Development of a Workbench for Knowledge-Based Systems Using the ECMA Reference Model for CASE Frameworks

Yaron Shavit
CAP GEMINI INNOVATION
118, rue de Tocqueville. 75017 Paris. France
e-mail: shavit@capsogeti.fr

Abstract
The European Computer Manufacturers Association Reference Model for Computer Assisted Software Engineering Environments Frameworks (ECMA RM) provides a description of the services which should be provided by a Software Engineering Environments (SEEs) as well as of the relations between these services. Over the past year, the ECMA RM has become a de facto European standard reference in the field of SEEs. This paper constitutes one of the first hands-on experience-reports on the use of the ECMA RM in the design of SEEs. The ECMA RM's salient features are discussed and evaluated on the pragmatic basis of their useability in the development of a software engineering workbench to support KBS development. Concrete proposals are then made for enhancing the useability of the RM.

1 Background
The primary purpose of the ECMA RM is to provide developers of SEEs with a comprehensive set of criteria for evaluating or defining the functionalities of existing and future SEEs. Over the past year, the ECMA RM has become a de facto European standard reference in the field of SEEs. The ESPRIT project KADS-I1's aim is to produce a standard methodology and supporting software tools for the development of knowledge-based applications. Within KADS-II, the ECMA RM has been used as the basis for the Functional Specification of the KADS-II Workbench: a KBS-specific SEE.

This section consists of a slightly abridged description of the ECMA RM's origins, a definition of its scope and of its structure as provided by A. Earl [1], one of the central figures in the elaboration of the RM and the author of the RM document itself. A rationale for this paper, on the utilization of the ECMA RM, is then provided.

Origins
The Portable Common Tool Environment (PCTE) project, begun in 1983, has produced a public tool interface to be used as a portability interface and to provide support for the integration of software engineering tools. When the ECMA Technical Committee for PCTE [2] standardization was formed, one of the decisions was to aim to create a CASE environments Reference Model to assist the standardization process. During 1989 the first full version of the Reference Model was created, and during 1990 the Model was refined further. It is currently being evaluated by applying it to a number of CASE integration frameworks. The Reference Model is now an official ECMA technical report [3].

Scope
A Reference Model is a conceptual framework which allows experts to work productively and independently on the development of standards for each part of the domain of SEEs. Some of the major aims for a CASE environments Reference Model are:

- The Reference Model should be suitable for describing, comparing and contrasting existing and
proposed environment frameworks.

- The Reference Model should provide a framework for the smooth and coordinated evolution of future standards.
- The Reference Model should cover all system aspects irrespective of implementation techniques or software development methods employed.

Structure

The CASE environment frameworks reference model is based on grouping sets of services together. Figure 1 shows the overall structure of the reference model (also known as "the Toaster"...). The purpose of grouping the services as they are is that standards will cover at least a whole group. Thus the important interfaces between existing and forthcoming standards can be identified. Some of the groupings also enable various kinds of integration to be discussed: presentation integration (user interface services); control integration (task management services plus the message services); data integration (data repository plus data integration services).

This reference model identifies many of the accepted relationships between services. It does not dictate a set of relationships which must exist but rather the relationships identified within it should be regarded as sensible ones which can be found in existing systems.

The tool slots reserve a place for extending the facilities provided by the environment with integrated tool sets.

In addition to the ECMA RM break-down of CASE environment frameworks into about 30 possible services, it was found to be very useful to use another structuring device in the description of those services in order to ensure that descriptions of systems under review were compatible and comparable, and were clear and precise about what their system did and did not provide. Each service is therefore fully described through partial complementary descriptions of four dimensions.

The first dimension's aim is to identify the set of data manipulated by a service, the operations used to do that, and the constraints upon the state of the data. The second dimension's aim is to allow separate discussion of what a service is (conceptually), how it is implemented (internally), and the ways in which it is made available (externally). The third dimension simply aims to ensure that a proper distinction is made between instances, types and information about types (metadata). The fourth dimension captures some further points about a service: how well its function and role are understood, how it relates to other services, and what justification it has for being included in a particular environment.

Rationale for this paper

The ECMA RM is one of the sole available references for SEE development which enjoys some kind standard-like status. Moreover, in the short period since its publication it has won acceptance by some of the major software engineering technology vendors in Europe, in particular by members of ECMA (e.g., HP, IBM).

The high technical quality of the ECMA RM and its immediate acceptance in Europe have provoked a reaction in the U.S which is exceptionally rapid in relation to European technology; a version of the RM is currently being put to test by the National Institute of Standards and Technology in Washington D.C. [4].

In the light of the RM's rapid and significant penetration it appears interesting to study the results and conclusions of the application of the RM in the development of the KADS-II Workbench, one of the first applications of the RM anywhere (January through May 1991).

2 The Reference Model and KADS-II

The KADS-II project

The overall aim of the project is to develop a methodology for building knowledge-based systems which will become a European standard. KADS is a structured methodology based on conventional system development methodologies. It supports the distinct phases of analysis, design, coding, and testing. In addition, it provides techniques for the acquisition and modelling of human expertise and modelling user-computer interaction. KADS is already in widespread use in Europe and has been successfully applied in several application domains, from credit card fraud
Figure 1: The ECMA Reference Model structure
detection to car part moulding design. Of particular interest is KADS compatibility with many conventional system development methodologies. The emerging KADS standard will therefore be of particular interest to the companies wishing to integrate KBS with existing, conventional systems. The KADS-II project is based on the work of a previous ESPRIT project (KADS) which it is extending to cover the entire development life cycle and to formalize the expertise modelling language.

Use of the RM in KADS-II

Within KADS-II the ECMA RM was used as the principal support for the structuring and formalization of the functional requirements for the KADS-II Workbench. Our starting point was a 40 page informal description of the functionalities that ought to be supported by the Workbench. These functionalities had been derived from the CommonKads Methodology which the Workbench was to support. We then went on to map these functional requirements into the RM's various services and dimensions. We thus obtained a comprehensive functional description of the Workbench that we were to develop, which had the advantage of strongly suggesting the actual software mechanisms that needed to be implemented. Although the ECMA RM was not originally intended by its authors as a basis for the actual implementation of SEEs we think that our utilization of it as a basis for implementation is justified for two very pragmatic reasons. Firstly, virtually no standard frameworks are available today for this purpose. The ECMA RM's great advantage is that it is available! In addition serious work to enhance it is on-going within a recognized international standardization body. Secondly, we found that the RM's “disclaimer” about its use as implementation support was overly modest; some exploratory attempts in this direction which bore fruitful results convinced us that the RM could indeed be used as support for SEE development.

We found further motivation for using the ECMA RM, when we considered the framework in which the KADS-II project is being carried out, namely the CEC's ESPrIT program. Using the ECMA RM clearly fits in with some of the major goals common to all ESPrIT projects:

- utilize novel techniques when their use can provide the project results with a leading edge over "purely industrial" projects, which cannot take the "risk" of using this kind of technology;
- promote the use of promising nascent standardization efforts;
- promote the use of European technology.

Our conclusions concerning the use of the ECMA RM within the KADS-II project are discussed in section 5.

3 Evaluation of RM useability

3.1 Advantages of using the Reference Model

Interoperability support: The SEE market is at the threshold of a fully industrial era. The past few years have seen the concretization of several significant efforts in the field, in Europe (e.g., EAST, Eureka Software Factory) and in the U.S (e.g., HP's Softbench). Each of these systems has been developed according to its own proper paradigm. Nevertheless, one of the major conditions for the success of these, and other SEEs, is their ability to achieve data and tool interoperability within a real life, heterogeneous development environment. The existence of a single, comprehensive, common reference framework allowing to align existing systems and develop new compatible ones is a significant step towards achieving interoperability.

Exhaustivity: The ECMA RM, as it stands, covers most of the generic services and mechanisms which have been identified within various SEEs. It is therefore a comprehensive framework.

Quality: The ECMA RM, as it has been made available [3] is a high-quality document; its structure is clearly defined and consistent throughout, the rationale for most significant statements is made explicit, the description of the various services and dimensions is given at a level of abstraction which is relevant to most SEEs. As a result, the ECMA RM, in its “paper" form, is immediately usable.
Penetration: Since its adoption as an official ECMA document in December 1990, the RM has achieved significant acceptance within the European software engineering community. A permanent committee within ECMA is working on its enhancement. The elaboration of PCTE+, the standard European SEE framework, is being carried out on the basis of the ECMA RM [1].

Wide scope: The ECMA RM's approach to defining SEE functionality is to describe the services which need to be provided and then to examine each of them through 4 complementary and/or orthogonal aspects i.e., “dimensions” (broken down in fact into 12 “subdimensions”). This is a very realistic and open approach which takes into account the complex nature of SEEs, our fragmentary current knowledge of what makes them tick, and the high probability that novel services and the resulting new dimensions will continue to be identified in the near future. The ECMA RM therefore naturally lends itself to future enhancements.

An assertive approach: Although defined in terms which are general enough to be of widespread applicability, the ECMA RM takes an assertive, albeit implicit stance on many of the functional issues related to SEE development, thereby suggesting useful, high-level guidelines for SEE design.

Support for communication: The structure and style of the ECMA RM have proven to be natural support to members of a SEE development team for reasoning about and discussing their SEE’s functionalities and services.

The distinction between tools and services: The distinction between SEE functionalities supported by general-purpose, generic services (e.g., UI, data integration) and those provided by specific tools is a useful one. If followed rigorously, it promises to yield highly modular SEEs which naturally lend themselves to functional enhancements and extensions.

3.2 Deficiencies of the Reference Model

Fuzzy scope definition: The restrained scope definition of the RM as defined by its authors (i.e., the RM is a basis for elaborating standards but not for designing SEEs, the RM is more suitable for evaluating existing SEEs than for proposing new ones) is problematic:

- Because it is based on a structural paradigm, the ECMA RM implicitly, yet forcefully, suggests some architectural design principles, a good thing in itself, for these principles are sound. The ECMA RM is not, in fact, as paradigm- and implementation-independent as its authors would like (us) to believe; the architectural and conceptual framework “suggested” by the RM is, in fact, very much akin to that of PCTE.

- The “Using the Reference Model” section in [3] proposes a process for applying the RM to examine existing SEEs (e.g., “obtain interface specifications, User Manual, internal documentation”) whereas it is previously stated that the RM should not be applied to evaluate implementations. The user-reader of the RM is therefore at a loss in trying to discern what the precise scope and intended usage of the RM are. This leads to uncertainty as to what is “good” and what is “bad” utilization of the RM. Besides remaining “fuzzy” about the field of applicability, the section in the RM document providing guidelines for its application is also far too brief (i.e., hardly a page long!)

Lack of metrics: The developer-evaluator having acquired, through application of the RM, much useful and detailed information about his SEE’s different services and dimensions, is provided with no support for evaluating the meaning of this information nor on putting it to use e.g., what is the relative importance of two dimensions when applied to a given service, how is one to determine whether a certain dimension is “well covered” within a given SEE, does a particularly strong Data Integration Service make up in any way for loosely integrated User Interface Service.

Lack of interface description: The RM through e.g., its use of the “toaster” metaphor, puts a strong emphasis within its paradigm on the definition of inter-service interfaces. Yet, nearly no information explicitly concerned with this aspect is directly pro-
vided within the RM. Services and dimensions are described very much in a “stand-alone”, isolated manner, whereas their role within a SEE depends to a great extent on their interaction with the rest of the SEE’s components.

Paradigm dependence: The author of the RM claims that the RM “is not biased towards PCTE” [1]. Nevertheless, anyone familiar with PCTE and its components will easily recognize its fingerprints all over the ECMA RM. Far from being a disadvantage, the PCTE influence can even be considered as being a positive element inasmuch as PCTE is a standard, state-of-the-art framework for SEE development. Recognizing the PCTE influence surely seemed strategically risky to the authors of the RM, but this risk can be outweighed by the advantages of putting to use the experience acquired through several European PCTE-based projects (e.g., EAST, PACT, ATMOSPHERE).

Brevity of description: Many service descriptions provided within the RM are far too brief and “sketchy” to be of any practical use. The current RM document reads like an excellent draft version for a full-blown first version of the RM.

Lacking services and dimensions: No mention is made within the RM of several services which have their place in any SEE. Three prominent examples are:

- Help and Guidance support for the user of the SEE and its tools,
- Support for user modelling,
- Support for tool integration (dealt with in an all too brief manner (a single page)).

The RM in the development life-cycle: Perhaps the RM’s greatest weakness is the lack of clarity regarding its utilization within the global SEE development life-cycle. Indeed, no guidelines are provided regarding what stage of the process of software development should the RM be applied. Consequently it is never made very clear what input precisely feeds into the RM nor how the output resulting from its application can be further exploited. Surprisingly enough, our own experience was that, contrary to the authors claims, the RM is far more useful in elaborating an architectural design for a SEE than in defining its functionalities.

4 Utilization of the Reference Model in the development of the KADS-II Workbench

Within the development cycle of the KADS-II Workbench we applied the RM as the principal means of formalizing informal functional requirements within a structured framework as explained in section 2. This approach has proven to be a constructive and useful one. It has allowed us within the Workbench development team to define and discuss the Workbench functionalities which we chose to support on the basis of a common framework. Within this framework it became possible to comprehend any and all functionalities within a broad context, immediately as the first idea of a functionality was brought up and to evaluate its implications on other functionalities. In this respect the RM proved to be a valuable support: a high-quality comprehensive, structured checklist of services and considerations to reckon with when designing a domain-specific SEE.

The following difficulties were encountered in applying the RM to the design of the KADS-II Workbench:

In describing existing services:

- The extensive care taken so that the RM remains generally applicable has resulted in “excessive open-endness”. This is due to an emphasis put on describing the services themselves and a lack of attention paid to the definition of the RM’s structuring elements (e.g., inter-service interfaces). As it turns out, almost any odd grouping of CASE tools can be made to fit artificially the RM’s service definitions, giving the impression that it actually complies with the implicit PCTE-like architecture. This was probably not the authors’ intention, yet their insistence on making the RM “architecture and paradigm independent” has had this perverse side effect.

In defining new SEES:

- the descriptions of most services and dimensions is given in a language which lends itself only with
great difficulty to the application of these descriptions to the actual definition of services in a SEE implementation. Indeed, many of the descriptions that we have come up with in our Workbench specification document read more like extensions and enhancements of the RM text rather than as their application to the KADS-II Workbench's functional requirements.

- no room is provided within the RM for the definition of new generic services available across the environment, such as the existing data integration and UI services. For instance, for the KADS-II Workbench we were required to introduce generic Reasoning Services necessary for supporting the development of any KBS. As these could not be considered as mere "tools" we were obliged to take a decision concerning the generic structure of the RM itself in order to implement a specific functionality in the Workbench.

Having concluded the application of the RM to KADS-II functional requirements we were faced with the dilemma of how to proceed with development on the basis of the voluminous RM-based description of the Workbench i.e., how was this 100-page document to feed into the Architectural Design of the Workbench. For lack of guidelines within the RM we decided to use the results of the RM in two ways:

- as an informal source of inspiration for the architectural design inasmuch as many of the Workbench's software components had suggested themselves through the RM-based functional description,
- as a cross reference with which to validate the components of architectural design.

We believe that application of the RM was a very useful contribution towards defining the functionalities needed in a state of the art special-purpose SEE. Nevertheless, the RM, as it stands today, can hardly be used as anything more than a detailed functional checklist, albeit an excellent one. Taking the following actions can enhance the useability of the RM:

Focus the scope: We suggest that those in charge of the ECMA RM's evolution take a clear stand on the RM's scope e.g., should it be applied only to the evaluation of existing SEEs or can it serve as the basis for specifying new SEEs? should it be used only in relation with the definition of Frameworks or should it be used to evaluate and specify specific SEEs? is the RM truly "architecture-independent" or should it "push" its users towards making some inevitable design decisions?

We strongly feel that instead of being simply an evaluation vehicle the RM should be used proactively. Carried further this may even become a methodology for SEE development.

Enhance the text: Most of the Service and Dimension descriptions need to be significantly enhanced. As mentioned previously, most dimension descriptions are too short (2-3 lines) to be of any practical use.

Provide tool support: Providing simple tool support for using the RM will go a long way towards increasing its audience within the software engineering community. A good start would be the use of a commercially available hypertext package as the basis for a simple RM CASE support (editor/browser/query) allowing to acquire, classify and relate relevant information according to the RM "dimensions". Thus, for instance, the user could obtain a view of the Conceptual dimension of all services or of the metadata dimension of all data-related services.

Provide guidelines for utilization: A separate document is necessary which defines, with examples and illustrations, how the RM should be used for evaluating and specifying SEEs.

Acknowledgements

The work described in this paper was carried out as part of the KADS-II project (P5248) partially funded
by the CEC's ESPRIT program. The Partners in this project are Cap Gemini Innovation (F), Cap Gemini Logic (S), IBM France (F), Ilog (F), Netherlands Energy Research Foundation ECN (NL), ERITEL SA (E), Lloyd's Register (UK), The Swedish Institute of Computer Science (S), SIEMENS AG (D), Touche Ross MC (UK), University of Amsterdam (NL) and Free University of Brussels (B).

The KADS-II Workbench was developed under the supervision of C. Toussaint (CGInn). An important contribution to the Workbench's architectural framework was made by J. Wielemaker (UVA). Mari Georges (CGInn), KADS-II project leader, helped with very useful comments throughout the development of the Workbench. M. Schlumberger (CGInn) contributed useful comments to this paper. A special thanks to I. Simmonds, of SFGL (F), for many enlightening discussions on Software Engineering Environments.

References


