

# Policy-based Management for Resource-Specific Semantic Services

David Lewis, John Keeney, Declan O'Sullivan  
*Knowledge and Data Engineering Group, Trinity College Dublin*  
{Dave.Lewis|John.Keeney|Declan.OSullivan}@cs.tcd.ie

## Abstract

*Semantic Web Services are the subject of intense scrutiny as they promise to address problems of dynamically discovering, selecting, composing and interoperating web services for e-commerce. However, there has been relatively little attention paid to the application of semantic service models to non-web services. Services in telecommunications and enterprise networks can equally be modeled as individual or composed semantic services, but with alternative grounding to the underlying, domain-specific service mechanisms. These domains reveal a further interesting requirement for the modeling of semantic services in that they often represent services on specific devices. As these devices house resources that must be managed as part of the operation of the network, the semantic model of the service must be integrated with the model that allows the resources to be efficiently managed. This paper presents a scheme via which policy-based management directives for a resource can be applied to a device offering semantically annotated services. We show that this integration can be achieved using existing service choreography expressions and illustrate this using an enterprise networking example based on existing standardized semantics for networked printers.*

## 1. Introduction

The popularity of Service-Oriented Architectures in integrating distributed systems, and the recent standardization of description logic languages for describing ontologies under the World Wide Web consortium's Semantic Web initiative [berners-lee] has resulted in intense research into languages for expressing and manipulating Semantic Web Services. These typically aim to integrate with

existing web service languages such as the Web Service Description Language (WSDL) and thus aim to exploit the array of WSDL compatible service execution technologies. Semantic Web Service languages typically incorporate composite service features from existing web service languages such as BPEL4WS [bpel4ws]. This allows them to express complex service interactions between a web service provider and its service consuming client, a modeling approach termed *choreography*. Alternatively, composite services may be expressed using a business process abstraction which describes how one service is provided by control and data flows between a set of constituent services, each of which may be in turn further decomposed. This latter approach to modeling composite services is termed *service orchestration*.

Semantic Web Service languages also introduce the modeling of conditional expressions detailing the state of the world in which the service is executed, before and after the services invocation. Conditional expression can also operate on the knowledge taken in and emitted by a service, i.e. its input and outputs.

Applying ontology-based semantics to web service descriptions offers the possibility of exploiting automated reasoning using off-the-shelf logic engines to assist in service discovery and service composition [mcilraith]. This offers the possibility of automating or semi-automating what are currently human-led engineering tasks. For instance in discovering and selecting a service, ontological queries can match terms in a service request to terms in service descriptions that are defined in an ontology to be sub-classes or super-classes of the requested terms. Alternatively, existing AI planning and situation calculus techniques have been applied to

sequential composition of services based on their semantic descriptions.

Currently, as this activity is led by the W3C, web-based e-commerce is seen as the primary profitable application area. Thus there has been relatively little attention paid to the application of semantic service models to non-web services. Nevertheless several researchers have highlighted that the above benefits could also be exploited in other areas where service-oriented solutions are sought amid a level of service heterogeneity. Alternative applications include: pervasive computing [masuoka], telecommunication networks or enterprise networks [duke], where hardware and software from multiple vendors need to be integrated rapidly to respond to changing value chain requirements. Such applications require groundings from semantic service descriptions to other service oriented mechanisms than those of the Web, e.g. CORBA, JXTA, or any of the many application layer communication protocols deployed directly on networks, e.g. SS7 or SIP. Fortunately, though much research on ontology-based service semantics focus on WSDL groundings, the languages typically do not exclude grounding to multiple service mechanisms, though few alternatives have been addressed in practice.

These non-web application domains reveal a further interesting requirement for the modeling of semantic services in that they often represent services on specific to certain types of physical devices. These device types are characterized by physical resources, e.g. toner level in a printer or routing tables in an IP router, that typically play a role both in delivering the value of services offered by the device and in administrating the operation of the device by its owner. This differs from the models recently being examined for the management of web services, e.g. by OASIS Web Service Distributed Management Technical committee ([www.oasis-open.org](http://www.oasis-open.org)), in that these resources are related to the operation of the device rather than to the operation of the Web Service. The increasing tendency to operate web services from sophisticated, commoditized server farms means that the management of the computing and network resources underlying the service is not closely integrated to the semantics of the service

itself. Grid technologies, due to the specialized nature of the services offered, sometimes provide a higher level of integration in the view of the service offered and how it is managed [foster]. However, when considering the operation of individual devices on a network, the value of the service that device offers is more closely linked to the resources that characterize the device, rather than being a web service using a pool of generic computing resources on a server farm. In other words, for the devices in which we are interested offer specific resources that underpin the value provided by the device's service, rather than general purpose computing and storage resources used to delivery a range of web services. The significance of this is that the latter resources can (and are) being standardized, e.g. by Open Grid Service Architecture ([www.globus.org/ogsa/](http://www.globus.org/ogsa/)) and the Web Services Distributed Management technical committee at OASIS ([www.oasis-open.org](http://www.oasis-open.org)). However, service-specific device resources will continue to demonstrate a higher degree of heterogeneity due to their specialized nature. It is therefore important when considering the operation of devices that offer services, that the engineering of the management of that device is closely integrated to the semantics of the service being offered.

Typically, in networked devices the management of resources has been handled using the manager-agent paradigm where the resources are modeled as a set of managed objects that can be manipulated by a managing application via a well defined management protocol (e.g. through CMIP, SNMP etc.). Such management information modeling has largely been a manual task requiring good knowledge of the services (typically expressed as communication protocols) offered by the device.

Increasingly, however, the availability of increased computing power on an individual device means that management decision making can be delegated to the device itself, without recourse to remote managing applications, and the architectural centralization and communication overhead that it typically entails. Where the required management actions for the occurrence of a particular operational state, e.g. a partial failure or performance dip, is well understood, the

binding between that state and the action that needs to be taken can be encoded in a declarative policy rule, which can be down loaded to the device for local evaluation.

This paper presents a scheme via which policy-based management directives for a resource can be applied to a device offering semantically annotated services. Policy-based management allows management decisions to be encoded and thus executed in a decentralized manner. To support an increasingly autonomous mode of policy-based management, policies must be engineered in a manner consistent with the semantics of the services offered by a device and of the resources that underpin those services. We show that this integration can be achieved using existing service choreography expressions and illustrate this using an enterprise networking example based on existing standardized semantics for networked printers.

## 2. Semantic Web Services

The Semantic Web Service working group of the W3C has identified a number of semantic web service frameworks. Some, such as WSDL-S, simply enable the referencing of external semantic files from within WSDL [akkiraju]. This for example allows an ontological description of a service parameter to be defined separately using the W3C's standardized Web Ontology Language (OWL) [owl]. Two other approaches that appear to offer a more comprehensive approach to working with semantic web services are OWL for Services (OWL-S) and the Web Service Modeling Ontology (WSMO).

### 2.1 OWL-S

OWL-S is an OWL ontology for describing services, thus reflecting the W3C approach of building more advanced Semantic Web features upon a 'stack' of standards [martin]. OWL-S aims to support the automated discovery, invocation, composition and management of web services. It consists of a number of interlinked models:

- the service profile which is used in advertising and selecting web services,
- the service model which is a process-oriented view of how services can be

composed (or orchestrated) in a nested manner;

- the grounding model that defines how the ontological service model is mapped onto a concrete communications mechanism (though only a WSDL grounding has been defined to date)
- a resource model offering shared semantics for underlying resources.

OWL-S defines a service in terms of its input and output parameters and in terms of preconditions that must be true before the service is invoked and effects which may become true once the invocation is completed. For the conditional terms, OWL-S requires an additional rule language, and currently allows a number of languages to be used while awaiting the standardization of the Semantic Web Rule Language ([www.w3.org/Submission/2005/01/](http://www.w3.org/Submission/2005/01/)) by the W3C. Similarly, conditional expressions are required in the service model in several of the process flow specification primitives used to define process models, e.g. if-then and while-do control flow constructs.

### 2.2 WSMO

WSMO builds on a previous non-semantic web service framework and is more focused on service discovery and service interoperability [wsmo]. It therefore explicitly includes the modeling of the goals of a service user, against which service offerings are matched. WSMO also includes a range of mediation types that can be used in binding semantic expressions between services, goals, ontologies and groundings. At its most basic WSMO describes services in terms of pre-conditions and post-conditions that apply to information that passes in and out of the service. It separately defines assumptions and effects, which express pre-invocation and post-invocation conditions that must apply to the environment, or world model, in which the service exists. These expressions are described using ontologies. However, WSMO does not subscribe the W3C stack approach to defining semantic languages, so although its ontology language can be mapped to OWL, it provides direct support for the expression of axioms and rules.

### 2.3 OWL-S vs. WSMO

In modeling complex, composite services, WSMO has focused on supporting choreography, i.e. the externally visible behavior of a service, rather than, as in OWL-S, upon orchestration, where the emphasis is on modeling the internal breakdown of a service into sub processes. The relative merits of OWL-S and WSMO are currently a topic of intense debate in the W3C Semantic Web Services working group and elsewhere. However, there has been little consideration as to how OWL-S and WSMO may be used to integrate semantic service models with semantic models of the resources that underpin them. Though OWL-S attempts to model resources, it is primarily with the aim to managing the sharing of resources between service invocations. It is currently the least developed part of the OWL-S specification, with little guidance and few examples on its use. WSMO does not aim to explicitly model the underlying resource of a service, but it does use the concept of an abstract state machine to model service choreographies. It thus supports state-oriented semantics which we will examine in more detail below for its potential for modeling physical resources.

### 3. Modeling Operational State and Adaptive Behavior

There is an increasing emphasis in pervasive computing [mccann], as there is in networking [smirnov] and distributed systems [kephart] in general, for systems to exhibit self-managing, or autonomic behavior. A prerequisite for any managed system is that it offers well defined management state to a managing application. Network and system management have a strong track-record of explicitly modeling management information as the basis for manager-agent management protocols, e.g. SNMP, CMIP. However, the move to self-managing systems implies that management decision making is delegated away from the human administrators using manager applications and toward the components being managed. The most common approach to delegating such management decision making is through the use of policy-based management, where a declarative rule that embodies the management decision is executed as

close to the managed resource as possible [strassner].

Thus we have previously proposed that a suitable component model for autonomic systems should combine semantic web services with existing management information semantics and policy rules defining adaptive behavior [lewis]. Figure 1 below depicts a reference model showing the relationship between the aspects of an adaptive service element.

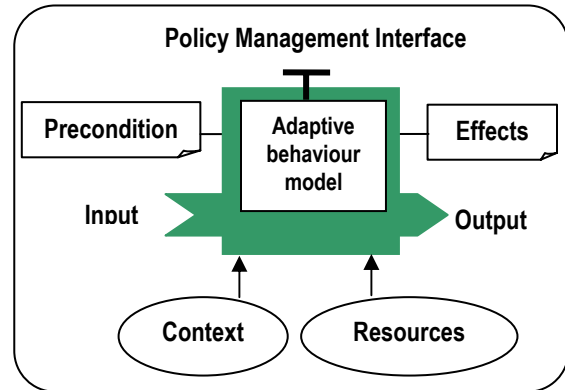


Figure 1: Reference model for an adaptive service element

These policies modify the behavior of the service offered by components, based on the state of the component's resources and the external context. An approach where semantics are shared between definitions of the service, the resources, the context and the policies could offer advantages of increased cohesiveness and reduced cognitive load when engineering autonomic systems from such components. To this end, in the remainder of this section we examine the current state of the art in using ontology based semantics for defining managed resources and policy rules. We then go on in the subsequent section to examine how semantics definitions of service, resources and policies could potentially be modeled in an integrated fashion using the abstract state machine expression of WSMO.

#### 3.1 Management Information Semantics

Attempts have been made to define new management information modeling and service modeling languages to act as a lingua franca between heterogeneous management models. Notable amongst these are the Distributed

Management Task Force's (DMTF) Common Information Model (CIM) schema [cim] and the TeleManagement Forum's NGOSS technology neutral architecture [tmf053]. However, a lack of a strong semantic interoperability mechanism and reliance on conformance to poorly subscribed industrial agreements effectively render these as yet further management knowledge formats with which other schemes needed to interoperate. Recent work [lopezdevergara] has shown directly how the modeling management information models in the OWL ontological format can be used to ease the interoperation between models originally conceived in different management information languages, i.e., GDMO, SMI, CIM.

### 3.2 Semantics for Policies

Policies, in their simplest form, are event-condition-action rules. They are regarded as being performed on behalf of subjects, e.g. a person or agent wishing to operate on a resource, and acting on a target, e.g. the resource upon which an operation is sought. Policies have been employed in system access control, defining authorization rules about whether a particular subject is permitted or denied access to a particular target resource [sloman]. General purpose policy languages address both authorization policies and obligation policies, the latter being rules about what and when a particular subject is required to do or not do to a particular target [damianou]. Policy languages assist administrators with the task of managing large policy rule sets through abstractions such as roles, used for grouping users, and domains, used for grouping subjects, targets and sub-domains. The engineering of policies is assisted through the reuse of policy specification elements, thus also encouraging consistent understanding of similar policy rules. Though some of these features have been incorporated into a policy model agreed by the Internet Engineering Taskforce and the Distributed Management Taskforce [moore] and the TeleManagement Forum's DENng [strassner], the definition of policy languages is still an active area of research. Recent research has begun to exploit ontologies for more extensible expression of subjects and targets as well as exploiting existing ontology-based inference engines to ease

policy engineering problems such as policy authoring, de-confliction and distributed enforcement [tonti][kagal]. In our scenario, such an ontology-based approach to specifying policies must be combined with semantic service models of the managed system.

## 4. Case Study

We now present a short case study to show how WSMO could be used to capture operational resource semantics and its behavior in relation to services offered by a device. This case study examines how a service component model for a network printer could be modeled in WSMO to integrate a semantic description of the services offered, of the resources used and of the policy-driven management behavior of the component. To ensure that the case study is representative of real world semantics, models extracted from existing industry standards were used to model the service component. Three aspects make up this model, firstly a model of the service offered by the printer, secondly a model of the state information available within the service component and lastly the model of policies that are applied to the service component and the effect this has on its operation. This follows the structure in figure 1, but in this case no contextual information is considered. The following WSMO concept definitions capture the semantics of a Printer device, the PrintQueues that such a device can support and the PrinterJobs that populate such a queue. As indicated by the WSMO namespace declaration, these semantics are taken directly from the DMTF's CIM model for a printer, capturing the managed object class as concepts and their attributes and relations as concepts attributes, with corresponding maximum and minimum cardinalities in parentheses.

```
namespace
  _"http://www.dmtf.org/CIM_Device29/Printers#" {
}

concept PrinterJob
  JobId ofType (1 1) _string
  JobSize ofType (1 1) _integer
  MimeTypes ofType MimeType
  RequiredPaperType ofType (1 1) PaperType
  Copies ofType (1 1) _integer
  DefaultNumberUp ofType (0 1) NumberUp
  HorizontalResolution ofType (0 1) _integer
  NumberUp ofType (0 1) NumberUp
  PrintJobstatus ofType (1 1) _string
```

```

TimeComplete ofType (0 1) time
RequiredJobSheets ofType (1 1) _integer
OwningPrinterQueue ofType (1 1) _PrintQueue
PrinterServiceJob ofType (1 1) Printer

concept PrintQueue
  NumberOnQueue ofType (1 1) _integer
  MaxJobSize ofType (1 1) _integer
  OwnPrintJobs ofType PrinterJob
  PrinterServicingQueue ofType Printer

concept Printer subConceptOf LogicalDevice
  PrinterStatus ofType (1 1) _string
  DetectedErrorState ofType (0 1) _string
  ErrorInformation ofType (0 1) _string
  PaperSizesSupported ofType PaperSize
  PaperSizesAvailable ofType PaperSize
  DefaultPaperType ofType (0 1) PaperType
  CurrentPaperType ofType (0 1) PaperType
  MimeTypesSupported ofType MimeType
  CurrentMimeType ofType (0 1) MimeType
  DefaultMimeType ofType (0 1) MimeType
  JobCountSinceLastReset ofType (1 1)
    _integer
  TimeOfLastReset ofType (0 1) time
  MaxCopies ofType (0 1) _integer
  DefaultCopies ofType (0 1) _integer
  MaxNumberUp ofType (0 1) NumberUp
  DefaultNumberUp ofType (0 1) NumberUp
  HorizontalResolution ofType (0 1)
    _integer
  VerticalResolution ofType (0 1) _integer
  MaxSizeSupported ofType (0 1) _integer
  AvailableJobSheets ofType (0 1) _integer
  Capabilities ofType Capability
  DefaultCapabilities ofType (0 1)
    Capability
  CurrentCapability ofType (1 1) Capability

```

For the service definition this example is based upon the createPrintJob service defined by the UPnP forum [upnp]. This forum defines simple service interfaces to devices that enable them to be accessed as part of a pervasive computing environment. Typically, the parameters of a request to this service (uPnPPr#CreatPrintJobReq) would be defined in the same name space as the service and a WSMO mediator would be defined to map term to terms from the CIM printer ontology, but in this case we have, for simplicity, performed the mapping directly by using the latter's terms directly in the request.

The WSMO example below just includes the ontological definition of the CreatePrintJobReq, but omits the WebService definition and the grounding. For more detail on these aspects of web service definite using WSMO the reader is referred to [wsmocho]. Instead the example below focuses solely on how the same expression used for the

choreography can be used to link the incoming request to the configuration of the state represented by the CIM model, in particular the cimPr#PrinterJob concept and its attributes. For brevity, we assume single instance of Printer and PrinterQueue, thisPrinter and thisQueue respectively exist. The structure of the CreatePrintJobReq message is taken from the UPnP message CreatePrint action in [albright]. Note that though UPnP standards have their own definitions of device state semantics, we are marrying this service definition with the CIM printer device semantics to demonstrate the integration possible by working at a semantic level.

Under the stateSignature, the in clause specifies concepts whose instances can be written by the environment and read by the choreography, and the out clause indicates concepts whose instances can be written by the choreography and only read by the environment. This clause allows the device design to restrict the visibility of internal state and the changes that can be made by a choreography subsequently added as a policy by an administrator.

The adaptive service behaviour of this device is expressed by the transitionRules clause in the choreography element of this WSMO example. This is in effect a configuration policy, taking parameters from the request (expressed in UPnP semantics) and applying it resources (expressed using CIM semantics). This could be considered as default behaviour for this device. However the if-then statement following restricts the number of copies in the print job to the maximum number specified by the MaxCopies attribute of the thisPrinter instance. This thereby gives an example of a device specific operational policy. A more discretionary policy then follows that forces all jobs over a certain size to use 2up printing.

```

namespace { _"http://example.org/ont#",
  dc _"http://purl.org/dc/elements/1.1#",
  cimPr "http://www.dmtf.org/CIM_Device29/
    Printers",
  uPnPPr "http://www.upnp.org/services/wsmo/
    PrintBasicv1_01"
}

```

```
ontology uPnPPr#CreatePrintJob
```

```

concept uPnPPr#CreatPrintJobReq
  uPnPPr#JobName ofType (0 1) _string
  uPnPPr#JobOriginatingUserName ofType
    (1 1) string
  uPnPPr#DocumentFormat ofType
    (1 1) cimPr#MimeType
  uPnPPr#Copies ofType (0 1) integer
  uPnPPr#Sides ofType (0 1) cimPr#SideTypes
  uPnPPr#NumberUp ofType
    (0 1) cimPr#NumberUp
  uPnPPr#OrientationRequested ofType (0 1)
    uPnPPr#PaperOrientation
  uPnPPr#MediaSize ofType
    (0 1) cimPr#PaperSize
  uPnPPr#MediaType ofType
    (0 1) cimPr#PaperType
  uPnPPr#PrintQuality ofType
    (0 1) cimPr#Capability

interface CreatePrinterJobIntf
choreography CreatePrinterJobChor
  importsOntology {
    "http://www.upnp.org/services/wsmo/
    PrintBasicv1_01",
    "http://www.dmtf.org/ont/CIM_Device29/
    Printers"}
  stateSignature CreatePrinterJobSig
    in uPnPPr#CreatPrintJobReq
    out cimPr#PrinterJob

  transitionRules PrinterPolicies
    forall (?request) with
      (?request memberOf
        uPnPPr#CreatPrintJobReq
        and ?request[uPnPPr#DocumentFormat
          hasValue ?format]
        and ?request[uPnPPr#Copies
          hasValue ?copies]
        and ?request[uPnPPr#NumberUp
          hasValue ?numup]
        and ?request[uPnPPr#MediaSize
          hasValue ?size]
        and ?request[uPnPPr#MediaType
          hasValue ?type]
        and ?request[uPnPPr#PrintQuality
          hasValue ?qual]
        and thisQueue[
          cimPr#PrinterServicingQueue
          hasValue thisPrinter]
        and thisPrinter[cimPr#MaxCopies
          hasValue ?maxcopies]
      ) do
      add(?job [
        cimPr#MimeType hasValue ?format,
        cimPr#RequiredPaperType
          hasValue ?type,
        cimPr#Copies hasValue ?copies,
        cimPr#NumberUp hasValue ?numup,
        cimPr#PrinterServicingJob
          hasValue thisPrinter,
        cimPr#OwningPrinterQueue
          hasValue thisQueue
      ] memberOf cimPr#PrinterJob)
    | if ?copies > ?maxcopies then
      update(?job[cimPr#Copies
        hasValue ?maxcopies])
    | if ?copies > 100 then
      update(?job[cimPr#NumberUp
        hasValue 2]).

```

## 4. Conclusions and Further Work

In this paper we explore how WSMO can be used to support the representation of operational state typical of managed devices supporting a management agent. We also explore how policy rules can be used to define adaptive service behavior in concert with the semantic expression of operational state. We show that the existing choreography element of WSMO provides sufficient expressiveness for such policy-based behavior, at least in simple cases.

In our further work we will examine additional case studies where a policy enabled semantic service can be applied in a network or pervasive computing management setting. For the former, we will build on the suggestion in [duke] and examine the TeleManagement Forum's NGOSS specification to assess the benefits a semantic approach based on WSMO can provide to its technology neutral architecture. We will also examine how readily the execution environment for WSMO can support policy execution and also whether specialized development tools are needed when authoring policies rather than choreographies. Finally, we will extend the use of choreography to the integration of context sensitivity into the operation of a service and its resource management policies.

## Acknowledgements

This work was partially supported by Science Foundation Ireland under the CTVR project and by the Irish Higher Education Authority under the M-Zones programme. We would like to thank Tomas Vitvar and Paavo Kotinurmi of DERI and Kris McGlenn of TCD for their assistance in using WSMO.

## References

- [akkiraju] Akkiraju, R, et al, "Web Service Semantics - WSDL-S", W3C Member Submission 7 November 2005, Version 1.0
- [albright] Albright, S. et al "PrintBasic1: Service Definition Template v1.01", for UPnP version 1.0, approved standards, April 2002

- [berners-lee] Berners-Lee, T., Hendler, K., Lassila, O. (2001), 'The Semantic Web', Scientific American, pp 35-43, Issue 284 (3), 17th May 2001
- [bpel4ws] Business Process Execution Language for Web Services, version 1.1, May 2003
- [cim] Common Information Model v2.5, DMTF 2000: [http://www.dmtf.org/spec/cim\\_schema\\_v25.html](http://www.dmtf.org/spec/cim_schema_v25.html)
- [damianou] Damianou, N., Dulay, N., Lupu, E., Sloman, M., (2001) "The Ponder Policy Specification Language", Proc. Policy 2001: Workshop on Policies for Distributed Systems and Networks, Bristol, UK, 29-31 Jan. 2001, Springer-Verlag LNCS 1995, pp. 17-28
- [duke] Duke, A., Davies, J., Richardson, M., Kings, N., "A Semantic Service Oriented Architecture for the Telecommunications Industry", in proc of IFIP Int'l Conf on Intelligence in communications Systems, Bangkok, Thailand, Nov 2004 (INTELLICOM 2004), Springer LNCS 3283, pp 236-245
- [foster] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid ServiceInfrastructure WG, Global Grid Forum, June 2002.
- [lewis] Semantic Interoperability for an Autonomic Knowledge Delivery Service, D. Lewis, D. O'Sullivan, R. Power, J. Keeney, in Proc of 2nd IFIP WG6.6 International Workshop on Autonomic Communication - Autonomic Communication Principles, Oct. 3-5, 2005, Vouliagmeni, Athens, Greece
- [lopezdevergara] López de Vergara, J.E., Villagrà, V.A., Berrocal, J., "Applying the Web Ontology Language to management information definitions", IEEE Communications Magazine, Vol. 42, Issue 7, July 2004, pp. 68-74. ISSN 0163-6804
- [mcilraith] McIlraith, S.A., Son, T.C., Honglei Zeng, H. (2001), 'Semantic Web Services', IEEE Intelligent Systems, 16(2), March/April 2001
- [martin] David Martin, Mark Burstein, Jerry Hobbs, Ora Lassila, Drew McDermott, Sheila McIlraith, Srin Narayanan, Massimo Paolucci, Bijan Parsia, Terry Payne, Evren Sirin, Naveen Srinivasan, Katia Sycara, "OWL-S: Semantic Markup for Web Services", W3C Member Submission 22 November 2004
- [masuoka] Masuoka, R., Labrou, Y., Parsia, B., Sirin, E., "Ontology-Enables Pervasive Computing Applications", IEEE Intelligent Systems, Sept/Oct 2003, pp 68-72
- [mccann] J. A. McCann and M. C. Huebscher. "Evaluation issues in autonomic computing". In Proceedings of Grid and Cooperative Computing Workshops (GCC), LNCS 3252, 597-608. Wuhan, China. October 21-24, 2004.
- [moore] Moore, B., Ellesson, E., Strassner, J., Westerinen, A., (2001), "Policy Core Information Model -- Version 1 Specification", IETF RFC 3060
- [kagal] Kagal, L., Finin, T., Joshi, A., "A Policy Language for A Pervasive Computing Environment", IEEE 4th International Workshop on Policies for Distributed Systems and Networks, June 04, 2003
- [kephart] Kephart, J., Chess, D., "The Vision of Autonomic Computing", IEEE Computer, Jan 2003, pp 41-50
- [owl] OWL Web Ontology Language Reference, available at: <http://www.w3.org/TR/owl-ref/>.
- [sloman] Sloman, M., Lupu, E., "Security and Management Policy Specification", IEEE Network, vol.16 No. 2, March/April 2002. pp 10-19.
- [smirnov] M. Smirnov, "Autonomic Communication: Research Agenda for a New Communications Paradigm", Fraunhofer FOKUS, November 2004. ([http://www.fokus.gmd.de/web-dokumente/Flyer\\_engl/Autonomic-Communication.pdf](http://www.fokus.gmd.de/web-dokumente/Flyer_engl/Autonomic-Communication.pdf))
- [strassner] Strassner, J., (2004) "Policy-based Network Management - Solutions for the Next Generation", Elsevier
- [tmf053] NGOSS Architecture, Technology Neutral Specification, Membership Evaluation Version 1.51, TeleManagement Forum, July 2001
- [tonti] Tonti, G., Bradshaw, J.M., Jeffers, R., Montanari, R., Suri, N., Uszok, A., "Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder" Proceedings of 2nd International Semantic Web Conference (ISWC2003), October 20-23, 2003, Sanibel Island, Florida, USA
- [wsmocho] D14v0.2 Ontology-based Choreography and Orchestration of WSMO, WSMO Final Draft, Feb 2006
- [wsmo] D3.1v0.1 WSMO Primer, WSMO Final Draft April 2006