Understanding as We Roam

Increasingly, the problem for roaming users isn't discovering local Web services or local context information or even getting their applications to interoperate with them at a syntactic level, but rather achieving interoperation at a semantic level. Developers are beginning to use Semantic Web technologies such as ontologies to expose the semantics of context models and Web services. However, different ontologies arise due to natural human diversity in modeling domains. It's therefore important to develop mappings between ontologies. This article describes a practical ontology-mapping process specified in the Unified Modeling Language, as well as its deployment.

Declan O'Sullivan, Vincent Wade, and David Lewis
Trinity College Dublin

Imagine you arrive for a meeting after a long-haul flight with a couple of hours to spare. You want a good lunch to replenish lost energy, but when you download your email, you find a document you must read before your meeting. You hate reading anything large on a PDA screen, so you ask your PDA to locate an eatery that serves the kind of food you like, has a free printing service, is located near the meeting venue, and will let you make a reservation that gives you enough time to walk to the meeting on schedule. To do this, your personal applications will have to interact with a diverse set of applications and services in the local environment.

Roaming users bring with them applications that increasingly must interact with the local computing environment via local Web services or by accessing local context information. Although developers have proposed several uniform platforms for retrieving context information, service providers continue to publish this information in a wide variety of formats and semantics. Although emerging standards such as the Semantic Markup for Web Services (OWL-S) and the Web Service Modeling Ontology (WSMO) represent Web service semantics, it's unlikely that service developers will use exactly the same semantic models in their Web service signatures.

Achieving an effective user experience in roaming situations will require mapping the semantic models used to represent Web context models and local Web service semantics to the semantic models used by the roaming users' applications. Yet, as Tim-Berners Lee noted in a recent review of the progress of semantic Web technologies,[1] ontology mapping remains a key challenge. Currently, most ontology-mapping systems are proprietary, with process and implementation closely intertwined, leading to a lack of consensus and momentum in the development of practical processes and tools in this crucial area for roaming.

To address this gap, we've developed a practical ontology-mapping process specified in a technology-neutral manner by using the Unified Modeling Language (UML). Although implementations show that our process and tools can successfully identify mappings and structural mismatches and help reduce effort and mapping errors, several challenges must be overcome before ontology mapping for roaming can become a widespread activity.

Ontology Mapping: Why, What, and How

Ontologies aim to let people and application systems share and communicate an understanding of a domain.[2] However, human diversity means that different ontologies will arise in modeling a domain — that is, people typically create ontologies from a particular perspective. Thus, to achieve roaming, we'll need to map ontologies that represent the semantics of personal information and the applications on the mobile platform to ontologies in the visited environment. Creating such mappings for ordinary roaming users will benefit mobile application vendors and local service environment managers (such as local tourist boards) as well as large organizations with roaming users.

Mappings between ontology elements are usually expressed as pairs of related entities in some mapping expression, which can range from simple equivalences to complex correspondences. An example simple equivalence is when *eatery* in the user's ontology is equivalent to *restaurant* in the city's tourist Web service ontology. An example complex correspondence is when a *price* property in the user's ontology is equivalent to the *cost* property plus the *local-tax* property in the restaurant's ontology. The mapping system typically outputs these expressions as separate documents, so it can manage the mappings independently of the ontologies.

Ontology mapping as a field of study is much younger than either ontology engineering or database schema mapping. Consequently, tools and techniques to support it have emerged primarily as extensions to database schema mapping approaches or ontology editors, or as developments supporting only the mapping activity . Subsequently, no commonly agreed-upon life-cycle model of ontology mapping yet exists. We've introduced a four-phase ontology-mapping life cycle to assist in analyzing the current state of play.

The *characterization phase* encompasses activities that discover ontologies for mapping, analyze the potential difficulties involved in undertaking a mapping, and generate candidate matches between the ontologies. Practically, the characterization phase helps organizations avoid wasted human effort or tool processing when attempting to map incompatible or poor-quality ontologies. However, current ontology-mapping systems concentrate mainly on finding matches between ontology elements. They use matching techniques that range from simple lexical matching to semantic model matching.[3,4] Some of these techniques operate on schema information alone, some on instance information, and some on both. Few of the current ontology-mapping systems help users analyze ontology quality or potential difficulties in mapping particular ontologies.

The *mapping phase* covers the activities involved in detecting mappings from the matches found in the characterization phase. Although fully automatic generation of mappings from ontology match information is feasible in some circumstances, it's generally considered impractical. Any automatic approach to matching two ontologies has a degree of uncertainty arising from the syntactic representation of ontologies, the combination of similarity measures, and the fact that most matching approaches are heuristic in nature. Thus, mapping detection has primarily relied on a human user examining the machine-generated matching information through a GUI. Systems such as Coma++[5] and Swoop,[6] for example, provide graphical support for presenting matching information and creating mappings by pointing and clicking. Coma++ also lets users browse mappings created in other mapping sessions. However, such systems rarely provide explicit user support for identifying mappings. In contrast, systems such as Protégé lead users through mapping decisions based on the iterative algorithm Anchor-Prompt,[7] and a recent proposal documents ontology-mapping patterns to assist decision making.[8]

In the *execution phase*, the system renders the mappings identified in the mapping phase in different formats, which it interprets to support semantic interoperability between two applications. Most current mapping systems express mappings in a proprietary format (such as the Mapping Framework [Mafra][9]) that's typically aligned with the technology used. Developers are increasingly recognizing the need for an open mapping format, though, and proposals have started to emerge. For

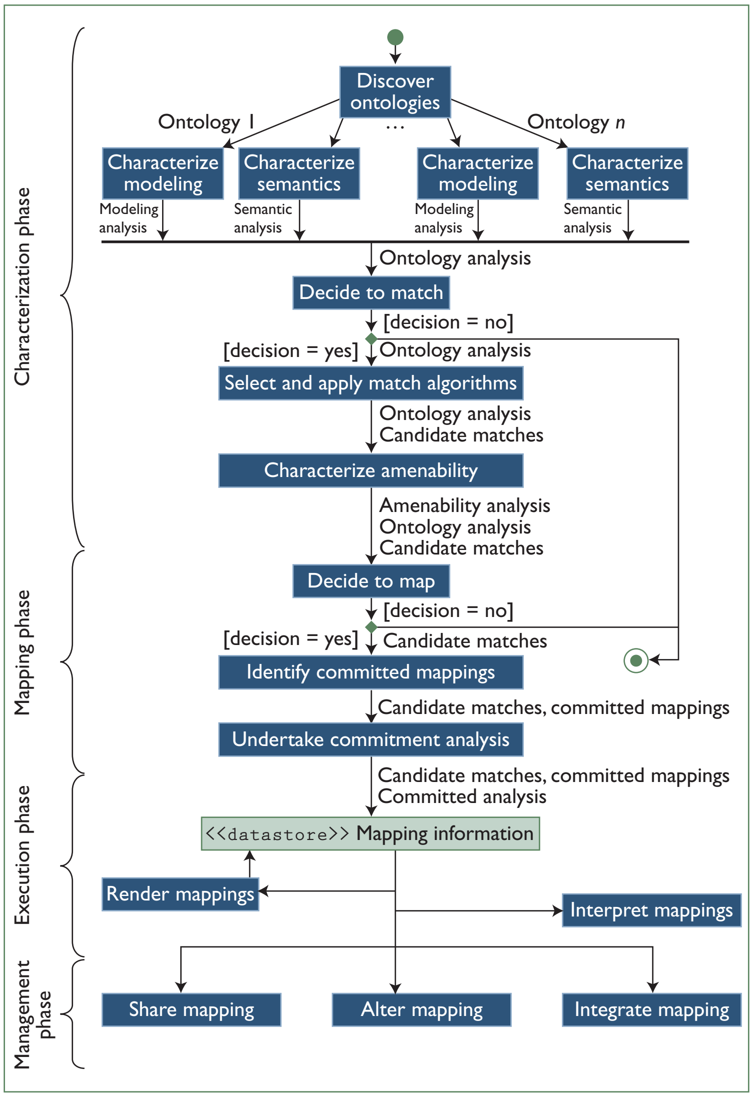


Figure 1. Summary Unified Modeling Language activity diagram for the ontology interoperability in support of semantic interoperability process. The diagram of the full OISIN process[12] includes Unified Modeling Language signals and data stores, showing how different roles can perform activities at different times.

example, the Information Interpretation and Integration Conference (I^{3}CON) and the Evaluation of Ontology-based tools (EON) contests featured XML-based formats for comparing the output of a variety of matching tools. Experience from these contests proved positive[10] and led to the development of an ontology-alignment format[11] that can be rendered into several other formats (Semantic Web Rule Language, OWL, and so on) for interpretation. Other researchers have proposed a generic mapping language that must be grounded in a declarative logical language and so requires a reasoner.[8] It's too early to tell which of these proposals will dominate or whether, instead, the Rule Interchange Format emerging from W3C will prove attractive for expressing the ontology mappings.

The *management phase* is responsible for the ongoing maintenance and management of the mappings once they're deployed. Although notable exceptions exist (such as Mafra and Coma++), ontology-mapping systems often treat mapping management as an afterthought. In addition, few researchers have explored the challenges associated with managing shared of ontology mappings.

The Process

With the significant progress in developing some of the technologies needed for mapping systems, many of the current challenges involve these systems' real-world deployment. Moving to real-world deployments requires moving away from closed ontology-mapping systems and toward ontology-mapping frameworks comprising process and tools. A framework approach can help organizations situate the ontology-mapping process within their current processes and communicate with contributors of supporting solutions.

Specifying the process independently of the tools (for example, using UML) lets you tailor the process to specific parts of an organization to support different tasks and allows for deployment of tools from different vendors. Moreover, you can implement individual parts of the process in various ways: manually, semiautomatically, and automatically.

In current ontology-mapping systems, the process isn't independent of the implementation. So, before creating our ontology-mapping system, we specified our process in UML. The UML activity diagram in Figure 1 summarizes the *ontology interoperability in support of semantic interoperability* (OISIN) process. In particular, the figure shows all activities following directly from each other and the data entities flowing between the activities. In reality, different roles are likely to perform the activities at different times, so the more detailed documented process[12] includes UML signals and data stores.

No two organizations are the same — even in the same organization, different use cases for ontology mapping can exist — so the process's design was nontrivial. The process must support the organization's choice of activities to include as well as the organization's choice of each activity as manual, semiautomatic, or automatic. Take, for example, a large organization with roaming users.

Such an organization's technical department will want to ensure that the ontologies of context-sensitive corporate applications that the corporate sales force uses will map to the context applications provided by the public authorities, such as location-based services in particular sales territories. Given the mappings' importance, such an organization will likely need a mix of manual, semiautomatic, and automatic activities. A contrasting example is a local tourist board that wants to support as many roaming applications as possible, but doesn't consider this a core objective. This organization would likely attempt a "best-effort" approach, implementing a fully automatic process that would attempt mappings for ontologies submitted by mobile application vendors or would attempt to reuse mappings created by sister tourist boards nationally or internationally.

Characterization Phase

The *discover ontologies* activity varies depending on the use case that the ontology mapping supports within an organization. For a systems-integration use case, the ontologies are likely to be known and presented to the integrator. For an open corpus-integration use case, the integrator might need to discover ontologies through a Web search.

The *characterize modeling* activity characterizes the ontologies using their quality, dimensions, and modeling style. From an automation perspective, modeling style is the most difficult of these three characteristics to determine.

The *characterize semantics* activity aims to analyze the ontology's content by characterizing how amenable the ontology is to either human or tool-based matching. It does this by examining the nature of the terms used in the ontology modeling for naming classes and properties, which can vary greatly. Understanding the terms' nature can positively impact the next activity.

The *decide to match* activity analyzes relevant information and decides whether to attempt to match the ontologies. If it makes a positive decision, the system performs the *select and apply match algorithms* activity to analyze the degree of overlap between the ontologies. Depending on the type of approach desired, the system can choose from a wide range of match algorithms — from lexical-match schemes to semantic-model-based schemes.[3] Match algorithms can also perform differently depending on the characteristics of the schemas involved.

The final activity in the characterization phase — the *characterize amenability* activity — examines the characteristics of the ontologies and the nature of the candidate matches discovered in previous activities; it then elicits key information items indicating the expected difficulty in mapping. This activity provides a natural boundary between the characterization and mapping phases, letting the organization perform the different phases in different departments or at different times. A separate activity also lets the organization choose relevant characteristics for consideration, especially if it uses third-party tools that output a wide range of ontology characteristics.

Mapping Phase

Organizations use policies to govern human and computing resource expenditures, and the organization will have to decide whether a mapping effort should occur. The *decide to map* activity applies organizational policies governing this decision making. In some cases, for example, the organization will have no choice but to undertake the mapping, so this activity will simply record information related to the mapping task to be allocated (it might record an estimate of time needed based on the task's degree of difficulty).

The mapping system must allow for two use cases in the mapping phase:

- In the first, an expert user determines the mappings that would generally satisfy an application set's needs.
- In the second, an individual application determines the mappings depending on the particular application's perspective and usage context.

Of course, the problem with an application determining its own mappings is that it needs some certainties or reference points to provide a context for deriving mappings from the available matching information. In the expert user use case, the *identify committed mappings* activity aims to identify as many definite correspondences as are relevant to the situation. For the individual-application use case, this activity aims to help the user identify the minimum number of definite correspondences, in order to leave the rest of the mapping determination to the application. Committed mapping is the term used to represent these definite correspondences between entities in the ontologies.

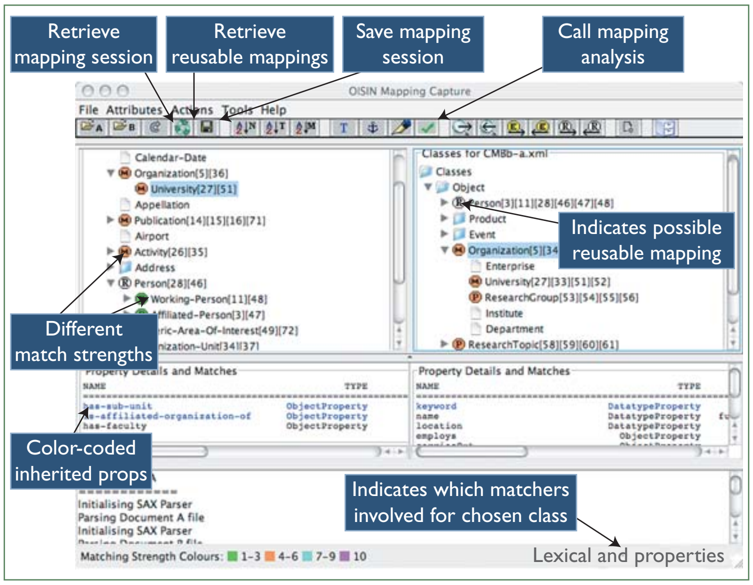

Retrieve mapping session
Retrieve reusable mappings
Save mapping session
Call mapping analysis
OISIN Mapping Capture
File Attributes Actions Tools Help
Indicates possible reusable mapping
Different match strengths
Color-coded inherited props
Indicates which matchers involved for chosen class
Lexical and properties
Matching Strength Colours: 1–3 4–6 7–9 10

Figure 2. OISIN mapping tool. This graphical-capture-style tool identifies and analyzes committed mappings.

Because entities in an ontology are highly interrelated, the availability of information about the committed mappings has implications for other match information. Thus, the *undertake commitment analysis* aims to analyze and enhance the other match information in light of the committed mappings.

Execution Phase

The mapping process's ultimate aim is to enable semantic interoperability for a diversity of situations. To support this goal, the mapping information must be available in human and software interpretable form. In the execution phase, an application interprets the mapping information (consisting of the candidate matches, committed mappings, and committed analysis entities) generated in the mapping phase to create mappings that are relevant to the usage context. In addition, in line with our requirements, the mapping information should be in a widely interpretable format. A data store stores the mapping information for retrieval and management purposes. The *render mapping* activity renders mapping information in different mapping formats. The *interpret mapping* activity interprets the mapping information to enable semantic interoperability between applications or people.

Management Phase

The *share mapping* activity lets the system share mapping information with other systems that require mappings for semantic interoperability purposes or that undertake an ontology-mapping task.

Conversely, the *integrate mapping* activity focuses on managing the receipt and integration of mappings that are being shared in the local mapping system. Finally, an *alter mapping* activity lets the system alter or retire existing mapping information.

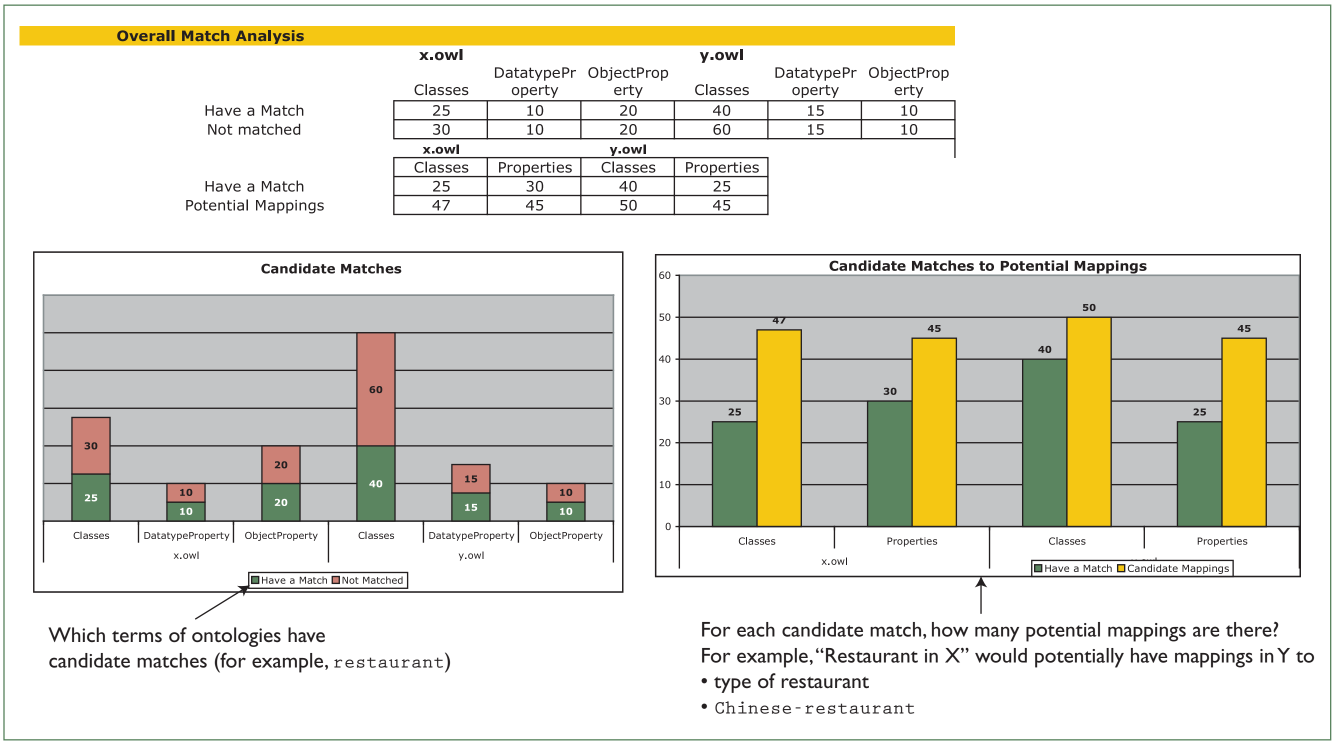
Implementation Experiences

Our implementation's scope was to support the ontology-mapping process's characterization, mapping, and execution phase activities. Eight different applications support the process. The applications don't call each other directly but depend on information processed by other applications in a persistent store. This approach facilitates the addition or substitution of applications. It also lets us execute the process's different activities at different times and execute them manually, semiautomatically, or automatically. We use XML to represent the information entities consumed or produced by each of the tools, and the HP Jena toolkit to convert the initial OWL representation of the ontologies into a canonical form. The Java-based tools manipulate the XML models by executing XQueries using Fraunhofer's IPSI-XQ engine.

To examine whether our process could support a continuum of deployment strategies, we implemented two organizational deployment scenarios. In the first, some key activities are semiautomated. To do this, we implemented semiautomated tools to support the mapping decision, identify committed mappings, and undertake commitment analysis activities, and used automated tools to support the other activities. In the second deployment scenario, the organization requires a fully automated ontology-mapping process, which was made possible through the availability of preexisting committed mappings that guided the rest of the mapping generation.

Figure 2 is a screenshot of the tool we used to identify and analyze committed mappings. It's similar in style to the graphical capture tools used elsewhere for capturing mappings.

We used Microsoft Excel to support the decide-to-map activity because it can aid decision making and most professionals are familiar with it. This approach lets us easily import and display characteristic information numerically and graph-



Overall Match Analysis

	x.owl			y.owl		
	Classes	DatatypeProperty	ObjectProperty	Classes	DatatypeProperty	ObjectProperty
Have a Match	25	10	20	40	15	10
Not matched	30	10	20	60	15	10

	x.owl		y.owl	
	Classes	Properties	Classes	Properties
Have a Match	25	30	40	25
Potential Mappings	47	45	50	45

Which terms of ontologies have candidate matches (for example, `restaurant`)

For each candidate match, how many potential mappings are there?
For example, "Restaurant in X" would potentially have mappings in Y to
• type of restaurant
• `Chinese-restaurant`

Figure 3. Example page of the decide-to-map spreadsheet. The page gives matching information in both numeric and graphical forms.

ically, as well as embed organizational policies and calculations related to decision making in the spreadsheet.

Figure 3 shows one of the spreadsheet pages. The page shows candidate matches of terms and composite terms in table and graphical format by type (that is, classes, data type properties, and object properties), indicating the number of terms having a candidate match in the other ontology. The "Candidate Matches to Potential Mappings" table and chart shows the number of potential mappings. The class term `Restaurant`, for example, might potentially map to a class term `Chinese-Restaurant` or the `Hotel` class property `type_of_restaurant`, showing that one candidate match could potentially have two mappings. So, this information can indicate the number of choices the user will need to consider during the mapping stage and help suggest the amount of effort that might be required.

We evaluated our process and software tools through two experiments (described in detail elsewhere[12]) and the ongoing use of OISIN in several research projects. The first experiment showed that the information generated by the characterization phase activities was useful in identifying mapping difficulties that might be involved in a mapping task. The second experiment showed that the mapping phase's semiautomated tools, artifacts, and guidance were useful in identifying mappings and structural mismatches. The tools also helped reduce the effort required and any mapping errors.

We're using the mappings generated from the tools in a wide range of domains where we're researching the achievement of dynamic semantic interoperability through semantic mappings. In the pervasive computing domain, for example, we use the mappings to support both the querying of context information over a range of heterogeneous and widely distributed information sources[13] and to support the dynamic composition of Web services.[14] In the telecommunications domain, we use the mappings to route, via a knowledge-based network (that is, a publish–subscribe network in which the routers use ontological reasoning), heterogeneous alarm messages, and configuration policies.[15] More recently, we've started to research how we can learn ordinary people's personal ontologies and their mappings to other ontologies through their interaction with Internet communication channels (such as RSS feeds and online communities) rather than through explicit engineering.

Our near-term goal is to offer roaming users an interaction experience identical to that they would experience in their native environment. This requires shielding users from semantic interoperability problems between personal applications and services in the roamed environment. Organizations have typically solved the semantic interoperability problem by adopting standards. However, with the increased diversity of services and a growing number of service providers, the semantic interoperability problem will likely become significantly worse, and commonly agreed-upon standards are unlikely to keep pace with the rapid pace of growth and innovation. Ontologies are one way to model application and service semantics that will allow for mapping specifications that can potentially overcome the semantic interoperability problem. Thus the need for ontology mapping is likely to increase in line with the growth in the number and diversity of applications and services.

It's unlikely that roaming users will undertake such ontology mapping themselves because current tools assume use by ontologically aware experts. Rather, roaming users will download the mappings to their devices either prior to travel (perhaps from their company's Web site) or upon first contact with a service provider in the new environment (such as in conjunction with the local tourist board). These mappings will be simple text documents that users can easily download and store. Another advantage of the ontology-based approach is that mappings can cater for language differences during roaming.

Thus, we envisage that in the near term, ontologically aware people will be the primary users of ontology-mapping tools. An organization with many roaming users might even deploy specialist mapping staff and computing resources. In contrast, a local tourist board might outsource automatic mapping management to a third-party IT support company.

Although researchers have made progress in the ontology-mapping field, several challenges remain before we can realize its benefit for roaming.

First, we need a commonly agreed practical ontology-mapping life cycle to stimulate the emergence of a vibrant market for software solutions to support ontology mapping. Our experience with OISIN has shown that having the process defined in UML provides tremendous flexibility for deployment in a variety of organizational types. Without such a practical process, the uptake by organizations involved in supporting roaming users will be insufficient.

Second, developers must agree on a common way to specify the results of matching algorithms and mapping systems. Without this agreement, the use of downloaded mappings by roaming users will quickly become limited.

Third, we need solutions to spot conflicts when mappings are downloaded into the roaming user's personal devices, especially when being downloaded from the service provider of a roamed environment.

We believe, however, that these challenges are surmountable, and a roaming experience that can easily cope with the wide variety of Web services and context information can be achieved through the use of ontology mappings.

References

1. N. Shadbolt, T. Berners-Lee, and W. Hall, "The Semantic Web Revisited," *IEEE Intelligent Systems*, vol. 21, no. 3, 2006, pp. 96–101.
2. D. Fensel, *Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce*, 2nd ed., Springer-Verlag, 2003.
3. P. Shvaiko and J. Euzenat, "A Survey of Schema-Based Matching Approaches," *J. Data Semantics*, vol. 4, no. 3730, 2005, pp. 146–171.
4. E. Rahm and P. Bernstein, "A Survey of Approaches to Automatic Schema Matching," *VLDB J.*, vol. 10, no. 4, 2001, pp. 334–350.
5. D. Aumüller et al., "Schema and Ontology Matching with COMA++," *Proc. 2005 ACM SIGMOD Int'l Conf. Management of Data*, ACM Press, 2005, pp. 906–908.
6. A. Kalyanpur et al., "Hypermedia Inspired Ontology Engineering Environment: Swoop," *3rd Int'l Semantic Web Conf.* (ISWC), poster session, 2004; www.mindswap.org/papers/SWOOP-Poster.pdf.
7. N. Noy and M. Musen, "Anchor-PROMPT: Using Non-Local Context for Semantic Matching," *Proc. IJCAI Workshop Ontologies and Information Sharing*, AAAI Press, 2001, pp. 63–70.
8. J. deBruijn, D. Foxvog, and K. Zimmerman, "Ontology Mediation Patterns Library," EU-IST Integrated Project IST-2003-506826 SEKT, D4.3.1, Feb. 2005; www.sekt-project.org/rd/deliverables/wp04/sekt-d-4-3-1-Patterns%20library.pdf.
9. A. Maedche et al., "MAFRA — A Mapping Framework for Distributed Ontologies in the Semantic Web," *Proc. Workshop Knowledge Transformation for the Semantic Web* (KTSW), *European Conf. Artificial Intelligence*, IOS Press, 2002, pp. 60–68.

10. J. Euzenat, “Evaluating Ontology Alignment Methods,” *Semantic Interoperability and Integration*, Dagstuhl Seminar Proc., Schloss Dagstuhl, 2005; http://drops.dagstuhl.de/opus/volltexte/2005/36/.
11. J. Euzenat, “An API for Ontology Alignment (version 1.3),” June 2005, http://co4.inrialpes.fr/align/align.pdf.
12. D. O’Sullivan, *The OISIN Framework: Ontology Interoperability in Support of Semantic Interoperability*, PhD thesis, Dept. of Computer Science, Trinity College Dublin, Univ. of Dublin, 2006.
13. R. Power and D. O’Sullivan, “Cashua: A Context Awareness Service for Heterogeneous Ubicomp Applications,” *Proc. 3rd Int’l Workshop Managing Ubiquitous Computing and Services* (MUCS), Cork Inst. Technology Press, in press.
14. D. O’Sullivan and D. Lewis, “Semantically Driven Service Interoperability for Pervasive Computing,” *Proc. 3rd ACM Int’l Workshop Data Eng. for Wireless and Mobile Access*, ACM Press, 2003, pp. 80–92.
15. D. Lewis et al., “Towards a Managed Extensible Control Plane for Knowledge-Based Networking,” *Proc. 17th IFIP/IEEE Int’l Workshop Distributed Systems: Operations and Management Large Scale Management* (DSOM 06), LNCS 4269, Springer, 2006, pp. 98–111.

Declan O’Sullivan is a lecturer and research team leader at Trinity College Dublin for the Higher Education Authority-funded M-Zones research program into the management of smart spaces, and for the Knowledge and Data Engineering Group’s (KDEG) involvement in the self-management activity of the Center for Telecommunications Value Chain Research (CTVR). His research interests are in telecommunications, their management, and their use of ontology-driven semantic interoperability. O’Sullivan has a PhD and an MSc in computer science from Trinity College Dublin. Contact him at declan.osullivan@cs.tcd.ie.

Vincent Wade is research director for KDEG in the Department of Computer Science, Trinity College Dublin and director of Trinity’s Center for Academic Practice and Student Learning (CAPSL). Wade has a Phd in computer science from Trinity College Dublin. He received a visiting scientist position in IBM’s Centre for Advanced Studies for his research in adaptive hypermedia and knowledge management. Contact him at vincent.wade@cs.tcd.ie.

David Lewis is a research lecturer at KDEG. His research interests include the engineering of open distributed systems for network and service management and the knowledge-driven engineering of autonomic pervasive computing and communication systems. Lewis has a PhD in computer science from University College London. Contact him at dave.lewis@cs.tcd.ie.