# Efficient Integration of Complex Information Systems in the ATM Domain with Explicit Expert Knowledge Models

**Thomas Moser[1] Richard Mordinyi[1] Alexander Mikula[2] Stefan Biffl[1]**

[1] Complex Systems Design & Engineering Lab, Vienna University of Technology, Favoritenstr 9/188, A-1040, Vienna, Austria
{thomas.moser, richard.mordinyi, stefan.biffl}@tuwien.ac.at
[2] Frequentis AG, Innovationsstr. 1, A-1100 Vienna, Austria
alexander.mikula@frequentis.com

**Abstract.** The capability to provide a platform for flexible business services in the Air Traffic Management (ATM) domain is both a major success factor for the ATM industry and a challenge to integrate a large number of complex and heterogeneous information systems. Most of the system knowledge needed for integration is not available explicitly in machine-understandable form, resulting in time-consuming and error-prone human integration tasks. In this chapter we introduce and evaluate a knowledge-based approach, "Semantically-Enabled Externalization of Knowledge" for the ATM domain (SEEK-ATM), which explicitly models a) expert knowledge on specific heterogeneous systems and integration requirements; and b) allows mapping of the specific knowledge to the general ATM problem domain knowledge for semantic integration. The domain-specific modeling enables a) to verify the integration knowledge base as requirements specification for later design of technical systems integration and b) to provide an application program interface (API) to the problem space knowledge to facilitate tool support for efficient and effective systems integration. Based on an industry case study, we evaluate effects of the proposed SEEK-ATM approach in comparison to traditional system integration approaches in the ATM domain. Major advantages of the novel approach are the efficient derivation of technical configurations and automated quality assurance of the expert knowledge models.

## 1. Introduction and Motivation

In the Air Traffic Management (ATM) domain complex information systems need to cooperate to provide data analysis and planning services, which consist in the core of safety-critical ATM services and also added-value services for related

businesses. ATM is a relevant and dynamic business segment with changing business processes that need to be reflected in the integration of the underlying information and technical systems.

A major integration challenge is to explicitly model the knowledge embedded in systems and ATM experts to provide a machine-understandable knowledge model for integration requirements between a set of complex information systems (CIS). CIS consist of a large number of heterogeneous subsystems. Each of these subsystems may have different data types as well as heterogeneous system architectures. In addition, CIS typically have significant quality-of-service demands, e.g., regarding security, reliability, timing, and availability. Many of today's ATM CIS were developed independently for targeted business needs, but when the business needs changed, these systems needed to be integrated into other parts of the organization [12]. Most of the system knowledge is still represented implicitly, either known by experts or described in human-only-readable sources, resulting in very limited tool support for systems integration. The process of establishing and/or maintaining integration solutions of business systems is traditionally a human-intensive approach of experts from the ATM and technology domains.

Making the implicit expert knowledge explicit and understandable for machines can greatly facilitate tool support for systems integrators and engineers by providing automation for technical integration steps and automatic validation of integration solution candidates. The overall process for systems integration consists of three phases (see [19]): 1. the elicitation and validation of systems integration requirements (problem space knowledge); 2. the description of the architecture and the modeling of the capabilities of technical solution candidates (solution space knowledge) [17]; and 3. the bridging of the knowledge models of problem and solution space to identify the most suitable solution candidates [20].

In this chapter we focus on the first phase of system integration to provide the foundation for the later phases. We propose a knowledge-based approach, "Semantically-Enabled Externalization of Knowledge" for the ATM domain (SEEK-ATM), which a) explicitly models specific heterogeneous system and expert knowledge on integration requirements using a three-layered ontology architecture for storing knowledge, b) allows mapping of the specific knowledge to the general ATM problem domain knowledge for enabling semantic integration, and c) facilitates tool support for e.g., requirements validation by means of providing homogeneous access to heterogeneous integration knowledge. The knowledge base provides tool access to knowledge models based on a common problem domain model, allowing queries or validation of heterogeneous knowledge sources. The output of this phase is a validated knowledge base of business requirements for integration as input to technical design steps.

We evaluate the effectiveness and efficiency of the SEEK-ATM approach in an industrial case study in the ATM domain. Based on two integration scenarios, we determine key performance indicators, like integration effort, integration duration, quality assurance efficiency, model complexity, and level of automation support in

order to compare the SEEK-ATM approach with traditional system integration approaches in the ATM domain.

The remainder of this chapter is structured as the following: section 2 motivates research issues and pictures the use case, section 3 summarizes related work, section 4 and 5 describe the SEEK-ATM approach; section 6 presents evaluation results. Finally, section 7 concludes and gives an outlook on future research work.

## 2. Objectives and Contribution

Recent projects with industry partners from the safety-critical ATM domain raised concerns about the verification of modern technology-driven integration environments. For certification a major goal was to improve the capability of engineers to verify an integration solution by facilitating team work and tool support.

The data-driven SEEK approach [19] has been developed in order to explicitly model the semantics of the problem space, the solution space, and provide a process to bridge problem and solution spaces. The SEEK approach, described in [19] in more details, consists of 6 process steps: 1. legacy system description, 2. domain knowledge description, 3. model QA, 4. derivation and selection of integration partners, 5. generation of transformation instructions, and 6. configuration QA. For a typical systems integration scenario, the problem space is described as integration requirements and capabilities, the solution space consists of connectors and data transformation instructions between legacy systems, while the bridging process between both spaces is concerned with finding feasible integration solutions, e.g., with minimal integration costs.

In this chapter, we apply the original SEEK process to a use case example from the ATM domain and describe the resulting variant of the SEEK process, SEEK-ATM, with a main focus on the first three process steps, namely the modeling of integration requirements and capabilities for integration knowledge elicitation and QA, resulting in the following research issues.

**RI-1. Foundations for tool support for automation of integration steps.** Investigate to what extent (e.g., effort saved during process execution) the explicit and machine-understandable semantic modeling of integration knowledge helps to automate time-consuming systems integration steps. Investigate the effect of the automated integration process steps regarding the quality assurance efficiency. As precondition for RI-1, we needed to ensure that a) the knowledge is complete enough for relevant tool support, and b) the knowledge can be accessed by tools e.g., by means of an API.

**RI-2. More efficient and effective systems integration process steps.** Investigate whether the SEEK-ATM approach provides an overall more efficient and effective systems integration process regarding key performance indicators like integration effort and duration, QA efficiency, model complexity and level of automation support.

For empirical evaluation we determine the integration effort needed for each process step to compare the steps in the new SEEK-ATM approach with traditional methods and measure the effectiveness and efficiency of the available methods and tools.

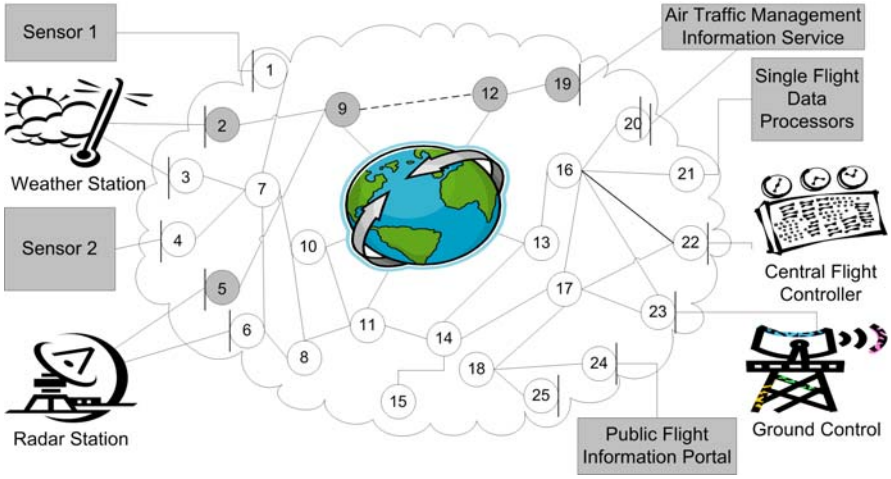


**Figure 1: Overview Use Case Example:**
**Network between information providers and consumers.**

A requirement of the ATM domain is to provide timely and correct data analyses from a web of heterogeneous legacy applications. The high number of distributed legacy applications with heterogeneous interfaces to their services on the one hand and the need to dramatically improve the flexibility in order to provide new ways of systems integration in a safety-critical environment on the other hand, demanded for a innovative approach like the SEEK-ATM.

The ATM use case (Figure 1) represents information that is typically extracted from participants in workshops on requirements elicitation for information systems in the aviation domain. The business system ATM Information Service (ATMIS) has to provide information services about flights to business partners via a Public Flight Information Portal (PFIP). ATMIS needs to collect and refine information from at least two other systems: the Central Flight Controller (CFC) and the Single Flight Data Processors (SFDPs). As input to integration process each data provider, in our case CFC and SFDPs, defines the data content and format he can provide and the quality of service, e.g., the frequency of incoming data such as radar signals; each data consumer, in our case ATMIS, similarly defines his needs for data content, format and quality of service, and may additionally require conditions such as data coming from a defined geographical area and within a defined time window. Finally, the network provider describes the capacity of connectors between the data provider and consumer nodes, and the quality of service of these

connectors, e.g., security levels, reliability. All systems have requirements on reliability, timeliness, safety, service quality, failover, performance, auditability, maintainability, and flexibility. An additional requirement regarding a possible systems integration solution is the capability of agile reaction to any kind of changes due to altered business needs.

Figure 2 illustrates traditional approaches to systems integration in the ATM domain: There are database-style and/or UML models of the systems interfaces, which work well together in a homogeneously designed set of systems. However, in typical domains the systems often exhibit heterogeneous semantics, i.e., similar meaning can be expressed in several ways. Currently, highly skilled domain experts in the ATM problem space and the technical solution space bridge these semantics as there are so far no machine-readable models available to facilitate comprehensive tool support. However, the limited availability of these experts slows down the pace of strategically desirable integration projects.

## 3. Related Work

This section summarizes related work on semantic integration using ontologies.

### *3.1. Semantic Data Integration*

Semantic Integration is defined as the solving of problems originating from the intent to share data across disparate and semantically heterogeneous data [12]. These problems include the matching of ontologies or schemas, the detection of duplicate entries, the reconciliation of inconsistencies, and the modeling of complex relations in different sources [24]. Over the last years, semantic integration became increasingly crucial to a variety of information-processing applications and has received much attention in the web, database, data-mining and AI communities. One of the most important and most actively studied problems in semantic integration is establishing semantic correspondences (also called mappings) between vocabularies of different data sources [6].

Goh [11] identified three main categories of semantic conflicts in the context of data integration that can appear: confounding conflicts, scaling conflicts, and naming conflicts. The use of ontologies as a solution option to semantic integration and interoperability problems has been studied over the last 10 years. In [27], Wache et al. reviewed a set of ontology-based approaches and architectures that have been proposed in the context of data integration and interoperability.

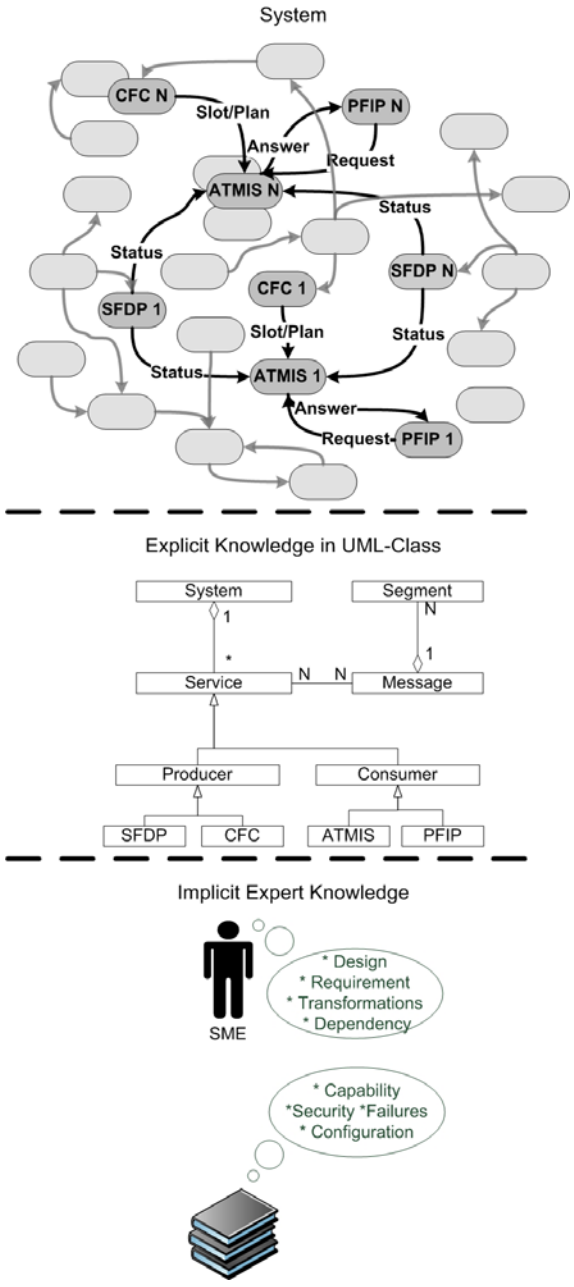


**Figure 2: Air Traffic Management Systems Integration – Explicit and Implicit Expert Knowledge.**

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. Two of the ontologies that are built specifically with the purpose of being formal top-level ontologies are the Suggested Upper Merged Ontology (SUMO) [22] and DOLCE [8]. 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 [3, 5]. The declarative formal representation of mappings is facilitated by the higher expressive power of ontology languages which provide the opportunity to represent mappings themselves in more expressive terms. There exists a large spectrum of how mappings are represented. Bridging axioms relate classes and properties of the two source ontologies and can be seen as translation rules referring to the concepts of source ontologies and e.g., specifying how to express a class in one ontology by collecting information from classes in another ontology. Another mapping representation is the declarative representation of mappings as instances in an ontology. This ontology can then be used by tools to perform the needed transformations. Then a mapping between two ontologies constitutes a set of instances of classes in the mapping ontology and can be used by applications to translate data from the source ontology to the target. Naturally, defining the mappings between ontologies, either automatically, semi-automatically, or interactively, is not a goal in itself. The resulting mappings are used for various integration tasks: data transformation, query answering, or web-service composition, to name a few. Given that ontologies are often used for reasoning, it is only natural that many of these integration tasks involve reasoning over the source ontologies and the mappings.

## *3.2. Ontologies for Semantic Integration*

Ontologies can support data integration processes by providing a continuous data model [4] that helps bridging semantic gaps between systems and/or processes. Compared to traditional common data models like UML Class Diagrams or Entity Relationship Diagrams (ERDs), ontologies both a) provide methods for integrating data models using automated transformation and b) support the concurrent modeling of different systems [14]. There is a wealth of research reports on the extension of UML to support Ontology Engineering for the Semantic Web [1]. For Quality Assurance (QA) ontologies can check whether a model has knowledge missing or inconsistent knowledge.

There has been ample research [13] on the use of ontologies for supporting typical software engineering processes like systems integration. Ontology-Driven Ar-

chitecture (ODA) is introduced, serving as a starting point for the W3C to elaborate a systematic categorization of the different approaches for using ontologies in Software Engineering. The current MDA-based [16] infrastructure provides architecture for creating models and meta-models (e.g. models of the systems to be integrated), define transformations between those models (e.g., transformations between integrated systems), and managing metadata. Though the semantics of a model is structurally defined by its meta-model, the mechanisms to describe the semantics of the domain are rather limited compared to knowledge representation languages. In addition, MDA-based languages do not have a knowledge-based foundation to enable reasoning (e.g., for supporting QA) [2]. System integration can benefit from the integration with ontology languages such as RDF and OWL [9, 10] in various ways, e.g., by reducing language ambiguity, enabling validation and automated consistency checking.

Uschold and Gruninger [26] identified four main categories of ontology application to provide a shared and common understanding of a domain that can be communicated between people and application systems [7]: Given the vast number of non-interoperable tools and formats, a given company or organization can benefit greatly by developing their own neutral ontology for authoring, and then developing translators from this ontology to the terminology required by the various target systems. To ensure no loss in translation, the neutral ontology must include only those features that are supported in all of the target systems. The trade-off here is loss of functionality of some of the tools; since certain special features may not be usable. While it is safe to assume there will not be global ontologies and formats agreed by one and all, it is nevertheless possible to create an ontology to be used as a neutral interchange format for translating among various formats. This avoids the need to create and maintain $O(N^2)$ translators for N systems and it makes it easier for new systems and formats to be introduced into an existing environment. In practical terms, this can result in dramatic savings in maintenance costs - it has been estimated that 95% of the costs of enterprise integration projects is maintenance [25].

There is a growing interest in the idea of “Ontology-Driven Software Engineering” in which an ontology of a given domain is created and used as a basis for specification and development of some software. The benefits of ontology-based specification are best seen when there is a formal link between the ontology and the software. This is the approach of Model-Driven Architecture (MDA) [15] created and promoted by the Object Modeling Group (OMG) as well as ontology software which automatically creates Java classes and Java Documents from an ontology. A large variety of applications may use the access functions of the ontology. Not only does this ensure greater interoperation, but it also offers significant cost reduction for software evolution and maintenance. A suite of software tools all based on a single core ontology are semantically integrated for free, eliminating the need to develop translators. To facilitate search, an ontology is used as a structuring device for an information repository (e.g., documents, web pages, names of experts); this supports the organization and classification of repositories

of information at a higher level of abstraction than is commonly used today Using ontologies to structure information repositories also entails the use of semantic indexing techniques, or adding semantic annotations to the documents themselves. If different repositories are indexed to different ontologies, then a semantically integrated information access system could deploy mappings between different ontologies and retrieve answers from multiple repositories.

## 4. Making Integration Knowledge Explicit

This section pictures the semantic modeling of heterogeneous knowledge using a set of ontologies as model. The ontology architecture [18] is described in detail as well as the distribution of the modeled information among the layers.

The ontologies used as input models for the derivation of the system configuration are organized using a subdivided architecture, consisting of three different types of ontologies. The ontology types building the semantic model for a specific scenario are the Abstract Integration Scenario Ontology (AISO), the Domain-specific Ontologies (DSO), and the Integration System Ontologies (ISO) (see Figure 3). The DSOs extend the AISO by adding concepts describing the common domain knowledge used. In addition, the ISO uses the other two ontologies for aligning its concepts with the more general concepts defined in either the AISO or DSO.

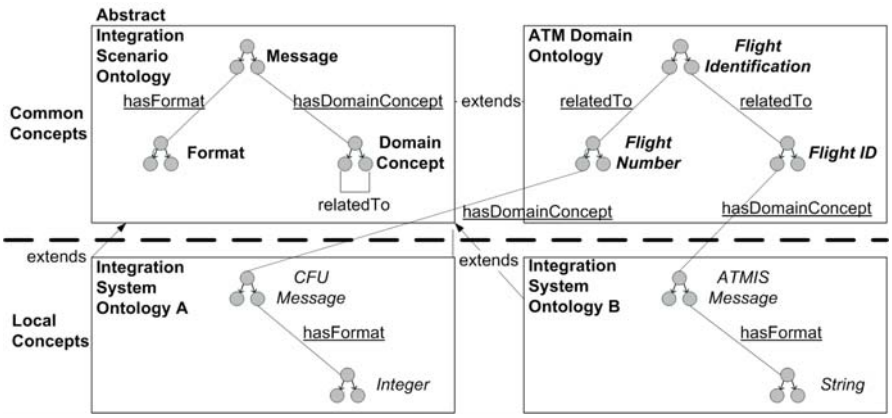


**Figure 3**: **Simplified Ontology Architecture Example [18, 21].**

### *4.1. Abstract Integration Scenario Ontology*

The Abstract Integration Scenario Ontology (AISO) is defined in an application-domain-independent manner, allowing its use across different domains. This domain independent definition is a powerful mechanism to provide a flexible base for information sharing scenarios, completely independent of a particular domain. The terms in the AISO are defined in an abstract way to simplify the conceivability of the use in different domains.

### *4.2. Domain-specific Ontology*

The Domain-specific Ontology (DSO) includes the main shared knowledge between stakeholders of the particular domain (e.g., ATM domain) and hence represents the collaborative view on the information exchanged in an integration scenario. In addition, the DSO is the place to model standardized domain-specific information. The customers map their proprietary information, which is defined in the integration system ontologies, to the standardized information in order to allow the interoperability with other participants.

This domain-specific information is used for the detection of semantically identical information provided or consumed by participating applications or organizations, independent of the format or identifiers used for the information, and therefore improves or enables the communication between these organizations. The identification of possible integration partners is simplified and the tool-supported transformation of semantically identical information existing in different formats allows further communication between new partners.

This particular domain-specific knowledge described in the DSO can easily be updated or transferred to other SEEK-ATM approach-based integration scenarios residing in the same domain. This allows a broad spectrum of new applications in a particular domain to benefit from the described domain knowledge. Instead of modeling the domain knowledge from scratch it is also possible to use as starting point a description of the problem domain, a so-called “world model”. The advantage of this approach is the reduced effort for modeling the domain knowledge; however a tradeoff exists in the complexity of typical “world model” ontologies, resulting in a longer waiting time when searching for concrete domain knowledge.

### *4.3. Integration System Ontology*

The Integration System Ontology (ISO) defines the customer-specific, proprietary view on the information exchanged in an integration scenario. This includes the

view on the format of the information (as required by the legacy application), but can also describe the meaning or the use of the specific view on the existing information, since there can exist multiple views for the same information. The ISO defines the structure of the legacy applications, services and messages, i.e., the services provided by a legacy application, the messages provided or consumed by a service and the message segments a message consists of, by adding instances of the concepts defined in either the AISO or the DSO.

The most important part of this description is the definition of the exchanged information, i.e., the definition of the messages either provided or consumed by the legacy applications. The ISO describes the semantic context and the format of each message segment, supported by the domain expert. Each message segment is mapped to exactly one particular domain concept. This defines the semantic context of the information contained in the segment and allows the detection of possible collaborations for an integration scenario. In addition, the format of the information is described, enabling automated transformation between formats.

## 5. SEEK-ATM Process Description

This section summarizes the key factors of the SEEK-ATM approach. Figure 4 gives a short overview of the SEEK-ATM process steps for requirements elicitation and validation in comparison with a traditional integration approach.

**Traditional Integration Approach.** In the traditional integration approach, for each legacy information system to be integrated, the Subject Matter Expert (SME) responsible for the particular system describes the requirements and capabilities of the system using human-readable (but typically not machine-readable) language. The outcome of this process step is a set of legacy systems interface description documents. The QA step is performed mostly by humans and mainly consists of a) a comparison of the knowledge represented in the legacy systems interface description documents with the knowledge captured implicitly by the SMEs; and b) a comparison of the accepted set of integration partners and the needed transformation instructions with the knowledge represented in the legacy systems interface description documents and again with the knowledge captured implicitly by the SMEs. In the traditional integration process, there are 2 QA steps performed mostly by humans: a) comparison of the knowledge represented in the legacy systems interface description documents with the knowledge captured implicitly by the SMEs; and b) comparison of the accepted set of IPs and the needed transformation instructions with the knowledge represented in the legacy systems interface description documents and again with the knowledge captured implicitly by the SMEs. As key parts of this knowledge are not available in machine-readable form, tool support for QA is very limited and takes much effort from scarce human experts.

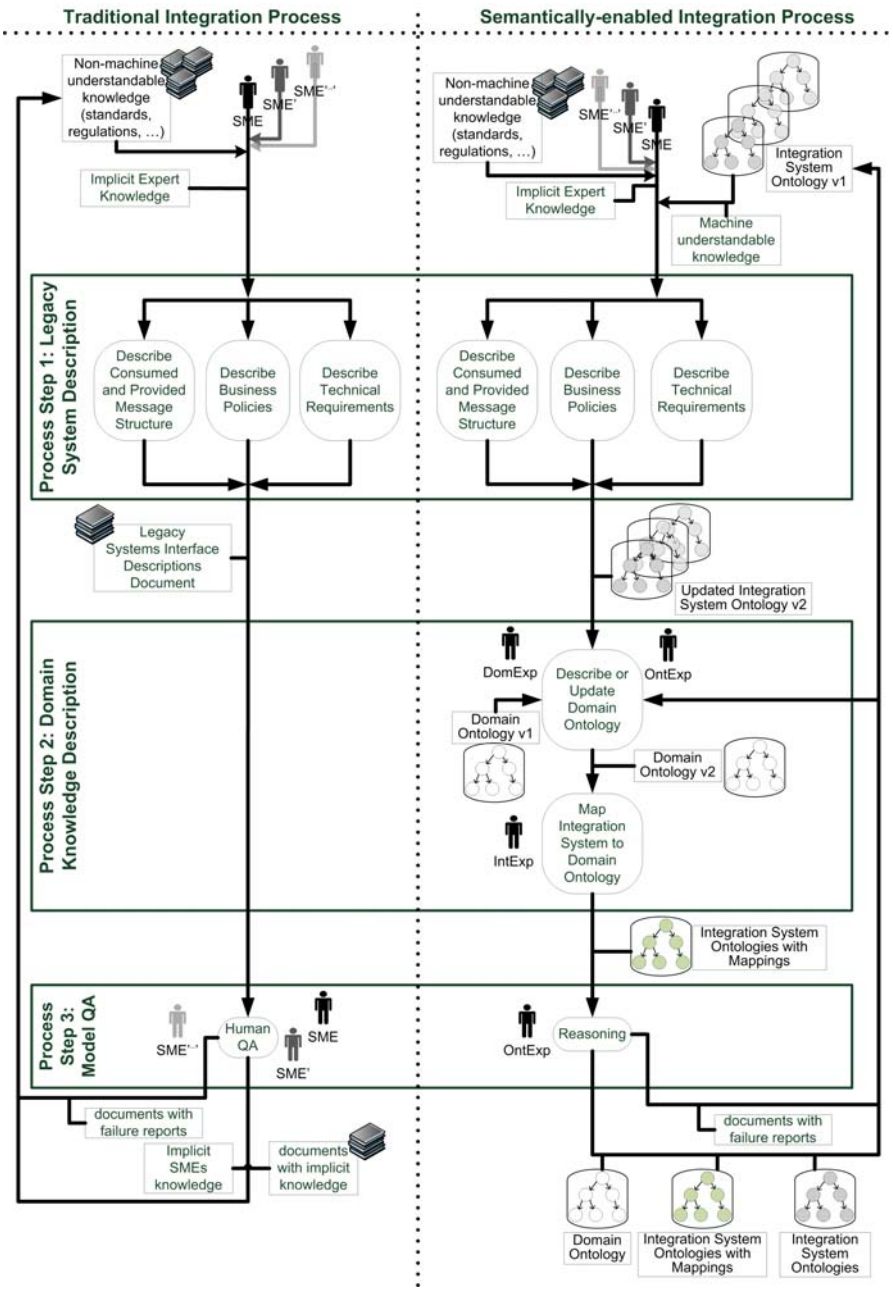


**Figure 4: Side-by-side comparison of the traditional and the SEEK-ATM integration process steps.**

**SEEK-ATM Integration Approach**. In the SEEK-ATM approach, for each legacy information system to be integrated, the SME responsible for the particular system describes the requirements and capabilities (R&Cs) of the system using machine-readable notations. In addition to these R&Cs, the semantic meaning of the exchanged information is externalized by mapping information to more general knowledge represented in the domain ontology. In comparison to the traditional integration process, the outcome of this process step is a set of ontologies describing the R&Cs of the legacy information system to be integrated, as well as the mapping of the information to general domain knowledge. In addition to the description of the R&Cs of the participating systems, the domain expert (DE) describes the common knowledge of the problem domain used in the integration scenario.

This externalized domain knowledge is used by the SMEs while describing the particular legacy systems. The outcome of this process step is an ontology describing the shared domain knowledge of the problem domain used in the integration scenario. This domain ontology can be reused for a set of different integration scenarios in a domain. The QA step in the SEEK-ATM integration approach can be very well supported with tools based on ontology-based reasoning. Reasoning allows checks for consistency (e.g., whether information entered in different input masks is consistent) and completeness (e.g., whether all needed information is entered). This allows a much faster and more reliable QA compared to the traditional integration process and allows relieving scarce experts from tedious work.

To summarize the process description, for both the traditional integration process and the SEEK-ATM process the input is the same, but the output differs.

While the output of the traditional integration process still consists of mainly implicit knowledge, the output of the SEEK-ATM process consists of explicit and machine-understandable knowledge.

## 6. Added Value from Explicit Knowledge

This section pictures usage scenarios for heterogeneous knowledge integrated using the SEEK-ATM approach. In addition, the real usage scenarios from the exemplary ATM use case are described shortly. The knowledge can be used for a set of queries like checking the consistency of the integrated data (e.g., by measuring type similarity between concepts), or checking the completeness of the mapped concepts (e.g., whether it is possible to fulfill the given requirements with the modeled knowledge).

In the use case from the ATM domain, the integrated knowledge can be used for the auto-mated identification of integration partner candidates, the generation of transformation instructions and for the generation of system integration configurations. This allows a much faster and more reliable QA compared to the tradi-

tional integration process and relieves scarce experts from tedious work. The following paragraphs summarize these usage examples.

**Automated Identification of Integration Partner Candidates.** For every consumer service, the set of possible provider services providing the required information is calculated. These sets of pairs of a consumer service and at least one provider service, together with the required transformation instructions are called collaboration candidates. The Domain Expert (DomExp) and the customer SMEs choose one or if applicable more desired collaborations from these collaboration candidates. Then the system integration configuration for these chosen collaborations is calculated by the SEEK-ATM Approach. The externalized knowledge of the SMEs, the DomExp, and the Network Administrator (NA) which is captured in the ontologies created in the previous steps is used to automatically derive the set of possible integration partners using ontology-based reasoning, allowing an easier and less error prone identification of possible integration partners compared to the traditional integration process. The outcome of this process step is a set of possible integration partners. The Integration Expert (IntExp) is responsible for choosing the wanted integration partners from the set of possible integration partners derived in the previous step. The outcome of this process step is a set of accepted integration partners.

**Generation of Transformation Instructions**. After these integration partners are selected, the transformation instructions for these collaborations need to be created. This generation process is semi-automatic and supervised by the DomExp. The DomExp reviews the generated transformation instructions and has to accept it, in order to be functional.

**Generation of System Integration Configuration.** The information derived in the previous steps is used to create the final system integration configuration. The configuration is stored in an XML files containing information all the needed instructions to run the system, such as routing tables, transformation instructions, and binding descriptions for connecting to particular legacy systems.

## 7. Evaluation

In the previous section RI-1 has been addressed. To discuss the RI-2, we started an evaluation by means of the proposed entire SEEK-ATM approach. Therefore, we derived four parameters (see Table 1) to compare the proposed approach with the traditional one. Table 1 summarizes the effort and duration needed for integration, the quality assurance efficiency, the complexity of the used models, and finally the level of automation support both approaches provide.

The evaluation is based on two scenarios within the ATM use case. The first scenario (Sc. 1) determines the results based on an integration project from the

scratch. The second scenario (Sc. 2) assumes that an initial integration project has been accomplished providing a first integration solution, but due to changing business requirements some system adaptations have to be performed, like the need to update the domain model. Sc. 1 within the ATM use case has the following characteristics: 5 systems (applications) with 30 integration points (services) and 100 data structures (logical entities). In case of Sc. 2, 10 integration points of 3 different systems have been updated resulting in 2 new data structures and 10 updated ones. The overall integration effort for scenario 1 using the traditional approach was 415 PDs[1] and for scenario 2 76 PDs. When using the SEEK-ATM approach, the overall integration effort for scenario 1 was 435 PDs, compared to 32 PDs for scenario 2.

**Integration effort.** The results of the evaluation show that the overall integration effort is similar for both approaches in case of small number of systems to be integrated and slightly higher for the SEEK-ATM approach in case of larger systems. The higher effort comes from the need to manage the domain model, since additional mappings between the integration system ontology and the domain model are needed. The effort to create the integration system ontology or the interface description is similar since in both approaches the conducted SMEs has to cope with the same problem of finding the right information describing the system interfaces with its semantics. The SEEK-ATM has the advantage that in case of adaptation the knowledge already gathered is explicitly given and can be reused in further discussions compared to the traditional approach where this knowledge exists implicitly only. In case of reconfiguration issues the SEEK-ATM process has proven to be more efficient than the traditional approach since once the knowledge has been externalized, it can be reused with little extra effort. Furthermore, in case of the traditional approach each system expert has to be contacted for any kind of changes resulting in discussions.

In case of the SEEK-ATM approach the domain expert is needed in major changes only where the mapping of the integration system ontology to the domain ontology has to be altered as well. In case of minor changes, affecting the characteristics of the system only, the SMEs are needed. Additionally, performing changes, like structure modifications, based on documents is more difficult and time consuming than compared with ontologies where you deal with classes. Changes can be performed much faster and can be done during the discussion concerning the integration project as well. The duration of the traditional approach tends to be higher due to error-prone mainly manual process steps resulting in additional efforts to discuss error sources and possible solutions. The proposed SEEK-ATM approach reports errors or missing information immediately due to in-time consistency and completeness checks based on ontology reasoning. In case of describing systems, parallel processing is possible in both approaches. However, the following SEEK-ATM processing steps are running mainly automated

[1] PD: Person Day (Full Time Equivalent).

from the third processing step on, while the traditional approach is still human-driven resulting in time consuming and error-prone processing steps. Therefore, the duration depends strongly on of automation support.

**Table 1: Comparison of the traditional and the SEEK-ATM approaches.**

| Evaluation parameters | Traditional approach | SEEK-ATM approach |
|---|---|---|
| Integration effort | System knowledge is described in human-readable documents by Subject Matter Experts (SMEs).<br><br>No explicit domain knowledge used. | System knowledge is externalized in a machine-readable ontology by SMEs.<br><br>Domain knowledge is incrementally externalized in a machine-readable ontology by the Domain Expert (DomExp). |
| QA efficiency | Low<br>Manual checks of documents and models needed (time consuming and error prone). | High<br>Automated ontology reasoning allows quickly locating inconsistent knowledge in the model. |
| Model complexity | High and distributed | High and centralized |
| Level of automation support | Low<br><br>Exhaustive communication between SMEs, DomExp, and IntExp is needed to clarify dependencies and integration partners.<br><br>DomExp coordinates the generation of transformation instructions with the affected SMEs.<br><br>Manual checks of documents and system configuration needed (time consuming and error prone). | High<br><br>Automated derivation of possible integration partners by means of ontology based reasoning.<br><br>Automated derivation of transformation instructions by means of ontology based reasoning.<br><br>Automated ontology reasoning allows quickly locating invalid system configurations. |

**QA efficiency.** Since the traditional approach focuses on manual validity checks, it is therefore more time consuming and error-prone. This also results in the fact that missing information is often detected in a later integration step. The

quality assurance efficiency is measured by the number of failures detected in each system description weighted by the time of detection. The later the failure detected the higher the weighting rate. The SEEK-ATM approach uses ontology-based reasoning. This allows performing consistency and completeness checks in-time automatically, resulting in a lower failure rate and in-time notification of the SME about missing/incorrect information. Additionally, since the SEEK-ATM approach is mainly automated, it allows returning to any processing state in order to e.g., reproduce errors or revise decisions taken.

**Model complexity.** The model used in the traditional approach is smaller and therefore less complex compared to the model used in the SEEK-ATM approach, since a considerable part of the integration knowledge is not described explicitly. In the SEEK-ATM approach, the number of relations, i.e., the number of mappings from the integration system ontology to the domain ontology introduces a higher structural complexity. The benefit of a more complex ontology model lies in the way how later integration steps can be supported by a higher level of automation. From the SME's point of view the complexity remains the same in both approaches. For the domain expert the SEEK-ATM approach reduces his efforts to the task of managing the structural complexities of the ontologies and to support the SMEs in mapping. In the traditional way the domain experts need to cope with the major part of the complexity, since he is responsible for ensuring the consistency and completeness as well as managing the integration of the SMEs' legacy system descriptions.

**Level of automation support.** The SEEK-ATM approach supports the user while entering the data with consistency and completeness checks. Additionally, it influences the integration process in later steps by automatically deriving integration partner candidates and automatically generating transformation instructions for message exchange between the integrated systems.

Within a research project with two industry partners, the approach has been evaluated by means of several different scenarios from the ATM domain. We determine the effort for both process step variants and compare the overall outcome. The following paragraphs summarize the effort needed to perform the particular process steps. The effort estimations are based on the expertises of the integration experts from both companies.

**Step 1. Legacy System Description:** The externalization of legacy system knowledge using ontologies needs slightly more effort than the traditional approach using only human-readable artifacts like documents because the knowledge needs to be transformed from implicit expert or system knowledge into machine-readable ontology models.

**Step 2. Domain Knowledge Description:** In the traditional integration process the domain knowledge is not made explicit but implicitly captured by domain experts and documents in a non-machine-readable way requiring no additional effort. Additionally, the integration network knowledge (i.e., the architec-

ture and capabilities of the underlying network infrastructure) are described, which again represents an additional effort compared to the implicit knowledge of the traditional integration process. Using the *SEEK* approach the domain and integration network knowledge has to be incrementally externalized by the domain expert and the network administrator resulting in medium effort in the first instance. This effort is reduced due to reuse within similar integration scenarios or additional process iterations triggered by reconfiguration issues.

**Step 3. Model QA:** The traditional approach requires a lot of effort to check the consistency and completeness of the documents since it has to be done manually. The *SEEK* approach uses automated ontology based reasoning techniques to assure consistent models leading to comparatively low model QA effort.

## 8. Conclusion and Future Work

In this chapter, we introduced and evaluated a domain-specific approach for ATM to make expert knowledge on heterogeneous systems and system integration requirements explicit to facilitate tool-support for design and QA. An important contribution of the chapter is to enable new research and application areas for semantic techniques that help control complex information system. Major results of our research evaluation of SEEK-ATM in an industrial case study were:

**1. Tool support for automation of integration steps.** The explicit and machine-understandable knowledge in SEEK-ATM helps to automate time-consuming systems integration steps like consistency and completeness checks. Furthermore, it allows automating later integration processing steps, like deriving integration partner candidates or automatically generating transformation instructions for message exchange between the integrated systems.

**2. More efficient and effective systems integration.** The evaluation showed that the integration effort needed with the SEEK-ATM approach is slightly higher in case of integration from the scratch, but comparatively a lot smaller when adaptations due to changing business needs have to be performed. In addition, the advantage of centrally storing the domain ontology together with the mappings of individual system knowledge lies in the possibility of an automated QA and automation of further integration steps resulting in less integration efforts and less failures.

Further work will extend the semantic modeling of the problem space to the technical solution space and ultimately ways to bridge problem and solution spaces, as well as include a large-scale evaluation of the SEEK-ATM approach using scenarios and integration effort measurements in real-world integration projects. Additionally, the feasibility of ontology-based reasoning for usage in QA will be evaluated.

## References

1. K. Baclawski, M. Kokar, P. Kogut, L. Hart, J. Smith, W. Holmes, J. Letkowski, and M. Aronson, "Extending UML to Support Ontology Engineering for the Semantic Web," *4th International Conference on UML*, Springer, pp. 342-360.
2. K. Baclawski, M.K. Kokar, P.A. Kogut, L. Hart, J. Smith, J. Letkowski, and P. Emery, "Extending the Unified Modeling Language for Ontology Development," *International Journal of Software and Systems Modeling (SoSyM)*, vol. 1, no. 2, 2002, pp. 142-156.
3. S. Bergamaschi, S. Castano, and M. Vincini, "Semantic integration of semistructured and structured data sources," *SIGMOD Rec.*, vol. 28, no. 1, 1999, pp. 54-59.
4. C. Calero, F. Ruiz, and M. Piattini, *Ontologies for software engineering and software technology*, Springer, 2006.
5. I.R. Cruz, X. Huiyong, and H. Feihong, "An ontology-based framework for XML semantic integration," *International Database Engineering and Applications Symposium (IDEAS '04)*, IEEE, pp. 217-226.
6. 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.
7. D. Fensel, *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*, Springer, 2003.
8. A. Gangemi, N. Guarino, C. Masolo, and A. Oltramari, "Sweetening WordNet with DOLCE," *AI Magazine*, vol. 24, no. 4, 2003, pp. 13-24.
9. D. Gasevic, D. Djuric, and V. Devedzic, "Bridging MDA and OWL Ontologies," *Journal of Web Engineering*, vol. 4, no. 2, 2005, pp. 118-143.
10. D. Gasevic, D. Djuric, and V. Devedzic, *Model driven architecture and ontology development*, Springer, 2006.
11. C.H. Goh, "Representing and Reasoning about Semantic Conflicts in Heterogeneous Information Systems," MIT, 1996.
12. A. Halevy, "Why your data won't mix," *Queue*, vol. 3, no. 8, 2005, pp. 50-58.
13. H.J. Happel, and S. Seedorf, "Applications of Ontologies in Software Engineering," *2nd International Workshop on Semantic Web Enabled Software Engineering (SWESE)*.
14. M. Hepp, P. De Leenheer, A. De Moor, and Y. Sure, *Ontology Management: Semantic Web, Semantic Web Services, and Business Applications*, Springer-Verlag, 2007.
15. J. Miller, and J. Mukerji, "Model Driven Architecture (MDA)," *Object Management Group, Draft Specification ormsc/2001-07-01, July*, vol. 9, 2001.
16. J. Miller, and J. Mukerji, "Model Driven Architecture (MDA) Object Management Group Draft Specification," 2001; http://www.omg.org/docs/ormsc/01-07-01.pdf.
17. R. Mordinyi, T. Moser, E. Kühn, S. Biffl, and A. Mikula, "Foundations for a Model-Driven Integration of Business Services in a Safety-critical Application

Domain," *35th Euromicro Conference Software Engineering and Advanced Applications*.
18. T. Moser, A. Andjomshoaa, and A. Mikula, "FISN Semantic Architecture Document," *Book FISN Semantic Architecture Document*, Series FISN Semantic Architecture Document, ed., Editor ed.^eds., Frequentis AG, 2007.
19. 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)*, pp. 56-63.
20. T. Moser, R. Mordinyi, W.D. Sunindyo, and S. Biffl, "Semantic Service Matchmaking in the ATM Domain Considering Infrastructure Capability Constraints," *21st International Conference on Software Engineering and Knowledge Engineering*, pp. 222-227.
21. 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)*, pp. 724-729.
22. I. Niles, and A. Pease, "Towards a standard upper ontology," *2nd International Conference on Formal Ontology in Information Systems*, ACM, pp. 2-9.
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. J. Pollock, "Integration's Dirty Little Secret: It's a Matter of Semantics," *Whitepaper, Modulant: The Interoperability Company*, 2002.
26. M. Uschold, and M. Gruninger, "Ontologies and semantics for seamless connectivity," *SIGMOD Rec.*, vol. 33, no. 4, 2004, pp. 58-64.
27. 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)*, pp. 108-117.