[8] M. Shapiro, "Structure and encapsulation in distributed systems: the Proxy Principle," in *Proc. 6th Intl. Conf. on Distributed Computing Systems*, (Cambridge, Mass. (USA)), pp. 198–204, IEEE, May 1986.

[9] M. Makpangou, Y. Gourhant, J.-P. Le Narzul, and M. Shapiro, "Structuring distributed applications as fragmented objects," Rapport de Recherche 1404, Institut National de la Recherche en Informatique et Automatique, Rocquencourt (France), Jan. 1991.

[10] Y. Gourhant and M. Shapiro, "FOG/C++: a fragmented-object generator," in *C++ Conference*, (San Francisco, CA (USA)), pp. 63–74, Usenix, Apr. 1990.

[11] M. Shapiro, P. Gautron, and L. Mosseri, "Persistence and migration for C++ objects," in *ECOOP'89*, (Nottingham (GB)), July 1989.

[12] P. Gautron and M. Shapiro, "Two extensions to C++: A dynamic link editor and inner data," in *Proceedings and additional papers, C++ Workshop*, (Berkeley, CA (USA)), USENIX, Nov. 1987.

[13] J. L. Narzul and M. Shapiro, "Un service de nommage pour un système à objets répartis," in *Actes Convention Unix 89*, (Paris), pp. 73–82, AFUU, Mar. 1989.

[14] M. Makpangou and M. Shapiro, "The SOS object-oriented communication service," in *Proc. 9th Int. Conf. on Computer Communication*, (Tel Aviv (Israel)), October–November 1988.

[15] M. M. Makpangou, *Protocoles de communication et programmation par objets : l'exemple de SOS*. PhD thesis, Université Paris VI, Paris (France), Feb. 1989.

[16] M. Shapiro, "Soul: An object-oriented OS framework for object support," in *Workshop on Operating Systems for the Nineties and Beyond*, (Dagstuhl Castle, Germany), Springer-Verlag, July 1991.

marshalling/unmarshalling between layers. Currently, the toolbox contains mainly classes of primitive FOs such as RPC and multicast communication channels.

Every system service in SOS is a fragmented object, and SOS has plenty of support for user-level fragmented objects. There is also language support if you use FOG [10], an extension of C++ for fragmented objects.

SOS also supports local ("elementary") objects of course. You can declare elementary objects to the system, store them persistently [11], or migrate them between addressing domains or between machines. Migration and storage preserve the data, the structure, and *the type* of the object: even though SOS as an OS has no concept of type, it maintains "prerequisites", allowing the C++ run-time to implement dynamic type checking and linking. The structure and management of elementary objects are decribed in [2, 4]. The mechanics of the dynamic type checker and linker are described in [12].

SOS has been implemented in C++, on top of Unix. It supports object migration, persistent object, dynamic linking, and arbitrarily complex user-defined objects (written in C++ or FOG). Existing system services (all structured as fragmented objects) are: a distributed object manager [4], a user-centered name service [13, 9], a storage service [11], a communication service (allowing groups to choose from a library of protocol types: datagrams, RPC, multicast, atomic multicast, etc.) [14, 15]. Applications built using SOS include a multimedia document manager and a UIMS.

SOS is thouroughly documented with a reference manual and an introductory programmer's manual. Most of the code is in the public domain, except for a few components derived from AT&T code, for which an ATT licence (Unix and C++) is necessary.

COOL is also an object-support OS. It is less ambitious and more efficient than SOS. Here objects are supported by a manager implemented as a kernel server above the Chorus micro-kernel [4, 5]. COOL was developed specifically for the SEPT "Cidre" application of intelligent circulating documents.

Our current work, Soul [16] is to generalize the previous, to have an object-oriented object-support system, as powerful as SOS and as efficient as COOL.

Many of the references are available by anonymous FTP on nuri.inria.fr [128.93.1.26], directory INRIA/publication/SOR.

# References


[1] M. Shapiro, "Object-support operating systems," in *Workshop on Operating Systems and Object Orientation at ECOOP/OOPSLA 1990*, (Ottawa (Canada)), Oct. 1990.

[2] M. Shapiro, Y. Gourhant, S. Habert, L. Mosseri, M. Ruffin, and C. Valot, "Sos: An object-oriented operating system — assessment and perspectives," *Computing Systems*, vol. 2, no. 7, 1989.

[3] M. Shapiro, "Prototyping a distributed object-oriented OS on Unix," in *Workshop on Experiences with Building Distributed and Multiprocessor Systems* (E. Spafford, ed.), (Ft. Lauderdale FL (USA)), USENIX, Oct. 1989. Also available as Rapport de Recherche INRIA no. 1082.

[4] S. Habert, *Gestion d'Objets et Migration dans les Systèmes Répartis*. PhD thesis, Université Paris VI, Paris (France), Dec. 1989.

[5] S. Habert, L. Mosseri, and V. Abrossimov, "COOL: Kernel support for object-oriented environments," in *Joint ECOOP/OOPSLA Conference*, (Ottawa (Canada)), Oct. 1990.

[6] V. Abrossimov, M. Rozier, and M. Shapiro, "Generic virtual memory management for operating system kernels," in *Proc. 12th ACM Symp. on Operating Systems Principles*, (Litchfield Park AZ (USA)), ACM SIGOPS, Dec. 1989.

[7] M. Shapiro, O. Gruber, and D. Plainfossé, "A garbage detection protocol for a realistic distributed object-support system," Rapport de Recherche 1320, Institut National de la Recherche en Informatique et Automatique, Rocquencourt (France), Nov. 1990.

# Systèmes d'Objets Répartis
# (Distributed Object-Support Systems)

INRIA (Institut National de Recherche en Informatique et Automatique)

## Personnel


| | | |
|---|---|---|
| Principal Investigator | Marc Shapiro | shapiro@sor.inria.fr |
| Senior Researcher | Mesaac Makpangou | mak@sor.inria.fr |
| Graduate Students | Pierre Collet | collet@sor.inria.fr |
| | Yvon Gourhant | gourhant@sor.inria.fr |
| | Jean-Pierre Le Narzul | lenarzul@sor.inria.fr |
| | Julien Maisonneuve | julien@sor.inria.fr |
| | David Plainfossé | dp@sor.inria.fr |
| | Michel Ruffin | ruffin@sor.inria.fr |
| | Hervé Soulard | soulard@sor.inria.fr |
| | Céline Valot | valot@sor.inria.fr |


## Contact

Marc Shapiro
INRIA
BP 105
78153 Rocquencourt Cédex
France

Telephone +33 (1) 39-63-55-11
Facsimile +33 (1) 39-63-53-30

## Project Description

The general theme of our work is operating system support for object-oriented systems, especially in distributed environments [1]. We have developed two systems: SOS [2, 3, 4] and COOL [5, 4]. In addition we have done work on structuring operating systems, especially virtual memory support [6], and OS support for distributed garbage collection [7].

A key concept for structuring distributed systems is "Fragmented Objects" [8, 9]. Fragmented Objects (FOs) extend the object concept to a distributed environment. The abstract view of a FO is a single, shared object, of which the distribution is hidden to clients. In the concrete view the FO designer controls (if wished) the distribution of data and function and of the communication between fragments.

FO programming is supported by the FOG language, an extension of C++, and by a toolbox of predefined FOs. The FOG compiler ensures distributed type-safety of both the external and internal interfaces, verifies the encapsulation of FO instances, and automatically generates whatever coercions are necessary for