Ábrahám, Erika; **Grabe, Immo**; **Grüner, Andreas**; **Steffen, Martin**

Behavioral interface description of an object-oriented language with futures and promises.

(English) Zbl 1187.68130
J. Log. Algebr. Program. 78, No. 7, 491-518 (2009).

Summary: This paper formalizes the observable interface behavior of a concurrent, object-oriented language with futures and promises. The calculus captures the core of Creol, a language, featuring in particular asynchronous method calls and, since recently, first-class futures.

The focus of the paper are open systems and we formally characterize their behavior in terms of interactions at the interface between the program and its environment. The behavior is given by transitions between typing judgments, where the absent environment is represented abstractly by an assumption context. A particular challenge is the safe treatment of promises: The erroneous situation that a promise is fulfilled twice, i.e., bound to code twice, is prevented by a resource aware type system, enforcing linear use of the write-permission to a promise. We show subject reduction and the soundness of the abstract interface description.

MSC:

68N19 Other programming paradigms (object-oriented, sequential, concurrent, automatic, etc.)

Keywords:

concurrent object-oriented languages; Creol; formal semantics; concurrency; futures and promises; open systems; observable behavior

Software:

Scala; Java Jr; Featherweight Java; Creol; ABCL; Multilisp

Full Text: DOI

References:

[1] Abadi, M.; Cardelli, L., A theory of objects, Monographs in computer science, (1996), Springer-Verlag

[2] E. Ábrahám, I. Grabe, A. Grüner, M. Steffen, Behavioral Interface Description of an Object-Oriented Language with Futures and Promises, Technical Report 364, University of Oslo, Dept. of Computer Science, October 2007.

[3] E. Ábrahám, A. Grüner, and M. Steffen, Abstract interface behavior of object-oriented languages with monitors, in: R. Gorrieri, H. Wehrheim [49], pp. 218-232.

[4] Ábrahám, E.; Grüner, A.; Steffen, M., Abstract interface behavior of object-oriented languages with monitors, Theory comput. syst., 43, 3-4, 322-361, (2008) · Zbl 1162.68475

[5] Ábrahám, E.; Grüner, A.; Steffen, M., Heap-abstraction for open, object-oriented systems with thread classes, J. softw. syst. modell. (sosym), 7, 2, 177-208, (2008), May

[6] ACM, Object oriented programming: systems, languages, and applications (OOPSLA)'86, 1986, in: SIGPLAN Notices, vol. 21(11).

[7] ACM, in: 23rd Annual Symposium on Principles of Programming Languages (POPL) (St. Petersburg Beach, Florida), January 1996.

[8] ACM, ACM Conference on Programming Language Design and Implementation, May 1999.

[9] ACM, Object oriented programming: systems, languages, and applications (OOPSLA)'99, 1999, in: SIGPLAN Notices.

[10] ACM, Object oriented programming: systems, languages, and applications (OOPSLA)'02 (Seattle, USA), November 2002, in: SIGPLAN Notices.

[11] ACM, eighteenth object oriented programming: systems, languages, and applications (OOPSLA)'03, 2003, in: SIGPLAN Notices.

[12] ACM, in: 31th Annual Symposium on Principles of Programming Languages (POPL), January 2004.

[13] ACM, Nineteenth object oriented programming: systems, languages, and applications (OOPSLA)'04, 2004, in: SIGPLAN Notices.

[14] ACM, Twentieth object oriented programming: systems, languages, and applications (OOPSLA)'05, 2005, in: SIGPLAN

Notices.

[15] Agha, G.A., Actors: A model of concurrent computation in distributed systems, (1986), MIT Press Cambridge, MA

[16] G.A. Agha, I.A. Mason, S.F. Smith, C.L. Talcott, Towards a theory of actor computation (extended abstract), in: R. Cleaveland [34], pp. 565-579.

[17] Alice project home page, 2006. <www.ps-uni-sb.de/alice>.

[18] B. Alpern, C.R. Attanasio, J.J. Barton, A. Cocchi, S.F. Hummel, D. Lieber, T. Ngo, M. Mergen, J.C. Sheperd, S. Smith, Implementing Jalapeno in Java, in: OOPSLA'99 [9], in: SIGPLAN Notices, pp. 313-324.

[19] America, P., Issues in the design of a parallel object-oriented language, Formal aspects comput., 1, 4, 366-411, (1989) · Zbl 0694.68012

[20] V. Arslan, P. Eugster, P. Nienaltowski, S. Vaucouleur, Scoop – concurrency made easy, in: J. Kohlas et al. [63], pp. 82-102.

[21] Baker, H.; Hewitt, C., The incremental garbage collection of processes, ACM sigplan notices, 12, 55-59, (1977)

[22] G. Barthe, P. Dybjer, L. Pinto, J. Saraiva (Eds.), Applied Semantics, International Summer School, APPSEM 2000, Caminha, Portugal, September 9-15, 2000, Advanced Lectures, Lecture Notes in Computer Science, vol. 2395, Springer-Verlag, 2002. · Zbl 0993.00046

[23] Benton, N.; Cardelli, L.; Fournet, C., Modern concurrency abstraction for C#, ACM trans. program. lang. syst., 26, 5, 769-804, (2004)

[24] M.M. Bonsangue, E.B. Johnsen (Eds.), Proceedings of the 8th IFIP International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS'07), Cyprus, Greece, Lecture Notes in Computer Science, vol. 4468, Springer-Verlag, June 2007.

[25] M.G. Burke, J.-D. Choi, S.F. Fink, D. Grove, M. Hind, V. Sarkar, M. Serranon, V.C. Sreedhar, H. Srinivasan, J. Whaley. The Jalapeno dynamic optimizing compiler, in: Proceedings of the ACM Java Grande Conference, San Francisco, 1999, pp. 129-141.

[26] Caromel, D., Service, asynchrony and wait-by-necessity, J. object-oriented program., 2, 4, 12-22, (1990)

[27] Caromel, D., Towards a method of object-oriented concurrent programming, Commun. ACM, 36, 9, 90-102, (1993)

[28] Caromel, D.; Henrio, L., A theory of distributed objects, Asynchrony – mobility – groups – components, (2005), Springer-Verlag · Zbl 1084.68012

[29] D. Caromel, L. Henrio, B.P. Serpette, Asynchronous sequential processes, Research Report RR-4753 (version 2), INRIA Sophia-Antipolis, May 2003. · Zbl 1170.68026

[30] D. Caromel, L. Henrio, B.P. Serpette, Asynchronous and deterministic objects, in: POPL'04 [12], pp. 123-134. · Zbl 1325.68052

[31] D. Caromel, W. Klauser, J. Vayssière, Towards seamless computing and metacomputing in Java, Conc., Pract. Exper., 10 (11-13) (1998) 1043-1061. ProActive available from: <www.infria.fr/oasis/proactive>.

[32] R. Chandra. The COOL Parallel Programming Language: Design, Implementation, and Performance, PhD thesis, Stanford University, April 1995.

[33] Chandra, R.; Gupta, A.; Hennessy, J.L., COOL: A language for parallel programming, (), 126-148

[34] R. Cleaveland (Ed.), Third International Conference on Concurrency Theory (CONCUR'92, Stony Brook, NY), Lecture Notes in Computer Science, vol. 630, Springer-Verlag, 1992. · Zbl 0825.00073

[35] M.J. Compton. SCOOP: An investigation of concurrency in Eiffel. Master's thesis, Department of Computer Science, The Australian National University, 2000.

[36] Proceedings of Coordination Languages and Models (Proceedings of COORDINATION'07), Lecture Notes in Computer Science, vol. 4467, Springer-Verlag, 2007.

[37] The Creol language, 2007. <http://heim.ifi.uio.no/creol>.

[38] F.S. de Boer, M.M. Bonsangue, A. Grüner, M. Steffen. Test driver generation from object-oriented interaction traces (extended abstract), in: Proceedings of the 19th Nordic Workshop on Programming Theory (NWPT'07). University of Oslo, Dept. of Computer Science, Technical Report, vol. 366, 2007, pp. 52-54.

[39] F.S. de Boer, D. Clarke, E.B. Johnsen, A complete guide to the future, in: de Nicola [40], pp. 316-330.

[40] R. de Nicola (Ed.), ESOP'07, Lecture Notes in Computer Science, vol. 4421, Springer-Verlag, 2007.

[41] The E language, 2007. <www.erights.org>.

[42] C. Flanagan, M. Felleisen, The semantics of future, Technical Report TR94-238, Department of Computer Science, Rice University, 1994. · Zbl 0926.68075

[43] C. Flanagan, M. Felleisen, Well-founded touch optimization of parallel scheme, Technical Report TR94-239, Department of Computer Science, Rice University, 1994.

[44] Flanagan, C.; Felleisen, M., The semantics of future and an application, J. funct. program., 9, 1, 1-31, (1999) · Zbl 0926.68075

[45] J. Foster, M. Fändrich, A. Aiken, A theory of type qualifiers, in: ACM Conference on Programming Language Design and Implementation [8], pp. 192-203.

[46] C. Fournet, G. Gonthier, The reflexive chemical abstract machine and the join-calculus, in: POPL'96 [7], pp. 372-385.

[47] C. Fournet, G. Gonthier, The join calculus: a language for distributed mobile programming, in: Barthe et al. [22], pp. 268-332. · Zbl 1065.68071

[48] Girard, J.-Y., Linear logic, Theoret. comput. sci., 50, 1, 1-102, (1987) · Zbl 0625.03037

[49] R. Gorrieri, H. Wehrheim (Eds.), Proceedings of the 8th IFIP International Conference on Formal Methods for Open Object-Based Distributed Systems (FMOODS'06), Bologna, Italy, Lecture Notes in Computer Science, vol. 4037, Springer-Verlag, 2006.

[50] Gul, I.A.M.; Agha, A.; Smith, S.F.; Talcott, C.L., A foundation for actor computation, J. funct. program., 7, 1, 1-72, (1997) · Zbl 0870.68091

[51] P. Haller, M. Odersky, Actors that unify threads and events, in: COORDINATION'07 [36], A longer version is available as EPFF Technical Report, pp. 171-190.

[52] Halstead, R.H., Multilisp: a language for concurrent symbolic computation, ACM trans. program. languages syst., 7, 4, 501-538, (1985) · Zbl 0581.68037

[53] IEEE, Seventeenth Annual Symposium on Logic in Computer Science (LICS) (Copenhagen, Denmark), Computer Society Press, July 2002.

[54] A. Igarashi, B.C. Pierce, P. Wadler. Featherweight Java: A minimal core calculus for Java and GJ, in: OOPSLA'99 [9], SIGPLAN Notices, pp. 132-146.

[55] Io, A small programming language, 2007. <www.iolanguage.com>.

[56] G.S. Itzstein, D. Kearney, Join Java: an alternative concurrency semantics for Java, Technical Report ACRC-01-001, University of South Australia, 2001.

[57] G.S. Itzstein, D. Kearney, Applications of join Java, in: Proceedings of the Australian Conferences in Research and Practice in Information Technology, vol. 6, 2002, pp. 37-46.

[58] A. Jeffrey, J. Rathke, A fully abstract may testing semantics for concurrent objects, in: LICS'02 [53], pp. 101-112. · Zbl 1078.68107

[59] A. Jeffrey, J. Rathke, Java Jr.: a fully abstract trace semantics for a core Java language, in: Sagiv [78], pp. 423-438. · Zbl 1108.68349

[60] Johnsen, E.B.; Owe, O.; Yu, I.C., Creol: a type-safe object-oriented model for distributed concurrent systems, Theoret. comput. sci., 365, 1-2, 23-66, (2006) · Zbl 1118.68031

[61] ()

[62] JSR 166, Concurrency utilities, 2007. <www.jcp.org/en/jsr/detail?id=166>.

[63] J. Kohlas, B. Meyer, A. Schiper (Eds.), Dependable Systems: Software, Computing, Networks, Research Results of the DICS Program, Lecture Notes in Computer Science, vol. 4028, Springer, 2006.

[64] L. Kornstaedt, Alice in the land of Oz – an interoperability-based implementation of a functional language on top of a relational language, in: Proceedings of the First Workshop on Mulit-Language Infrastructure and Interoperability (BABEL'01), Electronic Notes in Theoretical Computer Science, September 2001.

[65] H. Liebermann, A preview of ACT-1. AI-Memo AIM-625, Artificial Intelligence Laboratory, MIT, 1981.

[66] H. Liebermann, Concurrent object-oriented programming in ACT1, in: A. Yonezawa, M. Tokoro [90], pp. 9-36.

[67] Liskov, B.; Shrira, L., Promises: linguistic support for efficient asynchronous procedure calls in distributed systems, SIGPLAN notices, 23, 7, 260-267, (1988)

[68] D.A. Manolescu, Workflow enactment with continuation and future objects, in: OOPSLA'02 [10], SIGPLAN Notices, pp. 40-51.

[69] Meyer, B., Systematic concurrent object-oriented programming, Commun. ACM, 36, 9, 56-80, (1993)

[70] Moreau, L., The semantics of scheme with future, (), 146-156 · Zbl 1345.68066

[71] Niehren, J.; Sabel, D.; Schmidt-Schauß, M.; Schwinghammer, J., Observational semantics for a concurrent lambda calculus with reference cells and futures, Electron. notes theoret. comput. sci., 173, 313-337, (2007) · Zbl 1316.68034

[72] Niehren, J.; Schwinghammer, J.; Smolka, G., A concurrent lambda-calculus with futures, Theoret. comput. sci., 64, 3, 338-356, (2006) · Zbl 1110.68023

[73] M. Odersky, L. Spoon, B. Venners, Programming in Scala, A comprehensive step-by-step guide, Artima Developer, 2008.

[74] A. Poetzsch-Heffter, J. Schäfer, A representation-independent behavioral semantics for object-oriented components, in: M.M. Bonsangue, E.B. Johnsen [24], pp. 157-173. · Zbl 1202.68100

[75] P. Pratikakis, J. Spacco, M.W. Hicks, Transparent proxies for Java futures, in: OOPSLA'03 [13], SIGPLAN Notices, pp. 206-233.

[76] R.R. Raje, J.I. William, M. Boyles, An asynchronous method incocation (ARMI) mechanism for Java, in: Proceedings of the ACM Workshop on Java for Science and Engineering Computation, 1997, pp. 1207-1211.

[77] A. Rossberg, D. Le Botlan, G. Tack, T. Brunklaus, G. Smolka, Alice through the looking glass, in: Trends in Functional Programming, vol. 5, Intellect Books, Bristol, 2006, pp. 79-96[Chapter 6].

[78] M. Sagiv (Ed.), Proceedings of ESOP 2005, Lecture Notes in Computer Science, vol. 3444, Springer-Verlag, 2005.

[79] J. Schwinghammer, A concurrent λ-calculus with promises and futures, Diplomarbeit, Universität des Saarlandes, February 2002.

[80] M. Steffen, Object-connectivity and observability for class-based, object-oriented languages, Habilitation thesis, Technische Faktultät der Christian-Albrechts-Universität zu Kiel, July 2006.

[81] Stroustrup, B., The C++ programming language, (1986), Addison-Wesley · Zbl 0609.68011

[82] T. Sysala, J. Janecek, Optimizing remote method invocation in Java, in: DEXA, June 2001, pp. 29-35.

[83] É. Tanter, J. Noyé, D. Caromel, P. Cointe, Partial behavioral reflection: spatial and temporal reflection of reification, in: OOPSLA'03 [11], SIGPLAN Notices, pp. 27-46.

[84] K. Taura, S. Matsuoka, A. Yoneazawa. ABCL/f: a future-based polymorphic typed concurrent object-oriented language – its design and implementation, in DIMACS Workshop on Specification of Parallel Algorithms, American Mathematical Society, 1994, pp. 275-292.

[85] P.L. Wadler, Linear types can change the world, in: C.B. Jones and M. Broy [61], pp. 347-359.

[86] A. Welc, S. Jagannathan, A. Hosking, Safe futures in Java, in: OOPSLA'03 [14], SIGPLAN Notices, pp. 439-453.

[87] Y. Yokote, M. Tokoro, Concurrent programming in concurrent SmallTalk, in: A. Yonezawa, M. Tokoro [90], pp. 129-158.

[88] Yonezawa, A., ABCL: an object-oriented concurrent system, (1990), MIT Press

[89] A. Yonezawa, J.-P. Briot, E. Shibayama. Object-oriented concurrent programming in ABCL/1, in: OOPSLA'86 [6], SIGPLAN Notices 21(11), pp. 258-268.

[90] ()

[91] Y. Yonezawa, E. Shibayama, T. Takada, Y. Honda, Modelling and programming in an object-oriented concurrent language ABCL 1, in: A. Yonezawa, M. Tokoro [90], pp. 55-89.