Deriving Knowledge for the Knowledge Plane

Stephen Quirolgico, Kevin Mills and Doug Montgomery
National Institute of Standards and Technology
Advanced Network Technologies Division
Gaithersburg, MD 20899-8920

June 10, 2003

Abstract

The knowledge plane will be highly dependent upon knowledge derived from information contained in participating nodes. Such information will comprise data generated by a participating node's hardware (e.g., CPU and network devices) and software (e.g., operating system and network applications) that conforms to one of many distinct ontologies or information models. Before node information can be utilized by the knowledge plane, it must first be acquired and translated from its native forms. In this paper, we suggest possible alternative architectures for acquiring, translating and representing sensor information that could ultimately serve as knowledge for the knowledge plane. We posit these architectures in order to identify and discuss the pros and cons of various approaches for deriving knowledge for the knowledge plane.

1 Introduction

Recently, the *knowledge plane* has been proposed as a distributed and decentralized construct within the Internet to gather, aggregate and act upon information about network behavior and operation [2, 3, 14]. The architecture of the knowledge plane comprises a "central nervous system" that embodies various "core cognition" engines to allow for reasoning and learning about the network. These engines are applied to information that is received and aggregated from participating nodes within the network.

Given the knowledge plane's dependence on information acquired from nodes within the network, it is important for such nodes to be able to derive the information required by the knowledge plane. In some cases, such information may include raw data generated by a single hardware or software sensor (e.g., the number and distribution of packet collisions detected by a 802.11 card or the response delay to a pattern of ping messages). In other cases, however, a node may need to acquire information from many of its sensors and transform this information into knowledge that may be used by the knowledge plane. For example, the knowledge plane may require information from various nodes to identify the cause of network timeouts. Here, each node may need to assess its network state, configuration, behavior and performance by acquiring data from multiple sensors including its network applications, network interface

cards, drivers and operating system. A node may even need to acquire supplementary information, such as its CPU load, active processes, and memory usage.

In this paper, we suggest possible alternative architectures for acquiring, translating and representing node information that could ultimately serve as knowledge for the knowledge plane. We posit these architectures in order to identify and discuss the pros and cons of various approaches for deriving knowledge for the knowledge plane. We begin by presenting a brief overview of ontologies and information models that may be exploited to derive information about sensors. This overview is followed by a detailed discussion about the relationships between knowledge requirements of the knowledge plane and its impact on architectures for deriving knowledge from node information.

2 Ontologies and Information Models

Given the myriad formats, models and semantics of information generated by hardware and software components, the task of acquiring, translating and representing related information is challenging. Over the years, a number of standards organizations have attempted to alleviate this problem by developing standardized ontologies and information models that describe extensive information about systems, devices, applications and networks. The goal of these standards is to facilitate the development, and enhance the functionality, of network management systems by restricting the variability in the type, format and semantics of system-generated information.

In this section, we describe some standardized ontology and information models. These standardization efforts exist in the context of substantial investment among equipment and software suppliers to represent system management information. Further, some of the standardized information models (particularly, the newer, emerging standards) may not yet be widely adopted. The limited uptake and deployment of standardized solutions to management information models (MIMs) increases the challenge associated with acquiring, translating, and representing node information, which must be obtained (at least in the near term) from proprietary MIMs supplied by various vendors. For these reasons, an architecture for acquiring, translating and representing node information will most likely need to consider a variety of standardized and proprietary information sources, ontologies and information models.

2.1 Common Information Model

The Distributed Management Task Force (DMTF) Common Information Model (CIM) is a conceptual information model for describing overall computer and network management information. The CIM comprises a specification that defines the language, naming and mapping techniques for integration with other management models including the Simple Network Management Protocol (SNMP), Management Information Bases (MIBs) and DMTF Management Information Formats (MIFs) [7]. The CIM specification also includes a *meta schema* that defines the terms used to express the model and its semantics. Elements of the meta schema include `Classes`, `Properties`, `Methods` and `Associations`.

The CIM schema [12] defines models for systems and network components through a series of increasingly specific building blocks:. a *core schema*, *common schemas* and *extension schemas*. The core schema captures notions applicable to all areas of system

and network management including general component information, statistics, location and configurations. The common schemas capture vendor-neutral notions common to particular areas of computer and network management. Common schemas include models for describing systems, devices, networks, applications, metrics, databases, the physical environment, event definitions and handling, management of a CIM infrastructure, users and security, policy and trouble ticketing/knowledge exchange. Extension schemas represent vendor-specific extensions of the common schema. These schemas can be specific to particular environments (e.g., UNIX). Instances of the CIM specification and schemas may be represented in a variety of languages including the Extensible Markup Language (XML).

The DMTF has also defined a number of adjunct specifications that use the CIM specification and schemas to facilitate computer and network management. These adjunct specifications include: (1) the Desktop Management Interface (DMI) specification for managing and tracking components on a machine [6], (2) the Web-Based Enterprise Management (WBEM) specifications for unifying the management of enterprise computing environments [10, 11], (3) the Directory Enabled Network (DEN) specifications for providing building blocks for intelligent management by mapping concepts from CIM (such as systems, services and policies) to a directory and then integrating this information with other WBEM elements in the management infrastructure [9, 8] and (4) the System Management BIOS (SMBIOS) specification for defining how motherboard and system vendors present management information about their products [13].

2.2 Management Information Base

The IETF Management Information Base (MIB) is a collection of specifications that define the properties of a managed device (e.g., network interface card) [1]. These properties are defined in groups including System, Interfaces and IP groups. MIBs are generally associated with the use of SNMP for retrieving or updating MIB property values [4, 17]. For example, SNMP may be used to query System properties of a device to determine the device's name, vendor, location and purpose. Similarly, SNMP may be used to query Interface properties of a device to determine the number of sent and received packets, number of broadcasts, number of packet collisions, frame sizes and the current output queue size of the device. MIBs, which are represented using ASN.1, conform to the Structure Management Information specification defined in RFC 1155 [15].

MIBs have been adopted by equipment suppliers during the past several years and are currently available in numerous networking products. However, MIBs suffer from a number of issues that hinder their use in (effective) network management [16]. One issue concerns the focus of MIBs, and by extension SNMP, which concentrate solely on detailed characteristics of network devices making it difficult to abstract managerial tasks. Fortunately, these problems might be resolved by using richer representation models such as CIM, which can then be mapped to MIBs.

2.3 Standard Upper Ontology

The IEEE Standard Upper Ontology (SUO) is an emerging ontology standard that is general and covers a broad range of domain areas. Concepts specific to given domains

are not included in the SUO; however, the SUO standard provides a structure and a set of general concepts upon which domain ontologies (e.g., medical, financial and engineering) could be constructed. One such domain ontology is the Quality of Service (QoS) ontology that describes computer and network concepts including (hardware and software) products, processes, networks, and configurations. The QoS ontology is represented using the Knowledge Interchange Format (KIF) and can be easily mapped to a XML representation.

2.4 Non-Standard Ontologies and Information Models

In addition to standardized ontologies and information models, non-standardized information models are also used to define sensor information. Such models may include proprietary models (e.g., Microsoft event logs or Apache HTTP server logs).

3 Architectures and Knowledge Requirements

An architecture for the knowledge plane must consider the requirements of knowledge at different levels in the system as well as the relationships among such knowledge. In the knowledge plane, there may be three primary levels at which knowledge requirements could be defined: the cognitive level, node level and translation level. At the cognitive level, knowledge must be specified such that it satisfies the reasoning goals of the knowledge plane; this includes not only having the appropriate knowledge, but also ensuring that this knowledge (1) conforms to specified ontologies and (2) is encoded using specified representation languages. When specifying knowledge requirements at the cognitive level, it will be important to ensure that such knowledge can be derived from lower levels (e.g., the sensor level). In addition, knowledge at the cognitive level may be required to retain the semantics of knowledge derived at lower levels.

At the node level, knowledge requirements are defined by the node's information model (whether standard or proprietary) and populated by a node's sensors and the information that these sensors generate, along with configuration settings established for the node. At this level, it will be important to ensure that information required by higher-levels can be derived from a node's information model.

The translation level maps models from instantiations of disparate node information models to a form that may be used at the cognitive level. Thus, the models and mechanisms for translation are crucial in deriving knowledge for the cognitive level.

In this section, we posit architectures to derive knowledge for the knowledge plane from node information.[1] We begin this section by describing the general knowledge requirements of the knowledge plane and suggest possible architectures for supporting these requirements. We follow this with a discussion of translation issues and posit components for acquiring, translating and representing node information. We conclude this section with a discussion on possible translation schemes.

3.1 General Architecture

The general knowledge requirements of the knowledge plane require that low-level node information be propagated to higher-level, cognitive processes in a form suited

[1]Note that this paper focuses on deriving knowledge and not on asserting controls.

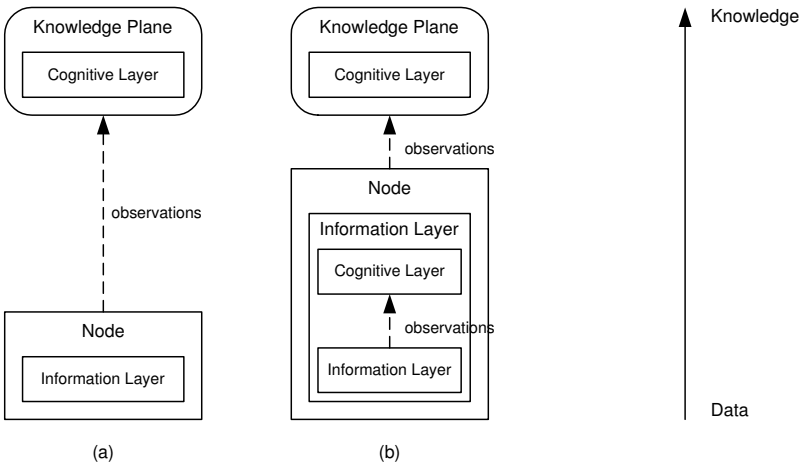

Figure 1: General architectures (a) base architecture and (b) architecture with recursively-defined information layer.

to the particular problem that a specific cognitive process is trying to resolve, or to the management service that a specific cognitive process is attempting to provide. In addition, node information must be propagated in a form suited to the knowledge representation and reasoning requirements of the specific cognitive process that requires the information. Thus, a general architecture for the knowledge plane must contain at least two fundamental layers: an *information layer* and a *cognitive layer*. The information layer represents the set of entities of a node that provides configuration information and produces observations for use by the knowledge plane. The cognitive layer embodies the knowledge plane's reasoning mechanisms and cognitive services, as targeted to specific knowledge-based applications (e.g., K-apps).

An architecture to support the general requirements of the knowledge plane may be designed in several ways. The simplest design may comprise an information layer for each participating node and a cognitive layer in the knowledge plane. We refer to this architecture as a *base architecture*. In a base architecture, the information layer would comprise the set of sensors and information models for each participating node. In addition to the base architecture, an alternative architecture to support the general requirements of the system may comprise recursively defined information and cognitive layers. For example, a participating node may comprise both an information layer and a cognitive layer to support local reasoning (at whatever scope the node exists, such as LAN manager, enterprise manager or autonomous-system manager). Here, information from a node may be processed by the node's own cognitive layer, which in turn propagates this knowledge (as observations) to the knowledge plane. This recursive architecture might facilitate scalability by reducing the volume of data and knowledge to transmit and store, and also by reducing associated processing requirements. These alternative architectures are shown in Figure 1. For the sake of brevity, we herein simplify discussion by considering only the base architecture. We note, however, that the base architecture may be easily extended to realize recursively-defined information and cognitive layers.

In the base architecture, sensors and configuration variables in the information layer

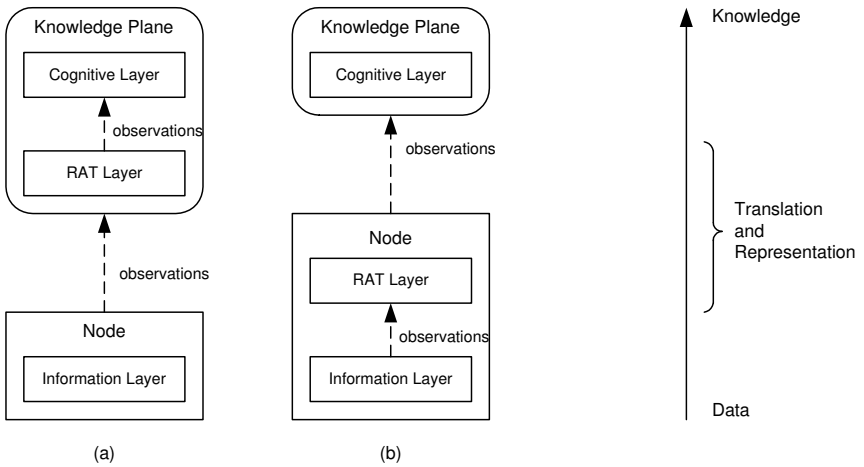

Figure 2: Base architecture with RAT layer (a) as part of knowledge plane (b) as part of participating node.

provide observations to the cognitive layer. However, a means is required to transform this information into a form suitable for use by the cognitive layer. Transformation might be required because the cognitive layer (1) works at a higher level of abstraction, (2) considers aggregations over sensor observations, or (3) requires a particular form of knowledge representation to facilitate its reasoning mechanism. In this paper, we posit the use of an additional layer to facilitate these tasks. We refer to this layer as the Registration, Acquisition and Translation (RAT) layer. As shown in Figure 2, the RAT layer may exist within the knowledge plane or within each of the knowledge plane's participating nodes. Note that the location of the RAT layer may affect the performance of the distributed system as well as how node information is acquired, translated and represented.

An architecture for deriving knowledge for the knowledge plane must also support communication technologies for propagating node information to higher-layer, cognitive processes. Such technologies may include components for transmitting information via primitive communication protocols (e.g., HTTP) and APIs (e.g., CORBA). Supported communication technologies may also comprise components for facilitating more intelligent forms of communication. Such components may include agent-based components for providing peer-to-peer communication, coordination and collaboration. Such agents may utilize a standardized agent communication infrastructure (e.g., FIPA [5]) to facilitate intelligent knowledge sharing between heterogeneous agents in the knowledge plane and participating nodes. Agents may also communicate knowledge using a variety of knowledge representation languages including the Resource Description Framework Schema (RDFS), the DARPA Agent Markup Language (DAML), and the Web Ontology Language (OWL). Other components for facilitating communication may include service discovery components that allow participating nodes to acquire knowledge plane service (e.g., translation services) or that allow the knowledge plane to access services for a participating node (e.g., data acquisition services).

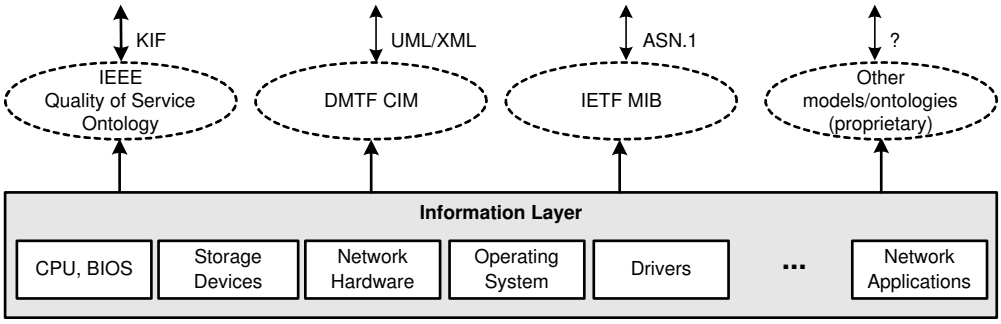

Figure 3: Information layer.

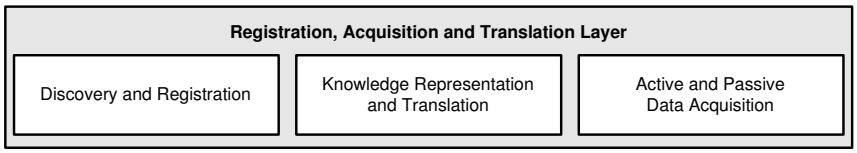

Figure 4: RAT layer.

3.2 Information Layer

In the base architecture, the information layer defines the set of sensors, configuration variables and local information models for a participating node. As shown in Figure 3, this set may include CPUs, BIOSs, operating systems, networking hardware, storage devices, drivers and applications. The ontology or information model used to define node-local information may be vendor-dependent but conform to an existing standard (e.g., MIB, CIM and SUO). In other cases, however, the ontology or information model used to define node-local information will be application-specific or proprietary, and conform to an information model that is non-standard. Note that a node will likely encompass sensors, variables, and actuators that conform to a variety of ontologies or information models — each of which may be encoded using a distinct representation language (e.g., MIB, CIM and SUO may be encoded using ASN.1, XML and KIF, respectively).

3.3 Registration, Acquisition and Translation Layer

Before node-local information can be used by a cognitive layer, it must first be acquired, translated and represented in a suitable form. These tasks are handled by the Registration, Acquisition and Translation (RAT) layer. The RAT layer comprises three components including (1) a Discovery and Registration component, (2) an Active and Passive Data Acquisition component and (3) a Knowledge Representation and Translation component. The RAT layer is shown in Figure 4.

3.3.1 Discovery and Registration Component

The Discovery and Registration (DREG) component allows monitorable and controllable sources to register their participation in some scope (e.g., node or LAN) of the

management infrastructure. Information registered by a source may include device type, driver version, supported ontology, actuation interface and other attributes. Such information may be used by higher-layer applications (e.g., cognitive, management, or agent-based applications) to search for specific sources and query for information. The DREG component also permits information consumers to register requirements for information, which can then be fulfilled as information sources register.

3.3.2 Active and Passive Data Acquisition Component

The Active and Passive Data Acquisition (APDA) component allows higher-layer applications to actively or passively acquire data from sources. This component provides APIs for direct querying of sources as well as APIs for providing passive querying through callback mechanisms.[2] Such callbacks may be invoked by specifying trigger events and pattern matches.

3.3.3 Knowledge Representation and Translation Component

The Knowledge Representation and Translation (KRT) component maps the ontology or information models used to define node-local information into knowledge that may be utilized by various higher-layer, cognitive processes. Depending on the requirements of the knowledge plane, there are at least two approaches to defining such knowledge: the *domain ontologies* approach and the *composite ontology* approach.

Domain Ontologies Approach In the domain ontologies approach, node-local information is mapped directly to one or more domain- or application-specific ontologies used by the knowledge plane. We refer to these ontologies as *domain ontologies*. Because domain ontologies are tightly coupled to the reasoning requirements of the knowledge plane, they must not only reflect the semantic information generated by nodes, but must also conform to domain- or application-specific schemas defined by the knowledge plane.

The use of domain ontologies is motivated primarily by a need to provide the rich semantic models that describe the relationships between node information and the knowledge plane's domains or applications (e.g., K-apps). Such models may facilitate cognitive processes by providing all of the information required for a specific domain or K-app in a single model. Figure 5 shows an example derivation of domain ontology instances. An example translation of domain ontologies is shown in Figure 6. Note that domain ontologies may or may not share underlying schemas or utilize the same encoding schemes.

The main disadvantage to using domain ontologies is that models for representing node information may differ from node to node requiring a separate translator for mapping information from a specific node to a specific domain ontology. This problem is illustrated in Figure 7.

Composite Ontology Approach In the composite ontology approach, information about all sources including the relationships among sources, is first mapped to a single, common ontology before being mapped to domain ontologies. We refer to

[2]Note that such sources may supply these APIs themselves.

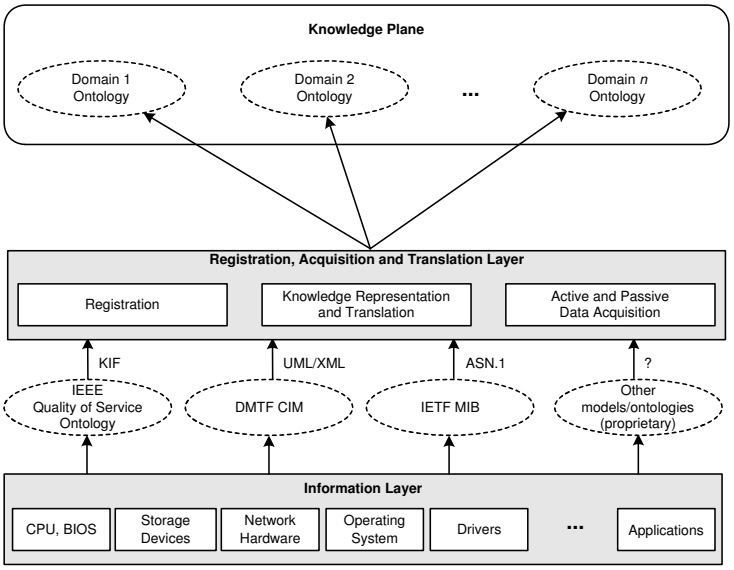

Figure 5: Derivation of domain ontologies.

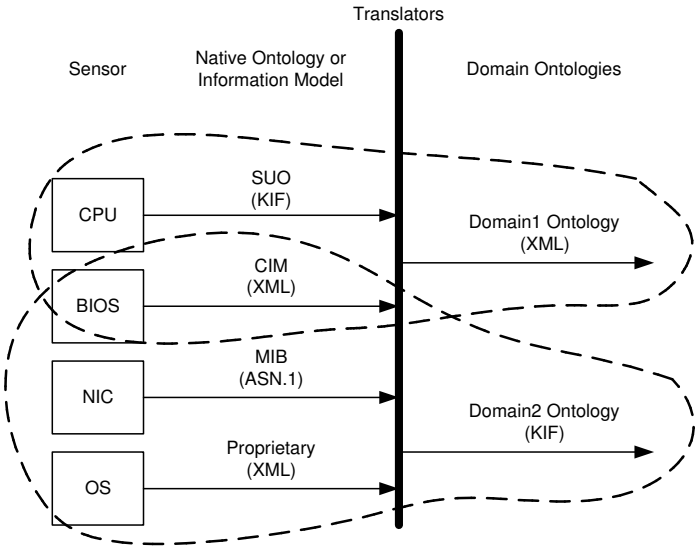

Figure 6: Translations of domain ontologies.

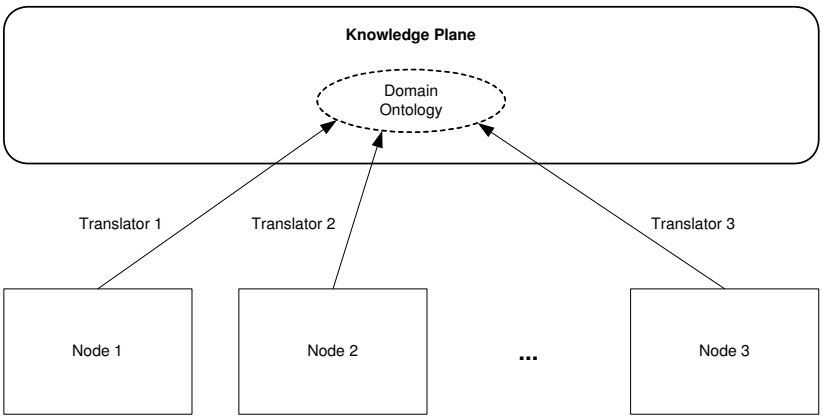

Figure 7: Multiple translators for heterogeneous nodes.

this single, common ontology as a *composite ontology*. Because the composite ontology provides a single, comprehensive model for specifying information about all sources including relationships among sources, it must reflect the appropriate level of semantic richness required to capture the semantics of all of the models used to specify source information. A possible motivation for using a composite ontology is twofold. First, a composite ontology limits the number of required translators to (1) one large translator from the sources to the composite ontology and (2) a number of smaller translators to extract application-specific knowledge from the composite ontology and convert this knowledge into a specific representation suited to the reasoning mechanisms of the K-app that uses the knowledge. At present, it appears unclear which approach (i.e., the composite or domain ontologies approach) requires the fewest translators; however, using a composite ontology will require two translation steps for each K-app, while using domain-specific ontologies will require only one translation step for each K-app.[3] Previous research into knowledge-based automated programming provides some evidence that a common-ontology approach can prove unwieldy and unworkable, while a domain-specific approach can provide more focused results in support of targeted problems and immediate needs. A second possible motivation for using a composite ontology is to facilitate aggregation of source information by higher-layer, knowledge based management systems by providing a single, consistent model for all source information. For example, suppose a knowledge base management system maintains a knowledge base of device-driver attributes. In the composite ontology, knowledge about each device driver in a node would conform to a single model allowing the knowledge based management system to easily maintain this knowledge. Without a composite ontology, however, the knowledge based management system might need to understand possibly numerous models for representing device-driver knowledge. Figure 8 shows the use of a composite ontology for deriving domain ontologies. Note that a composite ontology is concerned solely with representing the semantics of node-generated information and not with specifying the relationships between node-generated information and specific domains or applications.

[3]Note, however, that if the composite ontology is standardized, this will likely ease the development of translators between the source-generated models and the composite ontology, as well as between the composite ontology and the domain ontologies.

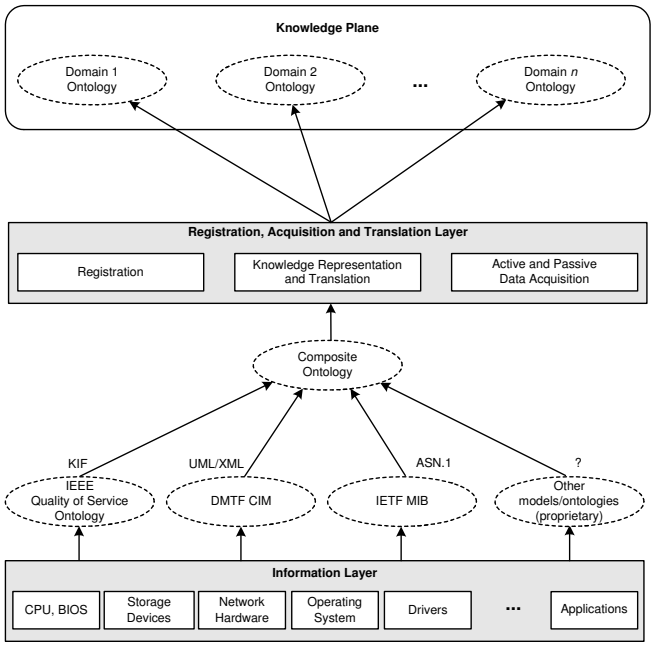

Figure 8: Use of composite ontology for deriving domain ontologies.

The placement of the RAT layer may affect the selection of the ontology that serves as the composite ontology. If the RAT layer exists on a participating node, the composite ontology need only reflect the semantic richness required to model the node's source information. However, if the RAT layer exists within the knowledge plane, the composite ontology might need to reflect the semantic richness required to model source information from multiple nodes.

The selected composite ontology must be able to appropriately capture the semantics of a variety of standardized and non-standardized models. Although a composite ontology could be developed from scratch, the effort to build such an ontology could be enormous. Instead, we posit the use of an existing standardized ontology to serve as the composite ontology. By using an existing standardized ontology, we may accelerate the development of higher-layer knowledge based systems. In addition, the use of an existing standardized ontology may improve performance if there are several sources that already generate information that conform to the same ontology. For example, if CIM is defined as the composite ontology and several sensors generate instances of CIM and MIB information, then little or no translations are required before this information is mapped to domain ontologies.[4] Translation from sensor information to a composite ontology, and from the composite ontology to domain ontologies, is shown in Figure 9. Here, CIM is used as the composite ontology.

One disadvantage of using a composite ontology is that it may not be able to completely or accurately capture the semantics of all the models used by sources. Another disadvantage to using a composite ontology is that there may exist competing ontologies that serve as the composite ontologies on different nodes. In such cases, the knowledge plane will require translators to translate between these competing ontologies and the ontologies used by the knowledge plane's cognitive processes.

[4]Note that CIM defines mappings to MIBs.

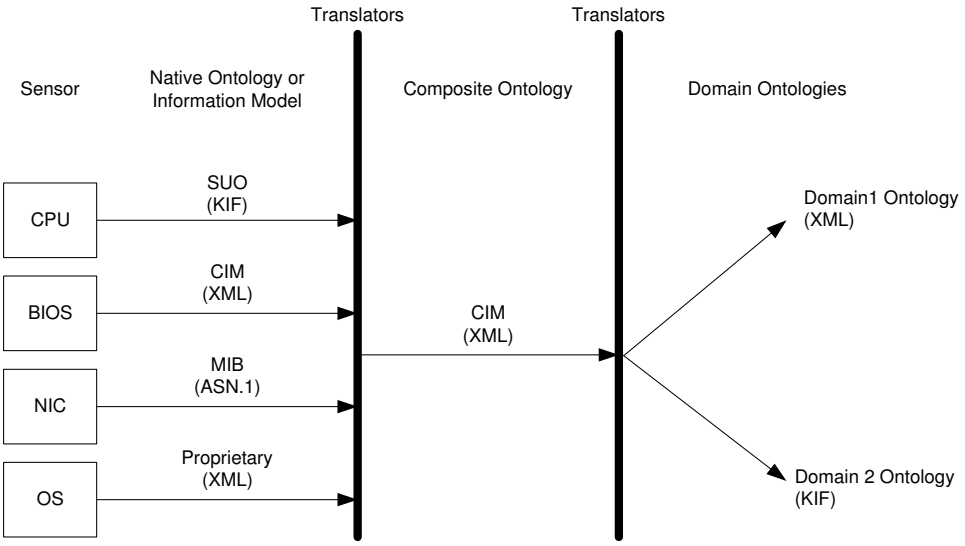

Figure 9: Composite ontology translation.

3.4 Cognitive Layer

The cognitive layer is associated with the knowledge plane's cognitive services and applications (e.g., K-apps). Here, the knowledge requirements are substantially more open-ended and variable than at the node-information layer where knowledge may be more constrained by existing sources. Although the cognitive layer may embody a single ontology to represent all source data, it will instead likely comprise several domain ontologies. Note that domain ontologies will specify other types of knowledge in addition to node-local knowledge including assertions and explanations [3].

At the cognitive layer, communication mechanisms will be needed to communicate with participating nodes. Such technologies will likely require support for primitive communication protocols (e.g., HTTP) and APIs (e.g., CORBA) as well as more sophisticated infrastructures (e.g., FIPA agent communication infrastructures or scalable publish-subscribe event systems).

4 Conclusion

In this paper, we suggested possible alternative architectures for acquiring, translating and representing sensor information that could ultimately serve as knowledge for the knowledge plane. We posited these architectures in order to identify and discuss the pros and cons of various approaches for deriving knowledge about a node, as well as to illuminate other issues surrounding the derivation of knowledge for the knowledge plane. We described a general architecture for the knowledge plane and posited mechanisms for acquiring, translating and representing sensor information. We described the concept of composite and domain ontologies and described how the use of such ontologies could facilitate the translation and representation of sensor information into a form suitable for use by the knowledge plane. We also described the relationships between composite and domain ontologies and suggested ways of exploiting the features

of each.

Although we touched on several issues concerning the derivation of knowledge for the knowledge plane, other issues will likely need to be addressed. Some of these issues include (1) determining which ontology approach (i.e., the composite or domain ontologies approach) is best suited for use in the knowledge plane, (2) identifying alternative approaches to modeling and representing node-local information, (3) determining how to appropriately coalesce information from disparate nodes into a consistent composite or domain ontology, and (4) deciding which encoding schemes and knowledge representation languages should be used at various levels in the system.

References

[1] T. Cikoski. SNMP FAQ, January 2003.

[2] D. Clark. A new vision for network architecture, September 2002.

[3] D. Clark, C. Partridge, and C. Ramming. A knowledge plane for the internet. In *SIGCOMM 2003*, February 2003.

[4] S. Feit. *SNMP: A Guide to Network Management*. McGraw-Hill, 1995.

[5] Foundation for Intelligent Physical Agents. Frequently asked questions about fipa, 2002.

[6] Distributed Management Task Force. Desktop management interface specification, June 1998.

[7] Distributed Management Task Force. Common information model specification, June 1999.

[8] Distributed Management Task Force. Guidelines for CIM-to-LDAP directory mapping, May 2000.

[9] Distributed Management Task Force. DMTF LDAP schema for the CIM v2.5 core information model, April 2002.

[10] Distributed Management Task Force. Specification for CIM operations over HTTP, May 2002.

[11] Distributed Management Task Force. Specification for the representation of CIM in XML, May 2002.

[12] Distributed Management Task Force. CIM schema v2.7.2, May 2003.

[13] Distributed Management Task Force. System management BIOS reference specification, v2.3.4, January 2003.

[14] C. Ramming. The knowledge plane, January 2003.

[15] M. Rose and K. McCloghrie. RFC 1155: Structure and identification of management information for TCP/IP-based internets, May 1990.

[16] J. Schoenwaelder. Overview of the 2002 iab network management workshop, May 2003.

[17] A. Tanenbaum. *Computer Networks*. Prentice Hall, third edition edition, 1996.