LNCS 3278

Akhil Sahai
Felix Wu (Eds.)

Utility Computing

15th IFIP/IEEE International Workshop
on Distributed Systems: Operations and Management, DSOM 2004
Davis, CA, USA, November 2004, Proceedings

IFIP TC6

Springer

Akhil Sahai Felix Wu (Eds.)

Utility Computing

15th IFIP/IEEE International Workshop
on Distributed Systems: Operations and Management, DSOM 2004
Davis, CA, USA, November 15-17, 2004
Proceedings

Springer

Volume Editors

Akhil Sahai
Hewlett-Packard Labs
1501 Page Mill Road, Palo-Alto, CA 94304, USA
E-mail: akil.sahai@hp.com

Felix Wu
University of California, Department of Computer Science
One Shields Avenue, Davis, CA 95616, USA
E-mail: wu@cs.ucdavis.edu

Library of Congress Control Number: 2004113913

CR Subject Classification (1998): C.2.4, C.2, D.1.3, D.4.4, K.6, K.4.4

ISSN 0302-9743
ISBN 3-540-23631-7 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by PTP-Berlin, Protago-TeX-Production GmbH
Printed on acid-free paper SPIN: 11339861 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3278

Preface

This volume of the Lecture Notes in Computer Science series contains all the papers accepted for presentation at the 13th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM 2004), which was held at the University of California, Davis during November 15–17, 2004.

DSOM 2004 was the fifteenth workshop in a series of annual workshops and it followed in the footsteps of highly successful previous meetings, the most recent of which were held in Heidelberg, Germany (DSOM 2003), Montreal, Canada (DSOM 2002), Nancy, France (DSOM 2001), and Austin, USA (DSOM 2000). The goal of the DSOM workshops is to bring together researchers in the areas of networks, systems, and services management, from both industry and academia, to discuss recent advances and foster future growth in this field. In contrast to the larger management symposia, such as IM (Integrated Management) and NOMS (Network Operations and Management Symposium), the DSOM workshops are organized as single-track programs in order to stimulate interaction among participants.

The focus of DSOM 2004 was "Management Issues in Utility Computing." Increasingly there is a trend now towards managing large infrastructures and services within utility models where resources can be obtained on demand. Such a trend is being driven by the desire to consolidate infrastructures within enterprises and across enterprises using third-party infrastructure providers and networked infrastructures like Grid and PlanetLab. The intent in these initiatives is to create systems that provide automated provisioning, configuration, and lifecycle management of a wide variety of infrastructure resources and services, on demand. The papers presented at the workshop address the underlying technologies that are key to the success of the utility computing paradigm.

This year we received about 110 high-quality papers of which 21 long papers were selected for the 7 long paper sessions and 4 short papers were selected for the short paper session. The technical sessions covered the topics "Management Architectures," "SLA and Business Objective Driven Management," "Policy-Based Management," "Automated Management," "Analysis and Reasoning in Management," "Trust and Security," and "Implementation, Instrumentation, Experience."

This workshop owed its success to all the members of the technical program committee, who did an excellent job of encouraging their colleagues in the field to submit a total of 110 high-quality papers, and who devoted a lot of their time to help create an outstanding technical program. We thank them profusely. We would like to thank Hewlett-Packard and HP Laboratories, the DSOM 2004 Corporate Patron.

September 2004

Akhil Sahai,
Felix Wu

DSOM 2004

November 15–17, 2004, Davis, California, USA

Sponsored by
Hewlett-Packard
IFIP

in cooperation with
IEEE Computer Society
Computer Science Department, University of California, Davis

Program Chairs

Akhil Sahai, Hewlett-Packard, Palo Alto, California, USA

Felix Wu, University of California at Davis, USA

Program Committee

Anerousis, Nikos .... IBM T.J. Watson Research Center, USA
Boutaba, Raouf .... University of Waterloo, Canada
Brunner, Marcus .... NEC Europe, Germany
Burgess, Mark .... University College Oslo, Norway
Cherkaoui, Omar .... Université du Québec á Montréal, Canada
Clemm, Alexander .... Cisco, California, USA
Feridun, Metin .... IBM Research, Switzerland
Festor, Olivier .... LORIA-INRIA, France
Geihs, Kurt .... TU Berlin, Germany
Gentzsch,Wolfgang .... Sun Microsystems, USA
Hegering, Heinz-Gerd .... Institut für Informatik der LMU, Germany
Hellerstein, Joseph .... IBM T.J. Watson Research Center, USA
Jakobson, Gabe .... Smart Solutions Consulting, USA
Kaiser, Gail .... Columbia University, USA
Kar, Gautam .... IBM T.J. Watson Research Center, USA
Kawamura, Ryutaro .... NTT Cyber Solutions Labs, Japan
Keller, Alexander .... IBM T.J. Watson Research Center, USA
Lala, Jaynarayan .... Raytheon, USA
Lewis, Lundy .... Lundy Lewis Associates, USA
Liotta, Antonio .... University of Surrey, UK
Lupu, Emil .... Imperial College London, UK
Lutfiyya, Hanan .... University of Western Ontario, Canada
Martin-Flatin, J.P. .... CERN, Switzerland
Maughan, Douglas .... DHS/HSARPA,USA
Mazumdar, Subrata .... Avaya Labs Research, Avaya, USA

Nogueira, Jose .... Federal University of Minas Gerais, Brazil
Pavlou, George .... University of Surrey, UK
Pras, Aiko .... University of Twente, The Netherlands
Quittek, Juergen .... NEC Europe, Germany
Raz, Danny .... Technion, Israel
Rodosek, Gabi Dreo .... Leibniz Supercomputing Center, Germany
Schoenwaelder, Juergen .... International University Bremen, Germany
Sethi, Adarshpal .... University of Delaware, USA
Singhal, Sharad .... Hewlett-Packard Labs, USA
Sloman, Morris .... Imperial College London, UK
Stadler, Rolf .... KTH, Sweden
State, Radu .... LORIA-INRIA, France
Stiller, Burkhard ..... UniBW Munich, Germany & ETH Zurich, Switzerland
Torsten, Braun .... University of Bern, Switzerland
Tutschku, Kurt .... University of Wuerzburg, Germany
Wang, Yi-Min .... Microsoft Research, USA
Becker, Carlos Westphall .... Federal University of Santa Catarina, Brazil
Yoshiaki, Kirha .... NEC, Japan

Lecture Notes in Computer Science

For information about Vols. 1–3193

please contact your bookseller or Springer

Vol. 3305: P.M.A. Sloot, B. Chopard, A.G. Hoekstra (Eds.), Cellular Automata. XV, 883 pages. 2004.

Vol. 3302: W.-N. Chin (Ed.), Programming Languages and Systems. XIII, 453 pages. 2004.

Vol. 3299: F. Wang (Ed.), Automated Technology for Verification and Analysis. XII, 506 pages. 2004.

Vol. 3293: C.-H. Chi, M. van Steen, C. Wills (Eds.), Web Content Caching and Distribution. IX, 283 pages. 2004.

Vol. 3292: R. Meersman, Z. Tari, A. Corsaro (Eds.), On the Move to Meaningful Internet Systems 2004: OTM 2004 Workshops. XXIII, 885 pages. 2004.

Vol. 3291: R. Meersman, Z. Tari (Eds.), On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE. XXV, 824 pages. 2004.

Vol. 3290: R. Meersman, Z. Tari (Eds.), On the Move to Meaningful Internet Systems 2004: CoopIS, DOA, and ODBASE. XXV, 823 pages. 2004.

Vol. 3287: A. Sanfeliu, J.F.M. Trinidad, J.A. Carrasco Ochoa (Eds.), Progress in Pattern Recognition, Image Analysis and Applications. XVII, 703 pages. 2004.

Vol. 3286: G. Karsai, E. Visser (Eds.), Generative Programming and Component Engineering. XIII, 491 pages. 2004.

Vol. 3284: A. Karmouch, L. Korba, E.R.M. Madeira (Eds.), Mobility Aware Technologies and Applications. XII, 382 pages. 2004.

Vol. 3281: T. Dingsøyr (Ed.), Software Process Improvement. X, 207 pages. 2004.

Vol. 3280: C. Aykanat, T. Dayar, İ. Körpeoğlu (Eds.), Computer and Information Sciences - ISCIS 2004. XVIII, 1009 pages. 2004.

Vol. 3278: A. Sahai, F. Wu (Eds.), Utility Computing. XI, 270 pages. 2004.

Vol. 3274: R. Guerraoui (Ed.), Distributed Computing. XIII, 465 pages. 2004.

Vol. 3273: T. Baar, A. Strohmeier, A. Moreira, S.J. Mellor (Eds.), <<UML>> 2004 - The Unified Modelling Language. XIII, 454 pages. 2004.

Vol. 3271: J. Vicente, D. Hutchison (Eds.), Management of Multimedia Networks and Services. XIII, 335 pages. 2004.

Vol. 3270: M. Jeckle, R. Kowalczyk, P. Braun (Eds.), Grid Services Engineering and Management. X, 165 pages. 2004.

Vol. 3269: J. Lopez, S. Qing, E. Okamoto (Eds.), Information and Communications Security. XI, 564 pages. 2004.

Vol. 3266: J. Solé-Pareta, M. Smirnov, P.V. Mieghem, J. Domingo-Pascual, E. Monteiro, P. Reichl, B. Stiller, R.J. Gibbens (Eds.), Quality of Service in the Emerging Networking Panorama. XVI, 390 pages. 2004.

Vol. 3265: R.E. Frederking, K.B. Taylor (Eds.), Machine Translation: From Real Users to Research. XI, 392 pages. 2004. (Subseries LNAI).

Vol. 3264: G. Paliouras, Y. Sakakibara (Eds.), Grammatical Inference: Algorithms and Applications. XI, 291 pages. 2004. (Subseries LNAI).

Vol. 3263: M. Weske, P. Liggesmeyer (Eds.), Object-Oriented and Internet-Based Technologies. XII, 239 pages. 2004.

Vol. 3262: M.M. Freire, P. Chemouil, P. Lorenz, A. Gravey (Eds.), Universal Multiservice Networks. XIII, 556 pages. 2004.

Vol. 3261: T. Yakhno (Ed.), Advances in Information Systems. XIV, 617 pages. 2004.

Vol. 3260: I.G.M.M. Niemegeers, S.H. de Groot (Eds.), Personal Wireless Communications. XIV, 478 pages. 2004.

Vol. 3258: M. Wallace (Ed.), Principles and Practice of Constraint Programming – CP 2004. XVII, 822 pages. 2004.

Vol. 3257: E. Motta, N.R. Shadbolt, A. Stutt, N. Gibbins (Eds.), Engineering Knowledge in the Age of the Semantic Web. XVII, 517 pages. 2004. (Subseries LNAI).

Vol. 3256: H. Ehrig, G. Engels, F. Parisi-Presicce, G. Rozenberg (Eds.), Graph Transformations. XII, 451 pages. 2004.

Vol. 3255: A. Benczúr, J. Demetrovics, G. Gottlob (Eds.), Advances in Databases and Information Systems. XI, 423 pages. 2004.

Vol. 3254: E. Macii, V. Paliouras, O. Koufopavlou (Eds.), Integrated Circuit and System Design. XVI, 910 pages. 2004.

Vol. 3253: Y. Lakhnech, S. Yovine (Eds.), Formal Techniques, Modelling and Analysis of Timed and Fault-Tolerant Systems. X, 397 pages. 2004.

Vol. 3252: H. Jin, Y. Pan, N. Xiao, J. Sun (Eds.), Grid and Cooperative Computing - GCC 2004 Workshops. XVIII, 785 pages. 2004.

Vol. 3251: H. Jin, Y. Pan, N. Xiao, J. Sun (Eds.), Grid and Cooperative Computing - GCC 2004. XXII, 1025 pages. 2004.

Vol. 3250: L.-J. (LJ) Zhang, M. Jeckle (Eds.), Web Services. X, 301 pages. 2004.

Vol. 3249: B. Buchberger, J.A. Campbell (Eds.), Artificial Intelligence and Symbolic Computation. X, 285 pages. 2004. (Subseries LNAI).

Vol. 3246: A. Apostolico, M. Melucci (Eds.), String Processing and Information Retrieval. XIV, 332 pages. 2004.

Vol. 3245: E. Suzuki, S. Arikawa (Eds.), Discovery Science. XIV, 430 pages. 2004. (Subseries LNAI).

Vol. 3244: S. Ben-David, J. Case, A. Maruoka (Eds.), Algorithmic Learning Theory. XIV, 505 pages. 2004. (Subseries LNAI).

Vol. 3243: S. Leonardi (Ed.), Algorithms and Models for the Web-Graph. VIII, 189 pages. 2004.

Vol. 3242: X. Yao, E. Burke, J.A. Lozano, J. Smith, J.J. Merelo-Guerv[illegible] B[illegible]linaria, J. Rowe, P. Tiňo, A. Kabán, H.-P. [illegible]wefel (Eds.), Parallel Problem Solving from Nature - PPSN VIII. XX, 1185 pages. 2004.

Vol. 3241: D. Kranzlmüller, P. Kacsuk, J.J. Dongarra (Eds.), Recent Advances in Parallel Virtual Machine and Message Passing Interface. XIII, 452 pages. 2004.

Vol. 3240: I. Jonassen, J. Kim (Eds.), Algorithms in Bioinformatics. IX, 476 pages. 2004. (Subseries LNBI).

Vol. 3239: G. Nicosia, V. Cutello, P.J. Bentley, J. Timmis (Eds.), Artificial Immune Systems. XII, 444 pages. 2004.

Vol. 3238: S. Biundo, T. Frühwirth, G. Palm (Eds.), KI 2004: Advances in Artificial Intelligence. XI, 467 pages. 2004. (Subseries LNAI).

Vol. 3236: M. Núñez, Z. Maamar, F.L. Pelayo, K. Pousttchi, F. Rubio (Eds.), Applying Formal Methods: Testing, Performance, and M/E-Commerce. XI, 381 pages. 2004.

Vol. 3235: D. de Frutos-Escrig, M. Nunez (Eds.), Formal Techniques for Networked and Distributed Systems - FORTE 2004. X, 377 pages. 2004.

Vol. 3232: R. Heery, L. Lyon (Eds.), Research and Advanced Technology for Digital Libraries. XV, 528 pages. 2004.

Vol. 3231: H.-A. Jacobsen (Ed.), Middleware 2004. XV, 514 pages. 2004.

Vol. 3230: J.L. Vicedo, P. Martínez-Barco, R. Muñoz, M. Saiz Noeda (Eds.), Advances in Natural Language Processing. XII, 488 pages. 2004. (Subseries LNAI).

Vol. 3229: J.J. Alferes, J. Leite (Eds.), Logics in Artificial Intelligence. XIV, 744 pages. 2004. (Subseries LNAI).

Vol. 3226: M. Bouzeghoub, C. Goble, V. Kashyap, S. Spaccapietra (Eds.), Semantics of a Networked World. XIII, 326 pages. 2004.

Vol. 3225: K. Zhang, Y. Zheng (Eds.), Information Security. XII, 442 pages. 2004.

Vol. 3224: E. Jonsson, A. Valdes, M. Almgren (Eds.), Recent Advances in Intrusion Detection. XII, 315 pages. 2004.

Vol. 3223: K. Slind, A. Bunker, G. Gopalakrishnan (Eds.), Theorem Proving in Higher Order Logics. VIII, 337 pages. 2004.

Vol. 3222: H. Jin, G.R. Gao, Z. Xu, H. Chen (Eds.), Network and Parallel Computing. XX, 694 pages. 2004.

Vol. 3221: S. Albers, T. Radzik (Eds.), Algorithms – ESA 2004. XVIII, 836 pages. 2004.

Vol. 3220: J.C. Lester, R.M. Vicari, F. Paraguaçu (Eds.), Intelligent Tutoring Systems. XXI, 920 pages. 2004.

Vol. 3219: M. Heisel, P. Liggesmeyer, S. Wittmann (Eds.), Computer Safety, Reliability, and Security. XI, 339 pages. 2004.

Vol. 3217: C. Barillot, D.R. Haynor, P. Hellier (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2004. XXXVIII, 1114 pages. 2004.

Vol. 3216: C. Barillot, D.R. Haynor, P. Hellier (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2004. XXXVIII, 930 pages. 2004.

Vol. 3215: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems. LVII, 906 pages. 2004. (Subseries LNAI).

Vol. 3214: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems. LVIII, 1302 pages. 2004. (Subseries LNAI).

Vol. 3213: M.G.. Negoita, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems. LVIII, 1280 pages. 2004. (Subseries LNAI).

Vol. 3212: A. Campilho, M. Kamel (Eds.), Image Analysis and Recognition. XXIX, 862 pages. 2004.

Vol. 3211: A. Campilho, M. Kamel (Eds.), Image Analysis and Recognition. XXIX, 880 pages. 2004.

Vol. 3210: J. Marcinkowski, A. Tarlecki (Eds.), Computer Science Logic. XI, 520 pages. 2004.

Vol. 3209: B. Berendt, A. Hotho, D. Mladenic, M. van Someren, M. Spiliopoulou, G. Stumme (Eds.), Web Mining: From Web to Semantic Web. IX, 201 pages. 2004. (Subseries LNAI).

Vol. 3208: H.J. Ohlbach, S. Schaffert (Eds.), Principles and Practice of Semantic Web Reasoning. VII, 165 pages. 2004.

Vol. 3207: L.T. Yang, M. Guo, G.R. Gao, N.K. Jha (Eds.), Embedded and Ubiquitous Computing. XX, 1116 pages. 2004.

Vol. 3206: P. Sojka, I. Kopecek, K. Pala (Eds.), Text, Speech and Dialogue. XIII, 667 pages. 2004. (Subseries LNAI).

Vol. 3205: N. Davies, E. Mynatt, I. Siio (Eds.), UbiComp 2004: Ubiquitous Computing. XVI, 452 pages. 2004.

Vol. 3204: C.A. Peña Reyes, Coevolutionary Fuzzy Modeling. XIII, 129 pages. 2004.

Vol. 3203: J. Becker, M. Platzner, S. Vernalde (Eds.), Field Programmable Logic and Application. XXX, 1198 pages. 2004.

Vol. 3202: J.-F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), Knowledge Discovery in Databases: PKDD 2004. XIX, 560 pages. 2004. (Subseries LNAI).

Vol. 3201: J.-F. Boulicaut, F. Esposito, F. Giannotti, D. Pedreschi (Eds.), Machine Learning: ECML 2004. XVIII, 580 pages. 2004. (Subseries LNAI).

Vol. 3199: H. Schepers (Ed.), Software and Compilers for Embedded Systems. X, 259 pages. 2004.

Vol. 3198: G.-J. de Vreede, L.A. Guerrero, G. Marín Raventós (Eds.), Groupware: Design, Implementation and Use. XI, 378 pages. 2004.

Vol. 3196: C. Stary, C. Stephanidis (Eds.), User-Centered Interaction Paradigms for Universal Access in the Information Society. XII, 488 pages. 2004.

Vol. 3195: C.G. Puntonet, A. Prieto (Eds.), Independent Component Analysis and Blind Signal Separation. XXIII, 1266 pages. 2004.

Vol. 3194: R. Camacho, R. King, A. Srinivasan (Eds.), Inductive Logic Programming. XI, 361 pages. 2004. (Subseries LNAI).

Table of Contents

Analysis and Reasoning

Trust and Security

Implementation, Instrumentation, Experience

Short Papers

Requirements on Quality Specification Posed by Service Orientation

Markus Garschhammer and Harald Roelle

Munich Network Management Team
University of Munich
Oettingenstr. 67
D-80538 Munich, Germany
{markus.garschammer,harald.roelle}@ifi.lmu.de

Abstract. As service orientation is gaining more and more momentum, the need for common concepts regarding Quality of Service (QoS) and its specification emerges. In recent years numerous approaches to specifying QoS were developed for special subjects like multimedia applications or middleware for distributed systems. However, a survey of existing approaches regarding their contribution to service oriented QoS specification is still missing.
In this paper we present a strictly service oriented, comprehensible classification scheme for QoS specification languages. The scheme is based on the MNM Service Model and the newly introduced LAL–brick which aggregates the dimensions **L**ife cycle, **A**spect and **L**ayer of a QoS specification. Using the terminology of the MNM Service Model and the graphical notation of the LAL–brick we are able to classify existing approaches to QoS specification. Furthermore we derive requirements for future specification concepts applicable in service oriented environments.

Keywords: QoS specification, service orientation, classification scheme

1 Introduction

In recent years Telco and IT industries have been shifting their business from monolithic realizations to the composition of products by outsourcing, which results in creating business critical value chains. This trend has had its impact on IT management and paved the way for concepts subsumed under the term service (oriented) management. Now, that relations involved in providing a service are crossing organizational boundaries, unambiguous specifications of interfaces are more important than ever before. In federated environments they are a fundament for rapid and successful negotiation as well as for smooth operation. Here, not only functional aspects have to be addressed, but quality is also an important issue.

In the context of service management, the technical term Quality of Service (QoS) is now perceived in its original sense. Prior to the era of service orientation, the term QoS was mainly referred to as some more or less well defined technical criterion on the network layer. Nowadays, QoS is regaining its original meaning of describing a service's quality in terms which are intrinsic to the service itself. Furthermore, QoS now

A. Sahai and F. Wu (Eds.): DSOM 2004, LNCS 3278, pp. 1–14, 2004.

reflects the demand for customer orientation, as QoS should be expressed in a way that customers understand, and not in the way a provider's implementation dictates it.

In the past, a number of QoS specification concepts and languages have been proposed. Unfortunately, when applied to real world scenarios in a service oriented way, each shows weaknesses in different situations. For example, considering negotiations between a customer and a provider, some are well suited regarding customer orientation, as they are easily understood by the customer. But they are of only limited use for the provider's feasibility and implementation concerns. Other specification techniques suffer from the inverse problem.

Apparently there is room for improvement in service oriented specification of service quality. This paper contributes to the field by introducing a classification scheme for quality specification techniques which is strictly service oriented. This is accomplished by considering e.g. the service life cycle, different roles and types of functionality. By applying the classification scheme to representative examples of quality specification techniques, the current status in the field is outlined. To contribute to the advancement of service orientation, we derive requirements for next generation specification techniques by two ways: First we analyze today's works' flaws and second we deduce requirements from our classification scheme.

The paper is organized as follows. In the next section (Sec. 2) the classification scheme for QoS specification languages and concepts is introduced. Application to typical examples of QoS specification techniques is discussed in Sec. 3. Using these results, the following section (Sec. 4) identifies requirements for future quality specification approaches. Section 5 concludes the paper and gives an outlook on further work.

2 Classification Scheme

In this section a classification scheme for quality specification concepts and languages is developed. In doing so, the paradigm of service orientation is strictly followed. Multiple aspects of services are covered, functional aspects of a specification as well as its expressiveness in relation to service properties.

In order to develop the classification, the MNM Service Model [GHH+02,GHK+01] is used as the foundation for the classification. It is a bottom–up developed model which defines a common terminology in generic service management, specifies atomic roles and denotes the major building blocks a service is composed of. Doing so, it offers a generic view on the building blocks rather than a specification for direct implementation. As the MNM Service Model is a generic model, which is not focusing a certain scenario, it serves well as a starting point for our classification, in respect to develop a model where completeness, generic applicability and scenario independency is ensured.

The second ingredient for our classification is the set of common concepts that can be found in various quality specification schemes. This set was derived from the survey of Jin and Nahrstedt [JN04] and is an enhancement of the taxonomy presented there.

2.1 The MNM Service Model as a Map for Specification Concepts

The MNM Service Model offers two major views (Service and Realization View) which group a service's building blocks into different domains, according to their roles and related responsibilities. Figure 1 combines the two views. One major characteristic of the model is the so called *Side Independent Part*[1]. Beside the *service* itself, it depicts additional building blocks which should be specified independently from realization details on either the *provider side* and the *customer side*.

The MNM Service Model decomposes the specification of a service's quality in two parts. The first part describes quality relevant properties of a *service* (class *QoS Parameters* in Fig. 1). The second part poses constraints on those specified properties which have to be met by the provider and are agreed upon in an *service agreement*. For both parts, relevant properties of the system have to be defined in an unambiguous manner.

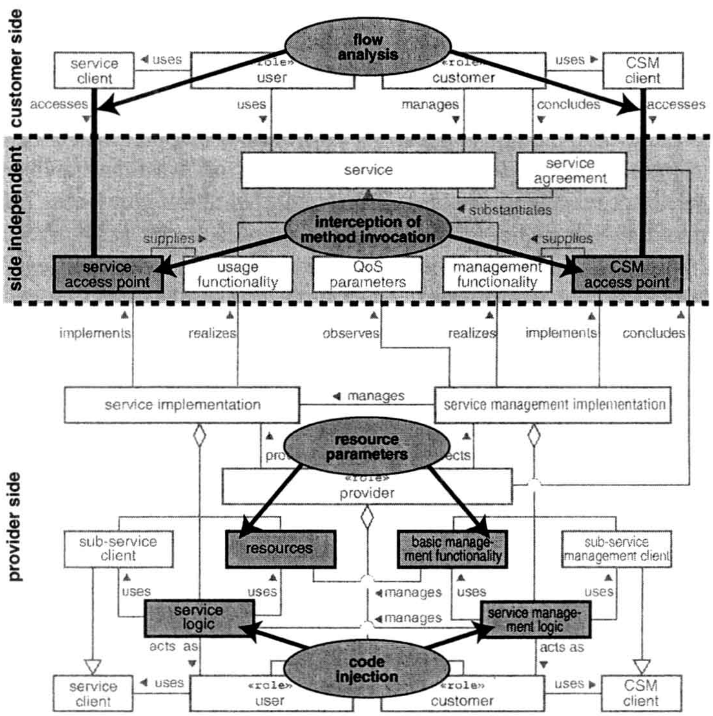

Fig. 1. Reference Points located in combined view of the MNM Service Model

QoS parameters may be specified against different reference points. Thus, even when they bear the same name, they may have different semantics. For example, a delay specification could be measured at the *user's client* or inside the *service implementation*, which results in different QoS parameters. In fact, our extension of the taxonomy presented in [JN04] describes such possible reference points for quality specification.

[1] In the following, parts of the model will be printed in italics

By locating the reference points in the MNM Service Model characteristics of these reference points can be identified. First of all, the model's part where the reference point is located, enables us to identify the affected roles. Furthermore, this allows us to draw conclusions on dependencies to other parts of a service. Thus we can identify typical properties of reference points, like limitations regarding portability to different realizations or the applicability in situations, when service chains come into play.

The following paragraphs first describe those reference points. In each paragraph's second part the reference point is located in the MNM Service Model as depicted in Fig. 1. By this, basic characteristics of specification techniques using the respective reference point can be pointed out later on by simply marking the corresponding reference point in the MNM Service Model.

Flow/Communication. Most of today's common QoS parameters, such as throughput or delay, are measured and thus specified from a communications point of view. Quality related properties of a service are derived from properties of a data stream or, in general, a flow. Constraints on the quality of a service are simply mapped onto constraints of the flow (e.g. "the transmission delay must not exceed 10ms"). So the quality of a service is only implicitly defined by properties of the communication it induces. This definition is therefore at the risk of being too coarse in respect to the service's functionality. However, this way of expressing quality is widespread because properties of a flow can be easily derived in real world scenarios. A typical example would be an ATM based video conferencing service where its properties are described as ATM QoS parameters.

In the MNM Service Model a communication flow in general can be observed between a client and the corresponding service access point (SAP). This relation exists between the *service client* and *service access point* (when accessing the service's *usage functionality*) as well as between the *customer service management (CSM) client* and the *customer service management (CSM) access point* (when accessing the *management functionality*). Hence, a quality specification has to be applied not only to the usage functionality but also to the management side.

As can be seen in Fig. 1, the relation between the *service client* and the *service access point* crosses the boundary between the *customer side* and the *side independent* part of the model (the same applies for the management side). Any analysis of flows depend on the service clients, thus, it cannot be implementation independent. In consequence, specifications using the technique of flow analysis depend on a client's implementation as well.

Method/API Invocation. Another technique to derive quality relevant properties of a service is motivated by object oriented (OO) design, programming and middleware architectures. Here, quality is specified as properties of a method invocation, e.g. the time it takes a method for encoding a video frame to finish. Constraints on these properties can be posed as easily as in the former case. This method of quality measurement and description requires the interception of method invocations. As this is naturally done in component oriented middleware, this technique is mostly used there.

Method invocation may occur at almost any functional component of the MNM Service Model. However, the invocation interception concept used in OO environments or middlewares can be mapped best to the service's access points where methods in the sense of service (management) functions can be invoked. The idea of interception of method invocations is therefore depicted in Fig. 1 at the *service access point* and the *customer service management (CSM) access point*. As this concept only uses blocks of the model which are located in the *side independent* part, it does not depend on any implementation, neither on the *customer* nor on the *provider side*.

Code Injection. The idea of code injection is to directly integrate constraints on quality into a service's implementation — into its executable code. Steps of execution monitored by injected code yield service properties (such as processing time or memory consumption). Constraints on these properties are inferred by directly coding conditional actions to be executed when a constraint is satisfied or violated. For example, information on memory usage during the decoding of a video stream is measured. If it exceeds a certain value, a less memory consuming but also worse performing decoding method is used. This procedure automatically assures a certain quality, in this case a guaranteed maximum amount of memory used.

The MNM Service Model divides a service's implementation into three parts: *subservice client*, *service logic* and *resources*. The *service logic* orchestrates resources and subservices to implement the service's *usage functionality* as a whole. The idea of code injection, in the sense of the service model, is to enhance the *service logic* with inserted code to automatically derive properties of the running service. Observation of these properties and reaction to changes are directly coupled. As the Service Model distinguishes between a service's *usage functionality* and its *management functionality*, this concept is shown in both the *service logic* and the *service management logic*. As one can easily see, the idea of code injection depends directly on a service's implementation by a provider. It is therefore an instrument for providers to assure a certain quality, but obviously should not be used in service negotiation when customer and provider need to establish a common understanding of a service's quality parameters.

Resource Parameters. Quality relevant properties of a service can also be derived from the parameters of resources the service is realized with. For this purpose, resource parameters can be aggregated in various ways.

However, details of the gathering and aggregation process have to be defined after service deployment, because relevant details of concrete resources used are unknown before deployment. Even worse, the specification may have to be adapted on a service instance basis because different service instances might use different resources, whose concrete characteristics might be needed for specification. Constraints on these resource oriented properties can be posed at various aggregation levels, but their derivation from constraints posed on the service itself is not a trivial task.

In the MNM Service Model, information about resources can be directly gathered from the *resources*, but can also be obtained via the class *basic management functionality*. When specifying quality aspects of the *management functionality* the *basic management functionality* itself is targeted in a QoS specification. As the location of both *resources* and *basic management functionality* inside the provider's domain illustrates, even with a suitable aggregation method, this concept of specification can only express quality that