as data series names in the MIDAS database, or unique IDs in ORACLE tables.

It was therefore critical to provide full reflexivity, i.e. the ability to inspect the structure of an object at run-time, obtain a list of attributes and attribute types, change the inheritance, and so on. In order to accomodate these needs with a minimum of syntactic sugar, the ADAM language contains some off-the-common-path features.

Adam is an object-oriented language falling in the category of prototype languages, since the class-instance duality was deemed to be unnecessary in our case. Specifically, prototype languages allow derivation through delegation to a prototype which may be a concrete object. In our case, for instance, we could choose a process to be exemplar for all related processes, without the necessity, for the user, to specify an abstract class and all its instances.

Adam is focused on data modeling; rules for the enforcer module are specified as collections of Adam objects, but the specification of the language does not encompass procedural semantics.

Objects are specified with a syntax vaguely resembling SGML; single inheritance is provided, and the part-of relationship is a primitive of the language. Objects share a common ancestor, named `thing`; attributes may belong to quite standard types, such as string, integer, double, date.

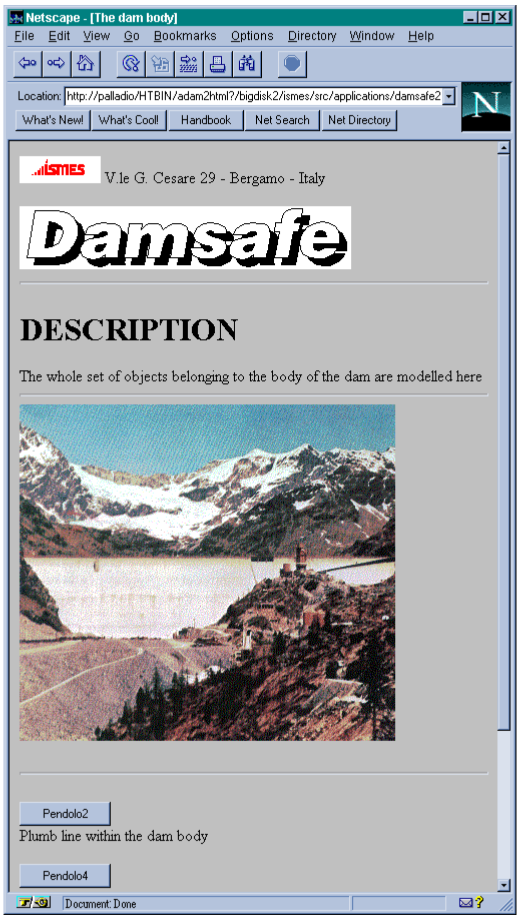
The following fragment describes the fact that instruments are things and that there is a instrument which is a hydrostatic balance; the database reference for the specific instrument is specified as a Web Uniform Resource Locator (URL).

```
<thing>
Identifier      TString Instrument
</thing>

<Instrument>
Identifier      TString Livellostato
description     TString "Hydrostatic balance"
name    TString Livellostato
ColonnaMidas    TString http://g50/DAMCGI_MDS/midas_str.sh?Diga%20di%20AlpeGera+
SOICMI+ALPE+1"
</Instrument>
```

ADAM supports quantity spaces as built-in types; they are heavily used in the description of discrete-state physical components, as well for representation of state and attributes of processes.

ADAM databases are contained in text files, which are read by an interpreter written in C++; the interpreter builds a set of C++ objects and provides an application program interface to the rest of the system, offering read and write access to the attributes, creation of new objects, changing of the parenthood status at run-time, as well as full reflexivity. This interface is used both by the reasoning agents and by the WWW server. Figure 1 illustrates a result.


Netscape - [The dam body]
File Edit View Go Bookmarks Options Directory Window Help
Location:
What's New! What's Cool! Handbook Net Search Net Directory
ismes

http://palladio/HTBIN/adam2html?/bigdisk2/ismes/src/applications/damsafe2

V.le G. Cesare 29 - Bergamo - Italy

Damsafe

DESCRIPTION

The whole set of objects belonging to the body of the dam are modelled here

Pendolo2

Plumb line within the dam body

Pendolo4

Document: Done

Figure 1: sample output

The server supports also modification of the database, through rewriting of the text files.

Since many data processing components in our company were based on an internal graphical format called IPF, we defined an additional MIME type (`application/x-ipf`), with associated viewers; therefore, the Midas system was able to transmit data in this format directly to the client workstations, having it correctly displayed.

Appendix A contains the complete formal Backus-Naur specification for Adam.

5. ACKNOWLEDGEMENTS

The development of DAMSAFE was supported by ENEL/DSR/CRIS (Research Centre on Hydraulics and Structures of the Italian Electrical Company ENEL).

We wish to thank Valerio Mazzoleni and Luca Varvello for the implementation of DAMSAFE and ADAM.

6. REFERENCES

1. D. Grime, T. Phillips and M. Waage, "The Kelly system: on-line expertise", *Hydro Review*, August 1988, pp. 36-41.

2. B. M. Franck and T. Krauthammer, *Preliminary safety and risk assessment for existing hydraulic structures - an expert system approach*, Report ST-87-05, Univ. of Minnesota, Dept of Civil and Mineral Eng., Inst. of Technology, 1987.

3. J.B. Comerford, J.H. Martin, D.I. Blockley and J.P. Davis, "On aids to interpretation in monitoring civil engineering systems", *Proc. of the IABSE Colloquium on Expert Systems in Civil Engineering*, International Association for Bridge and Structural Engineering, Zürich, Switzerland, 1989, pp. 219-228.

4. P. Salvaneschi, M. Cadei and M. Lazzari, "Applying AI to structural safety monitoring and evaluation", *IEEE Expert*, **10**(3), June 1996.

7. APPENDIX A.

The ADAM grammar is formally defined below.

Any input file may contain blank lines and comments, which arelines beginning with a '#'; we assume that blanks and tabs have been removed.

```
start ::= Definition*

Definition ::= QSpaceDef | ThingDef

QspaceDef ::=
        '<QSPACE>' CR 'Identifier' QSpaceId CR 'Description' Id CR QValueDef* '</QSPACE> CR

QValueDef ::= 'value' QSpaceValue CR

ThingDef ::= '<' Id '>' CR SomeDef* '</' Id '>'

SomeDef ::= AttributeDef | ThingDef

AttributeDef ::=
        Id          'TString'   Id CR
                    |
        Id          'TString'   '"[^"]*"' CR
                    |
        Id          'TDate'                 int int int CR
                    |
        Id          'TInteger'  int CR
                    |
        Id          'TDouble'   double CR
                    |
        Id          QSpaceId    QSpaceValue

QSpaceId::= Id
QSpaceValue         ::= Id

Id                  ::= '[a-zA-Z][a-zA-Z0-9_]*'
int                 ::= '{-}[0-9][0-9]*'
double              ::= '{[+-]}[0-9]*\.[0-9]*{[eE]{[+-]}[0-9]+}'
```