Historique des travaux

Marc Shapiro
Senior Researcher, Microsoft Research, Cambridge UK
Directeur de Recherche INRIA

8 octobre 2002

Ce document présente de façon concise et historique l'ensemble de mes travaux. La présentation, forcément linéaire, est bien sûr une simplification d'un cheminement plus complexe.

1 Travaux de thèse : structuration de programmes CSP

Dans le cadre d'un travail de thèse mené au LAAS dans l'équipe de Michel Diaz, mes travaux ont concerné la mise en œuvre d'un système réparti de contrôle de processus [8, 2, 3]. Dans ce cadre, j'ai élaboré une méthode de conception des systèmes répartis, par enrichissements successifs du contrôle, basé sur quelques propriétés particulières du langage CSP. Cette méthode, qui a fait l'objet de ma thèse [49] et d'une publication [50] n'a plus qu'un intérêt historique.

2 Travaux post-doctoraux

Grâce à une bourse post-doctorale de l'INRIA j'ai travaillé 18 mois au *Laboratory for Computer Science* du MIT dans l'équipe de Mme. Barbara Liskov. Le groupe commençait alors la conception du successeur de CLU, Argus, un langage et un environnement d'exécution réparti persistant. Cela a constitué mon premiers contacts avec la programmation par objets, que j'ai embrassé avec enthousiasme.

De retour en France, j'ai été embauché par le tout nouveau Centre Mondial Informatique et Ressources Humaines (CMIRH) comme chercheur et ingénieur système. J'y ai continué mes travaux sur la structuration des systèmes répartis, cette fois sous l'angle de objets répartis [51, 52]. Dans le même temps nous avons conçu un réseau local bon marché à l'intention des écoles [5, 53, 54] qui n'a malheureusement jamais été mis en œuvre.

Par la suite, j'ai rejoint le GIPSI SM-90, où j'étais responsable des logiciels réseau. Mon travail consistait à mettre en œuvre TCP/IP et Ethernet, alors des protocoles nouveaux, sur la station de travail française.

3 SOS et les concepts de mandataire et d'objet fragmenté

J'ai ensuite rejoint le projet Chorus de l'INRIA, en tant que responsable du système d'exploitation prévu pour une station de travail de bureautique, dans par le contrat Esprit SOMIW. Le système d'exploitation s'appelait SOS, pour «SOMIW Operating System». Après le passage de Chorus dans l'industrie, je suis resté responsable du groupe restant, qui est peu à peu devenu le projet de recherche INRIA «Systèmes d'Objets Répartis» (SOR).

Inventeur de la notion de mandataire [55], j'ai appelé cette structuration des systèmes répartis le *principe du mandataire (Proxy Principle)*. Par la suite, selon la terminologie de Banâtre [4], j'ai désigné sous le nom d'*objet fragmenté (Fragmented Object)* l'objet réparti fournisseur de service.

Le système SOS est spécialement conçu pour faciliter la mise en œuvre et l'exécution des objets fragmentés et est lui-même structuré en objets fragmentés [68, 68, 59, 56, 57, 67, 65, 64, 30]. Il a permis une riche recherche sur les protocoles de communication [32, 33, 34, 35], sur les langages de programmation pour la répartition [17, 63, 19, 29, 30], sur la migration et la persistance des objets [17, 70, 63]. Un certain nombre d'applications réparties basées sur les objets fragmentés ont été développés [18, 76], en particulier le «Service d'Accointances», le gestionnaire d'objets de SOS [65] et son service de nommage [23, 24, 29, 30].

L'expérience accumulée dans SOS a servi par la suite dans la conception du système COOL (Chorus Object-Oriented Layer) [20, 21, 25].

Un certain nombre de leçons importantes apprises de l'expérience SOS ont été consignées, certaines sur le moment [65, 21], certaines plus tard (*cf.* § **??**). L'une d'elles étaient la difficulté pour une petite équipe de recherche de développer un système d'exploitation complet. Il s'ensuit la priorité donnée au développement de composants autonomes donnant la fonctionnalité des objets fragmentés comme valeur ajoutée aux systèmes existants [31, 26, 9, 73, 10, 28, 58].

4 Automatisation de la gestion des objets

Les données partagées sont très difficiles à administrer manuellement. Lorsque ces données sont persistantes, toute erreur d'administration se perpétue dans le temps. C'est pourquoi le modèle de persistance «par atteignabilité» exige que toute donnée atteignable depuis une donnée persistante soit elle-même persistante. Les points ci-dessus justifient l'importance du GC.

Le GC pose des problèmes non triviaux dans un système réparti grande échelle [71], et les algorithmes de la littérature sont insatisfaisants [40]. Par ailleurs c'est un domaine assez négligé par la communauté système, et dans lequel l'INRIA-Rocquencourt a des bonnes compétences. C'est donc une excellente niche du point de vue recherche.

L'étude du GC implique aussi celui des mécanismes de référence et de liaison.

D'abord avec Olivier Gruber du projet Rodin, nous avons suivi une approche plutôt descendante, de l'expression des besoins à la spécification du protocole, puis à sa mise en œuvre. Nous avons publié le premier protocole de GC réaliste pour un système réparti asynchrone [66], et un système de références, les *Chaînes de Paires Souche-Scion (CPSS)*, adapté aux systèmes de grande échelle [61]. De nombreuses autres publications décrivent ces travaux [66, 37, 36, 60, 39, 38, 61, 71, 40]. Les CPSS ont connu plusieurs mise en œuvre sans changement notable de la spécification, car les performances et l'utilisabilité de la réalisation sont des points importants dans leur acceptation. Aujourd'hui nous disposons de réalisations performantes et interopérables en C++, Java, et OCAML.

5 Persistance par atteignabilité : GC et gestion de mémoire partagée

Paradoxalement, les CPSS ne résolvent pas le problème initial, celui de la persistance, qui pose le problème du ramasse-miettes en présence de réplication et de cohérence. Ce problème se pose, plus généralement, dans toute architecture répartie comportant des caches, par exemple une DSM. Le GC doit alors éviter les entrées-sorties, les envois de message, et la synchronisation, toutes choses à priori inhérentes aux principes du traçage et aux algorithmes de la littérature. Nous avons donc posé le principe d'une coopération de GC partiels, et établi que ceux-ci n'ont pas besoin du même degré de cohérence que les applications.

Avec l'appui d'une représentation des références similaires aux CPSS, on peut donc approximer le traçage global par une série de traces locales non coordonnées entre elles. Ce résultat fait l'objet de publications [12, 11, 62, 13, 14, 15, 16, 69].

6 Entrepôt Persistant Réparti PerDiS

Des contacts avec des utilisateurs, du monde du traitement des images, de la finance et du bâtiment m'ont convaincu que leurs besoins, d'environnements d'exécution réparties simples pour leurs applications à objets, ne peuvent être satisfaits par les environnements alors existants. Ni SOS, trop complexe et manquant de maturité, ni Corba, trop restrictif et peu performant, ni les bases de données à objets, trop complexes et lentes ne répondent à leur demande. Par contre une mémoire partagée, efficace grâce à sa gestion de cache, simple d'utilisation grâce à l'adressage plat, et automatique grâce au ramasse-miettes, correspondait aux besoins. C'est justement ce qu'offriat l'architecture Larchant. C'est pourquoi nous avons proposé le projet européen PerDiS, un entrepôt persistent réparti pour des applications d'ingénierie coopératives dans l'industrie du bâtiment.

7 Cohérence optimiste de réplicats

En 1985, en disponiblité de l'INRIA, j'ai rejoint le tout nouveau laboratoire *Microsoft Research* à Cambridge (Royaume-Uni) sous la direction du prof. Roger

Needham, où j'ai monté le *Cambridge Distributed Systems Group* (Camdis). Celui-ci s'est focalisé sur les nouveaux défis du partage de l'information dans les SIRGE face à la fragmentation de l'Internet en sous-réseaux très faiblement connectés, à la complexité de l'administration des réseaux, et au développement du travail nomade.

Ce groupe est aujourd'hui mondialement connu pour ses travaux sur les systèmes *d'égal à égal (Peer-To-Peer)* avec à la base le protocole de gestion de table de hachage réparti Pastry [42], le protocole de diffusion Scribe [7], le système d'archivage réparti Past [43], et divers travaux associés.

Pour ma part je me suis spécialisé dans l'étude de la cohérence optimiste des réplicats.

Un protocole de réplication est dit pessimiste lorsqu'il vérifie avant toute écriture que celle-ci ne va pas entrer en conflit avec une autre, évitant ainsi à deux réplicats de diverger durablement. Un protocole pessimiste suppose un couplage fort entre réplicats.

Au contraire, un protocole optimiste autorise un couplage faible car il permet de modifier les réplicats sans vérification préalable. Les réplicats peuvent donc diverger ; ce problème est résolu à posteriori par un protocole de *réconciliation*. Cette approche autorise un plus grand parallélisme et permet de progresser même sans accès réseau. De plus elle est bien adaptée au travail collectif, puisque plusieurs utilisateurs peuvent faire des modifications «à l'essai» et décider plus tard lesquelles persisteront.

Dans le cadre de ces travaux j'ai collaboré, avec Yasushi Saito de HP Labs, à une étude globale des problèmes et techniques liées à la réplication optimiste [48].

Une difficulté de taille de l'approche est de définir un algorithme de réconciliation adapté. La réconciliation consiste à sélectionner parmi les mises à jour proposées par les différents sites un sous-ensemble non conflictuel et à les combiner d'une façon ou d'une autre.

Il est reconnu depuis longtemps (vori l'exemple du système Bayou [75]) que la notion de conflit dépend de la sémantique des objets partagés. Cependant tous algorithmes de combinaison existants ignoraient la sémantique et ne prenaient en compte que les propriétés «syntaxiques» des mises à jour, par exemple l'ordre de ceux-ci. Nous avons conclu quant à nous que les ordres syntaxiques sont d'un intérêt limité et que l'algorithme de combinaison lui-même devait prendre en compte la sémantique. Étant donné la diversité des sémantiques possibles il apparaît difficile de concevoir un algorithme de réconciliation d'intérêt général. C'est pourtant ce que nous avons accompli avec notre système IceCube [72, 22, 41].

8 Divers

8.1 Mémoire virtuelle Chorus

En 1988–1989, j'ai pu collaborer avec Dima Abrossimov et Marc Rozier à la conception de la mémoire virtuelle du noyau de système d'exploitation Chorus

[44], développé par la société Chorus-systèmes. J'y ai contribué quelques idées intiales, puis j'ai participé à l'effort de mise au propre des concepts, donnant lieu à une publication SOSP [1].

8.2 Autres travaux

Une importante activité du projet SOR de l'INRIA a été la mise en œuvre de mécanismes système de base pour le partage. Dans le cadre de SOS et de COOL nous avons développé des noyaux et des bibliothèques d'exécution [27], des adaptations langage [17, 19], compilateur, déboggueur, ou linker, des pilotes de mémoire virtuelle [27], et des protocoles de communication [33]. Dans nos activités Kitlog [45, 46, 47] et Boss [73, 74] nous avons développé des mécanismes de stockage.

Références

[1] V. Abrossimov, M. Rozier, and M. Shapiro. Generic virtual memory management for operating system kernels. In *Proceedings of the 12th ACM Symposium on Operating Systems Principles*, pages 123–136, Litchfield Park AZ (USA), December 1989. ACM.

[2] J. M. Ayache, B. Carrichon, M. Devy, M. Diaz, B. Potin, and M. Shapiro. A distributed control system for industrial plants. In *Euromicro 80 Conference*, London, Royaume-Uni, September 1980.

[3] J.M. Ayache, B. Carrichon, J.P. Courtiat, M. Diaz, B. Potin, and M. Shapiro. Fault tolerance in Rebus, a distributed system for industrial real time control. In *Proc. Symposium on Fault-Tolerant Computing FTCS-11*, Portland, Maine, USA, June 1981.

[4] Jean-Pierre Banâtre, Michel Banâtre, and Florimond Ployette. The concept of multi-function : a general structuring tool for distributed operating. In *The 6th International Conference on Distributed Computer Systems*, pages 478–485, Cambridge, Massachusetts, May 1986. IEEE.

[5] Guy Bernard, Pierre Bouchet, Thierry Fleury, Marc Shapiro, and Yves de Talhouet. Microrézo : le projet réseau local du Centre Mondial. In *1983 World Conference on Systems*, Caracas, Venezuela, 1983.

[6] Luis-Felipe Cabrera, Vince Russo, and Marc Shapiro, editors. *1991 International Workshop on Object Orientation in Operating Systems*, Palo Alto CA (USA), October 1991. IEEE, IEEE Computer Society Press. IEEE Computer Society Press Order Number 2265.

[7] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, and Antony Rowstron. Scribe : A large-scale and decentralised application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communication (JSAC)*, 20(8), October 2002. `http://www.research.microsoft.com/~antr/PAST/jsac.pdf`.

[8] Michel Devy, Michel Diaz, Miguel Menasche, and Marc Shapiro. A fault-tolerant virtual-ring algorithm for bus allocation. Note interne L.A.A.S. 80.I.35, Laboratoire d'Automatique et d'Analyse des Systèmes du CNRS, Toulouse, France, November 1980.

[9] Peter Dickman and Mesaac Makpangou. A refinement of the fragmented object model. In *1992 Int. Workshop on Object Orientation and Operating Systems*, pages 230–234, Dourdan (France), October 1992. IEEE Comp. Society, IEEE Comp. Society Press.

[10] Peter Dickman, Mesaac Makpangou, and Marc Shapiro. Contrasting fragmented objects with uniform transparent object references for distributed programming. In *5th European SIGOPS Workshop, on "Models and Paradigms for Distributed Systems Structuring"*, Mont Saint-Michel (France), September 1992. ACM SIGOPS, IRISA, INRIA-Rennes.

[11] Paulo Ferreira and Marc Shapiro. Garbage collection and DSM consistency. In *Proc. of the First Symposium on Operating Systems Design and Implementation (OSDI)*, pages 229–241, Monterey CA (USA), November 1994. ACM. `http://www-sor.inria.fr/publi/GC-DSM-CONSIS_OSDI94.html`.

[12] Paulo Ferreira and Marc Shapiro. Garbage collection of persistent objects in distributed shared memory. In *Proc. of the 6th Int. Workshop on Persistent Object Systems*, pages 176–191, Tarascon (France), September 1994. Springer-Verlag. `http://www-sor.inria.fr/publi/GC-PERS-DSM_POS94.html`.

[13] Paulo Ferreira and Marc Shapiro. Garbage collection in the Larchant persistent distributed store. In *Proc. of the 5th Workshop on Future Trends in Distributed Computing Systems (FTDCS'95)*, Cheju Island (Republic of Korea), August 1995.

[14] Paulo Ferreira and Marc Shapiro. Larchant : Persistence by reachability in distributed shared memory through garbage collection. In *Proc. 16th Int. Conf. on Dist. Comp. Syst. (ICDCS)*, Hong Kong, May 1996. `http://www-sor.inria.fr/publi/LPRDSMGC:icdcs96.html`.

[15] Paulo Ferreira and Marc Shapiro. Modelling a distributed cached store for garbage collection. In *12th Euro. Conf. on Object-Oriented Prog. (ECOOP)*, Brussels (Belgium), July 1998. `http://www-sor.inria.fr/publi/MDCSGC_ecoop98.html`.

[16] Paulo Ferreira, Marc Shapiro, Xavier Blondel, Olivier Fambon, João Garcia, Sytse Kloosterman, Nicolas Richer, Marcus Roberts, Fadi Sandakly, George Coulouris, Jean Dollimore, Paulo Guedes, Daniel Hagimont, and Sacha Krakowiak. PerDiS : design, implementation, and use of a PERsistent DIstributed Store. In S. Krakowiak and S. K. Shrivastava, editors, *Recent Advances in Distributed Systems*, volume 1752 of *Lecture Notes in Computer Science*, chapter 18, pages 427–452. Springer-Verlag, February 2000. `http://www-sor.inria.fr/publi/PDIUPDS_lncs1752.html`.

[17] Philippe Gautron and Marc Shapiro. Two extensions to C++ : A dynamic link editor and inner data. In *Proceeding and additional papers, C++ Workshop*, Berkeley, CA (USA), November 1987. USENIX.

[18] Yvon Gourhant. Un conférencier réparti sous SOS, et portage de l'éditeur de liens dynamique. Rapport de Recherche 785, Institut National de la Recherche en Informatique et Automatique, Rocquencourt (France), January 1988.

[19] Yvon Gourhant and Marc Shapiro. FOG/C++ : a fragmented-object generator. In *C++ Conference*, pages 63–74, San Francisco, CA (USA), April 1990. Usenix.

[20] Sabine Habert, Laurence Mosseri, and Vadim Abrossimov. COOL : Kernel support for object-oriented environments. Rapport de Recherche 1211, INRIA, Rocquencourt (France), April 1990.

[21] Sabine Habert, Laurence Mosseri, and Vadim Abrossimov. COOL : Kernel support for object-oriented environments. In *ECOOP/OOPSLA'90 Conference*, volume 25 of *SIGPLAN Notices*, pages 269–277, Ottawa (Canada), October 1990. ACM.

[22] Anne-Marie Kermarrec, Antony Rowstron, Marc Shapiro, and Peter Druschel. The IceCube approach to the reconciliation of divergent replicas. In *20th Symp. on Principles of Dist. Comp. (PODC)*, Newport RI (USA), August 2001. ACM SIGACT-SIGOPS. `http://research.microsoft.com/research/camdis/Publis/podc2001.pdf`.

[23] Jean-Pierre Le Narzul and Marc Shapiro. Un service de nommage pour un système à objets répartis. In *Actes Convention Unix 89*, pages 73–82, Paris, March 1989. AFUU.

[24] Jean-Pierre Le Narzul and Marc Shapiro. Un service de nommage pour un système répartis à objets. In *Séminaire Franco-Brésilien sur les Systèmes Informatiques Répartis*, pages 127–133, Florianopolis (Brazil), September 1989. LAAS and UFSC.

[25] R. Lea and G.S. Blair. The impact of distribution on the object oriented approach to software development. *IEE Software Engineering Journal*, March 1992. `http://www.chorus.com/Documentation/ref.html\#9240`.

[26] Julien Maisonneuve, Marc Shapiro, and Pierre Collet. Implementing references as chains of links. In *1992 Int. Workshop on Object Orientation and Operating Systems*, pages 236–243, Dourdan (France), October 1992. IEEE Comp. Society, IEEE Comp. Society Press.

[27] Julien Maisonneuve and Hervé Soulard. Structure de COOL V1. Rapport Technique 127, Institut National de la Recherche en Informatique et Automatique, Rocquencourt (France), March 1991.

[28] Mesaac Makpangou and Georges Brun-Cottan. Unification des paradigmes de partage en environnement réparti. In *Proc. of the 1st African conf. on Research in Computer Sc.*, volume III, Yaoundé (Cameroun), October 1992.

[29] Mesaac Makpangou, Yvon Gourhant, Jean-Pierre Le Narzul, and Marc Shapiro. Structuring distributed applications as fragmented objects. Rapport de recherche 1404, Institut National de la Recherche en Informatique et Automatique, Rocquencourt (France), January 1991.

[30] Mesaac Makpangou, Yvon Gourhant, Jean-Pierre Le Narzul, and Marc Shapiro. Fragmented objects for distributed abstractions. In T. L. Casavant and M. Singhal, editors, *Readings in Distributed Computing Systems*, pages 170–186. IEEE Computer Society Press, July 1994.

[31] Mesaac Makpangou, Yvon Gourhant, and Marc Shapiro. BOAR : A library of fragmented object types for distributed abstractions. In *Proc. of the International Workshop on Object-Orientation in Operating Systems*, Palo Alto, CA (USA), October 1991.

[32] Mesaac Makpangou and Marc Shapiro. The SOS object-oriented communication service. Rapport de Recherche 801, Institut National de la Recherche en Informatique et Automatique, Rocquencourt (France), March 1988.

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

[34] Mesaac Mounchili Makpangou. Invocations d'objets distants dans SOS. In Guy Pujolle, editor, *De Nouvelles Architectures pour les Communications*, pages 195–201, Paris (France), October 1988. Eyrolles.

[35] Mesaac Mounchili Makpangou. *Protocoles de communication et programmation par objets : l'exemple de SOS*. Thèse de doctorat, Université Paris VI, Paris (France), February 1989.

[36] David Plainfossé and Marc Shapiro. Distributed garbage collection in the system is good. In Cabrera et al. [6], pages 94–99. IEEE Computer Society Press Order Number 2265.

[37] David Plainfossé and Marc Shapiro. Un algorithme conservatif de détection des miettes en milieu réparti. In *Convention Unix 91*, pages 61–74, Paris, March 1991.

[38] David Plainfossé and Marc Shapiro. A distributed GC in an object-support operating system. In *1992 Int. Workshop on Object Orientation and Operating Systems*, pages 221–229, Dourdan (France), October 1992. IEEE Comp. Society, IEEE Comp. Society Press.

[39] David Plainfossé and Marc Shapiro. Experience with a fault-tolerant garbage collector in a distributed Lisp system. In *Proc. 1992 International Workshop on Memory Management*, pages 116–133, Saint-Malo (France), September 1992.

[40] David Plainfossé and Marc Shapiro. A survey of distributed garbage collection techniques. In *Proc. Int. Workshop on Memory Management*, Kinross Scotland (UK), September 1995. `http://www-sor.inria.fr/publi/SDGC_iwmm95.html`.

[41] Nuno Preguica, Marc Shapiro, and Caroline Matheson. Efficient semantics-aware reconciliation for optimistic write sharing. Technical Report MSR-TR-2002-52, Microsoft Research, Cambridge (UK), May 2002. `http://research.microsoft.com/scripts/pubs/view.asp?TR_ID=MSR-TR-2002-52`.

[42] Antony Rowstron and Peter Druschel. Pastry : Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Int. Conf. on Dist. Sys. Platforms (Middleware)*, pages 329–350, Heidelberg, Germany, December 2001. IFIP/ACM. `http://www.research.microsoft.com/~antr/pastry/pubs.htm`.

[43] Antony Rowstron and Peter Druschel. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In *18th ACM Symposium on Operating Systems Principles*, pages 188–201, October 2001.

[44] M. Rozier, V. Abrossimov, F. Armand, I. Boule, M. Gien, M. Guillemont, F. Herrmann, C. Kaiser, S. Langlois, P. Léonard, and W. Neuhauser. Chorus distributed operating systems. *Computing Systems*, 1(4) :305–367, 1988.

[45] Michel Ruffin. Kitlog : A generic logging service. In *Proc. of the 11th Symp. on Reliable Dist. Syst.*, pages 139–146, Houston TX (USA), October 1992.

[46] Michel Ruffin. *Kitlog : Un Service de Journalisation Générique.* Thèse de doctorat, Université Pierre et Marie Curie, Paris VI, Paris (France), September 1992. Available from INRIA as TU-205, ISBN-2-7261-0759-1.

[47] Michel Ruffin. Journalisation dans les systèmes distribués, kitlog : Une solution générique. *Ingénierie des Systèmes d'Information*, 1(2) :229–258, July 1993.

[48] Yasushi Saito and Marc Shapiro. Replication : Optimistic approaches. Technical Report HPL-2002-33, Hewlett-Packard Laboratories, March 2002. `http://www.hpl.hp.com/techreports/2002/HPL-2002-33.html`.

[49] Marc Shapiro. *Une méthode de conception progressive des systèmes parallèles utilisant le langage C.S.P.* Thèse de docteur-ingénieur, Institut National Polytechnique de Toulouse, E.N.S.E.E.I.H.T., Toulouse, France, September 1980.

[50] Marc Shapiro. An experiment in distributed program design, using control enrichment. In *Proc. 3d. Int. Conf. on Dist. Computing Syst.*, Miami-Ft. Lauderdale FL (USA), October 1982.

[51] Marc Shapiro. Un cadre pour la conception de logiciel réparti, sûr et modulaire. *Revue BIGRE*, (30), June 1982.

[52] Marc Shapiro. The contributions of an object-oriented approach to the programming of distributed applications. In Jean Bézivin and Pierre Cointe, editors, *Journée d'Étude sur les Langages Orientés Objet*, number 37 in BIGRE, pages 149–159, Le Cap d'Agde, October 1983. AFCET-Informatique.

[53] Marc Shapiro. Une proposition pour Microrézo : Girolle, un protocole d'appel de procédure distante. In *"Génie Logiciel : le second souffle ?", Journées Bigre 1983*, Le Cap d'Agde, France, October 1983.

[54] Marc Shapiro. Le service d'appel de procédure distante Girolle. *Technique et Science Informatiques*, 3(6) :435–442, December 1984.

[55] Marc Shapiro. Structure and encapsulation in distributed systems : the Proxy Principle. In *The 6th International Conference on Distributed Computer Systems*, pages 198–204, Cambridge, Massachusetts, May 1986. IEEE.

[56] Marc Shapiro. The design of a distributed object-oriented operating system for office applications. In *Proc. Esprit Technical Week 1988*, Brussels (Belgium), November 1988.

[57] Marc Shapiro. Prototyping a distributed object-oriented OS on Unix. In Eugene Spafford, editor, *Workshop on Experiences with Building Distributed and Multiprocessor Systems*, Ft. Lauderdale FL (USA), October 1989. USENIX. Also available as Rapport de Recherche INRIA no. 1082.

[58] Marc Shapiro. A binding protocol for distributed shared objects. In *Proc. Int. Conf. on Distributed Computing Systems*, Poznan (Poland), June 1994.

[59] Marc Shapiro, Vadim Abrossimov, Philippe Gautron, Sabine Habert, and Mesaac Mounchili Makpangou. SOS : un système d'exploitation réparti basé sur les objets. *Techniques et Sciences Informatiques*, 6(2) :166–169, 1987.

[60] Marc Shapiro, Peter Dickman, and David Plainfossé. Robust, distributed references and acyclic garbage collection. In *Symp. on Principles of Distri-*

buted Computing, pages 135–146, Vancouver (Canada), August 1992. ACM. Superseded by [61] : corrects a bug, more elegant, more informative.

[61] Marc Shapiro, Peter Dickman, and David Plainfossé. SSP chains : Robust, distributed references supporting acyclic garbage collection. Rapport de Recherche 1799, Institut National de la Recherche en Informatique et Automatique, Rocquencourt (France), November 1992. `http://www-sor.inria.fr/publi/SSPC_rr1799.html`.

[62] Marc Shapiro and Paulo Ferreira. Larchant-RDOSS : a distributed shared persistent memory and its garbage collector. In J.-M. Hélary and M. Raynal, editors, *Workshop on Distributed Algorithms (WDAG)*, number 972 in Springer-Verlag LNCS, pages 198–214, Le Mont Saint-Michel (France), September 1995. `http://www-sor.inria.fr/publi/LRDSPMGC_wdag95.html`.

[63] Marc Shapiro, Philippe Gautron, and Laurence Mosseri. Persistence and migration for C++ objects. In Stephen Cook, editor, *ECOOP'89, Proc. of the Third European Conf. on Object-Oriented Programming*, British Computer Society Workshop Series, pages 191–204, Nottingham (GB), July 1989. The British Computer Society, Cambridge University Society.

[64] Marc Shapiro, Yvon Gourhant, Sabine Habert, Jean-Pierre Le Narzul, Laurence Mosseri, Michel Ruffin, and Céline Valot. Un bilan du système réparti à objets SOS. Rapport de Recherche 1242, Institut National de la Recherche en Informatique et Automatique, Rocquencourt (France), May 1990.

[65] Marc Shapiro, Yvon Gourhant, Sabine Habert, Laurence Mosseri, Michel Ruffin, and Céline Valot. SOS : An object-oriented operating system — assessment and perspectives. *Computing Systems*, 2(4) :287–338, December 1989.

[66] Marc Shapiro, Olivier Gruber, and David 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), November 1990.

[67] Marc Shapiro and Nuno Guimarães. Software integration in SOMIW : Goals, accomplishments, and problems. In Commission of the European Communities, editor, *Esprit'89, Proceedings of the 6th Annual Esprit Conference*. North-Holland, November 1989.

[68] Marc Shapiro and Sabine Habert. Un système d'exploitation orienté objets pour SOMIW. In *3èmes Journées d'Étude Langages Orientés Objet*, Paris (France), January 1986. AFCET.

[69] Marc Shapiro, Fabrice Le Fessant, and Paulo Ferreira. Recent advances in distributed garbage collection. In S. Krakowiak and S. K. Shrivastava, editors, *Recent Advances in Distributed Systems*, volume 1752 of *Lecture Notes in Computer Science*, chapter 5, pages 104–126. Springer-Verlag, February 2000. `http://www-sor.inria.fr/publi/RAIDGC_lncs1752.html`.

[70] Marc Shapiro and Laurence Mosseri. A simple object storage system. In J. Rosenberg, editor, *Proc. Workshop on persistent object systems*, pages 320–327, Newcastle NSW (Australia), January 1989.

[71] Marc Shapiro, David Plainfossé, Paulo Ferreira, and Laurent Amsaleg. Some key issues in the design of distributed garbage collection and references. In *Unifying Theory and Practice in Distributed Systems*, Dagstuhl (Germany), September 1994.

[72] Marc Shapiro, Antony Rowstron, and Anne-Marie Kermarrec. Application-independent reconciliation for nomadic applications. In *Proc. SIGOPS European Workshop : "Beyond the PC : New Challenges for the Operating System"*, Kolding (Denmark), September 2000. ACM SIGOPS. `http://www-sor.inria.fr/~shapiro/papers/sigops-ew-2000-logmerge.html`.

[73] Hervé Soulard and Mesaac Makpangou. A generic FO-structured framework for persistence support in distributed settings. In *1992 Int. Workshop on Object Orientation and Operating Systems*, pages 57–65, Dourdan (France), October 1992. IEEE Comp. Society, IEEE Comp. Society Press.

[74] Hervé Soulard and Makpangou Mesaac. Une architecture pour la conception de système de stockage. In *Journées des Jeunes Chercheurs en Systèmes Répartis*, Grenoble (France), April 1993.

[75] Douglas B. Terry, Marvin M. Theimer, Karin Petersen, Alan J. Demers, Mike J. Spreitzer, and Carl H. Hauser. Managing update conflicts in Bayou, a weakly connected replicated storage system. In *Proc. 15th ACM Symposium on Operating Systems Principles*, Copper Mountain CO (USA), December 1995. ACM SIGOPS. `http://www.acm.org/pubs/articles/proceedings/ops/224056/p172-terry/p172-terry.pdf`.

[76] Céline Valot. Un moniteur d'objets pour un système réparti à objets. Rapport de Recherche 985, INRIA, Rocquencourt, March 1989.