Dynamic Ontology Mapping for Interacting Autonomous Systems

Steven Heeps[1], Joe Sventek[1], Naranker Dulay[2], Alberto Egon Schaeffer Filho[2], Emil Lupu[2], Morris Sloman[2], Stephen Strowes[1]

[1] Department of Computing Science, University of Glasgow
heeps,joe,sds@dcs.gla.ac.uk
[2] Department of Computing, Imperial College London
n.dulay,aschaeff,e.c.lupu,m.sloman@imperial.ac.uk

Abstract. With the emergence of mobile and ubiquitous computing environments, there is a requirement to enable collaborative applications between these environments. As many of these applications have been designed to operate in isolation, making them work together is often complicated by the semantic and ontological differences in the meta-data describing the data to be shared. Typical approaches to overcoming ontological differences require the presence of a third party administrator, an approach incompatible with autonomous systems. This paper presents an approach to automatic ontology mapping suitable for deployment in autonomous, interacting systems for a class of collaborative application. The approach facilitates the collaboration of application-level data collections by identifying areas of ontological conflict and using meta-data values associated with those collections to establish commonality. A music sharing application has been developed to facilitate the sharing of music between peers.

1 Introduction

Recent advances in ubiquitous and mobile computing have dramatically changed the role of the computer in users' lives and made mobile computing the new personal computing and communication paradigm. The overriding motivation is that computing systems should seamlessly integrate into the life of the user and interoperate with other systems to offer mobile services as and when desired.

We have previously proposed the concept of a Self-Managed Cell (SMC) as the fundamental management design pattern for autonomous systems [20]; an SMC is a policy-based architecture that provides autonomic management capabilities for ubiquitous computing environments [3, 6, 10, 19]. In ubiquitous environments, SMCs need to collaborate without having a pre-agreed schema, and it is also desirable that there is agreement and common semantics for applications and devices. The SMC architecture currently supports integration at the system and management level where the basics for SMC interaction are handled in terms of policy, data and event exchanges [17]. Successful SMC integration

at this level provides the mechanisms for services at the application level to collaborate.

This paper explores the challenge of integration at the application level. Semantic differences between collaborating applications are usually managed by an administrator who maps the differences or documents a strict ontology to which systems developers and users adhere. It is likely that ontological and semantic differences between individual applications will prove a barrier to application collaboration due to the autonomous nature of the environment.

To explore application level ontology conflict and develop suitable mapping mechanisms, we have investigated the use of SMCs in the domain of peer-to-peer music sharing. The ability to see and listen to the music of others became prominent when Apple Inc. released a version of iTunes that supported the sharing of music collections on the same sub-network through the DAAP protocol [1]. This change, from music players as a single-user jukebox application to a tool for music sharing, brings with it the potential for further study, particularly with regards to the divergence of meta-data used to describe the tracks within each player. The following example highlights this problem: Bob and Alice have streaming access to each other's music collection. Bob loves "Indie" music, and searches for this in Alice's collection. Disappointingly, no matching tracks are found as Alice has not defined the genre "Indie", despite having a number of tracks that Bob would commonly classify as "Indie". There is a clear semantic difference in the way Bob and Alice define their music collections; whilst this is a standard feature of personal music collections, overcoming these differences automatically would undoubtedly enhance the users music sharing experience.

The paper is organised as follows: Section 2 describes the automatic ontology mapping mechanism; Section 3 discusses a prototype implementation in a peer-to-peer wireless music sharing environment; Section 4 presents related work, with conclusions and directions for future work presented in Section 5.

2 Automatic Ontology Mapping Mechanism

Seamless collaboration at the application level is difficult. It is unlikely that discovered services and applications will adhere to a common language or naming structure. It is likely that different devices and applications will originate from different vendors who use different semantic descriptions. Alternatively, semantics are user-defined and thus subject to great variation [18].

Ontologies are used to solve the semantic difference problem between application and application content. Ontologies capture knowledge of a given domain in a generic yet formal way, so that it can be reused and shared across applications and users. Ontologies are generally created via a man-made, time-consuming process where humans attempt to define all aspects of a system in a very explicit fashion. Frequently, different ontologies define very similar knowledge. Mapping between ontologies associates terms defined in one ontology with terms in another. Currently, such mappings are identified manually [15]. This is extremely resource intensive, not always possible and susceptible to ontology change. Au-

tomatic ontology mapping covers a large number of fields from machine learning and formal theory to database schema and linguistics. Applications also range significantly, from academic prototypes to large scale industrial applications [5]. Most systems are fairly complex, resource intensive creations and, as such, are not deployable in resource-limited, ubiquitous computing environments [11, 13].

To confirm the need for ontology mapping in the music player context, we analysed the music collections of 17 users comprising 64,704 songs. There were a total of 6,040 artists and 462 distinct music genres in the libraries studied. The existence of 462 distinct genres indicates immediately that there are going to be vast ontological differences between the music of only 17 peers. Apple's iTunes, for example, only contains approximately 30 different default genres, indicating that user-defined genres are very popular. The analysis also highlighted that approximately one third of all artists had more than one genre associated with them across the libraries. Table 1 shows six popular Artists from the libraries studied and the number of unique genres with which they were associated. This was apparent for all track meta-data, such as Track Size, Length, Album, Format and Artist.

Table 1. Genres Associated with Artists

Artist	Number of Unique Genres	Genres
Miles Davis	3	Alternative and Punk, Jazz , No Genre
Mozart	3	Classical, Classicism, Concerto
Marvin Gaye	4	Dance, Electronica, RandB, No Genre
Bob Dylan	6	Folk, Pop, Rock, Soundtrack, Various, No Genre
The Beatles	7	Alternative Rock, Dance, Electronica, Pop Rock, Rock and Pop, Rock and Roll, No Genre
Oasis	8	Alternative, Alternative and Punk, Alternative Rock Brit Pop, Pop, Punk, Rock, No Genre

2.1 The Basic Mechanism

We restrict our considerations to applications that manipulate data that conform to a common schema - i.e. the application expects to access a data collection that can be modelled as a relational table; each row of the table corresponds to one object (e.g. a musical track), and each column corresponds to a metadata attribute for that type of object (e.g. Genre, Artist); finally, one, additional column containing the value of the object is included in each row (e.g. the actual encoding of a musical track).

Using the music player example, the collection of tracks used by a particular player can be represented as shown in Table 2.

Each user is associated with a "home" collection of objects; in the music sharing example, it is the collection associated with the users music player; difficulty can ensue when the application has access to one or more "foreign"

Table 2. An Example Home Collection

Title	Artist	Composer	Genre	Album	Size(mb)	...	Value
Son	Jethro Tull	Ian Anderson	Rock	Benefit	2.77		mt000001.mp3
Black Hole Sun	SoundGarden	Chris Cornell	Grunge	Superunknown	5.02		mt000002.mp3
Exsultate, jubilate	Kiri Te Kanawa	Mozart	Classical		14.11		mt000003.mp3
Rusty Cage	Johnny Cash	Chris Cornell	Country	Unchained	1.31		mt000004.mp3
Hush	Tool		Metal	Opiate	1.30		mt000005.mp3
Sleeping	The Band		Country Rock	Stage Fright	3.11		mt000006.mp3
Hello	Evanescence		Gothic Rock	Fallen	3.48		mt000007.mp3
...							

collections in addition to the "home" collection. The user is most familiar with navigation through the "home" collection; in order to effectively access objects in the "foreign" collections, it is important to map the metadata values that describe the "foreign" objects into values that have meaning to the user.

In general, the metadata attributes exhibit correlated values within a collection - i.e. many objects with $attribute_i = value_i$ also have $attribute_j = value_j$. The degree of correlation between $attribute_i$ and $attribute_j$ will depend upon: the attributes chosen, the nature of the collection, and the degree of consistency in value assignment when objects are added to the collection. For example, most artists are strongly correlated with a particular genre (e.g. all tracks produced by Pearl Jam are associated with the Grunge genre), while release dates are only weakly correlated with a particular genre (e.g. Grunge is correlated with release dates 1990 and beyond, but not before). Such correlations can be asymmetric due to the fact that some attributes have broader scope than others; the correlation strength is a measure of the predictive power of one value over the value of the other (e.g. Pearl Jam strongly predicts Grunge, but Grunge predicts Pearl Jam, Soundgarden, Alice in Chains, etc.).

Consider a collection of N objects, and each object has M metadata attributes associated with it. Let us focus upon two attributes, i and j. In a particular collection, $Attr_i$ takes on values $v_{i1}...v_{in}$; similarly, $Attr_j$ takes on values $v_{j1}...v_{jn}$. We can then analyze all of the tracks in the collection to yield the following matrix (Table 3):

Table 3. Pairwise Classification of Objects in a Collection

$Attr_i / Attr_j$	V_{j1}	V_{j2}	...	V_{jn}
V_{i1}	C_{11}	C_{12}		C_{1n}
...				
V_{im}	C_{m1}	C_{m2}		C_{mn}

where C_{k1} is the number of objects in the collection that have $Attr_i = V_{ik}$ and $Attr_j = V_{jl}$. It is informative to consider two limiting cases:

1. $Attr_i$ is strongly correlated with $Attr_j$: in this case, if there are N_{ik} objects with $Attr_i = V_{ik}$, then most of those objects will have $Attr_j = V_{jl}$ for some l; note that by definition, $N_{ik} > 0$.

2. $Attr_i$ is not correlated with $Attr_j$: in this case, the N_{jk} objects with $Attr_i = V_{i,k}$ are distributed over many different values for $Attr_j$.

We can sum over the pairwise matrix in Table 3 to determine the predictive power of $Attr_i$ for $Attr_j$ as well as the predictive power of $Attr_j$ for $Attr_i$. One such formulation is as follows:

$$\text{predictive power}_{i,j} = \sum_{k=1}^{m} \frac{max_l\{c_{kl}\}}{\sum_{l=1}^{n} c_{kl}} \tag{1}$$

Obviously, the predictive $power_{j,i}$ simply requires that we swap k for l and m for n in Equation (1). Performing this analysis for all pairs of attributes yields a correlation matrix of the form shown in Table 4. The value in the i, j^{th} cell indicates how strongly correlated values of $Attr_i$ are to values of $Attr_j$; obviously, the diagonal elements have a value of 1. Armed with this correlation information for the home collection, we now describe a protocol that uses this mechanism to dynamically map objects from a foreign collection into the home object ontology.

Table 4. Predictive Power

	$Attr_1$	$Attr_2$	$Attr_3$	...	$Attr_M$
$Attr_1$	1.000	0.357	0.771		0.467
$Attr_2$	0.953	1.000	0.849		0.121
...				1.000	
$Attr_M$	0.125	0.294	0.186		1.000

2.2 The Mapping Protocol

The general protocol is as follows: if one is interested in objects in the foreign collection with $Attr_i = Value_i$, and none exist, then one searches the i^{th} column of Table 4 from the home collection for the $Attr_j$ with the largest correlation value (excluding row i). One can then query for objects corresponding to known $Value_j$'s, and discover the $Value_i$'s that the foreign collection associates with those objects. One can then import objects with those particular $Value_i$'s, replacing the actual $Value_i$ with the value used by the home collection.

Assume that two peers are sitting on a train, each with a personal music player in the form of a PDA hosting a music streaming service; the two players have discovered each other, and the policies in the two players permit streaming of tracks from one player to the other. Once the players have bound together, the music services on each player can enter into the ontology mapping protocol. Bob's music service remotely performs a genre search on Alice's system for each value of the genre meta-data attribute defined for Bob's system; for example, suppose that one value of the genre attribute is "Grunge". Unfortunately Alice does not have any music defined as "Grunge", so the initial query returns a

negative. The ontology mapping mechanism in Bob's music player selects a meta-data attribute strongly correlated with Genre, namely Artist, and queries Alice's player with a list of Artists associated with the genre "Grunge". Alice's music service then searches for those Artists in her collection, and returns the most-prevalent genre value, if any, associated with each artist in her collection. The protocol has established a Bob-specific mapping from his genre values to those used by Alice. Bob's music service can now represent tracks in Alice's system using Bob-specific genre values. Besides enabling comfortable navigation over the other individual's collection and subsequent streaming, the mapping information can also be retained for future sharing with each other, or possibly to inform future negotiations with other peers. The current protocol maps Bob's genre value to multiple genre values in Alice's collection. Another approach would be to only solicit the Alice genre value for the artist in Bob's collection with the largest number of tracks with that particular value, or the largest percentage of tracks with that particular value. The current approach maximises the number of tracks mapped to facilitate human navigation; more study is needed to determine if other approaches yield more usable results.

Table 5. Predictive Power of Music Tracks

	Genre	Artist	Name	Album	Year	BitRate	Kind
Genre	1	0.579	0.25	0.57	0.475	0.646	0.885
Artist	0.818	1	0.623	0.861	0.855	0.865	0.921
Name	0.908	0.946	1	0.912	0.905	0.939	0.941
Album	0.857	0.893	0.275	1	0.793	0.888	0.964
Year	0.283	0.259	0.139	0.256	1	0.376	0.462
BitRate	0.238	0.188	0.187	0.234	0.184	1	0.939
Kind	0.18	0.13	0.039	0.035	0.064	0.299	1

The mapping factor (attribute strongly linked to "Genre" in the preceding example) is determined through analysis of music collections. The application of Equation (1) to the meta-data from 17 unique iTunes music libraries yielded Table 5. The mapping factors for music collections indicate, for example, that there is a close relationship between Artist and Genre (0.818). In other words, if the Genre is not known then Artist is a good aspect of meta-data to map from, as is, Name and Album. Kind and Year, however, would not be suitable search attributes.

Even though our discussion is dominated by music sharing examples, other types of data collections are accessed in this way; for example, the collection of books maintained by a library. Initial results from a study of the meta-data for multiple book libraries also shows similar disparities across Subject Headings.

3 Experimental Validation

The Self-Managed Cell architecture running a music sharing service has been implemented as a test platform for our automatic ontology mapping technique.

The music sharing service utilises core SMC services such as the discovery and policy service.

The SMC has been built to run on a PDA (HP iPAQ hx4700, with a 624MHz XScale PXA270 processor and 64MB RAM, running Familiar Linux 0.8.4 or Windows Mobile 5.0). The SMC is written in Java, and uses JamVM 1.4.3 [8] in a bid to cut down on memory usage. The policy service used is Ponder2 written in Java 1.4. The music player, built to run as a service on an SMC, is also written in Java 1.4. The player enables a user to search the music collection of other discovered music players and stream music found from their search via wifi to their music player. It uses the DAAP [2] which performs as an HTTP server for advertising and streaming requested songs to clients. At present the music player has been successfully tested and functions successfully under J2SE. Currently attempts are being made to run the player on a PDA under Windows Mobile 5.0 using the Mysaifu JVM [14]. The music player is approximately 4Mb in size and has a memory footprint of around 15-30mb depending on activity status i.e. idle, playing, streaming etc. The music service relies upon the mechanism documented in [17] for establishing the initial peer-to-peer binding between a pair of music players running as services on SMCs.

The ontology mapping mechanism, as used to enhance collaboration between peer music libraries, has been fully tested and evaluated. Analysis of collaborations using the 17 peers documented in Section 2 revealed significant use of the mapping system, with song returns frequently running into the hundreds where initial collaboration had revealed few or no artists. Genre-to-Artist mapping results from a peer-to-peer collaboration are shown in Table 6. Only genre searches where no song results were initially returned are shown.

Table 6. Genre-Artist Mapping

Peer 1	**Peer 2 Returns after Mapping**		
Genre Request	Genres	Artists	Songs
Blues	2	337	3594
Classic Rock	2	282	2352
Electronica	1	115	587
Folk	2	282	2352
Rock/Pop	2	337	3594
Soul	1	11	109
Top 40	1	40	467

4 Related Work

Automatic Ontology mapping has seen a surge of research interest in recent years. Formal ontology mapping approaches have modelled ontologies using graphs, logic and models with mappings being developed from viewing graph, logic and model convergence [11, 13]. Current software systems that automatically generate ontology mappings are ONION [13], MAFRA [4] and IFF [16]. ONION generates mappings using graph transformations. MAFRA combines

different similarity measures, both lexical and structural, to establish the mappings. IFF is based on convergence between logical theories [5].

Such ontology mapping mechanisms are unlikely to be suitable for use in our ubiquitous environment. They have primarily been designed to provide automated administrative assistance when mapping well defined but conflicting ontologies in traditional conflicting environments. They require considerable user input and tend to focus on the use of a bridging ontology, a resource unlikely to be available in the ubiquitous world. Furthermore, the mapping mechanisms would likely struggle in the undefined and uncontrolled ubiquitous world. Most mechanisms are also not suitably lightweight so as to be deployable on resource limited devices.

Online music based Information Retrieval mechanisms are also gaining prominence. Last.fm [9], for example, leverages each user's musical profile to make personalised recommendations and connect users who share similar tastes. The downside of such mechanisms is the need for a common software plug-in and a network connection.

5 Conclusions and Future Work

A novel automated ontology mapping mechanism has been described that supports application-level integration within ubiquitous systems. The mechanism facilitates the successful collaboration of data collections by using meta-data contained within the collections to identify areas of commonality between them. The commonality identified is then used to automatically generate a common ontology and map between the areas of conflict. By using the meta-data information stored within music tracks, for example, we were able to successfully share music between peers despite there being no outwardly visible signs or commonality for collaboration. The techniques establish the beginnings of a common ontology and enabled a reference regarding the mapping to be held for future sharing. The system is suitably lightweight and resource efficient that it is capable of running in constrained environments such as PDAs and mobile telephones using our Self-Managed Cell architecture .

The current prototype uses exact string match during the mapping protocol. Given the anarchy that exists within some distributed collections we will investigate similarity matches between attribute values in an attempt to understand if this provides improved matching results. Likewise, future work will investigate enhancements to the quality of the mapping mechanism, particularly in relation to ranking results based on the probability a user will like them and will define how the mapping factors are regenerated over time.

6 Acknowledgements

The authors wish to thank the UK Engineering and Physical Sciences Research Council for their support through grants GR/S68040/01, GR/S68033/01 and GR/N15986/01.

References

1. Apple. ipod and itunes. http://www.apple.com/itunes, 2007.
2. C. Boot. Digital audio access protocol. http://daap.source forge.net/, 2007.
3. N. Dulay, S. Heeps, E. Lupu, R. Mathur, O. Sharma, M. Sloman, and J. Sventek. Amuse: Autonomic management of ubiquitous e-health systems. In Proceedings of the UK e-Science Al l Hands Meeting, UK, 2005.
4. Y. Kalfoglou and M. Schorlemmer. IF-map: an ontology mapping method based on information flow theory. Journal on Data Semantics, pages 98127, 2003.
5. Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. The Knowledge Engineering Review, 18(1):131, 2003.
6. S. L. Keoh, K. Twidle, N. Pryce, A. E. Schaeffer-Filho, E. Lupu, N. Dulay, M. Sloman, S. Heeps, S. Strowes, J. Sventek, and E. Katsiri. Forthcomming: Policy-based management for body-sensor networks. In 4th International Work- shop on Wearable and Implantable Body Sensor Networks, 2007.
7. L. C. Y. Kong, C. L. Wang, and F. C. M. Lau. Ontology mapping in per- vasive computing environment. In International Conference on Embedded and Ubiquitous Computing, pages 10141023, 2004.
8. R. Lougher. Jamvm. http://jamvm.sourceforge.net/, 2007.
9. Last.fm. http://www.last.fm, 2007.
10. E. Lupu, N. Dulay, M. Sloman, J. Sventek, S. Heeps, S. Strowes, K. Twidle, L. Keoh, and A. E. SchaefferFilho. Amuse: autonomic management of ubiquitous systems for e-health. Special Issues of the Journal of Concurrency and Computation: Practice and Experience, 2006.
11. A. Maedche, B. Motik, N. Silva, and R. Volz. A mapping framework for dis- tributed ontologies. In 13th International Conference on Knowledge Engineering and Knowledge Management, 2002.
12. Microsoft. Zune. http://www.zune.net, 2007.
13. P. Mitra, G. Wiederhold, and M. Kersten. A graph-oriented model for ar- ticulation of ontology interdependencies. In 7th International Conference on Extending Database Technology, 2000.
14. Mysaifu. Mysaifu. http://sourceforge.jp/projects/mysaifuj vm/, 2007.
15. N. F. Nay and M. A. Musen. Prompt: Algorithm and tool for automated ontology merging and alignment. In AAAI, 2000.
16. M. Romn, C. Hess, R. Cerqueira, A. Ranganathan, R. Campbell, and K. Nahrstedt. Gaia:a middleware infrastructure to enable active spaces. IEEE Pervasive Computing, pages 7483, 2002.
17. A. Schaeffer-Filho, E. Lupu, N. Dulay, S. Keoh, K. Twidle, M. Sloman, S. Heeps, S. Strowes, and J. Sventek. Supporting interactions between self-managed cells. Submitted to International Conference on Self-Adaptive and Self-Organizing Systems, 2007.
18. S.Heeps, N.Dulay, E.Lupu, A. E. Schaeffer-Filho, M.Sloman, S.Strowes, and J.Sventek. The autonomic management of ubiquitous systems meets the semantic web. In The Second International Workshop on Semantic Web Technology For Ubiquitous and Mobile Applications, 2006.
19. S. Strowes, N. Badr, N. Dulay, S. Heeps, E. Lupu, M. Sloman, and J. Sventek. An event service supporting autonomic management of ubiquitous systems for e-health. In 5th International Workshop on Distributed Event-Based Systems, 2006.
20. J. Sventek, N. Badr, N. Dulay, S. Heeps, E. Lupu, and M. Sloman. Self-managed cells and their federation. In CAiSE Workshops, volume 2, pages 97107, 2005.