Knowledge-driven Architecture Composition: On the usage of case-based formalization methods for reliable and automated component coupling

Fabian Burzlaff [1]

Abstract: Automating component coupling has been around for various decades. In fact, in the last few years' interface and component matching progress seems not to be regarded as a hot-topic in research. However, reappearing paradigms such as decentralized and flexible production scenarios are again in need for automated system coupling. This is mainly due to the increasing number of heterogeneous devices. Building upon existing component integration research, this PhD project introduces case-based reasoning techniques for formalizing integration knowledge to overcome standardization requirements. As a consequence, integration knowledge becomes reusable.

Keywords: Knowledge-driven architecture composition, Interoperability, Semantic Integration, Dynamic Adaptable Systems, Industrial Internet of Things

1 Motivation and Problem statement

In current trends, such as Industrial Internet of Things (IIoT), information technology is postulated to merge with classical automation techniques. For example, so called "Cyber-Physical Systems" are therefore equipped with high-level programming interfaces. Current information modelling frameworks like OPC-UA[2] provide means to create industrial information models and establish a secure communication between heterogeneous machines. However, industry also still relies on informal integration standards. Especially integration tasks in automation scenarios are still executed mostly by humans as soon as syntax and/or semantics of two interfaces differ. Enforced by the rising demand for connecting decentralized production machines and executing dynamic, adaptable processes with low batch sizes, manual integration tasks may slow down or even hinder future process automation.
For long time, solutions for automated component coupling have been proposed in service-based and component-based research communities [Va16] [Pl16] [Vi07]. On the one hand, semantic service descriptions with formalized semantics such as W3C Recommendation for "Semantic Annotations" for Web-Service-Description-Language (e.g. SAWSDL[3] or WSDL-S[4]) have already been tested in various applications scenarios [Vi07]. On the other

[1] University of Mannheim, Institute for Enterprise Systems, L15 1-6, 68131 Mannheim, burzlaff@es.uni-mannheim.de
[2] https://opcfoundation.org/about/opc-technologies/opc-ua/
[3] https://www.w3.org/TR/sawsdl/
[4] https://www.w3.org/2005/04/FSWS/Submissions/17/WSDL-S.htm

doi:10.18420/in2017_240

hand, interface matching techniques and component models (e.g. CORBA[5]) have also produced a lot of valuable results [Va16] [Pl16]. In addition, other approaches for component discovery exist. Hummel [Hu08] has developed test-driven techniques for semantically identifying general software components.
Nevertheless, completely formalizing a service or interface (e.g. pre- and post-conditions or quality attributes) requires a distinct amount of manual work. As component providers can hardly anticipate for which usage-context a component will be requested, such formalization techniques quickly consume (too) much work with no direct benefit for component providers (i.e. their semantics must be "fully" formalized for each possible customer use-case). However, by interconnecting more and more devices industry is particularly in need for an automated and reliable component coupling mechanism to meet increasing market- and customer-demands.

Hence, there is a dilemma between huge efforts for creating complete, (un)-standardized semantic component specification and the rising amount of required point-to-point integrations (e.g. realized by adapters). Based on the assumption that an overall semantic standardization for certain industry branches (e.g. plant engineering) opposed to other branches (e.g. AUTOSAR[6] for automotive) is not feasible, current academic interface formalization approaches are not practical. This PhD project explicitly allows for incomplete, case-based interface formalization and can be summarized as:

> "How can software components be semantically coupled in an automated way based on partially incomplete integration knowledge?"

2 Solution Approach

Due to the increasing speed of technological innovations, it will be hard to come up with up-to-date semantic interoperability standards. Case-based reasoning (CBR) methods seems to be a reasonable approach for tackling the problem of slow semantic standardization processes from upside down [AgPl94]. However, as solution for new integration cases are only loosely coupled to previous solutions using similarity measures like "nearest neighbour" (c.f. Figure 1), a high reliability cannot be achieved within the traditional approach. This means that input and output descriptions in the knowledge base (KB) must be expressive but also reliably decidable (**RQ1**).

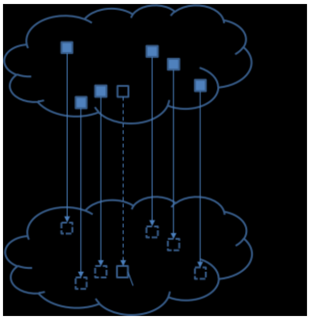
Figure 1: Problem-Solution Space

Although ontologies for meeting high reliability demands in production scenarios seem to be a promising approach for capturing integration cases, it is not clear whether deductive, closed-world reasoning approaches to derive new integration rules are usable (c.f. Fig. 2 - Revise). Similarity measures are not applicable for case revision as automated component coupling can only take place based on secured facts. Thus, efficiently deriving integration rules and KB consistency checks must be evaluated (**RQ2** and **RQ3**).

[5] http://www.omg.org/spec/CCM/
[6] https://www.autosar.org/

In order to reuse derived integration knowledge for automated component coupling of previously unknown components, a suitable software engineering approach is needed. This engineering approach must provide means for transforming declarative integration knowledge into imperative – at best even into deployable - code (**RQ4**).
As a restriction, we do not focus on technical or protocol specific integration issues as our approach assumes a suitable middleware (e.g. an Enterprise Service Bus). Furthermore, fuzzy interface matching techniques can be used for component suggestion but not for automating component coupling. Available adaptation techniques can be reused, but must be adapted for processing declarative integration rules (c.f. Fig.2 - Solution)

Despite the need for manual component integration executed by humans in the beginning, inserting formalized integration knowledge evolutionary in a suitable KB can result in automated component coupling without depending on slow standardization process. This can ultimately result in a minimized, manual point-to-point integration

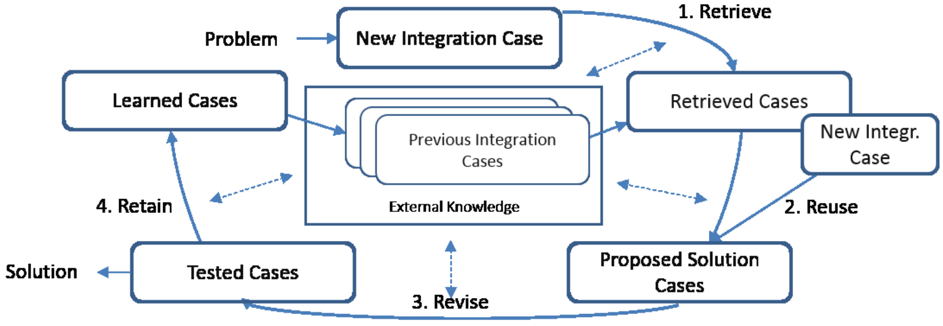

Figure 2: Traditional Case-based Reasoning Cycle, adapted from[AaPl94]

3 Related work

To tackle the proposed problem, several mature research streams can be considered. For this PhD project, especially CBR, schema matching for integration scenarios and component adaptation are currently regarded as relevant.

3.1 Case-based reasoning methods

CBR and its foundations were first mentioned by Aamodt and Plaza in 1994 [AgPl94]. They introduced a problem-based learning method that derives solution for new problems based on similar and already solved problems. As cases are retrieved based on a continuous similarity measure, proposed solution may only be partially correct. Although domain-specific adaptation knowledge can be learned and thus improve component adaptation [CNR06], the proposed solution must always be verified during application. Thus, deductive inference mechanisms (e.g. Pellet[7]) have been incorporated into the CBR

[7] https://github.com/stardog-union/pellet

process. As a result, case definitions as well as case retrieval have been significantly improved (i.e. high similarity measure) [BM03]. CBR supported by ontologies has been successfully applied in the field of medicine and biology, for instance to correctly diagnose multiple cancer types [Be11].

Although distinct component integration tasks can theoretically be interpreted as cases as well, explicitly defined integration rules, which can be reasoned about, are not present in the traditional CBR approach. Furthermore, the intended functionality to reason about the semantics of integration rules explicitly in a reliable way is new as *these* semantics are typically implicitly defined in similarity measures or domain-specific adaptation rules.

3.2 Schema matching for integration scenarios

Schema-matching techniques for information models evolved from database communities into various other information engineering discipline. For example, ontologies are often used for information integration scenarios as they offer features to reason about the semantics of schema elements [Wa01]. Furthermore, Niepert et al. [Ni11] developed an approach for ontology matching where semantic characteristics are already considered during the matching process. The logic- and probabilistic-based computation of the most "equal" ontology could be further improved by interactive integration tools [SNF12]. Researchers from component-based communities have dealt with different levels of formalized semantic service descriptions. Platenius [Pl16] developed a tool-based approach for combining different matching techniques for arbitrary formalized service specifications. They propose the concept of "fuzzy matching" for dealing with uncertainty where the produced results are then further processed by the user (i.e. service integrator).

Both, information integration based on ontologies as well as fuzzy service specification matching do need user input during their matching task to deal with uncertainty. However, this is not possible for automated component coupling scenarios.

3.3 Component adaptation

As soon as a component mismatch occurs (e.g. syntactic signature mismatch), engineering approaches for component adaptation are needed. These are typically executed by a system integrator or automated to a certain degree [BOR04] [HA10]. Becker et al. [Be06] describes basic steps of an engineering approach for component adaptation. They distinguish between the mismatch types "Technical-Signature-Protocols-Concepts" as well as "Quality Attributes". Next, they propose different imperative programming patterns for each mismatch type (e.g. Adapter). Hummel et al. [Hu08] [HA10] introduced an approach for automatically generating all syntactically feasible adapters for a provided and required interface constellation and select the "the most semantically" correct one by running test cases. This approach can already solve mismatches types like naming of methods or ordering of parameters.

Although these approaches propose multiple solutions to re-occurring integration problems, no sophisticated solution for automated and reliable component adaptation “on-the-fly” is known. Currently, (in-) formal standards, if available, are used for automating component coupling in dynamic adaptive systems. Otherwise, time-consuming point-to-point adapters must be programmed.

4 Research challenges

In order to tackle our research problem, we focus on the following research questions:

RQ1. What characteristics must a suitable description language fulfil for capturing integration cases as well as integration rules (i.e. representation of integration rules and integration cases)?

RQ2. How can integration rules be derived from an integrated ontology (i.e. not on feature-similarity but on reliable case facts)?

RQ3. What are suitable knowledge-management techniques for updating partially-incomplete integration KBs (e.g. interfaces with same syntax but different semantics)?

RQ4. Which steps must an integrator fulfil to reuse formalized integration knowledge?

5 Status quo and next steps

Our approach for case-based formalization of integration knowledge making integration knowledge reusable has been presented at ICSA [BB17]. Next, we will focus on a literature review answering **RQ1** and **RQ2** (until end of 2017). **RQ3** will deal with test-based assessments to ensure KB consistency, but not knowledge-management techniques in general (until mid of 2018). A suitable engineering approach will be constructed and evaluated by case studies (**RQ4**). Exemplary evaluation criteria are “Time to formalize integration knowledge against existing approaches” or “Covered integration cases” for measuring the expressiveness of our language and the amount of integration cases needed for automated component coupling. The technical feasibility of this approach will be estimated by a prototypical implementation (until end of 2019).

References

[AaPl94] Aamodt, Agnar, and Enric Plaza. "Case-based reasoning: Foundational issues, methodological variations, and system approaches." AI communications 7, no. 1 (1994): 39-59.

[BB17] Burzlaff, Fabian, and Christian Bartelt. "Knowledge-Driven Architecture Composition: Case-Based Formalization of Integration Knowledge to Enable Automated Component Coupling." In Software Architecture Workshops (ICSAW), 2017 IEEE International Conference on, pp. 108-111. IEEE, 2017.

[Be06] Becker, Steffen, Antonio Brogi, Ian Gorton, Sven Overhage, Alexander Romanovsky, and Massimo Tivoli. "Towards an engineering approach to component adaptation." In Architecting Systems with Trustworthy Components, pp. 193-215. Springer Berlin Heidelberg, 2006.

[Be11] Begum, Shahina, Mobyen Uddin Ahmed, Peter Funk, Ning Xiong, and Mia Folke. "Case-based reasoning systems in the health sciences: a survey of recent trends and developments." IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 41, no. 4 (2011): 421-434.

[BM03] Bergmann, Ralph, and Martin Schaaf. "Structural case-based reasoning and ontology-based knowledge management: A perfect match?." J. UCS 9, no. 7 (2003): 608-626.

[BOR04] Becker, Steffen, Sven Overhage, and Ralf H. Reussner. "Classifying software component interoperability errors to support component adaption." In International Symposium on Component-Based Software Engineering, pp. 68-83. Springer Berlin Heidelberg, 2004.

[CNR06] Craw, Susan, Nirmalie Wiratunga, and Ray C. Rowe. "Learning adaptation knowledge to improve case-based reasoning." Artificial Intelligence 170, no. 16-17 (2006): 1175-1192.

[HA10] Hummel, Oliver, and Colin Atkinson. "Automated creation and assessment of component adapters with test cases." In International Symposium on Component-Based Software Engineering, pp. 166-181. Springer Berlin Heidelberg, 2010.

[Hu08] Hummel, Oliver. "Semantic component retrieval in software engineering." PhD diss., Universität Mannheim, 2008.

[Ni11] Niepert, Mathias, Jan Noessner, Christian Meilicke, and Heiner Stuckenschmidt. "Probabilistic-logical web data integration." In Reasoning Web. Semantic Technologies for the Web of Data, pp. 504-533. Springer Berlin Heidelberg, 2011.

[Pl16] Platenius, Marie Christin. "Fuzzy matching of comprehensive service specifications." PhD diss., Dissertation, Paderborn, Universität Paderborn, 2016, 2016.

[SNF12] Stuckenschmidt, Heiner, Jan Noessner, and Faraz Fallahi. "A Study in User-centric Data Integration." In ICEIS (3), pp. 5-14. 2012.

[Va16] Vale, Tassio, Ivica Crnkovic, Eduardo Santana de Almeida, Paulo Anselmo da Mota Silveira Neto, Yguaratã Cerqueira Cavalcanti, and Silvio Romero de Lemos Meira. "Twenty-eight years of component-based software engineering." Journal of Systems and Software 111 (2016): 128-148.

[Vi07] Vitvar, Tomas, Adrian Mocan, Mick Kerrigan, Michal Zaremba, Maciej Zaremba, Matthew Moran, Emilia Cimpian, Thomas Haselwanter, and Dieter Fensel. "Semantically-enabled service oriented architecture: concepts, technology and application." Service Oriented Computing and Applications 1, no. 2 (2007): 129-154.

[Wa01] Wache, Holger, Thomas Voegele, Ubbo Visser, Heiner Stuckenschmidt, Gerhard Schuster, Holger Neumann, and Sebastian Hübner. "Ontology-based integration of information-a survey of existing approaches." In IJCAI-01 workshop: ontologies and information sharing, vol. 2001, pp. 108-117. 2001.