Empowering Associations in CIM for Active Data Modeling

Contact Author : **Arun Kumar**

IBM India Research Lab
Block-1, Indian Institute of Technology,
Hauz Khas, New Delhi - 110016. INDIA.
Phone: +91-11-26861100
Fax: +91-11-26861555
Email: kkarun@in.ibm.com

Number of Illustrations : 5

Empowering Associations in CIM for Active Data Modeling

Arun Kumar, Neeran Karnik
IBM India Research Lab.,
Block-1, Indian Institute of Technology,
Hauz Khas, New Delhi-110016, INDIA.
{kkarun, kneeran}@in.ibm.com

Abstract

This paper presents a mechanism to enhance the data modeling capabilities of DMTF's Common Information Model (CIM) by introducing *active associations*. This novel active data modeling concept enables systems management middleware to incorporate certain capabilities that are currently perforce implemented outside of it, by management applications. Active associations are associations empowered with *rules* that limit the scope of instances with which a particular instance can be associated. These rules are defined in the class model and are evaluated at runtime for instances. From a management application's point of view, the enhanced association construct enables valuable functionality that is not currently available. We motivate and demonstrate the utility of active associations through a scenario that involves the management of composite web services.
Keywords: CIM, active data modeling, composite web services, management systems, schema, web services

1 Introduction

Various management standards have evolved over last few years owing to the increased complexity of enterprise systems. Some examples include Simple Network Management Protocol (SNMP), Java Management Extensions (JMX), Common Management Information Protocol (CMIP) [1] etc. Recently, Common Information Model (CIM) [2] from Distributed Management Task Force[1] (DMTF) has emerged as a popular standard for generic management in network/enterprise environment. CIM uses object-oriented modeling for managing systems, software, users, networks etc. Web Based Enterprise Management[2](WBEM) technologies, built around CIM provide a powerful infrastructure for building generic management systems. Management applications can query and utilize information modeled through CIM to achieve various goals such as runtime application management [3], problem determination in enterprise environments [4] etc.

In the software services arena, web services [5] have emerged as the dominant component technology for distributed applications. Using standardized interfaces, clients can remotely access functionality or resources offered by service providers. The services offered could vary

[1] http://www.dmtf.org
[2] http://www.dmtf.org/standards/wbem

from low-end hardware rentals such as renting compute power, storage, bandwidth etc. to high-end complex business processes modeled as web services. Drawing upon economies of scale, service providers are able to offer better, cheaper and more reliable service to their customers while keeping them in step with rapid advancements in technology.

Rapid development of web services standards are increasing the gap between management requirements of web services and the capabilities of existing standards/solutions. The requirements include management components for Service Level Agreements (SLA), service monitoring and provisioning, platform monitoring, life cycle and configuration management, performance management, metering etc. The distributed nature of web services and their operation in autonomous environments only adds on to the expectations from a management system. Enabling web service management support for service providers is non-trivial and requires foresight about both how to decompose an automation problem and how to deliver it [6]. Several efforts are in progress to design infrastructures for management of web services some of which include Web Services Distributed Management [7], Web Services Management Framework [8] etc. Recently, CIM and WBEM based solutions have been used for management of different aspects of the web services environment [9, 10, 11].

In this paper, we exploit the dynamic modeling capabilities of CIM and enhance it by introducing a novel concept called *active association* to be incorporated into CIM. We then demonstrate how this powerful concept can be used to perform complicated management of a business process that spans across multiple web services. The rest of the paper is organized as follows. Section 2 provides an overview of CIM and introduces some aspects of web service management in CIM. Section 3 describes the concept of active associations and shows how it can be used for web service management. Section 4 discusses implementation issues for active associations. Finally, section 5 summarizes our work and presents opportunities for further research.

2 CIM for Web Services Management

2.1 CIM Overview

CIM is an object-oriented information model providing a conceptual view of a managed IT environment. It attempts to unify and complement the existing instrumentation and management standards (DMI[3], CMIP, SNMP [1] etc.) using object-oriented constructs and design. Being only an information model, it is independent of instrumentation and the repository format.

The CIM standard consists of a specification and a schema. The CIM Specification defines the details for integration with other management models, while the CIM Schema [12] provides the actual model descriptions. It consists of a large number of classes that capture the characteristics of various system entities. It is divided into three conceptual layers [13]:

- The *Core* model is an information model that captures notions that are applicable to all areas of management.

- The *Common* model is an information model that captures notions that are common to particular management areas, but independent of a particular technology or implementation. The common areas are systems, applications, databases, networks and devices. The information model is specific enough to provide a basis for the development of management applications. This model provides a set of base classes for extension into the area

[3] http://www.dmtf.org/standards/dmi/

of technology-specific schemas. The Core and Common models together are expressed as the CIM schema.

- The *Extension* models capture technology-specific aspects of management information. These are specific to environments such as operating systems.

A CIM/WBEM implementation consists of a CIM Object Manager (CIMOM) – middleware that hosts all classes and instances. Each managed element is represented by an instance in CIMOM and has associated with it, a *CIM Provider* that interfaces with the actual managed element. Whenever an application issues a query related to an instance, the CIMOM looks up its associated provider and forwards the query to it. The provider then responds to the query based upon the state of the managed element. The application may also update the state of the managed element in the same way.

2.2 Representing Web Services in CIM

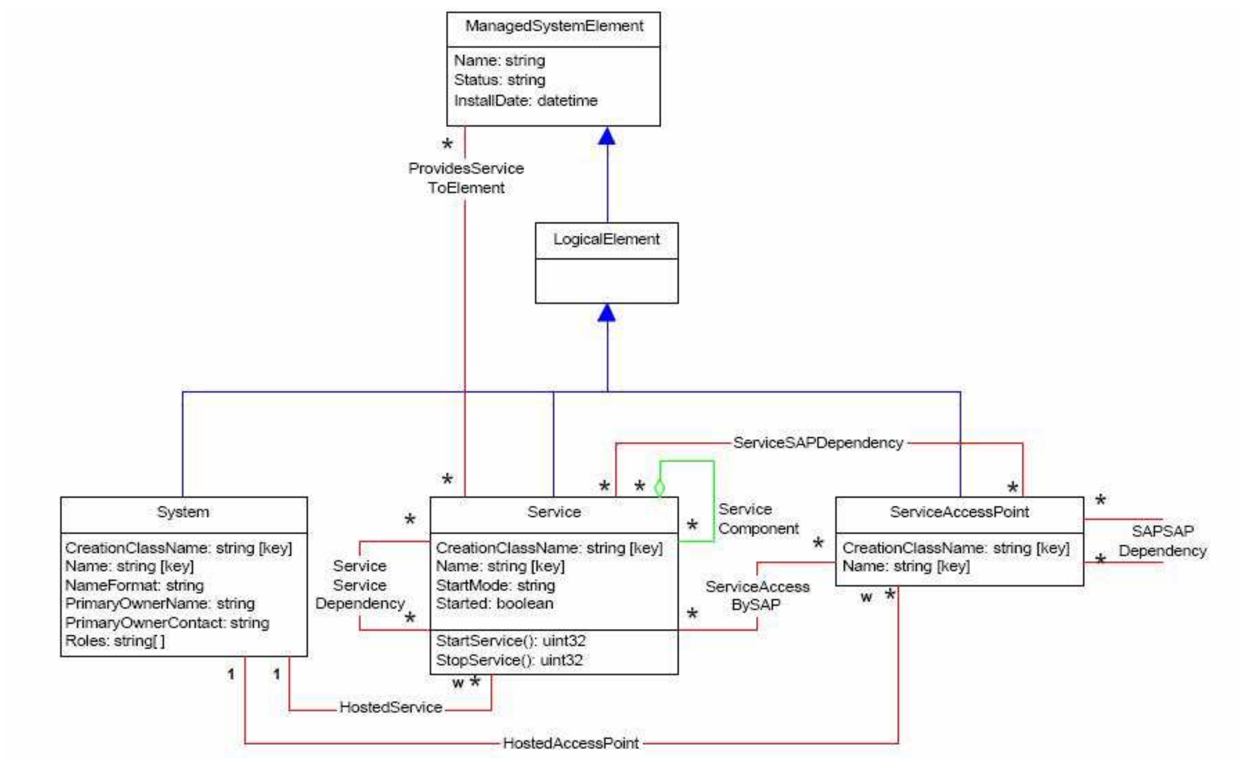


Figure 1: Service/Service Access Point Classes in CIM Core Model

The CIM core model [14] includes a data model for representing management interfaces of a service (refer Fig. 1). It defines a class *Service* along with its *ServiceAccessPoint* (which could be a *ProtocolEndPoint*, a *ServiceAccessURI* or a *RemoteServiceAccessPoint*). The *Service* class also supports methods for starting and stopping the service. Apart from these there are useful associations such as *ServiceComponent* that relates an aggregate service to its component services, *ServiceServiceDependency* that captures functional and existential dependencies between different services and so on. Web services could be modeled as a subclass of *Service* and the related classes mentioned above could be used or extended to represent the portTypes and protocol bindings of a web service.

Some aspects of web service management have been modeled in CIM recently without explicitly modeling web services themselves. The WSLA system [10] uses CIM to represent Web Service Level Agreement definitions and their values. These are then used to implement a system that provides SLA-driven management. Similarly, metering and accounting aspects of services have been modeled in [9] and the model is used to implement an accounting infrastructure for web services. The CIM metrics model has been applied for measuring and tracing application response times for distributed transactions [11]. A similar system could also be used to provide response time measurement and tracing for web services.

2.3 Composite Web Services

We now use the example of *business process management* to motivate the need for *active* associations in CIM. Often business processes, modeled as workflows using web services, result in implementations that span across multiple web services. Such business processes are rendered as *composite* web services – services that invoke other, simpler web services as part of their business logic. The *component* services may reside in different administrative domains and may be geographically distributed. This gives rise to several management issues. For example, business rules governing a business process hosted by service provider X may dictate that it cannot use components offered by its competitor Y. IT-level policies may specify constraints on the quality of service and cost associated with component services. The business process may need to be dynamically reorganized (by selecting different components, or by reordering the message flow between them) if these constraints are violated, or if a component fails, or if 'better' component services become available at runtime.

Note that a CIM model only captures class-level relationships using the association construct. It has no first-class construct to model instance-level constraints or assertions such as the examples above. The *ServiceComponent* association merely captures the composite-component relationship between two services; we cannot impose cost/QoS constraints on its membership. In the absence of this support from CIM (and correspondingly, CIM middleware), individual management applications need to implement such functionality themselves. By adding the notion of *active associations* (described in the next section), we can however push such generic management functionality into the middleware itself, thus simplifying the task of building management applications on top of CIM.

3 Active Associations

3.1 Active data modeling in databases

Database researchers adopted the rules paradigm from the field of Artificial Intelligence and, over the years, modified it to allow automatic response to database operations, occurrences of database states, and transition between states [15]. A rule in a database rule language takes the form:

```
ON event  ==>  IF condition ==>  THEN action
```

This is termed as the Event-Condition-Action (E-C-A) model [16]. It specifies that on the occurrence of an *event* E, a *condition* C is evaluated. If the result is true then the specified *action* A is executed. Here an event represents a point in time either specified explicitly or determined by some occurrence in the database [17]. Typically, events are triggered by data modification operations such as inserts, updates or deletes. Some systems allow events to be triggered at specific points in time or even at the time of data retrieval [15]. Further, composite events are also supported [17, 18]. Conditions are predicates or queries defined over the data in the database. A condition when evaluated verifies some property of the state of the database at the time of occurrence of the event. If the property is true when the event is triggered then the associated action is executed. Actions on the other hand could be arbitrary sequences of retrieval and modification commands over data in the database. The E-C-A model provides powerful capability to the databases for enforcing integrity constraints, derived data maintenance and version control etc.

CIM is an object oriented data model, and concepts from object oriented databases apply here. The events in an active OODBMS are related to actions that happen to objects and the state of the objects. Basic events applicable for OODBMSes are categorized as follows [18]:

1. Object State Events: triggered on creation/deletion of an object or when the object is read/updated through its public interface.
2. Method Execution Events : triggered on specific method invocations on an object.
3. Timing Events: triggered at specific points in time.
4. Transaction Events: triggered at different points in the life-cycle of a transaction e.g, at begin, at commit etc.

In a WBEM/CIM based system, the CIM Event Model [19] provides basic support for active data modeling. It defines CIM Indication hierarchy and the use of Indications to model events. While an event is the occurrence of a phenomenon of interest, an Indication is a record of the detection of an event. CIM Event Model enables the CIM clients to subscribe to event notifications of interest to them. CIM's support for active data modeling stops here. Beyond this, it is up to the clients to make use of these indications to implement the Condition-Action (C-A) portion of the E-C-A model. In other words, support for modeling of the Condition-Action portion is currently missing, and by introducing *active* associations we aim to start filling this gap in CIM.

3.2 Associations with Rules

An association in CIM represents a relationship between two or more objects. As shown in Figure 2, it is modeled as a class that contains two or more references. This is unlike traditional OODBMSes, where associations are handled as inverses with references attached to each class. Therefore in CIM, by defining an association it is possible to establish a relationship between two classes without affecting those classes themselves.

We enhance the association construct in CIM by allowing one or more *rules* (based upon E-C-A model) to be specified in an association class. The event specification in the rules would follow the CIM Event Model. Each specified rule would have one or more associated CIM Indications, the arrival of which would trigger the rule. The conditions could be arbitrary queries on the state of various objects in the CIMOM. Typically however, a condition in an association would refer to the properties of instances of classes participating in that association itself. The *action* could involve creation or deletion of an instance of that association, or modification of an

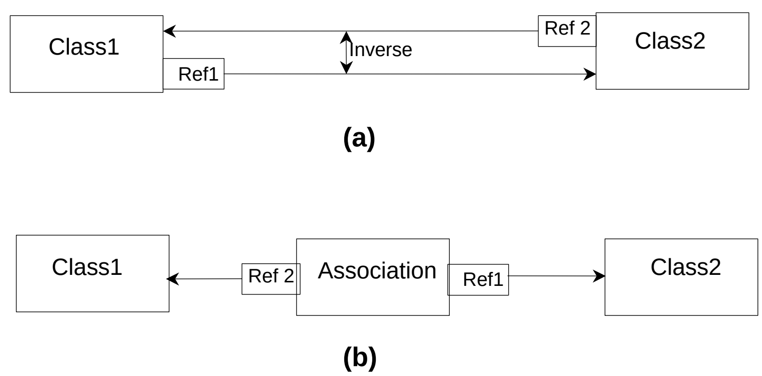


Figure 2: (a) Association using Inverses (b) Association using References

existing instance of that association, among other possibilities. Rule languages from existing systems such as [20] could be leveraged to define the rule language for CIM.

Therefore, rules in an association could be used to specify the following:

1. when to create or sever a relationship between instances of two (or more) classes.

2. which instances of those classes qualify to be a part of the relationship.

Each of these capabilities is of immense importance in various management applications. We demonstrate this through the business process management scenario introduced earlier.

Consider the scenario in which a business process modeled as a composite web service of type $S1$, functionally depends upon services of type $S2$ and $S3$. A service provider of $S1$ wishes to promise certain quality of service (QoS) guarantees to its customers through formal contracts. These guarantees translate into QoS requirements from component services of type $S2$ and $S3$. Now, given that multiple instances of these component services may exist at a point in time, the service provider offering $S1$ makes a choice to select one instance each of $S2$ and $S3$.

To represent these services in a WBEM/CIM based solution we subclass the CIM Service class as shown in Figure 3. Initially, the class model describing the services $S1$, $S2$, $S3$ and the relationships between them is loaded into CIMOM. Over time various instances of these classes get instantiated to reflect creation of new services in the actual world. When a service provider wishes to offer an instance of $S1$, a management application developed by her would have to search the CIM repository for instances of component services that match its criteria. Assuming $I2$ – an instance of $S2$ – and $I3$ – an instance of $S3$ – get selected, the application would then create the new composite service and represent it in CIMOM as instance $I1$ of class $S1$. Next, it would create an instance $A1$ of the CIM ServiceComponent aggregation that would relate $I1$ to $I2$. Similarly, an instance $A2$ of the CIM ServiceComponent aggregation would be created that would relate $I1$ to $I3$. At this point, this information in CIMOM reflects the current state of the composite service $I1$. A management client could query the CIMOM for this information or even initiate some change to this state if the services support it. The *CIM Providers* of these

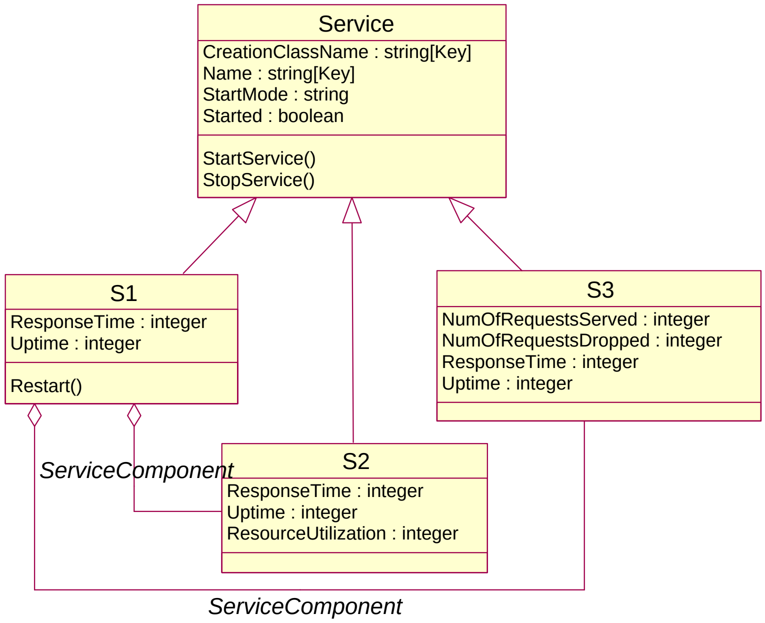


Figure 3: Business Process Management Scenario

instances are responsible for handling such queries, for which they communicate with the actual services.

Now, a problem that the service provider of $I1$ faces is that of ongoing monitoring of the QoS levels of component services $I2$ and $I3$. A drop in QoS of either $I2$ or $I3$ below promised levels may cause $I1$ to violate its contracts with its customers, and its service provider would be liable to pay penalties. In this situation, the service provider of $I1$ has to switch over to other instances of $S2$ and/or $S3$. The only solution possible currently is to develop a custom management application that implements this logic.

Using active associations, however, some decision making can be incorporated into the middleware itself. This can be achieved by allowing the CIM ServiceComponent aggregation instances $A2$ and $A3$ to subscribe to events reporting the state of component instances. Once the event notifying a change in some QoS parameter of a component service arrives, it could be used to determine drop in QoS levels, if any. If a drop beyond specified threshold limits is detected then an action may be initiated to replace the existing instances of component services with others. This ongoing monitoring functionality can be captured as one or more rules of aggregation associations $A2$ and $A3$. These rules can be populated at the time of instantiation of aggregations $A2$ and $A3$. CIM Event model would be used to define one or more CIM Indications that notify change in the state of services $I2$, $I3$. Therefore, the event part of the rule in $A2$ would specify subscription to the CIM Indications that notify a change in the state of $I2$. Similarly, the event part of the rule in $A3$ would specify subscription to the CIM Indications that notify a change in the state of $I3$. The condition part of the rules would be a boolean expression involving comparison of threshold values of QoS parameters with the values reported in the CIM Indication. The corresponding action could be one that modifies this association to replace the defaulting component service instance with another one. This decision could then be reflected into the environment through the CIM providers of the services involved.

4 Implementation Issues

In this section, we discuss the support required from the basic CIM infrastructure in order to provide active associations.

1. **Rule Specification**

   There are different alternatives for specifying rules in associations. One of them needs to be adopted for CIM.

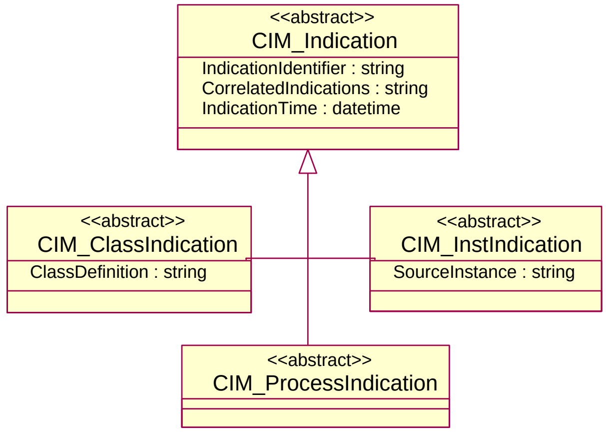
   - **A Rule Language**: An E-C-A model based rule language may be defined that can be used for specifying rules in CIM. We can adopt the rule language constructs from traditional OODBMSes directly but some CIM specific concepts may need to be added to make them more useful. For instance, a rule in an association instance should be able to refer to attributes of participating instances. To support active associations, this rule language would be used only in associations. However, utility of such rules outside of associations could also be exploited.
   - **Methods as Rules**: Since associations in CIM are modeled as classes, they too have properties and methods. Therefore, the C-A portion of the rule could be implemented as methods of that association. Each such method may be associated with one or more CIM Indications to specify the event on which it should be triggered. Also those methods maybe parameterized to supply information such as values of QoS parameters reported in the CIM Indication.

2. **CIM Event Model**

   As mentioned earlier in this paper, CIM Event Model [19] provides basic support for active modeling in CIM. It defines CIM Indication hierarchy and the use of Indications to model events. However, the CIM Event model as it stands currently is limited in its applicability. To illustrate this point we briefly describe the CIM Event model. It defines three different types of indications:

   - CIM_InstIndication is used to report instance life cycle events such as creation, deletion, method invocation etc. These are equivalent to Object State Events and Method Execution Events in traditional OODBMSes.
   - CIM_ClassIndication is used to report class life cycle events such as class creation, deletion, modification. These are provided in CIM since it allows creation of new classes and deletion/modification of existing ones at runtime.
   - CIM_ProcessIndication is used to report alert and notification type of events associated with objects that may or may not be completely modeled in CIM or do not correspond to a simple life cycle event. These include events like low-level instrumentation alerts, DMI alerts, SNMP traps, and time-based events etc.

   Subscription to an event is expressed by the creation of an IndicationSubscription association instance (see Fig. 4) that references an IndicationFilter instance, and an IndicationHandler instance. A Filter contains the query that selects an Indication class or classes. A Handler instance specifies how and where to send an indication. On the occurrence of



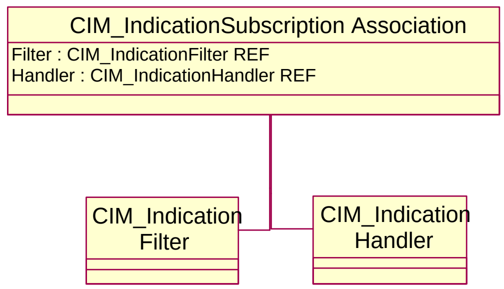


Figure 4: CIM Indication Schema

an event in a managed element, an appropriate Indication instance is created which gets delivered to its subscribers (refer Fig. 5).

By design, the CIM Event model can be used to deliver events to CIM clients that comply with DMTF's WBEM CIM-XML standard. Internal CIM Providers cannot currently benefit from it. For an active association to be able to subscribe to CIM Indications, it needs the ability to to specify the Filter directly in its rule. In such a case, the CIM IndicationSubscription association would specify the active association's CIM Provider as the destination where the Indication needs to be delivered. Therefore, either the current CIM Event model should be modified or an alternative mechanism to deliver Indications to internal CIM Providers may also be considered.

3. **Active Providers**

Currently, CIM Providers are passive entities that provide information when demanded by an application. However, various applications require active monitoring and reporting of information. For instance in [10], the authors use CIM to model a Service Level Agreement which is used by external management services to detect violations. They implemented an active provider to collect monitored metrics periodically and to carry out computation for detecting violations. This enabled them to avoid the inefficient method of polling. In general, active providers are needed as a basic facility in CIM especially to provide support for any kind of active data modeling. Such CIM Providers would be able to actively monitor the managed elements and update the corresponding state represented in CIM. CIM Event model could then utilize this functionality to trigger off generation

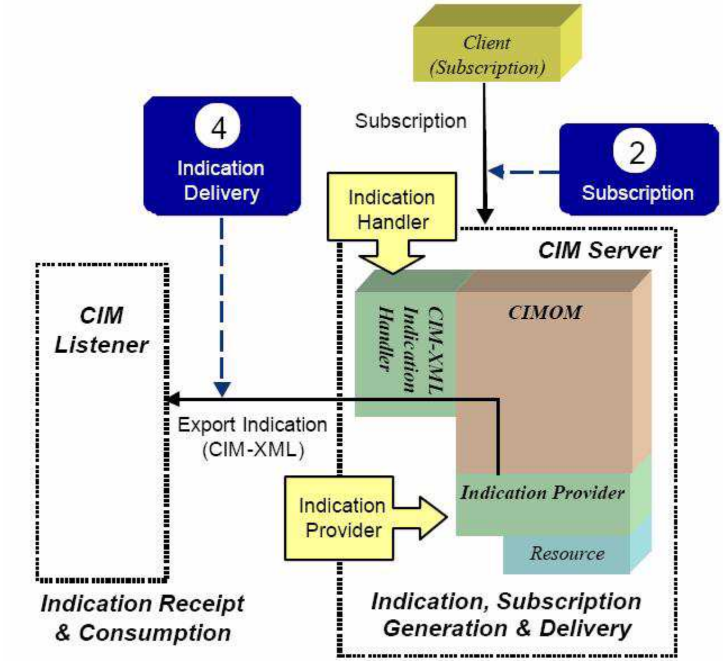


Figure 5: CIM Event Model (Source: CIM Event White paper)

of important Indications.

5 Conclusion

We have proposed a novel construct called active associations, to introduce active data modeling capabilities in DMTF's Common Information Model. The proposed construct enables various management tasks to be incorporated into the middleware itself. We showed this through the example of management of business processes modeled as composite web services. We also discussed implementation issues that need to be addressed before active associations can be fully realized in CIM. We plan to extend this work to explore other applications of active associations and more generally to study the usefulness of rules in CIM outside of associations. Another research direction to follow is enabling support for web service management in CIM. It would have to start with modeling of various aspects of web services and associated protocols.

References

[1] William Stallings. *SNMP, SNMPv2, and CMIP*. Don Mills: Addison-Wesley, 1993.

[2] Common Information Model. Distributed Management Task Force, http://www.dmtf.org/standards/standard_cim.php.

[3] Alexander Keller, Heather Kreger, and Karl Schopmeyer. Towards a CIM Schema for Runtime Application Management. In *Proceedings of the 12th IFIP/IEEE International*

Workshop on Distributed Systems: Operations & Management (DSOM), Nancy, France, October 2001.

[4] Manish Gupta, Anindya Neogi, Manoj K. Agarwal, and Gautam Kar. Discovering Dynamic Dependencies in Enterprise Environments for Problem Determination. In *14th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, DSOM 2003, Heidelberg, Germany*, pages 221–233, Oct 2003.

[5] S. Graham, S. Simeonov, T. Boubez, G. Daniels, D. Davis, Y. Nakamura, and R. Neyama. *Building Web Services with Java: Making Sense of XML, SOAP, WSDL, UDDI.* Sams, 2001.

[6] Christoph Schlueter Langdon. The State of Web Services. *IEEE Computer*, July 2003.

[7] OASIS. Web Services Distributed Management. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev̄wsdm.

[8] Nicolas Catania, Pankaj Kumar, Bryan Murray, Homayoun Pourhedari, William Vambenepe, and Klaus Wurster. Web Services Management, Web Service Mangement Framework version 2.0. http://devresource.hp.com/drc/specifications/wsmf/WSMF-WSM.jsp, July 2003.

[9] Vikas Agarwal, Neeran Karnik, and Arun Kumar. An Information Model for Metering and Accounting. In *Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS 2004), Seoul, Korea*, April 2004.

[10] Markus Debusmann and Alexander Keller. SLA-driven Management of Distributed Systems using the Common Information Model. In *Proceedings of the 8th IFIP/IEEE International Symposium on Integrated Network Management (IM), Colorado, US*, March 2003.

[11] Alexander Keller, Andreas Koppel, and Karl Schopmeyer. Measuring Application Response Times with CIM Metrics Model. In *Proceedings of the 13th IFIP/IEEE International Workshop on Distributed Systems: Operations & Management (DSOM), Canada*, Oct 2002.

[12] W. Bumpus, J.W. Sweitzer, P. Thompson, A.R. Westerinen, and R.C. Williams. *Common Information Model: Implementing the Object Model for Enterprise Management.* Wiley, 2000.

[13] Common Information Model (CIM) Specification, Version 2.2. Distributed Management Task Force, http://www.dmtf.org/standards/documents/CIM/DSP0004.pdf, June 1999.

[14] Common Information Model (CIM) Core Model, Version 2.4. Distributed Management Task Force, http://www.dmtf.org/standards/documents/CIM/DSP0111.pdf, August 2000.

[15] Eric N. Hanson and Jennifer Widom. An Overview of Production Rules in Database Systems. *The Knowledge Engineering Review*, 8(2), June 1993. pp. 121-143.

[16] D. R. McCarthy and U. Dayal. The architecture of an active database management system. In *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, pages 215–224, May-June 1989.

[17] Stella Gatziu and Klaus R. Dittrich. SAMOS: an Active Object-Oriented Database System. *IEEE Data Engineering Bulletin*, 15(1-4), Dec 1992. pp. 23-26.

[18] N. H. Gehani, H. V. Jagadish, and O. Shmueli. Event specification in an active object-oriented database. In *Proceedings of the 1992 ACM SIGMOD International Conference on Management of Data*, pages 81–90. ACM Press, 1992.

[19] CIM Event Model White Paper, Version 2.1. Distributed Management Task Force, http://www.dmtf.org/standards/documents/CIM/DSP0107.pdf, June 2003.

[20] Daniel F. Lieuwen, Narain H. Gehani, and Robert M. Arlein. The ode active database: Trigger semantics and implementation. In Stanley Y. W. Su, editor, *Proceedings of the Twelfth International Conference on Data Engineering, February 26 - March 1, 1996, New Orleans, Louisiana*, pages 412–420. IEEE Computer Society, 1996.