

Automated Synthesis of CONNECTORS to support Software Evolution

AmelBennaceur, Paola Inverardi, ValérieIssarny, Romina Spalazzese

Today's software systems are increasingly networked and are further characterized by ever-changing functionalities provided to/required from the networked environment because of the wide variety of dynamically available heterogeneous applications.

To manage evolution in this context, the CONNECT project pursues the automated synthesis of CONNECTORS enabling continuous composition and interoperability of applications.

Software evolution is a hot topic in today's distributed software systems, which are characterized by an increasing level of heterogeneity and dynamism. For instance, updated versions of software are continually made available, new applications are created and dynamically join the networked environment, old or broken applications are unavailable or removed, and new needs emerge.

Evolution management in this environment raises the need for novel computing paradigms able to ensure continuous run-time composition and interoperability of heterogeneous and dynamically available applications in an automated manner.

Heterogeneous applications populating this landscape, although in principle could interact since they have *compatible* (i.e., complementary) *functionalities*, can be characterized by *discrepancies* that may undermine their ability to seamlessly *interoperate* (i.e., communicate and coordinate). Discrepancies include *incompatible interaction protocols* and/or *different interfaces* meaning *different actions* and/or *data models*.

Examples of heterogeneous applications are shown in Figure 1(a) where the light blue entities show independently developed client and server for trips organization. The server exhibits separate searches for flight and hotel while the client would expect to have a single search with all the information.

To enable perpetual and automated composition and interoperability, the ICT FET IP European project CONNECT pursues the automated synthesis of CONNECTORS that drop the interoperability barriers. This responds to the evolution of functionalities provided to and required from the networked environment of today's software systems. Automatically synthesized CONNECTORS are concrete *emergent* entities that *mediate* the application discrepancies, i.e., translate and coordinate mismatching interaction protocols, actions, and/or data models, letting applications interact effectively. Examples of CONNECTORS are illustrated in the Figure 1(b) -green entities.

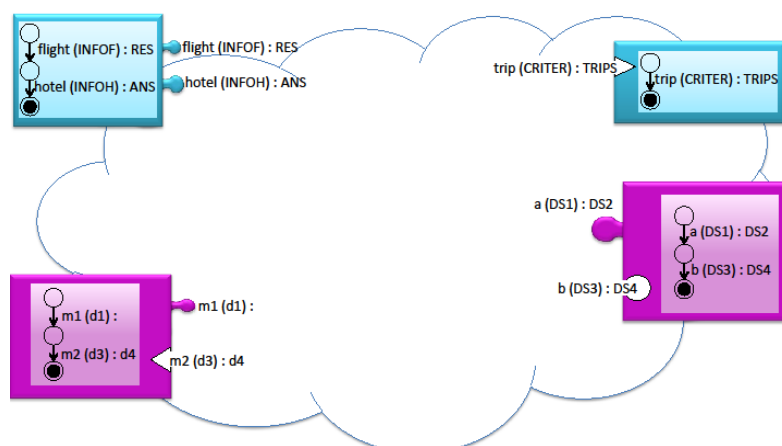


Figure 1(a). Examples of heterogeneous networked applications

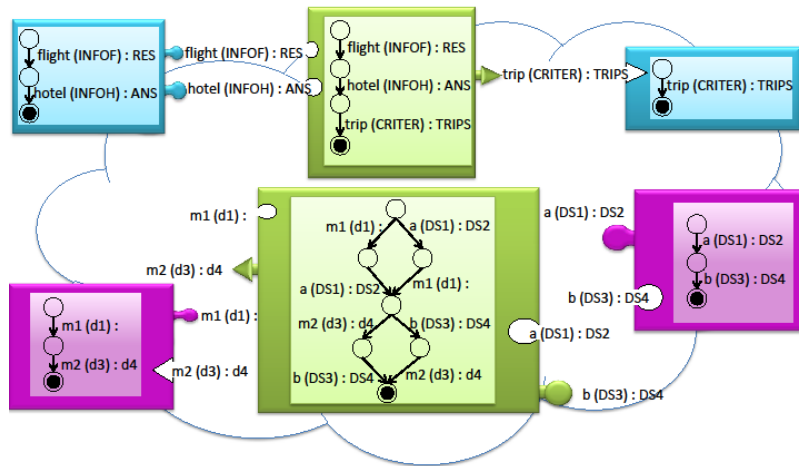


Figure 1(b). Example of CONNECTORS enabling composition and interoperability in the heterogeneous networking environment

Automated Synthesis of CONNECTORS.

To formally ground and automatically reason on the interoperability problem among heterogeneous applications, we developed a *theory* for the automated synthesis of application-layerCONNECTORS. The theory also served as principled basis for the development of a *prototype tool* that automatically generateCONNECTORS.

A TheoryofCONNECTORS.

Our theoryfor the automated synthesis of CONNECTORSassumes the descriptions of two Networked Systems (NSs) focusing on their application-layer protocol. Then,it defines a reasoning process on the provided protocols that produces as output the behavior of a CONNECTORthat let heterogeneous applications interact by mediating their discrepancies.

Specifically, the description of aNSs encompasses the following models:

- *interface definition*, in terms of actions required and provided by the NS, together with associated input/output data;
- *interaction protocol specification*, i.e., the behavioral automaton describing the process according to which actions from the interfaces can be invoked; and
- *ontological description*ofactions and datawhich refers to concepts of a widely shared application domain ontology. This offers a means to automatically and dynamically reason on the semantic interoperability of applications for which we do not have a *priori* knowledge.

TheCONNECTORtheory reasoning decomposes into three phases or steps: *Abstraction,Matching andMapping*.

- (1) *Abstraction* makes the Networked Systemmodels *comparable* and, if possible, reduces their size thus allowing to ease and speed up the reasoning on them.
- (2) *Matching* checks the Networked Systems*compatibility* identifying possible interaction protocols incompatibility and/or actions differences and/or data models differences preventing seamless interoperation.

- (3) *Mapping* (or *Synthesis*). A successful matching check identifies possible discrepancies preventing proper coordination of networked applications. Mapping solves such mismatches, leading to identify the existence of a CONNECTOR. Consequently, this step produces a proper CONNECTOR that interposes between application protocols and, while managing heterogeneity, allows applications to effectively exchange compatible sequences of actions with data.

Future perspectives.

This article has outlined our theory of CONNECTORS that supports software evolution and more specifically eternal interoperability of networked systems through on-the-fly synthesis of CONNECTORS.

This work is part of the larger CONNECT project effort, that revisits the middleware paradigm to face the challenges raised by the increasingly dynamic and heterogeneous distributed systems. Indeed, today's and even more future distributed systems, exhibit a complexity that is by far larger than the one tackled by state of the art middleware. Hence, the project focuses on sustaining interoperability in an ever diverse and continuously changing networking environment. As a result, CONNECT introduces the paradigm of "*emergent middleware*" that are dynamically created, thanks to supporting enablers for protocol synthesis but also learning. Emergent middleware is further grounded on the emerging Models@runtime paradigm due to the extensive use of networked systems models at runtime to be able to learn and reason about, as well as compose the protocols that are executed.

USEFUL LINKS

<http://www.connect-forever.eu/>

<http://www.di.univaq.it/romina.spalazzese/publications.html>

CONTACT ADDRESS

Romina Spalazzese

romina.spalazzese@univaq.it

romina.spalazzese@gmail.com

Valérie Issarny (CONNECT Project Co-ordinator)

Valerie.Issarny@inria.fr