A Guided Tour of the Fibonacci System: Taste of Time

Antonio Albano[1,*], Carla Brasini[2,*], Milena Diotallevi[2,*], Giorgio Ghelli[1] and Renzo Orsini[3,*]

[1]*Università di Pisa, Dipartimento di Informatica*
[2]*Engineering Ingegneria Informatica S.p.A., Laboratorio Ricerca & Sviluppo*
[3]*Università di Venezia, Corso di Laurea in Scienze dell'Informazione*

Abstract

At the beginning of the eighties, if you had to operate on persistent data with a complex structure, you had to go through a cicle of reading data from storage, converting it in main memory format, operate on the main memory format, convert back to storage format, save the converted data. Somebody had to do something about this.

1. Introduction: Galileo and Fibonacci

At the beginning of the eighties, if you had to operate on persistent data with a complex structure, you had to go through a cicle of reading data from storage, converting it in main memory format, operate on the main memory format, convert back to the storage format, save. At the Computer Science Department of Pisa University, Antonio Albano began the Galileo project with the aim of designing a new approach that could end all that hussle: a modern and robust language and system where you could just operate on complex persistent data as if it where a local variable of your code, without any need to convert, to load, to save [1] (other groups were studying the same idea, we do not claim he was the only one [2, 3]) .

This sounds ambitious, but Galileo was actually much more than this. First of all, it abandoned the relational data model that was dominating in that period in favour of a much more expressive "semantic data model". Then, it exhibited a *static* and *strong* type system, which means that every type error was detected at compile time, with no exception. It was a mostly functional programming language, meaning that side-effects were limited to a statically delimited amount of updatable locations, paving the way for simpler approaches to automatic verification of program properties and to powerful code optimization techniques [4].

The static type system of Galileo was inspired by the language ML, designed by Robin Milner in the seventies [5], but it substituted polymorphism with *subtyping*, in order to better support complex persistent data like those of conceptual models of databases.

The first paper on Galileo appears in 1983 [6], which is the same year when Stroustrup introduced C++. The connection between objects in the C++ style and Galileo subtyping was immediately clear, and Galileo played a big part in the frenetic research of that period about

SEBD 2023: 31st Symposium on Advanced Database Systems, July 02–05, 2023, Galzignano Terme, Padua, Italy
*We list here the authors of the original paper with their affilitiations in 1994; we have not been able to contact Carla Brasini and Milena Diotallevi for the redaction of this note

the idea of "object-oriented databases": in the middle of the eighties the Galileo dream of a database where the "impedence mismatch" between persistent data and program variables would disappear seemed, for some years, to be adopted by many researchers in the field, and it even appeared in some commercial products.

In the meanwhile, the Galileo project was moving along new lines. First of all, the quest for a better integration of object-oriented features with strong subtyping [7]. These were the years when graphical interfaces started their diffusion, and the Galileo project went in that direction as well [8].

An important contribution of the project was the design of new linguistic mechanisms, that were absent in traditional object-oriented languages but were important to allow Galileo to be used for conceptual models. In this context, the main contributions of the project were the notions of *roles*, *virtual objects*, and *named associations*.

Roles and *virtual objects* were introduced in order to give objects the ability to change thier class during their lifetime, and to extend to strongly typed objects the capabilities that *views* provide to relational databases [9, 10, 11]. Regarding *Named associations*, object-oriented databases use object references in order to represent associations, but references are directed while associations are bi-directional, and references represent unary associations, which association can also be many-to-many. *Named associations* were a typed mechanism parallel to the class mechanism designed to solve these problems [12].

While Galileo was evolving in order to incorporate these mechanism, its type system was showing its limits, which derived from the initial choice of trading parametric polymorphism for subtyping. Advances in the study of type theory, some of which arrived from the Galileo group itself, proved now that it was possible to have a language supporting the two mechanisms at the same time [13, 14].

The combination of parametric polymorphism and subtyping is difficult but is extremely powerful. Around 1991, the Galileo group, while continuing its work on new versions of the language and system Galileo [15], began the definition of a new Database Programming Language, called Fibonacci, which would inherit all the ideas and innovations introduced by Galileo, but would embed them into an extremely rich type system, able to support new mechanisms for code reuse and modularization [16, 17].

The SEBD community followed this stream of innovations, since the first Fibonacci papers were presented in 1993 and in 1994 to SEBD, before the publication of the Fibonacci definition in VLDB Journal in 1995 [18, 19].

Fibonacci was the apex of the Galileo project, and its conclusion. The century was closing down, new research themes were attracting the group interest, such as semi-structued data, XML data, mobile ambients, graph databases, data warehouses, peer-to-peer databases. In the world of commercial databases many new systems were now object-relational, mainstream languages, such as Java, were starting to combine subtyping with parametric polymorphism, some NoSQL system were trying to address the impedence mismatch problem, the group decided to move onwards.

We do not yet understand why so many programmers have still to load data, convert, operate, convert back, save the converted data, forty years after Antonio Albano and others studied many different approaches to free the programming world from this legacy, but, at least, we tried to make our part.

References

[1] A. Albano, L. Cardelli, R. Orsini, Galileo: A strongly-typed, interactive conceptual language, ACM Trans. Database Syst. 10 (1985) 230–260. URL: https://doi.org/10.1145/3857.3859. doi:10.1145/3857.3859.

[2] J. W. Schmidt, Some high-level language constructs for data of type relation (abstract), in: D. C. P. Smith (Ed.), Proceedings of the 1977 ACM SIGMOD International Conference on Management of Data, Toronto, Canada, August 3-5, 1977, ACM, 1977, p. 9. URL: https://doi.org/10.1145/509404.509407. doi:10.1145/509404.509407.

[3] M. P. Atkinson, K. Chisholm, W. P. Cockshott, PS-Algol: an Algol with a persistent heap, ACM SIGPLAN Notices 17 (1982) 24–31. URL: https://doi.org/10.1145/988376.988378. doi:10.1145/988376.988378.

[4] A. Albano, F. Giannotti, R. Orsini, D. Pedreschi, The type system of Galileo, in: M. P. Atkinson, P. Buneman, R. Morrison (Eds.), Data Types and Persistence. Edited Papers from the Proceedings of the First Workshop on Persistent Objects, Appin, Scotland, UK, August 1985, Topics in Information Systems, Springer, 1985, pp. 101–119.

[5] M. J. C. Gordon, R. Milner, F. L. Morris, M. C. Newey, C. P. Wadsworth, A metalanguage for interactive proof in LCF, in: A. V. Aho, S. N. Zilles, T. G. Szymanski (Eds.), Conference Record of the Fifth Annual ACM Symposium on Principles of Programming Languages, Tucson, Arizona, USA, January 1978, ACM Press, 1978, pp. 119–130. URL: https://doi.org/10.1145/512760.512773. doi:10.1145/512760.512773.

[6] A. Albano, R. Orsini, Dialogo: An interactive environment for conceptual design in Galileo, in: S. Ceri (Ed.), Methodology and Tools for Data Base Design, North-Holland, 1983, pp. 229–254.

[7] A. Albano, G. Ghelli, M. E. Occhiuto, R. Orsini, Object-oriented Galileo, in: K. R. Dittrich, U. Dayal, A. P. Buchmann (Eds.), On Object-Oriented Database Systems, Topics in Information Systems, Springer, 1991, pp. 87–104.

[8] A. Albano, L. Alfò, S. Coluccini, R. Orsini, An overview of Sidereus: A graphical database schema editor for Galileo, in: J. W. Schmidt, S. Ceri, M. Missikoff (Eds.), Advances in Database Technology - EDBT'88, Proceedings of the International Conference on Extending Database Technology, Venice, Italy, March 14-18, 1988, volume 303 of *Lecture Notes in Computer Science*, Springer, 1988, pp. 567–571. URL: https://doi.org/10.1007/3-540-19074-0_77. doi:10.1007/3-540-19074-0_77.

[9] A. Albano, R. Bergamini, G. Ghelli, R. Orsini, An object data model with roles, in: R. Agrawal, S. Baker, D. A. Bell (Eds.), 19th International Conference on Very Large Data Bases, August 24-27, 1993, Dublin, Ireland, Proceedings, Morgan Kaufmann, 1993, pp. 39–51. URL: http://www.vldb.org/conf/1993/P039.PDF.

[10] A. Albano, G. Antognoni, G. Ghelli, View operations on objects with roles for a statically typed database language, IEEE Trans. Knowl. Data Eng. 12 (2000) 548–567. URL: https://doi.org/10.1109/69.868907. doi:10.1109/69.868907.

[11] G. Ghelli, Foundations for extensible objects with roles, Inf. Comput. 175 (2002) 50–75. URL: https://doi.org/10.1006/inco.2001.2943. doi:10.1006/inco.2001.2943.

[12] A. Albano, G. Ghelli, R. Orsini, A relationship mechanism for a strongly typed object-oriented database programming language, in: G. M. Lohman, A. Sernadas, R. Camps

(Eds.), 17th International Conference on Very Large Data Bases, September 3-6, 1991, Barcelona, Catalonia, Spain, Proceedings, Morgan Kaufmann, 1991, pp. 565–575. URL: http://www.vldb.org/conf/1991/P565.PDF, Best Paper Award of the European Program Committee.

[13] P. Curien, G. Ghelli, Coherence of subsumption, minimum typing and type-checking in $F_{<=}$, Math. Struct. Comput. Sci. 2 (1992) 55–91. URL: https://doi.org/10.1017/S0960129500001134. doi:10.1017/S0960129500001134.

[14] G. Ghelli, Termination of system F-bounded: A complete proof, Inf. Comput. 139 (1997) 39–56. URL: https://doi.org/10.1006/inco.1997.2662. doi:10.1006/inco.1997.2662.

[15] A. Albano, G. Antognoni, G. Baratti, G. Ghelli, R. Orsini, Il Galileo95, in: A. Albano, S. Salerno, F. Arcelli, M. Gaeta, S. Rizzo, G. Vantini (Eds.), Atti del Terzo Convegno Nazionale su Sistemi Evoluti per Basi di Dati, SEBD 1995, Ravello (Costiera Amalfitana), Italy, 28-30 Giugno 1995, 1995, pp. 247–272.

[16] A. Albano, G. Ghelli, R. Orsini, Fibonacci: A programming language for object databases, VLDB J. 4 (1995) 403–444. URL: http://www.vldb.org/journal/VLDBJ4/P403.pdf.

[17] G. Ghelli, F. Nanni, G. Puglielli, A. Albano, Tipi e moduli nel linguaggio Fibonacci, in: P. Atzeni, L. Cabibbo, M. Panti (Eds.), Atti del Sesto Convegno Nazionale Sistemi Evoluti per Basi di Dati, SEBD 1998, Ancona, Italy, 23-25 Giugno 1998, 1998, pp. 377–397.

[18] A. Albano, R. Bergamini, G. Ghelli, R. Orsini, An introduction to the database programming language Fibonacci, in: D. Saccà (Ed.), Convegno SEBD'93, Proceedings of the Conference on Advanced Database Systems (Atti del Convegno Nazionale su Sistemi Evoluti per Basi di Dati, SEBD) 1993, Hotel Capo Suvero, Gizzeria, Italy, 14-16 June, 1993, Mediterranean Press (via S. Pellico, 13 - Tel. 0984-465645 - 87030 Rende (CS)), 1993, pp. 247–266.

[19] A. Albano, C. Brasini, M. Diotallevi, G. Ghelli, R. Orsini, A guided tour of the Fibonacci system, in: S. Bergamaschi, C. Sartori, P. Tiberio (Eds.), Atti del Secondo Convegno Nazionale su Sistemi Evoluti per Basi di Dati, SEBD 1994, Rimini, Italy, 1994, Editrice Esculapio Progetto Leonardo, via U. Terracini, 30, 40131 Bologna, 1994, pp. 371–394.