Towards a Formalization of Mediating Connectors for on the Fly Interoperability

Romina Spalazzese, Paola Inverardi, Valérie Issarny

Towards a Formalization of Mediating Connectors for on the Fly Interoperability

Romina Spalazzese
Università degli Studi dell'Aquila
via Vetoio I-67100 L'Aquila, Italy
romina.spalazzese@di.univaq.it

Paola Inverardi
Università degli Studi dell'Aquila
via Vetoio I-67100 L'Aquila, Italy
paola.inverardi@di.univaq.it

Valérie Issarny
INRIA-Rocquencourt
Domaine de Voluceau
78153 Le Chesnay, France
valerie.issarny@inria.fr

***Abstract*—Mediators stand as a core architectural paradigm for today's and future systems that increasingly need to be connected. The mediator concept has been used to cope with many heterogeneity dimensions. Still, a key challenge for today's systems architectures is to embed the necessary support for automated mediation, i.e., the connector concept needs to evolve towards the one of mediating connector. In this paper, we introduce a framework to formalize mediating connectors. The proposed characterization paves the way for automated reasoning about protocol matching and mapping, and thus for the dynamic synthesis of mediating connectors to enable eternal networked systems, which we investigate as part of the CONNECT European project.**

***Keywords*-mediating connector; protocol interoperability.**

I. INTRODUCTION

Our everyday activities are increasingly dependent upon the assistance of digital systems that pervade our living environment. However, the current ubiquity [1] of digital systems is technology-dependent. The efficacy of integrating and composing networked systems is proportional to the level of interoperability of the systems' respective behaviors, from application- down to network-layers. This leads to a landscape of islands of networked systems, although interoperability bridges may possibly be deployed among them. As a matter of fact, middleware is yet another technological block, which also creates islands of networked systems. A number of systems have been introduced to provide middleware protocols interoperability [2], [3], [4], [5], [6]. Still, interoperability between networked software systems further concerns the systems' interfaces and behaviors at the application layer [7].

Mediator then stands as a core architectural paradigm for today's and future systems that increasingly need to be connected. The mediator concept was early introduced to cope with the integration of heterogeneous data sources [8], [9], and as design pattern [10]. However, with the significant development of Web technologies [11] and given abilities to communicate openly for networked systems, many heterogeneity dimensions shall now be mediated. Heterogeneity effectively spans [12]: terminology, representation format and transfer protocols, functionality and application-layer protocols. The first heterogeneity dimension is addressed by data level mediation that primarily relies on ontology integration [13], while the second relies on a combination of data level and protocol mediations. Functional mediation depends on the reasoning about logical relationships between the functional descriptions of networked systems (that are expressed in terms of pre- and post-conditions over the resources' states [14]) similar to the notion of behavioral subtyping [15]. Protocol mediation is further concerned with behavioral mismatches that may occur during interactions. Examples of approaches to solve protocol mismatches in software architectures are [16], [17].

A key challenge for today's systems architectures is to embed the necessary support for automated mediation, i.e., the connector concept needs to evolve towards the one of *mediating connector*. Indeed, the actual systems with which communications will take place cannot be anticipated at design time due to today's open networking and to further continuous evolution of networked systems. As such, connectors not only coordinate the interaction behaviors of connected systems but also mediate mismatching behaviors to enable actual interactions.

Automated mediation has deserved a great deal of attention especially in the context of Semantic Web technologies and of Web services technologies (that is certainly one of today's most popular and enabling architectures for networked resources) [18], [19], [20], [21]. However, most solutions are discussed informally, making it difficult to assess their respective advantages and drawbacks. They further remain rather vague on the definition of enforced matching relationship. What is needed is a new and formal foundation for mediating connectors from which protocol matching and associated mapping relationships may be rigorously defined and assessed. To the best of our knowledge, such an effort has not been addressed in the past although proposed algorithms for automated mediation manipulates formally grounded process models.

In this paper, we introduce a framework to formalize mediating connectors (Section II). The proposed description paves the way for automated reasoning about protocol matching and mapping and thus for on the fly synthesis of mediating connectors (Section III). We conclude by identifying future work (Section IV). Overall, the proposed framework for the formalization of mediating connectors constitutes a first step towards the dynamic synthesis of

mediating connectors to enable eternal networked systems, which we investigate as part of the CONNECT[1] European project.

II. A FRAMEWORK FOR MEDIATING CONNECTORS FORMALIZATION

The approach we are proposing is a first step towards the automatic generation of connectors for on the fly interoperability between mismatching protocols (that are observable at component interface). We consider the basic case of two protocols. The approach first abstracts the behavioral description of the mismatching protocols highlighting some *structural characteristics* [22], then checks the possibility for the two protocols to communicate, and finally (if possible) synthesises a mediating connector.

A. Protocols as LTS

With respect to protocol mediation, we are interested in: i) the identification whether two *protocols share similar intent*, and if this is the case, ii) the synthesis of the mediating connector that enables them to interoperate despite mismatches.

With "protocols share similar intent" we mean that, given two interaction protocols P_1 and P_2, parts of their behaviors are *complementary* thus showing an interaction potentiality. That is P_1 and P_2 show the same complementary behavior with respect to send and receive.

We use Labelled Transition Systems (LTS)[23] to formally describe interaction protocols, and mediating connectors. Moreover the concept of trace, which is a sequence of actions of a given LTS, is used to describe a specific protocol behavior. Figure 1 illustrates an example of interaction protocol.

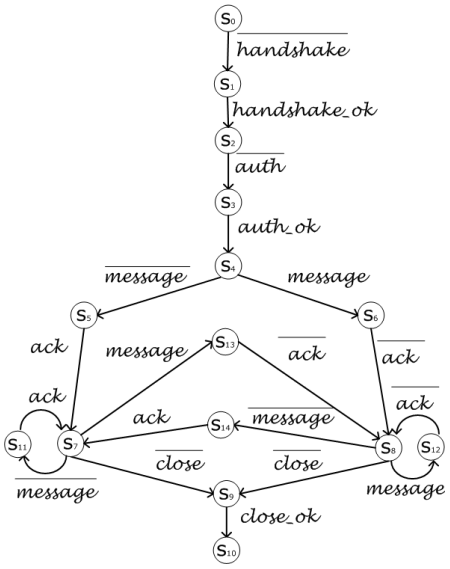

Figure 1. An example of interaction protocol.

[1]http://connect-forever.eu/

To establish i), that is if the protocols P_1, P_2 share similar intent, we first define the *abstracted LTSs* A_{P_1}, A_{P_2} of P_1, P_2 respectively, basing on structural characteristics. Then, we decide whether parts of A_{P_1}, A_{P_2} have complementary behaviors.

Figure 2 illustrate the actors of our approach and their relationships, that we will describe in the following.

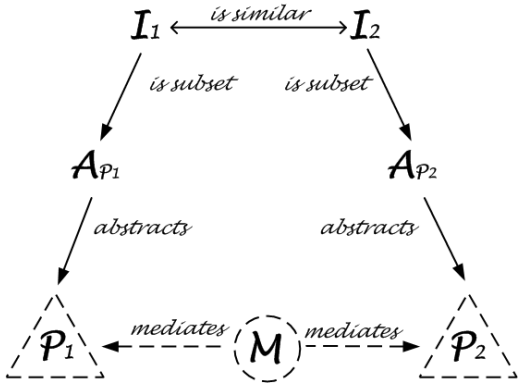

Figure 2. The actors of the approach and their relationships.

Given a protocol P, the abstracted LTS A_P, called structure in [22], is built by maintaining only a subset of *rich* states of P, and by collapsing into single transitions the traces of P between such states. Rich states are *branch states, entry cycle states, convergence states, and initial and final states*. Thus, the transitions of A_P may be labelled with sequence of actions of P. Note that at this stage we do not consider the LTS labels while abstracting but we just look at the LTS control structure.

The matching relationship, taking into consideration also the labels, defines the conditions to check in order to answer the above issue whether parts of A_{P_1}, A_{P_2} have complementary behaviors.

B. Protocol matching and mapping

To single out the parts of the abstracted LTSs A_{P_1}, A_{P_2} that are expected to be complementary, i.e., the parts that can possibly carry on the actual communication between P_1, P_2, we have to determine if there exists a common language between A_{P_1} and A_{P_2}. In order to do this we need an ontology mapping. We assume to have an ontology mapping O between the labels of P_1, P_2.

The *common language* between A_{P_1} and A_{P_2} is made by the set of labels that have the same meaning through O. In the open world setting we are assuming, the portions of the protocols that have not a common language are the parts of the protocols that synchronize with third entities.

We call *induced LTS* of A_{P_1} (A_{P_2}), the sub-LTS of the LTS labelled with labels belonging to the common language between A_{P_1} and A_{P_2}. We denote with I_1, I_2 the induced LTS of A_{P_1}, A_{P_2} respectively. Thus, we expect that I_1

and I_2 exibit complementary behaviors. Finally, in order to establish if the two protocols have complementary behaviors, we use a suitable equivalence relation that we call *similarity*. Informally this relation succeeds if the structure of A_{P_1} and A_{P_2} is the same and labels on each transition are complementary through O.

In essence, if the complementary check succeeds this means that it is possible to let P_1 and P_2 communicate. The problem is thus transferred into finding a suitable mediating connector, that coordinates P_1 and P_2, making their interoperability possible.

More precisely in the following we formally describe the above concepts.

Let L_1 and L_2 be the two alphabets of the protocols P_1 and P_2 respectively, and let the intersection between L_1 and L_2 be an empty set. We assume that actions in the LTS are only the observable (external) actions of the protocol (i.e., we do not consider its internal actions). We adopt the usual convention that an action represented by the label α synchronizes with the action $\overline{\alpha}$. Moreover, we use the usual progress property that the parallel composition of two LTS evolves only by means of synchronization.

A *mediated matching* between P_1 and P_2 exists if and only if: (i) the induced LTS I_1 of A_{P_1} is *similar* to the induced LTS I_2 of A_{P_2}, and (ii) there exists a mediator M which is a refinement of both I_1 and I_2.

Thus, supposing that I_1 is similar to I_2, the issue that remains is to identify, a protocol M that refines A_{P_1} and A_{P_2}. In order to solve this, the ontology has a central role because it maps one or more actions of P_1 into one or more actions of P_2.

As said before, we are considering a networked environment in which two mismatching protocols P_1 and P_2 want to interoperate. Actually, the environment can be populated by many protocols, and P_1 and P_2, during their evolution, can interact also with the remainder of the environment in order to reach a suitable state in which they are ready to synchronize with each other.

We therefore consider that the mediator, besides forwarding the synchronization messages between the two protocols (i.e., the ones belonging to the common language), also provides the needed complementary behaviors to P_1 and P_2 to let them evolve.
Hence, as far as the communication in between P_1 and P_2 is concerned, we can consider as τs, the actions that have to be exchanged with third parties.

In other words, the mediator M, while mediating between P_1 and P_2, implements a sort of hiding operation by providing to P_1 and/or P_2 the complementary portions that allow them to evolve and reach the right state from which they can synchronize with each other. The states of P_1 and P_2 from which they can synchronize are elicited from the common language. Indeed, if an action a_1 of P_1 is a synchronization action with a_2 of P_2, a_1 and a_2 are in the common language. We call *corresponding action* an action that is in both protocols, has the same meaning through O and if it is overlined in one protocol then is non-overlined in the other.

Due to the possibility for P_1 and P_2 to synchronize with third parties, both the ontology and the mediator are important. The former identifies the portions of the two protocols for mutual synchronization and identifies also the states that P_1 and P_2 have to reach in order to synchronize. The latter simulates the complementary protocols of portions of P_1 and P_2 leading them to their mutual synchronization states. It also "tunnels" the messages between the protocols to let them interoperate.

Summarizing, given two protocols, the protocol mapping checks if they match and ends either with the mediator abstract specification (if it exists) or by demonstrating its non existence (and thus the impossibility for P and Q to interoperate).

The mediator specification M forms the basis for the subsequent synthesis of its actual behavior (the actual implementation of M).

III. Towards mediating connector synthesis

The synthesis of a mediating connector between two protocols that want to interoperate is built, as presented in Section II-B. In addition to the actual synchronization between P_1 and P_2 it also contains portions of complementary protocols for both P_1 and P_2 for the portions of them that is not in the induced LTSs.

Given the mediator specification M, the process of automatically synthesize the actual code for M exploits the framework defined in [24].

IV. Conclusion

With the networked environment being increasingly open and dynamics, the seamless composition and related connection of systems becomes a prime requirement. However, the openness of the environment comes along with great heterogeneity in the networked systems, which challenges their connection. Mediating connectors then appear as a paradigm of choice to effectively overcome behavioral mismatches among the protocols run by the networked systems that need to coordinate. However, while the mediator paradigm has deserved much attention over the last few years, including research towards supported automated mediation, key principles remain loosely defined.

This paper proposes a first formal characterization of the mediating connector concept. The approach is a first attempt, it only partly covers the existing mismatches, and needs to be extended to cover a larger set. Future work concerns several investigation areas among which an assessment of the generality of the proposed framework and the realization of

semi-automatic support to the elicitation and the subsequent synthesis of the mediator.

ACKNOWLEDGMENT

The work is partly supported by: the CONNECT European project No 231167 of the Future and Emerging Technologies (FET) programme within the ICT theme of the Seventh Framework Programme for Research of the European Commission, the Italian PRIN D-ASAP, and French project INRIA associated team ASA. Romina Spalazzese would also like to thank Scuola Superiore Sant'Anna and ISTI CNR for presently hosting her.

REFERENCES

[1] M. Weiser, "The computer for the 21^{st} century," *Scientific American*, Sep. 1991.

[2] J. Nakazawa, H. Tokuda, W. K. Edwards, and U. Ramachandran, "A bridging framework for universal interoperability in pervasive systems," in *ICDCS '06: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2006, p. 3.

[3] Y.-D. Bromberg and V. Issarny, "Indiss: interoperable discovery system for networked services," in *Middleware '05: Proceedings of the ACM/IFIP/USENIX 2005 International Conference on Middleware*. New York, NY, USA: Springer-Verlag New York, Inc., 2005, pp. 164–183.

[4] D. Chappell., *Enterprise Service Bus*. O'Reilly, 2004.

[5] P. Grace, G. S. Blair, and S. Samuel, "A reflective framework for discovery and interaction in heterogeneous mobile environments," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 9, no. 1, pp. 2–14, 2005.

[6] Y.-D. Bromberg, V. Issarny, and P.-G. Raverdy, "Interoperability of service discovery protocols: Transparent versus explicit approaches," in *Proceedings of the 15th IST Mobile & Wireless Communications Summit*, Mykonos, June, Jun. 2006.

[7] V. Issarny, M. Caporuscio, and N. Georgantas, "A perspective on the future of middleware-based software engineering," in *FOSE '07: 2007 Future of Software Engineering*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 244–258.

[8] G. Wiederhold and M. Genesereth, "The conceptual basis for mediation services," *IEEE Expert: Intelligent Systems and Their Applications*, vol. 12, no. 5, pp. 38–47, 1997.

[9] G. Wiederhold, "Mediators in the architecture of future information systems," *IEEE Computer*, vol. 25, pp. 38–49, 1992.

[10] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Resusable Object-Oriented Software*. Addison-Wesley Professional, 1995.

[11] D. Fensel and C. Bussler, "The web service modeling framework wsm," *Journal of Electronic Commerce Research and Application*, vol. 1, no. 1, pp. 113–137, 2002.

[12] M. Stollberg, E. Cimpian, A. Mocan, and D. Fensel, "A semantic web mediation architecture," in *In Proceedings of the 1st Canadian Semantic Web Working Symposium (CSWWS 2006*. Springer, 2006.

[13] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," *SIGMOD Rec.*, vol. 33, no. 4, pp. 65–70, 2004.

[14] M. Stollberg, E. Cimpian, and D. Fensel, "Mediating capabilities with deltarelations," in *In Proceedings of the First International Workshop on Mediation in Semantic Web Services, co-located with the Third International Conference on Service Oriented Computing (ICSOC 2005*, 2005.

[15] B. Liskov and J. Wing, "Behavioral subtyping using invariants and constraints," 1999. [Online]. Available: citeseer.ist.psu.edu/article/liskov99behavioral.html

[16] B. Spitznagel and D. Garlan, "A compositional formalization of connector wrappers," in *ICSE '03: Proceedings of the 25th International Conference on Software Engineering*. Washington, DC, USA: IEEE Computer Society, 2003, pp. 374–384.

[17] D. M. Yellin and R. E. Strom, "Protocol specifications and component adaptors," *ACM Trans. Program. Lang. Syst.*, vol. 19, no. 2, pp. 292–333, 1997.

[18] R. Vaculín and K. Sycara, "Towards automatic mediation of owl-s process models," *Web Services, IEEE International Conference on*, vol. 0, pp. 1032–1039, 2007.

[19] R. Vaculín, R. Neruda, and K. P. Sycara, "An agent for asymmetric process mediation in open environments." in *SOCASE*, ser. Lecture Notes in Computer Science, R. Kowalczyk, M. N. Huhns, M. Klusch, Z. Maamar, and Q. B. Vo, Eds., vol. 5006. Springer, 2008, pp. 104–117.

[20] H. R. Motahari Nezhad, B. Benatallah, A. Martens, F. Curbera, and F. Casati, "Semi-automated adaptation of service interactions," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*. New York, NY, USA: ACM, 2007, pp. 993–1002.

[21] S. K. Williams, S. A. Battle, and J. E. Cuadrado, "Protocol mediation for adaptation in semantic web services," in *ESWC*, 2006, pp. 635–649.

[22] R. Spalazzese, P. Inverardi, and V. Issarny, "A formalization of mediating connectors: Towards on the fly interoperability." Tech. Rep., 2009. [Online]. Available: http://www.di.univaq.it/romina.spalazzese/publications.html

[23] R. M. Keller, "Formal verification of parallel programs," *Commun. ACM*, vol. 19, no. 7, pp. 371–384, 1976.

[24] M. Tivoli and P. Inverardi, "Failure-free coordinators synthesis for component-based architectures," *Sci. Comput. Program.*, vol. 71, no. 3, pp. 181–212, 2008.