Framework of Fully Integrated Hybrid Systems

A. Santos, J.J. Romero and A. Pazos

Department of Communications and Information Technologies, School of Computer Science, A Coruña University, 15071 A Coruña, Spain

A framework of fully integrated Hybrid Systems (HS) is proposed for the development and management of HS which involve Databases (DB), Advanced User Interfaces (AUI), Symbolic Systems (SS) and Artificial Neural Networks (ANN). This framework provides a common input-output interface among those HS modules developed on the framework, with a completely two-directional flow control and a highly parallel processing. This integration framework facilitates the incorporation of heterogeneous modules, together with their subsequent management and updating.

Keywords. Advanced user interfaces; Artificial neural networks; Databases; Data mapping; Symbolic systems

Correspondence to: A. Santos, Department of Communications and Information Technologies, School of Computer Science, A Coruña University, 15071 A Coruña, Spain. E-mail: nino@udc.es

1 Introduction

HSs can be divided into three categories [1-4]: loosely coupled, tightly coupled, and fully integrated.

Loosely coupled Hybrid Systems (HSs) include Artificial Neural Networks (ANNs) and Symbolic Systems (SSs) separated and interlinked by means of an one-directional information flow and serial processing. Usually, the relation between both approaches is limited to the fact that one of them carries out the other's pre or post information processing, or to a master slave relation. This kind of systems does not allow a parallel processing of the different types of modules involved, either symbolic or connectionist ones, it only provides a sequential type of processing. In addition, the one-directional communication flow among the modules does not allow a high degree of cooperation or integration, which would be desirable to improve the potential of the developed systems. On the other hand, a master-slave kind of configuration between the connectionist and the symbolic modules does not facilitate a thorough integration.

Tightly coupled systems include common representation structures which allow two-directional information exchanges between connectionist and symbolic modules enabling parallel processing. This level of integration does allow a parallel processing of the modules and a two-directional flow. Both loosely and tightly coupled systems usually include

a control system or module which rules the system's global functioning, and to a certain extent, its sequentialisation. Tightly coupled systems allow a certain parallelisation of the modules. One of their disadvantages is that these systems' functioning scheme is very difficult to control, which makes their development and maintenance terribly complicated. In addition, these control structures and modules are usually specific to each system and non-reusable.

Fully integrated systems incorporate both modules in the same system, which usually includes a common input-output interface. These systems' flow control is two-directional and it allows a highly parallel processing among the modules involved. This kind of system comprises the possibilities of loosely and tightly coupled ones. Therefore, it provides more advantages towards a general application of HSs. A greater effort must be made at this integration level to carry out a thorough integration which facilitates the development of HSs [5, 6].

One of the researched possibilities has been the use of a homogeneous interface for the various modules based on a set of function calls. Although this approach allows the development of fully integrated systems, it poses great difficulties for the subsequent maintenance. In addition, we should not only think about a symbolic-connectionist approach, other modules must also be taken into account, since they are

absolutely necessary for a correct software and knowledge engineering. In this regard, Databases (DBs) and Advanced User Interfaces (AUIs) are also included as modules of the integration framework.

In the field of HSs-DBs integration, we can generalize the *Al-Zobaidie* and *Grimson* classification according to their degrees of coupling and the control allocation [7]:

- Enhanced HSs. Extended data management facilities are incorporated into the HSs.
- Intelligent/deductive DBs. In this type of systems, deductive and connectionist components are embedded into DB management.
- HSs-DBs with communication. DBs management and HSs are independent with some form of communication between them. Also, the communication can be classified into three classes: data definition, DB maintenance and administration, and data manipulation.

According to the definition, only the third kind of approach will allow the development of fully integrated HSs. Its main advantage is that an existing DB can be used. The DB management system is an independent one, so it can also operate as a totally independent system.

For developing the framework a series of integration levels has been proposed, ranging from the most simple and easy to formalize ones to the most complex and subjective. Our framework constitutes the last

integration level. The specific characteristics of the different integration levels and the functioning pattern of the framework are reviewed next.

2 Level I: Information Systems. DBs-AUIs integration

This level refers to the integration of DBs with AUIs, providing the most adequate interface in each case. The following requisites constitute the basis for the development of this first integration level:

- Reusable interfaces
- Adaptable interfaces
- Development facilities
- Optimizing the development times
- Decrease of rejection possibilities
- Access control and security possibilities
- Possibility of integration at other levels

The interface modules must be reusable and adaptable to each user's specific characteristics reducing the interfaces' development costs and times. The framework's first level must facilitate the manipulation of the user's interfaces, improving considerably the design and AUIs integration possibilities. The proposed scheme must incorporate the DBs advantages (integrity, consistency, security and the possibility of query) to the user's interfaces [8, 9]. Figure 1 shows the elements of the first level.

Three elements are differentiated in this first level's framework:

- An AUI Management Server (AUIMS) in charge of managing the user's interfaces indexed in a Forms DB (FDB).
- The second element is constituted by the set of servers of those application domain DBs which are part of the client forms in the FDB.
- The third element is integrated by all those client applications which may apply interface services to the AUIMS.

The AUIMS is an independent system which stores the client applications interfaces in the FDB. The AUIMS is structured into two modules (Fig.1):

- The Forms Management Module (FMM)
- The Forms Runtime Module (FRM)

The FMM has been structured into three submodules:

- Development environment
- Lexicographic analyser
- FDB manager

The development environment allows the creation of data input and output forms, linked or not to one or more DBs. This module provides a graphic environment which allows the creation of new forms, the selection of the already existing ones, the runtime and deletion of forms, the selection of the different controls, the definition of the controls' properties and features and those of the associated code.

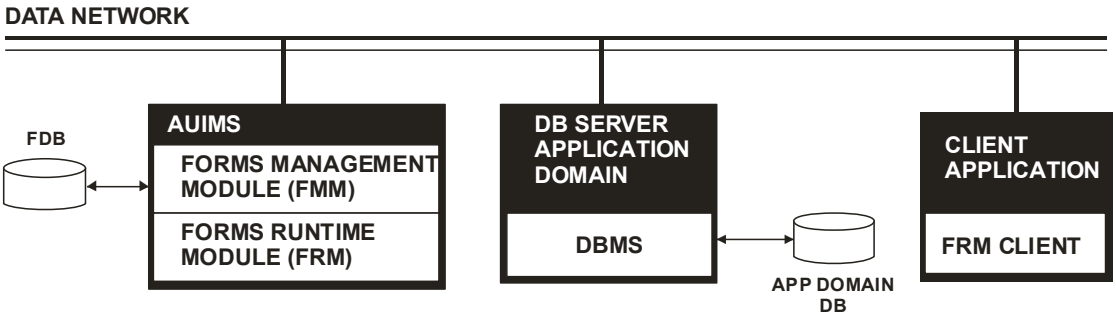

Fig.1 Integration level I

The lexicographic analyser allows the addition of code to the forms controls events. A language associated to the various control events has been designed. This module analyses and translates those expressions linked to the forms controls events. These expressions are translated into a format which can be interpreted by the runtime system. The compiled expressions are translated into a format which can be understood by the user. This module's analyser is a recursive descendant, carrying out a syntactic check by means of a downwards construction of the syntactic tree. A thorough description of the FDB model and of the syntax of the various sentences of the form definition language can be found in [10].

The FDB manager translates and interprets the forms' definition. This module translates the objects, properties, and relations of the developed forms into the FDB table registers. The FDB models all the information of the forms generated from the development environment; their properties, controls, restrictions, and relations. Therefore, the FDB

will have all the different forms generated by the AUIMS administrator, by the various client applications programmers or by the final users.

The FRM, the second AUIMS module, is in charge of the screens runtime, composing the different forms dynamically from their FDB definition. The client application programmes are structured as a series of calls to the FRM executed by the corresponding forms. This functioning pattern is oriented towards the events of the running forms. The FRM is composed of three submodules:

- The generator
- The runtime
- The DB access

The generator module interprets the definition of the FDB forms and it composes the screen in a dynamic way.

The runtime module runs the code linked to the forms definition. This module has the same lexicographic analyser as the FMM to do that. This module keeps a table of crossed references among the possible control events and the code which must be executed in each case.

The DB access module queries the data when the forms fields are linked to a DB. Most of the form elements in the FDB refer to the DBs, tables and fields involved in the client applications. This module

guarantees the access to the various DB servers which provide inputs and outputs to the system's forms.

The AUIMS integrates the corresponding forms into any final application. To achieve that, the FDB includes a relation to the entity which models the client applications and the one which models the forms. Thus, the AUIMS allows local and remote applications to use the generated forms on a client-server architecture. In case of remote client applications, the FRM must be included in the client hosts and then load and run the forms requested by the final applications. It makes this level independent from the client applications, facilitating the integration into fully integrated HSs, incorporating a high degree of parallelism and co-operation among heterogeneous systems by means of two-directional information flows.

3 Level II: Integration of SSs into Level I

Level II defines the integration of SSs into the Level I. The following requirements are contemplated for this second level:

- Facilities for developing and maintaining SSs, decreasing costs and development times.
- Capacities of integration into the Level I.
- Capacities of integration with other developments based on specific SSs or on knowledge engineering tools.

The aim must be the integration of both programmed SSs and those developed on knowledge engineering tools. Two different and non-exclusive functioning approaches can be defined, the first directed by the client application and the second directed by the AUIMS. The goal is to find a scheme which provides a two-directional flow between DBs and SSs, a single homogeneous interface and a high degree of parallel processing on a fully integrated HSs.

The approach directed by the client application comprises both programmed SSs and those developed on knowledge engineering tools. The SSs do not make use of the development and maintenance facilities of symbolic modules provided by the AUIMS. They only make use of the I Level by means of calls to the FRM for the execution of data input and output forms. The integration of I Level forms provides an information flow between SSs and DBs. The addition of calls to the AUIMS into this kind of system will allow the execution of forms as input source order of the knowledge bases during inference processes. This approach facilitates a quick integration of already existing SSs not developed on the AUIMS facilities. These developments cannot be considered to be fully integrated HSs with totally two-directional information flows and a totally parallel processing. This integration pattern can be considered loosely or tightly coupled nature.

The approach directed by the AUIMS is a clear extension of the previous one, incorporating a whole set of SSs management and maintenance tools which provide the HSs with the capacities of fully integrated ones. The AUIMS is in charge of suggesting to the client applications a set of possible symbolic modules which are relevant to the corresponding interfaces or call parameters. As support to the AUIMS decisions, the I Level FDB incorporates a whole set of entities which allow the modelling of those structures and relations involved in the SSs (knowledge bases, classes, objects, properties, slots, hypotheses, and rules). This second approach allows a high degree of parallel processing on a completely two-directional flow among the integrations' modules. Figure 2 shows the elements of the second integration level.

The second level is structured into the following elements:

- The Advanced User Interfaces and Symbolic Management Server (AUI-S-MS) which substitutes the I Level AUIMS.
- The set of DB servers which are part of the forms and symbolic modules managed by the AUI-S-MS.
- All those applications or client modules held by the AUI-S-MS, both conventional and symbolic ones.

The AUI-S-MS stores both the interface structures and those of the symbolic modules into the Forms and Symbolic Data Base (F-S-DB),

which substitutes the I Level FDB. The AUI-S-MS is structured, as may be seen in Fig. 2, into three modules:

- The Symbolic Definition Module (SDM), which is incorporated into the AUI-S-MS at this integration level.
- The Forms and Symbolic Management Module (F-S-MM), which substitutes the FMM in I Level.
- The Forms and Symbolic Runtime Module (F-S-RM), which substitutes the FRM, both in its server and client versions.

The SDM module is in charge of the whole management (additions, deletions, modifications and queries) of the F-S-DB tables which model the different entities of the SSs domain (knowledge bases, classes, objects, proprieties, slots, hypotheses, and rules).

The second level's F-S-MM incorporates the three submodules of the previous level's FMM (development environment, lexicographic analyser, and F-S-DB manager). Each and every one of the submodules is adjusted to serve the integration needs of the SSs [10].

The development environment includes the possibility of defining a new set of proprieties linked to the forms elements, considering the knowledge engineers as the system's users at this level. These proprieties are aimed at relating forms elements or fields with the properties or slots held by the F-S-DB after being defined by the SDM. It allows the establishment of

relationships among the forms objects and those elements which characterise symbolic modules. In this regard, the relations between forms and knowledge bases can be defined, forms and hypotheses, controls and knowledge bases, controls and hypotheses, fields and proprieties, etc [10].

The F-S-MM lexicographic analyzer incorporates new sentences with respect to that of the I Level. These sentences allow the definition of triggers on the relations defined among the specific elements of the forms and those which characterise symbolic modules. The expressions linked to the relations among the forms elements and those of the symbolic modules are analysed and translated in this module into a format which can be interpreted and executed by the F-S-RM.

The F-S-DB manager, apart from the tasks carried out in the level I, also have to translate each of the relations at a symbolic level, together with their proprieties, restrictions and triggers into new registers. It allows a total integration of the symbolic level with the level I.

The F-S-RM serves the interface service requirements generated by the client modules for their runtime, whether of a symbolic nature or based on classical approaches. The F-S-RM uses the defined relations available in the F-S-DB to determine the communication interface among those modules involved. The information flow among all the modules is two-directional and with parallel processing.

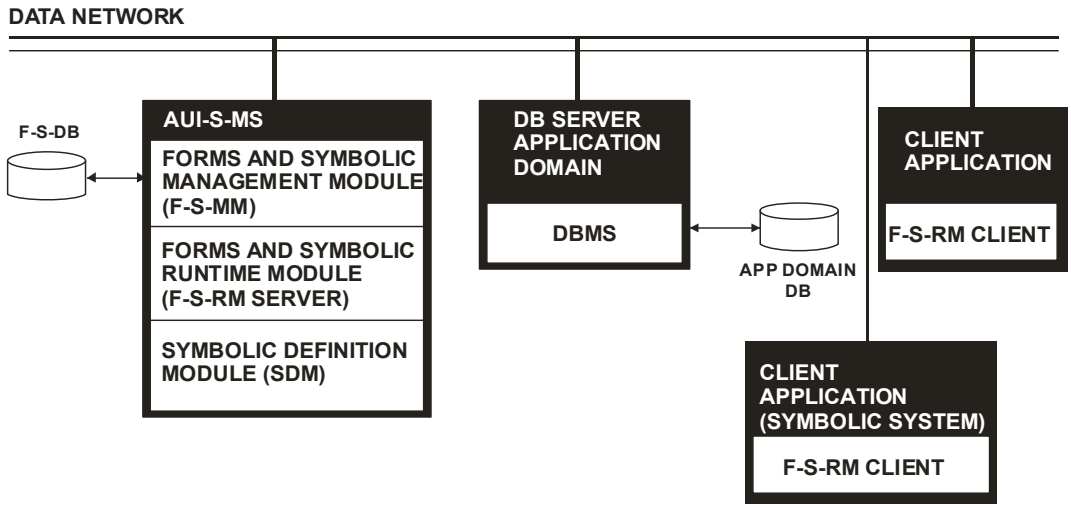

Fig.2 Integration level II

The F-S-RM incorporates the three modules from the previous level (generator, runtime and DB access) to support this new level. The runtime module integrates the new F-S-MM lexicographic analyser, including in the table of cross-references the list of calls among applications with their respective interface parameters. This module is in charge of running the corresponding calls with the right parameters among the modules or client applications, and of redirecting the various resulting information channels. When the client applications, both of a symbolic nature and based on classic approaches, are remote from the AUI-S-MS, the F-S-RM client module must be included in the remote host to access the services provided by the F-S-RM server.

As a result of this second integration level, the AUI-S-M incorporates into the framework a set of tools and languages which allow the definition

of the SSs structural and organic characteristics, facilitating their development and a complete integration with DBs and AUIs.

4 Level III: Integration of ANNs into Level II

Level III includes the advantages of integrating all the stages of the ANNs life cycle (design, training, test, and runtime) into I and II levels. It will have to support all those problems which require learning capabilities, fault-tolerant processing, and a certain generalizing capacity. These third level requirements can be summarized as folllows:

- Development facilities of all the ANNs life cycle stages.
- Integration facilities of ANNs into I and II levels.
- Management facilities of main ANNs architectures and models, both in local and remote environments.
- Reusable ANNs modules.
- Access control, integrity and security levels.

The possibilities of integrating the different stages of the ANNs life cycle with DBs, which represent the Level I, are apparent. DBs offer a great potential for supporting the structures of ANNs (weights, thresholds, outputs, error and learning rates, activation functions, learning rules, training, and test sets) [11]. The storing structures shall refer to the DBs of the work domains of each supported ANN, given that those structures

support the training and test stages. The framework must facilitate the management of great quantities of ANNs with many different models and architectures, generalizing their integration with every type of development to use in runtime. Level III provides access to the applications of the I and II levels to the framework connectionist modules, and vice versa. The integration framework ensures the information exchange among heterogeneous systems throughout a data network, homogenizing their interfaces, DB accesses, and information and co-operation exchanges. Figure 3 shows the elements of the III integration level.

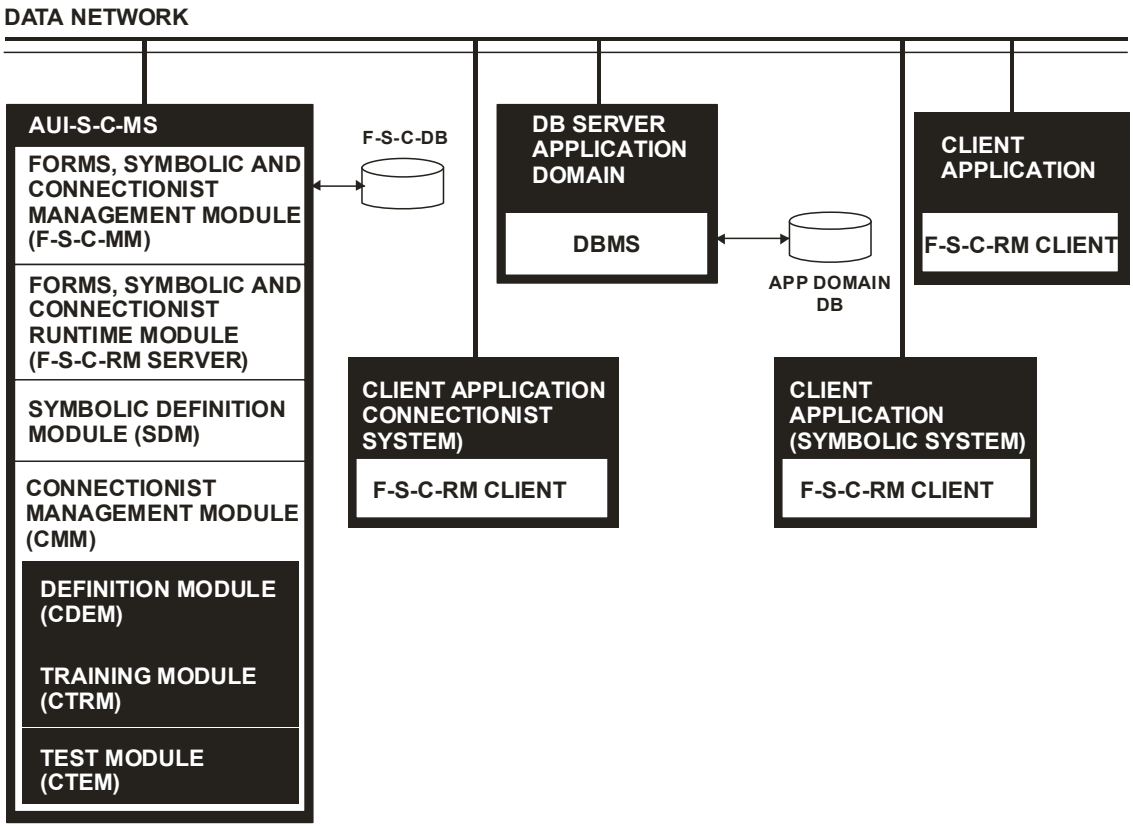

Fig.3 Integration level III

At this level the Advanced User Interfaces-Symbolic and Connectionist Management Server (AUI-S-C-MS) substitutes the AUI-S-MS and the Forms-Symbolic and Connectionist Database (F-S-C-DB) substitutes the F-S-DB.

The AUI-S-C-MS has been restructured into the following elements:

- A new module, the Connectionist Management Module (CMM).
- The Forms, Symbolic and Connectionist Management Module (F-S-C-MM), which substitutes the F-S-MM.
- The Forms, Symbolic and Connectionist Runtime Module (F-S-C-RM), which substitutes the F-S-RM.

The CMM has been structured into three submodules:

- The Connectionist Definition Module (CDEM).
- The Connectionist Training Module (CTRM).
- The Connectionist Test Module (CTEM).

The CDEM is oriented towards the design of ANN structures. This module manages the addition, deletion, modification, and query processes of the ANN structures supported by the new F-S-C-DB tables.

The second module or CTRM is oriented towards ANN training. This module includes a whole set of calls designed for training the structures developed by the CDEM. The CTRM manages the queries of the DBs in

the application domains which will integrate the training sets. For example, in a supervised learning scheme, the query must include all those fields which constitute both the input patterns and the desired output ones. In the case of a non-supervised scheme, the second type of patterns disappears. This query carries out a joint operation among the tables which are involved in the learning process and a projection on the relevant fields of the training set. A third type of field is included in the query, which provides the semantic descriptions of the input and output fields. This third type of field allows the learning process to use concepts and descriptions instead of the codes or values used by ANNs. This module has access to any application DB, both local and remote, which we desire to integrate into the process. In this sense, the I Level provides us the capacity of DB management.

The last module or CTEM is in charge of testing the ANNs. This module includes different forms oriented towards testing the framework's own ANNs. The test process requires a functioning scheme which is similar to the one used in the learning stage, with regard to the elaboration of the test set. This module determines the generalising capacity of the ANNs. The CTEM provides as output, depending on the case and after the test process, from an error rate to a sample confusion matrix [12].

To achieve the integration of this level into the global AUI-S-C-MS, the F-S-C-MM incorporates, with regard to the previous level, the possibility of defining a new set of proprieties linked to the forms elements. These proprieties are oriented towards relating form fields or elements with the ANN process elements, both input and output ones. All the ANN elements are defined in the F-S-C-DB. The F-S-C-MM is in charge of keeping all this information. Thus, various relations may be defined among the elements of the different forms of the client applications, both based on classic approaches, and symbolic or connectionist ones. The annex section of [10] includes a thorough description of the framework's interfaces, controls, proprieties and language.

The F-S-C-MM lexicographic analyzer incorporates to the II level's language a set of new sentences which allow the definition of triggers on the II level relations established by the F-S-C-MM. As in previous levels, the F-S-C-MM is in charge of analysing and translating the expressions linked to the different relations defined into a format which can be interpreted by the F-S-C-RM, so that it can be executed. Once again, the definition of these relations allows information to flow among every kind of module (based on classic approaches, and symbolic or connectionist ones), in a completely two-directional and parallel way, defining a whole set of possible communication channels among the applications.

In turn, the general DB manager of the framework or F-S-C-DB is in charge of translating each and every one of the ANN relations, their properties, restrictions, and triggers into the F-S-C-DB.

In runtime, client applications require services from the F-S-C-RM, either of interfaces and access to I Level DBs or of services to symbolic modules of the II level or connectionist ones of the III. As a result of the different forms runtime, the F-S-C-RM suggests the runtime of a whole set of possible ANN modules, symbolic ones or those based on classic approaches, according to the relations and triggers defined. The F-S-C-RM also uses the defined relations to determine the communication interface among the modules involved. The information flow among all the modules is two-directional, and with parallel processing, providing simultaneous support to client applications of diverse nature. The F-S-C-RM incorporates the F-S-C-MM lexicographic analyser to support the runtime of this third level including in its crossed reference table the various calls among applications with their respective interface parameters. The F-S-C-RM will still be in charge of running the code associated to the forms definition, and of running the corresponding calls with the right parameters among the various modules or client applications, and of redirecting the multiple information channels. When the client applications are remote

from the AUI-S-C-MS, the client F-S-C-RM version must be included into remote hosts to access the services provided by the F-S-C-RM server.

The main features of our third level framework have been tested in the environmental impact assesment domain [10, 11, 13]. The subsequent knowledge extraction of this core, and its generalisation including SSs, has resulted in our fully integrated framework of HSs.

5 Conclusions and future works

The three levels of our framework provide us the capacity of integrating different systems (modules based on classic approaches, DBs, AUIs, SSs, and ANNs). It facilitates the development, adaptability, and management of a great variety of fully integrated HSs. The use of a complex DB, the F-S-C-DB, as the framework's general catalogue, provides integrity, consistency, possibility of defining restrictions, reuse capacities, and different security levels for the HSs.

The client-server scheme of the AUI-S-C-MS, supported by the F-S-C-RM, approaches the integration possibilities to every kind of environment and user. From the point of view of the integrating nature, a high degree of co-operation and full integration is provided by means of the management of multiple two-directional intercommunication channels among the modules to be integrated. It provides a parallel support to all the integration

levels. The system's parallel capacities are double, on the multiple channels of each HS, and on the multiple channels of the various HSs.

The AUI-S-C-MS functioning scheme allows the addition of other approaches into the integration framework. The future incorporation for the full integration of genetic algorithms in our framework will incorporate the solution-search possibilities of these algorithms [14-16] into the developed HSs. This future incorporation will make a new framework of Advanced User Interfaces, Symbolic, Connectionist and Genetic Algorithm Manipulator Server (AUI-S-C-GA-MS) to substitute the present AUI-S-C-MS. The huge modularity and integrating capacity of our framework will allow the incorporation of other types of approaches in future versions, for example evolutionary strategies [17, 18], immune systems [19], etc.

Finally, it must be said that with each new HS, the F-S-C-DB will grow in very valuable information. For instance, it can be used for the research of the symbolic-connectionist integration from two different approaches; that is, the unified neural architectures in symbolic interpretations [20] and the transformation architectures between both representations [21]. Other possibilities of research could be oriented to the AUIs field, analysing aspects such as design and usability [22]. Once again, the available data in the F-S-C-DB and those to be incorporated with each development will allow to increase our knowledge in this and other research fields.

References

1. McGarry K, Wermter S, MacIntyre J (1999) Hybrid neural systems: from simple coupling to fully integrated neural networks. Neural Comput. Surv. 2:62-94

2. Wermter S, Sun R (2000) Hybrid neural systems. Springer, New York

3. Sun R (2002) Hybrid systems and connectionist implementationalism. In: Encyclopedia of Cognitive Science, Nature Publishing Group (MacMillan), London, pp 697-703

4. Sun R (1994) Integrating rules and connectionism for robust commonsense reasoning. Wiley, New York

5. Corchado E, Corchado JM, Abraham A (2007) Innovations in hybrid intelligent systems. Springer, Heidelberg

6. Christos C, Giua A, Seatzu C, Zaytoon J (2006) Analysis and design of hybrid systems. Elsevier, Alghero

7. Al-Zobaidie A, Grimson JB (1987) Expert systems and database systems: how can they serve each other?. Expert Syst. 4:30-37

8. Codd EF (1972) Derivability, redundancy and consistency of relations in large data banks. IBM research report RJ 599

9. Codd EF (1982) Relational database: A practical foundation for productivity. Commun. of ACM 25:109-117

10. Santos A (1998) Methodology for hybrid systems including expert systems, artificial neural networks and databases. PhD thesis. University of A Coruña, A Coruña

11. Santos A, Arcay B, Dorado J, Pazos A (2002) Artificial neural networks manipulation server. Research on the integration of data bases and artificial neural networks. Neural Comput & Appl 11:3-16

12. Weiss SM, Kulikowski CA (1991) How to estimate the true performance of a learning system. In: Weiss SM, Kulikowski CA (eds) Computer systems that learn. Morgan Kaufmann, San Francisco, pp 17-49

13. Walker LJ, Johnston J (1999) Guidelines for the assessment of indirect and cumulative impacts as well as impact interactions. Office for official publications of the European Communities, Luxembourg

14. Goldberg DE (1989) Genetic algorithms in search, optimization and machine learning. Wesley, Massachusets

15. Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Michigan

16. Holland JH (1996) Complex Adaptive Systems. In: Pazos A (ed) Artificial neural networks and adaptive systems. University of A Coruña Press, A Coruña, pp 259-295

17. Fogel LJ, Owens AJ, Walsh MJ (1966) Artificial intelligence through simulated evolution. Wiley, New York

18. Fogel DB (1995) Evolutionary computation: toward a new philosophy of machine intelligence. IEEE Press, New Jersey

19. Dasgupta D (1999) Artificial immune systems and their applications. Springer, Berlin

20. Feldman JA, Ballard DH (1982) Connectionist models and their properties. Cogn Sci 6(1):205-254
21. Tickle A, Maire F, Bologna G, Andrews R, Diederich J (2000) Lessons from past, current issues and future research directions in extracting the knowledge embedded in artificial neural networks. In: Wermter S, Sun R (eds) Hybrid neural systems. Springer, Heidelberg, pp 230-245
22. Teodorescu H (2000) Intelligent systems and interfaces. Kluwer Academic, Boston