Theoretical Computer Science ••• (••••) •••–•••

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

CoFI with Don Sannella

Peter D. Mosses

Department of Computer Science, Swansea University, Swansea, UK

ARTICLE INFO

Article history:
Received 28 July 2017
Received in revised form 15 January 2018
Accepted 26 February 2018
Available online xxxx

Keywords:
Algebraic specification
Software development
Formal methods

ABSTRACT

CoFI is the acronym of the *Common Framework Initiative for Algebraic Specification and Development*, which started in 1995. CoFI designed CASL, the *Common Algebraic Specification Language*.

This article first summarises the origins of CoFI and the motivation for CASL. It then recalls some of the crucial contributions to CoFI made by Don Sannella, and concludes with an indication of the impact of CoFI.

1. Origins and motivation

CoFI, the *Common Framework Initiative* for algebraic specification and development of software, started in 1995 as an open international collaboration. (Much of the rest of this section consists of excerpts from the information provided on the CoFI website http://www.cofi.info.)

The dozens of algebraic specification languages that have been developed all support the basic idea of using axioms to specify algebras, but differ in design choices concerning syntax (concrete and abstract) and semantics. Why not agree on a common framework? This was the provocative question asked at a WADT/COMPASS meeting in Santa Margherita, Italy, 1994 [1]. At least the main concepts to be incorporated were thought to be clear – although it was realised that it might not be so easy to agree on a common language to express these concepts.

The specification language subsequently developed by CoFI is called CASL: the Common Algebraic Specification Language. CASL is based on a critical selection of known constructs; it is expressive, yet simple, and pragmatic. Its main intended application is for specifying requirements and design for conventional software packages.

CASL consists of several major parts, which are quite independent and may be understood (and used) separately:

- basic specifications: declarations and definitions (of sorts, operations, predicates, datatypes), axioms (full first-order logic, sort generation constraints);
- structured specifications: translations, reductions, unions, extensions, freeness, named specifications, generic specifications, views;
- architectural specifications: implementation units, composition;
- specification libraries: local, distributed.

An early but key design decision was that CoFI should provide a coherent *family* of languages, all restrictions or extensions of a single main algebraic specification language. For instance, sublanguages have been obtained by disallowing declarations of

E-mail address: p.d.mosses@swansea.ac.uk.

https://doi.org/10.1016/j.tcs.2018.02.034
0304-3975/

Table 1
CASL development timeline.

1995	CASL design started *Don: initial CoFI Semantics Task Group coordinator*
1997	CASL initial design approved by IFIP WG1.3
1998	CASL version 1.0 released *Don: CoFI overall coordinator, applied for WG funding* ESPRIT CoFI WG started
2001	CASL version 1.0.1 approved by IFIP WG1.3 ESPRIT CoFI WG terminated
2004	CASL version 1.0.2 released CASL User Manual and Reference Manual published *Don: co-author and co-editor of* CASL *Semantics*

partial operations, predicates, and/or subsorts, or by restricting CASL's first-order logic axioms to universally-quantified positive conditional logic. Various extensions of CASL have been defined by subgroups of the CoFI participants: higher-order and coalgebraic extensions (HASCASL, COCASL), reactive extensions (CASL-LTL, SB-CASL, CSP-CASL), and extensions at the structured level (HETCASL and a refinement language).

CASL subsumes many previous languages for the formal specification of functional requirements and modular software design. Tools for CASL are interoperable, i.e., capable of being used in combination rather than in isolation. CASL interfaces to existing tools extend this interoperability. The Heterogeneous Tool Set (HETS, [2]) is the central parsing, analysis and prover integration tool for CASL and its extensions. HETS is free software, available from http://hets.eu.

Even though the intention was to base the design of CASL on a critical selection of concepts and constructs from existing specification languages, it was not easy to reach a consensus on a coherent language design. A great deal of careful consideration was given to the effect that the constructs available in the language would have on such aspects as the methodology and tools.

The CASL design effort started in September 1995, together with the creation of CoFI, as a joint effort of the COMPASS Working Group and IFIP WG1.3 on *Foundations of System Specification*. More than 40 experts in algebraic specification (primarily but not exclusively from European research groups) contributed to the CASL design – some over many years, others only occasionally. An initial design was proposed in May 1997 (with a language summary, abstract syntax, formal semantics, but no agreed concrete syntax) and tentatively approved by IFIP WG1.3. The report of the IFIP referees on the initial CASL design proposal suggested reconsideration of several points in the language design, and requested some improvements to the documents describing the design. Apart from a few details, the design was finalised in April 1998, with a complete draft language summary available, including concrete syntax. CASL version 1.0 was released in October 1998, and CASL version 1.0.1 was officially approved by IFIP WG1.3 in April 2001. The present version 1.0.2 was completed in 2004; it is documented in the CASL Reference Manual [3] and illustrated in the CASL User Manual [4].

2. Don's contributions

Don Sannella's expertise in algebraic specification languages and their semantics is well known: already in his PhD thesis [5] he gave a formal definition of the language CLEAR, and he was subsequently involved in the development of the seminal ASL [6] and EXTENDED ML [7] languages. He has made major contributions to CoFI, and in particular to the design of CASL and the definition of its semantics. He was also a primary CoFI coordinator throughout the entire CASL design period; see Table 1.

CoFI semantics task group Don was the initial coordinator of the CoFI Semantics task group (a rôle he subsequently shared with Andrzej Tarlecki). Here is an excerpt from a message from Don on the task group mailing list in September 1996,[1] proposing a crucial principle for factorising the semantics into independent parts:

> *A very good way to assess the tentative design for flaws would be to attempt to write down its semantics. Although this would be a big step towards completion of our main task, writing a complete semantics is too big a task to complete within the next six months or so, especially for a language that has not yet been finalized. So I think the best that we can do is to *sketch* the semantics, considering the features that seem potentially problematic with more care than the others. It should be possible to decompose this task, at least splitting work on the features of basic specifications from work on structured specifications and so-called architectural specifications. As we all know, it's necessary to agree on the interface between modules before getting down to work.*

The task group produced a semantics for the initial CASL design proposal in 1997, and completed the formal semantics for CASL 1.0 in 1998, in parallel with the later stages of the language design. The desire for a relatively straightforward

[1] http://www.cofi.info/old/MailingLists/cofi-semantics/96/msg00001.html.