Semantic Service Matchmaking in the ATM Domain Considering Infrastructure Capability Constraints

Thomas Moser, Richard Mordinyi, Wikan Danar Sunindyo, Stefan Biffl

Institute of Software Technology and Interactive Systems, Vienna University of Technology
Favoritenstrasse 9-11/188, Vienna, Austria
{thomas.moser, richard.mordinyi, wikan.sunindyo, stefan.biffl}@tuwien.ac.at

***Abstract*—In a service-oriented environment business processes (BPs) flexibly build on software services (SSs) provided by systems in a network. A key design challenge is the semantic matchmaking of BPs and SSs in two steps: 1. Find for one BP the SSs that meet or exceed the BP requirements; 2. Find for all BPs the SSs that can be implemented within the capability constraints of the underlying network, which poses a major problem since even for small scenarios the solution space is typically very large. In this paper we analyze requirements from mission-critical BPs in the Air Traffic Management (ATM) domain and introduce an approach for semi-automatic semantic matchmaking for SSs, the "System-Wide Information Sharing" (SWIS) BP integration framework. A tool-supported semantic matchmaking process like SWIS can provide system designers and integrators with a set of promising SSs candidates and therefore strongly reduces the human matching effort by focusing on a much smaller space of matchmaking candidates. We evaluate the feasibility of the SWIS approach in an industry use case from the ATM domain.**

I. INTRODUCTION

Safety-critical systems and business processes, e.g., in the Air Traffic Management (ATM) domain, have to become more flexible to implement changes due to new business environments (e.g., mergers and acquisitions), new standards and regulations. A promising approach follows the service-oriented architecture (SOA) paradigm that builds flexible new systems for business processes (BPs) based on a set of software services (SSs) provided by system nodes in a network. A key design challenge is the matchmaking of BPs and SSs, i.e., finding the SSs that a) best meet the requirements of the BPs under consideration and b) can be implemented with the available network capabilities. The solution space is typically large even for small problems and a general semantic solution to enable comprehensive tool support seems infeasible.

To provide a SOA solution for a set of BPs, meaning to identify suitable SSs for BPs, designers and system integrators need to overcome 3 integration challenges that build on each other:

1. Technical integration connects networked systems that use heterogeneous technologies, i.e., different protocols, operational platforms, etc. Current technical integration approaches like Enterprise Service Bus (ESB) [2] or Service Oriented Architecture (SOA) [17] need manual configuration on the technical detail level and tool support is typically focused on a single technology or vendor.

2. Semantic integration translates data content and format between systems that use heterogeneous semantics, i.e., different terminologies for service names, data formats, etc. For semantic integration, there is no standard or framework available, making the semantic transformations between multiple services inefficient and expensive.

3. Business process support builds on technically and semantically integrated systems that provide SSs the BP needs to fulfil its goal. The system integrator has to select SSs that really match the requirements of the BP, and check whether the infrastructure capabilities can support the communication requirements of the chosen solution.

Large BP and SS integration networks consist of hundreds of integration nodes; changes of SS properties and network capabilities make the correct and efficient identification of feasible BP and SS pairs a recurring complex and error-prone task. Current service matchmaking approaches focus on either technical or semantic integration issues [27, 29], while business process support is, to our knowledge, missing. Tool support for matchmaking of BPs and SSs need to make the requirements of BPs and SSs as well as the capabilities of SSs and the underlying infrastructure understandable for machines.

In previous work, we introduced a systems integration approach, the "system-wide information sharing" (SWIS) approach. The SWIS framework explicitly models the semantics of integration requirements and capabilities in machine-understandable form (semantic integration) [21]; and the connectors and transformations between heterogeneous legacy systems (technical integration) [19] to simplify systems integration (business process support) [20].

In this paper, we describe the semantic matchmaking of BPs and SSs and the optimization of the integration solution with respect to available network capabilities. Semantic matchmaking uses the machine-understandable SWIS models to describe BP and SS requirements and SS and network capabilities to derive 2 results: 1. Provide sets of possible SSs for each BP; 2. Optimize the set of selected SSs with multiple objectives (e.g., costs, delay) while observing the capabilities of the underlying network infrastructure, a variation of the knapsack problem [14]. We evaluate the feasibility of the SWIS approach in a use case from the ATM domain.

The remainder of this paper is structured as follows: Section 2 summarizes related work, Section 3 motivates the research issues, while Section 4 describes the use case. Section 5 elaborates the semantic service matchmaking approach and the optimization of the integration solution. Section 6 evaluates the approach and Section 7discusses the results with regard to the research issues. Finally, Section 8 concludes and identifies further work.

II. Related Work

This section summarizes related work on technical integration, semantic integration with semantic web services, and service matchmaking with multi-objective optimization.

A. Technical Integration

Technical system integration is the task to combine networked systems that use heterogeneous technologies to appear as one big system. There are several levels at which system integration could be performed [1], but there is so far no standardized integration process that explains how to integrate systems in general.

The need for integration over heterogeneous middleware technologies with different APIs, transportation capabilities, or network architecture styles implies either solutions like ESB [2] and SOA [17] or the development of static and therefore inflexible wrappers between each combination of middleware technologies, and thus increases the complexity of communication.

B. Semantic Integration with Semantic Web Services

Typical integration solutions focus only on either the semantic heterogeneity (i.e., heterogeneity on service level, e.g., heterogeneous service descriptions, message data fields or service policies) or technical heterogeneity (i.e., heterogeneity on network level, e.g., different technologies or protocols). Technical integration technology approaches provide syntactical transformation between services, while the semantic heterogeneity of services could be addressed by means of a common data schema [7]. The derived limitations for such kinds of integration approaches are on the one hand the need for a common data schema [7], which is often a hard and time consuming procedure, if not even impossible in integration scenarios with several different stakeholders.

Semantic integration is solving problems originating from the intent to share data across disparate and semantically heterogeneous data sources [7]. These problems include the matching of ontologies or schemas, the detection of duplicate entries, the reconciliation of inconsistencies, and the modelling of complex relations in different sources [24]. One of the most important and most actively studied problems in semantic integration is establishing semantic correspondences (or mappings) between vocabularies of different data sources. [3]

Goh [6] identified three main categories of semantic conflicts in the context of data integration that can appear: confounding conflicts (e.g., equating concepts are actually different), scaling conflicts (e.g., the use of different units for the same concept), and naming conflicts (e.g., synonyms and homonyms). The use of ontologies as a solution option to semantic integration and interoperability problems has been studied over the last 10 years. Ontologies promise to provide machine-understandable representation of knowledge, while allowing the mapping between certain facts as well as the derivation of new facts using reasoning approaches based on the modeled knowledge [28]. In a general domain, semantic integration has shown to be very hard if not unsolvable. However, in a specialized domain, like the ATM domain, semantic integration seems doable.

Noy [23] identified three major dimensions of the application of ontologies for supporting semantic integration: the task of finding mappings (semi-)automatically, the declarative formal representation of these mappings, and reasoning using these mappings. There exist two major architectures for mapping discovery between ontologies. On the one hand, the vision is a general upper ontology which is agreed upon by developers of different applications. On the other hand, there are approaches comprising heuristics-based or machine learning techniques that use various characteristics of ontologies (e.g., structure, concepts, instances) to find mappings. These approaches are similar to approaches for mapping XML schemas or other structured data.

In SOA the promise of Web Services and the need for widely accepted standards enabling them are by now well recognized [9]. In particular, the Web Services Description Language (WSDL)[1] is already well established as an essential building block in the evolving stack of Web Service technologies, allowing the specification of the syntax of the input and output messages of a basic service, as well as of other details needed for the invocation of the service. WSDL does not, however, support the specification of workflows composed of basic services. In this area, the Business Process Execution Language for Web Services (BPEL4WS)[2] has the most prominent status. With respect to registering Web services, for purposes of advertising and discovery, Universal Description, Discovery and Integration (UDDI)[3] has received the most attention to date.

At the same time, recognition is growing of the need for richer semantic specifications of Web Services, so as to enable fuller, more flexible automation of service provision and use, support the construction of more powerful tools and methodologies, and promote the use of semantically well-founded reasoning about services. Furthermore, richer semantics can help to provide fuller automation of activities as verification, simulation, configuration, supply chain management, contracting, and negotiation of services. [15]

To meet this need, researchers have been developing languages, architectures and related approaches for so called Semantic Web services [16]. The Ontology Web Language for Services (OWL-S)[4], which seeks to provide the building blocks for encoding rich semantic service descriptions in a way that builds naturally upon the Web Ontology Language (OWL)[5], supplies Web Service providers with a core set of markup language constructs for describing the properties and capabilities of their Web Services in unambiguous, computer-interpretable form [4]. OWL-S mark up of Web Services facilitates the automation of Web Service tasks, including automated Web Service discovery, execution, composition and interoperation. WSDL-S [18] is another approach for annotat-

[1] http://www.w3.org/TR/2001/NOTE-wsdl-20010315
[2] http://www.ibm.com/developerworks/library/specification/ws-bpel/
[3] http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm
[4] http://www.daml.org/services/owl-s/1.1.
[5] http://www.w3.org/2004/OWL/

ing current Web Service standards with semantic descriptions. In WSDL-S, the expressivity of WSDL is enriched with semantics by employing concepts similar to those in OWL-S [8], while being agnostic to the semantic representation language.

The Web Service Modeling Ontology (WSMO)[6][13] is a framework for Semantic Web Services which refines and extends the Web Service Modeling Framework (WSMF) to a meta-ontology for Semantic Web services. WSMF defines a rich conceptual model for the development and the description of Web Services based on two main requirements: maximal decoupling and strong mediation.

All three approaches, OWL-S, WSDL-S and WSMO, provide mechanism for semantically describing Web Services, with the major goal of allowing generic description of service functionality as well adding semantics to general service descriptions like provided/consumed messages or service bindings. This ambitious goal seems very useful for generic service descriptions; however its usage is limited in specific domains like in the ATM domain, since too specific features would complicate a generic approach too much. Therefore, we defined our own ontology-based architecture for describing the properties and features of the ATM services [21].

C. Service Matchmaking Approaches

Software components discovery and (Web) Service discovery can be classified into two categories: signature matching and semantic matching. Signature matching is an efficient means for software components retrieval, but two software components with similar signatures may have completely different behaviours. Semantic matching addresses this problem by comparing software components based on formal descriptions of the semantics of their behaviours. Semantic matchmaking can be seen as major feature of semantic integration which supports designers and system integrators by providing sets of possible integration partners regarding both structural and semantic attributes. However, the relevant semantic concepts are hard to define unambiguously for general domains, thus the focus on a well-defined domain like ATM provides semantic clarity.

Zaremski and Wing [29] extend their signature-matching work with a specification-matching scheme. Semantic matching identifies suitable services more precisely than signature-matching methods, but the cost of formally defining provided and required services is considerable. Paolucci et al. [26] propose a DAML-S based approach for a declarative description of web services outside the representation capabilities of UDDI and WSDL. They provide an upper-level ontology of service profiles consisting of service actors, functional service attributes, and function service descriptions.

Both approaches [26, 29] provided useful hints how a semantic service matchmaking framework could be implemented. We used the concept of identifying suitable service candidates based on the semantics of their provided/consumed messages, which we modelled using OWL. Additionally, we mapped each single segment of all messages to the particular concept of the domain knowledge it represents, allowing a more powerful identification of semantically equal message segments [19, 22]. Furthermore, we used multi-objective optimization [5] to rank the service candidate matches.

Kolovski et al. [10] provide a mapping of WS-Policy to OWL. WS-Policy[7] provides a general purpose model and syntax to describe the policies of a Web service. It specifies a base set of constructs that can be used and extended by other Web service specifications to describe a broad range of service requirements and capabilities. The main advantage of representing Web Service policies using OWL is that OWL is much more expressive than WS-Policy and thus provides a framework for exploring richer policy languages. Verma et al. [27] present an approach for matching the non-functional properties of Web Services represented using WS-Policy. To date, most policy matching has used syntactic approaches, where pairs of policies are compared for structural and syntactic similarity to determine compatibility. In their approach, the authors enhance the policies of a Web Service with semantics by creating the policy assertions based on terms from ontologies. Oldham et al. [25] present a framework for the matching of providers and consumers based on WS-Agreements. The WS-Agreement specification[8] defines a language and protocol for capturing the relationship with agreements between two parties. An agreement between a service consumer and a service provider specifies one or more service level objectives (SLO) which state the requirements and capabilities of each party on the availability of resources and service qualities.

Both WS-Policy and WS-Agreement define a generic framework for the representation of standard Web Service policies, however both frameworks seem too generic to be effectively used in a concrete scenario from a specialized domain like the ATM domain is. Therefore, we used the concept of describing Service policies using a knowledge representation language like OWL, but defined our own extendable policy representation language which is better suitable for the ATM domain [19, 21].

III. RESEARCH ISSUES

Recent projects with industry partners from the ATM domain raised the need for semi-automated BP integration support in technology-driven integration environments. Recently, we developed a data-driven approach [20] that explicitly models the semantics of the problem space, i.e., BP integration requirements and network infrastructure capabilities [21]; the solution space, i.e., the connectors, and data transformations between SSs [19]. Finally, we provide a process to bridge problem and solution spaces, i.e., identify feasible BP and SSs pairs while fulfilling business requirements and optimizing the chosen integration solution according to multiple objectives.

Figure 1 provides an overview on the integration layers, data flows between the integration layers, and the steps of the semantic service matchmaking process: SM1: For each BP, identify the suitable SSs sets, which fulfil all BP service and

[6] http://www.wsmo.org/

[7] http://www.w3.org/Submission/WS-Policy/

[8] www.ogf.org/documents/GFD.107.pdf

data requirements. From these possible BP and SSs sets, the system integrators choose the most promising sets, the so-called collaboration sets. SM2: The selected collaboration sets are then optimized regarding the original infrastructure requirements of both the business BPs and the SSs, as well as the available limited capabilities of the infrastructure's nodes and links. The outcome of SM2 is an optimized configuration of the integration solution, consisting of the selected collaboration sets as well as their grounding to the underlying integration network infrastructure.

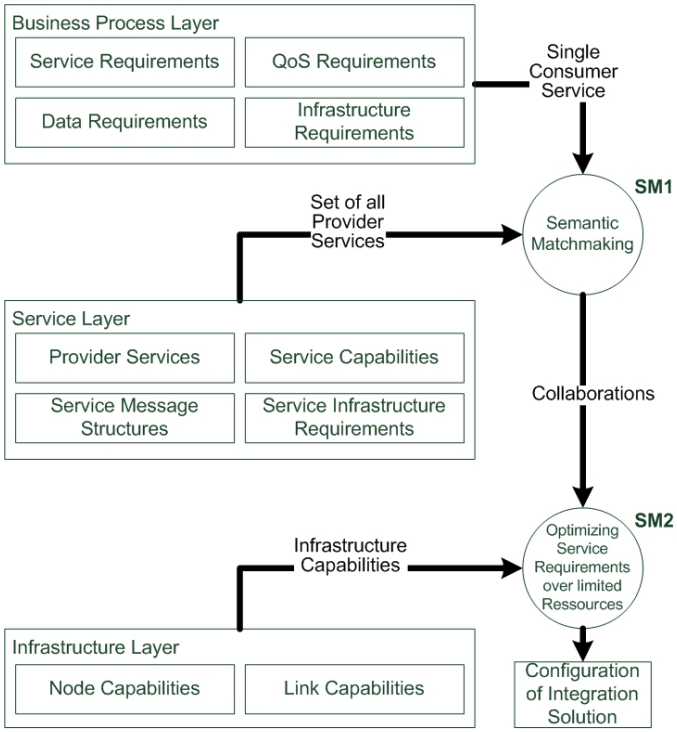


Figure 1: Semantic Service Matchmaking Process Steps.

Based on this process, we derive the following research issues:

RI-1: Semantic Matchmaking of SS candidates for one BP (*SM1*). Provide machine-understandable descriptions for BP and SSs requirements as well as SS and network capabilities to provide tool support for SM1 to make the search space reduction effective (low number of false negatives and false positives) and efficient (less human effort required) compared to the current human-based approach.

RI-2: Resource Feasibility Check and Optimization for all Collaborations (*SM2*). Provide a framework to enable a) checking the validity of a set of BPs and SSs with the infrastructure capability constraints and b) ranking valid solutions by multiple optimization criteria like network cost and service delay.

IV. ATM Scenario Description

This section describes the integration scenario from the ATM domain used throughout this paper. The ATM use case (Figure 1) represents information that is typically extracted from customers/participants in workshops on requirements and capabilities elicitation for information systems in the aviation domain. In safety-critical domains like ATM BP integration solutions have to pass certifications before deployment, which typical dynamic SOA solutions [2, 17] cannot fulfil regarding the current rigid integration network in the ATM domain designed to guarantee integration requirements even in case of partial failure.

The ATM domain consists of several hundreds of services, like the business system Air Traffic Management Information Service (ATMIS), which has to provide information services about flights to business partners via a Public Flight Information Portal (PFIP). On the other hand ATMIS needs to collect and refine information from at least 2 other services: the Central Flight Controller (CFC) and the Single Flight Data Processors (SFDPs). With respect to service matching, each data provider, in our case CFC and SFDPs, defines (see section V) the content and the format of the data it can provide and the quality of service, e.g. the frequency of incoming data such as radar signals. Similarly, each data consumer, in our case ATMIS, defines its needs for data content, the format it can handle, and quality of service. It may additionally require conditions such as data coming from a defined geographical area and within a defined time window.

In the ATM domain semantic matchmaking is an effort for scarce human experts who have to cope with a huge search space and often miss better solutions due to their simple heuristic search strategies. Tool-supported semantic matchmaking provides designers and system integrators with a set of promising integration partner candidates and therefore strongly reduces the human matching effort by focusing on a much smaller space of feasible matchmaking candidates that can be rated according to relevant optimization criteria.

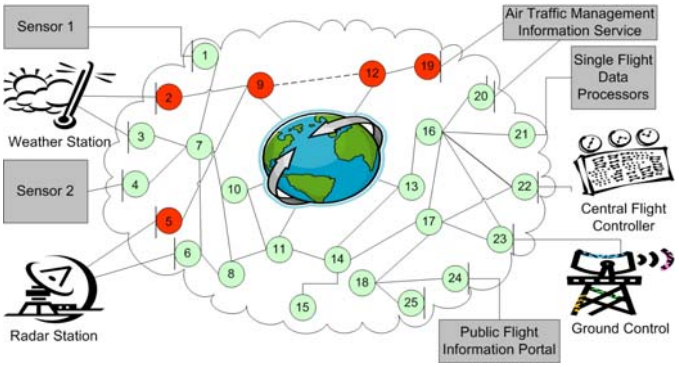


Figure 2: A Typical ATM Domain Integration Network.

As shown in Figure 2, the integration network consists of business services connected to integration network nodes. Between these nodes, there may exist different kinds of network links using different transmission technologies (e.g., radio or wired transmission) as well as different middleware technologies (e.g. SONIC, TIBCO, or IPX-based middlewares) for communication purposes. The capabilities of nodes and links, like throughput, availability, reliability, or security are explicitly modelled in order to be capable of selecting suitable communication paths for particular service requirements, e.g., the communication link between the red ATMIS Node and the

red Node 12 represents a reliable and secured communication path which may be requested by e.g., the ATMIS business service.

The high number of distributed legacy applications with heterogeneous interfaces to their services, given service requirements on reliability, timeliness, safety, service quality, failover, performance, auditability, maintainability, and flexibility, and the need to dramatically improve the capability of agile reaction to any kind of changes due to altered business needs in order to provide new ways of systems integration in a safety-critical environment, requires a platform like SWIS (other and more detailed aspects can be found in [19-22]).

V. Semantic Service Matchmaking

This section describes the semantic service matchmaking approach as well as the multi-objective optimization of the chosen integration services candidates. For every BP, the set of possible SSs providing the required information is calculated. These sets of a BP and at least one SS are called collaboration candidate sets. The system integrators and designers choose one or if applicable more desired collaborations from these collaboration candidates. Then the optimal integration system configuration for these active collaborations is calculated.

A. Identification of Possible Collaboration Candidate Sets

The identification of possible collaboration candidate sets is implemented as a heuristic algorithm. Step by step, the possible collaboration candidate sets are reduced by applying the rules described to the possible collaboration candidate sets. The heuristic rules that are applied during the source/sink matching are described in the following paragraphs.

Message mapping. During the description of the SS messages, each SS message segment was mapped to a domain concept, which has been specified in the common domain ontology. Therefore, for all segments of the message required by a certain BP, it is searched for messages of the SSs that contain segments, which are mapped to the same domain concept, and if possible, to the same message format.

Service Policies. In addition, SSs can define requirements (policies) regarding preferred or unwanted SS partners, as well as other non-functional requirements, e.g., QoS requirements regarding the underlying integration network. A policy is a restriction or a condition for a single collaboration or a set of collaborations, in order to allow the communication via the underlying integration network. In SWIS-based applications, there are two kinds of policies. On the one hand, there are policies which are valid for all collaborations. They specify global conditions that need to be fulfilled by all collaborations, e.g., a maximum time for the delivery of messages. On the other hand, there are policies which are required only for a specific subset of collaborations. These policies specify conditions that need to be fulfilled by the collaborations containing particular SSs, e.g., the communication has to use only secure links, or only a specified set of other SSs is allowed to participate in the collaboration. The SS policies that regard other SSs are evaluated by checking whether the attributes and tags of every SS of the particular collaboration candidate meet the service policies defined by the BP.

Format Translation. If a message segment is mapped to the same domain concept as the required message segment, but the formats of the two segments differ, check whether there is a converter defined for the two formats. A converter is used to convert the format of message segments from one basic data type to a different one. An explicit identifier is defined to allow the search for the converter at runtime (e.g., by using Java Reflection).

External Service Transformation. If the message segments differ in the domain concept they are mapped to, check if a service exists that consumes a segment mapped to the same domain concept as the segment of the message of the SS and provides a message with a segment mapped to the same domain concept of the segment of the message of the BP.

Route Deduction. As last rule it is checked whether there is an existing route between the nodes connecting the SSs and the node connecting the BP.

If all the rules mentioned above are successfully applied to a set of one or more SSs and a BP, then the particular set is accepted as collaboration candidate. If any of the rules cannot be met, the particular set is discarded as collaboration candidate.

B. Validity-Check and Optimization of Collaborations

Once all collaborations have been specified a Scenario is derived. A Scenario contains beside all collaborations a specification detailing how to configure the network infrastructure, so that the integration solution is optimized according to the given objectives. In order to optimize the configuration the process step needs a network infrastructure model (nodes, links and their properties like cost, delay, capacity and supported policies), the set of collaborations, the size of the messages to be exchange at each collaboration, global policies to be taken into account, and a list of parameters, like cost and delay, which are used as objectives for optimization purposes. In the following the process steps needed to optimize the scenario is explained.

Preliminary Checks. The process step checks whether there is at least one single network route for each collaboration satisfying all global and collaboration specific policies. If this step cannot be completely satisfied the process raises an exception. The system integrator either updates or removes the collaborations which cannot be mapped to a network route, and restart the process step, or adapts the semantic infrastructure model, by adding additional nodes and links.

Route Derivation. Once it has been verified that each collaboration can be mapped to at least one route in the network, the process step derives every possible route for each collaboration. The only restrictions are that no node is allowed to appear twice within the same route and all policies have to be satisfied. The valid ones are retained; the ones violating the restrictions are removed. At the end of this process step, each collaboration will have either a single route or a set of valid routes to choose from.

Creating Scenarios. The processing step combines each route of each collaboration with each other. This means that a scenario consists of a set of collaborations where each collaboration represents exactly one route. The more scenarios are created, the higher the probability to find a scenario that is well suited for achieving the stated optimization objectives.

Evaluation. The process iterates through all scenarios and calculates their fitness according to the optimization objectives. The fitness of a scenario is the fitness of all its containing collaborations, and represents the real values (e.g. the time a message needs and the costs along the chosen route) of the objectives. The fitness represents the trade-off of the configuration, the routes of each collaboration predetermine. The set of fitness values is then analyzed according to the Pareto Front approach [5]. The Pareto Front contains either a single Scenario or a set of Scenarios. In the latter case there may be several "nearly equivalent" configurations as integration solutions. Thus, the system integrator has to decide which one to pick for practical deployment.

Multi-Objective Optimization. We have accomplished the process of optimizing collaborations by implementing a Java version of the mPOEMS approach into the SWIS framework. mPoems is an evolutionary algorithm using the concept of dominance for multi-objective optimization. The results and explanations of the approach can be found at [11, 12].

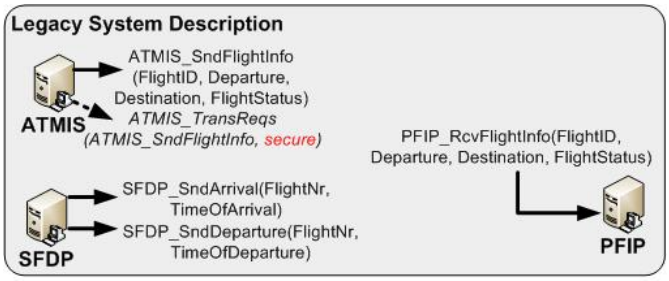
VI. Evaluation

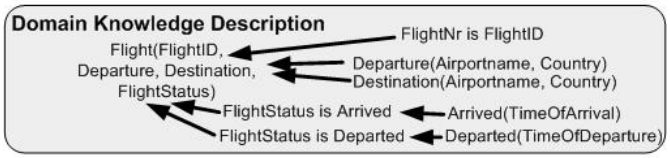
In this section, we evaluate the SWIS framework using a clear and comprehensible example to show the principles of our approach. We then identify key effort indicators and extrapolate them to a large example taken from an industry project to show the feasibility and strength of the SWIS framework.

An example for semantic service matchmaking in the SWIS framework is shown in Figure 3. There are three services of provided by legacy systems, two provider services (*ATMIS* and *SFDP*) and one consumer service (*PFIP*). The consumer service needs information that can be obtained from the provider services, i.e. *FlightID, Departure, Destination and FlightStatus*. This needed information is provided separately by the two provider services, so the system has to find the suitable information that match with the consumer service's needs. Additionally, the service *ATMIS_TransReqs* defines a policy for the underlying integration network, stating that only secure network links may be used for the communication.

Suppose the provider services provide the following data. The flight number is LH6351, the departure is a pair of airport code and country code (VIE,AT), the arrival is a pair of airport code and country code (CDG,FR), the time of departure is December 14th 2008 19:55 and the time of arrival is December 14th 2008 22:00. This would be represented using the following tuples:

```
ATMIS_SndFlightInfo(LH6351,(VIE,AT),(CDG,FR),
  Departed(2008-12-14-19:55)).
SFDP_SndDeparture(LH6351,2008-12-14-19:55).
SFDP_SndArrival(LH6351,2008-12-14-22:00).
```





Service Matchmaking

Provider Services	Consumer Service	Transformation
FlightNr	FlightID	[FlightNr → FlightID]
(Airportcode, Countrycode)	Departure(Airportname, Country)	Departure([Airportcode →AirportName], [Countrycode → Country])
(Airportcode, Countrycode)	Destination(Airportname, Country)	Destination([Airportcode →AirportName], [Countrycode → Country])
TimeOfArrival	Arrived(TimeOfArrival)	[TimeOfArrival → Arrived(TimeOfArrival)]
TimeOfDeparture	Departure (TimeOfDeparture)	[TimeOfDeparture → Departure (TimeOfDeparture)]
YYYY-MM-DD-HH:NN	HH:NN-DD-MM-YYYY	*{time formatting}*

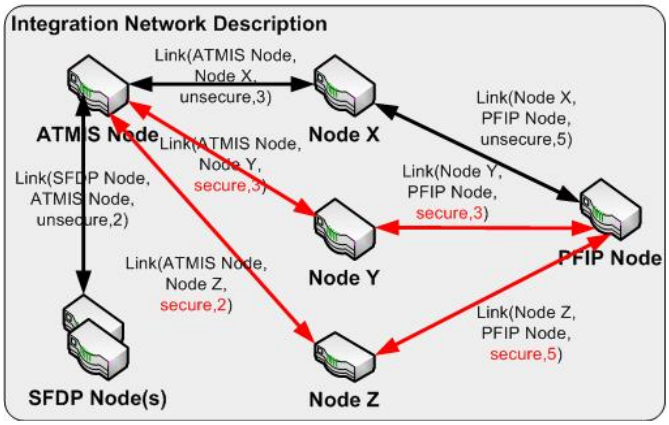


Figure 3: Service Matchmaking Example.

From the domain knowledge description, we know that *Flight ID* is a synonym for *Flight Number*, that *Departure* and *Arrival* are combinations of the airport code and country code of departure/arrival, and that the *FlightStatus* arrived or departed, can be derived by checking the occurrence of either *TimeOfArrival* or *TimeOfDeparture*.

The service matchmaking itself contains direct and indirect matches based on the domain knowledge description. Format transformations like time formatting are also possible during the service matchmaking process. One possible match can be displayed as follows:

```
FlightID = LH6351,
Departure = Departure(Vienna International
  Airport, Austria),
Destination = Destination(Paris-Charles de
  Gaulle Airport, France),
FlightStatus = Departed(19:55-14-12-2008)
```

Next, we calculate the network resources needed for sending messages from the *SFDP* Node to the *PFIP* Node with less capacity. From the integration network description, we can see several nodes connected by links. Each link contains information regarding source node and target node, support for secure transmissions and the transmission delay. From the legacy system description, we can see that communication between *ATMIS* to *PFIP* needs to be done using secure connections only. There are two possible connections, either via *Node Y* or via *Node Z*. The system will choose connection via *Node Y* because it has less delay (6) than connection via *Node Z* (7).

VII. DISCUSSION

The example shows that even for small problems the solution space is typically large. However, large BP and SS integration networks consist of hundreds of integration nodes; and changes of SS properties and network capabilities make the correct and efficient identification of feasible BP and SS pairs a recurring complex and error-prone task. By providing only sets of feasible/promising service provider and consumer candidates, semantic matchmaking supports designers and system integrators by providing sets of possible integration partners regarding both structural and semantic attributes. However, the relevant semantic concepts are hard to define unambiguously for general domains, thus the focus on a well-defined domain like ATM provides semantic clarity.

We used the concept of describing Service policies using a knowledge representation language like OWL, but defined our own extendable policy representation language which is better suitable for the ATM domain. We do not use standardized Web Service description frameworks because, since the strengths of Web Service description frameworks lies in the generality of the approach, however their weakness is that it may become very complicated to describe domain-specific issues. For specific domains, it may be useful to use the principles of web service descriptions but tailor them to the domain. Additionally, we defined our own ontology-based architecture for describing the properties and features of the ATM services.

We have developed a data-driven approach [20] that explicitly models the semantics of the problem space, i.e., BP integration requirements and network infrastructure capabilities [21]; the solution space, i.e., the connectors, and data transformations between SSs [19]. In this paper, we described a process to bridge problem and solution spaces, i.e., identify feasible BP and SSs pairs while fulfilling business requirements and optimizing the chosen integration solution according to multiple objectives. In order to evaluate the proposed process, we have derived two major research issues that will be discussed in the following paragraphs.

Semantic Matchmaking of SS candidates for one BP. Current service matchmaking approaches focus on either technical or semantic integration issues [27, 29], while business process support is, to our knowledge, missing. In the SWIS framework, we presented a combined service matchmaking approach that performs matching based on the data of the services (i.e., the content of their messages) and available service policies regarding other services (e.g., only certain types of consumers are allowed for a specific service). The "open-world" architectural style of ontologies allows the efficient inclusion of new or à priori not known policies in the ontology, the algorithmic "counterpart" of the ontology during the matchmaking process could be represented by a plug-in-architecture that allows additional policy plug-in to be loaded at runtime.

The SWIS framework's semantic service matchmaking enables an effective search space reduction (i.e., lower number of false negatives and false positives) and poses lower risk and effort compared to the current human-based approaches.

Resource Feasibility Check and Optimization for all Collaborations. The optimization process steps allow using existing resources efficiently. Out of all possible collaborations for a single business process which are creatable by means of the proposed semantic matchmaking approach, only those are desirable to be deployed in the integration solution which fulfills certain criteria. Those criteria are set up by the integration expert so that existing collaborations use the underlying integration network infrastructure with its limited resources as efficient as possible.

In addition, service policies of both provider and consumer policies regarding network capabilities or features are regarded during the optimization process step. This allows an effective modeling of network constraints for services, which are then obeyed during the optimization process step.

VIII. CONCLUSION AND FURTHER WORK

In this paper we presented an approach for semi-automatic semantic matchmaking for software services (SSs), the "System-Wide Information Sharing" (SWIS) Business Process (BP) integration framework. The SWIS BP integration frameworks uses the machine-understandable SWIS models to describe BP and SS requirements as well as SS and network capabilities to provide sets of possible SSs for each BP. Out of these possible sets, the system integrators choose the wanted sets. These wanted sets are then optimized with multiple objectives (e.g., costs, delay) while observing the capabilities of the underlying network infrastructure.

We evaluated the feasibility of the SWIS approach in an industry use case from the ATM domain. The example shows that even for small problems the solution space is typically large, and even bigger for large BP and SS integration networks consisting of hundreds of integration nodes. A tool-supported semantic matchmaking process like SWIS can provide system designers and integrators with a set of promising SSs candidates and therefore strongly reduces the human matching effort by focusing on a much smaller space of matchmaking candidates.

Further Work. Further work will include a detailed description of the semantic design to translate between matched services and an evaluation measuring the effectiveness and efficiency of deriving the semantic transformation with tool-support compared to a manual approach. Additionally, we will describe the optimization process of the selected collaborations in more details with special focus on the multi-objective optimization problem and present possibly approaches for performing this optimization.

ACKNOWLEDGMENT

The authors would like to acknowledge all project members of the SWIS (System-Wide Information Sharing) project performed from 2006-2008 at Vienna University of Technology together with Frequentis AG and Austro Control GmbH.

REFERENCES

[1] K. Balasubramanian, A. Gokhale, G. Karsai, J. Sztipanovits, and S. Neema, "Developing Applications Using Model-Driven Design Environments," *COMPUTER*, 2006, pp. 33-40.

[2] D.A. Chappel, *Enterprise Service Bus*, O'Reilly Media, 2004.

[3] A. Doan, N.F. Noy, and A.Y. Halevy, "Introduction to the special issue on semantic integration," *SIGMOD Rec.*, vol. 33, no. 4, 2004, pp. 11-13.

[4] J. Dong, Y. Sun, and S. Yang, "OWL-S Ontology Framework Extension for Dynamic Web Service Composition," *Eighteenth International Conference on Software Engineering & Knowledge Engineering (SEKE'2006)*, Knowledge Systems Institute Graduate School, 2006, pp. 544-549.

[5] M. Ehrgott, *Multicriteria Optimization*, Springer, 2005.

[6] C.H. Goh, "Representing and Reasoning about Semantic Conflicts in Heterogeneous Information Systems," MIT, 1996.

[7] A. Halevy, "Why your data won't mix," *Queue*, vol. 3, no. 8, 2005, pp. 50-58.

[8] K.M. Hansen, W. Zhang, and G. Soares, "Ontology-Enabled Generation of Embedded Web Services," *Twentieth International Conference on Software Engineering & Knowledge Engineering (SEKE'2008)*, Knowledge Systems Institute Graduate School, 2008, pp. 345-350.

[9] S. Herr, K. Läufer, J. Shafaee, G.K. Thiruvathukal, and G. Wirtz, "Combining SOA and BPM Technologies for Cross-System Process Automation," *Twentieth International Conference on Software Engineering & Knowledge Engineering (SEKE'2008)*, Knowledge Systems Institute Graduate School, 2008, pp. 339-344.

[10] V. Kolovski, B. Parsia, Y. Katz, and J. Hendler, "Representing Web Service Policies in OWL-DL," *4th International Semantic Web Conference (ISWC 2005)*, Springer, 2005, pp. 461-475.

[11] T. Kremmel, J. Kubalik, and S. Biffl, "Software Project Portfolio Optimization with Advanced Multiobjective Evolutionary Algorithms," *TBD*, 2008.

[12] J. Kubalík, R. Mordinyi, and S. Biffl, "Multiobjective Prototype Optimization with Evolved Improvement Steps," *Evolutionary Computation in Combinatorial Optimization*, 2008.

[13] H. Lausen, A. Polleres, and D. Roman, "Web Service Modeling Ontology (WSMO)," *W3C Member Submission*, vol. 3, 2005.

[14] S. Martello, and P. Toth, *Knapsack problems: algorithms and computer implementations*, John Wiley & Sons, 1990.

[15] D. Martin, M. Paolucci, S. McIlraith, M. Burstein, D. McDermott, D. McGuinness, B. Parsia, T. Payne, M. Sabou, and M. Solanki, "Bringing Semantics to Web Services: The OWL-S Approach," *First International Workshop on Semantic Web Services and Web Process Composition*, Springer, 2005, pp. 26-42.

[16] S.A. McIlraith, T.C. Son, and H. Zeng, "Semantic Web Services," *IEEE INTELLIGENT SYSTEMS*, vol. 16, no. 2, 2001, pp. 46-53.

[17] P.P. Mike, and H. Willem-Jan, "Service oriented architectures: approaches, technologies and research issues," *The VLDB Journal*, vol. 16, no. 3, 2007, pp. 389-415.

[18] J. Miller, K. Verma, P. Rajasekaran, A. Sheth, R. Aggarwal, and K. Sivashanmugam, "WSDL-S: Adding Semantics to WSDL-White Paper," 2004.

[19] R. Mordinyi, T. Moser, A. Mikula, and S. Biffl, "Foundations for a Model-Driven Integration of Business Services in a Safety-critical Application Domain. Technical Report.," 2009; http://www.complang.tuwien.ac.at/richard/techrep/MDIBSSAD.pdf.

[20] T. Moser, R. Mordinyi, A. Mikula, and S. Biffl, "Efficient System Integration Using Semantic Requirements and Capability Models: An approach for integrating heterogeneous Business Services," *11th International Conference on Enterprise Information Systems (ICEIS 2009)*, 2009, accepted for publication.

[21] T. Moser, R. Mordinyi, A. Mikula, and S. Biffl, "Making Expert Knowledge Explicit to Facilitate Tool Support for Integrating Complex Information Systems in the ATM Domain," *International Conference on Complex, Intelligent and Software Intensive Systems (CISIS 2009)*, 2009, accepted for publication.

[22] T. Moser, K. Schimper, R. Mordinyi, and A. Anjomshoaa, "SAMOA - A Semi-automated Ontology Alignment Method for Systems Integration in Safety-critical Environments," *2nd IEEE International Workshop on Ontology Alignment and Visualization (OnAV'09)*, 2009, accepted for publication.

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

[24] N.F. Noy, A.H. Doan, and A.Y. Halevy, "Semantic Integration," *AI Magazine*, vol. 26, no. 1, 2005, pp. 7-10.

[25] N. Oldham, K. Verma, A. Sheth, and F. Hakimpour, "Semantic WS-agreement partner selection," *15th International World Wide Web Conference*, ACM, 2006, pp. 697-706.

[26] M. Paolucci, T. Kawamura, T.R. Payne, and K. Sycara, "Semantic Matching of Web Services Capabilities," *First International Semantic Web Conference*, Springer, 2002, pp. 333-347.

[27] K. Verma, R. Akkiraju, and R. Goodwin, "Semantic Matching of Web Service Policies," *2nd International Workshop on Semantic and Dynamic Web Process (SDWP 2005)*, 2005.

[28] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, and S. Hübner, "Ontology-based integration of information-a survey of existing approaches," *Workshop on Ontologies and Information Sharing (IJCAI-01)*, 2001, pp. 108-117.

[29] A.M. Zaremski, and J.M. Wing, "Specification matching of software components," *ACM Trans. Softw. Eng. Methodology (TOSEM)*, vol. 6, no. 4, 1997, pp. 333-369.