Nomads - Enabling Distributed Analytical Service Environments for the Smart City domain

Johannes M. Schleicher, Michael Vögler, Christian Inzinger, Waldemar Hummer and Schahram Dustdar
Distributed Systems Group, Vienna University of Technology, 1040 Vienna, Austria
{lastname}@dsg.tuwien.ac.at

***Abstract*—The advent of the Smart City domain has led to the creation of massive amounts of diverse data. Stakeholders in this domain need to be able to analyze this data in order to make informed planning decisions. To address this complex task, Distributed Analytical Environments (DAEs) have emerged. These environments consist of different distributed analytical and data services, which are composed in a dynamic way to deliver insights that are crucial for stakeholders. Since these environments deal with business critical and sensitive information, strict compliance constraints apply. These constraints lead to situations where certain concrete services are not allowed to exchange data, even though their interaction is necessary to produce the desired results. Finding a valid solution in the space of possible instantiations is a non-trivial problem. In this paper we introduce Nomads, a framework that enables service mobility in such constrained dynamic composition environments to overcome aforementioned restrictions. The framework improves the overall satisfiability and therefore also the quality of constrained DAEs. We outline the requirements of a representative DAE scenario, provide a detailed problem formulation, and then discuss the service mobility framework along with our solution finding algorithm. The evaluation demonstrates that the Nomads framework considerably increases the number of successfully performed compositions even in highly constrained environments.**

I. Introduction

In order to successfully make informed decisions and, more importantly, to plan in the Smart City domain it is elementary to understand the massive amounts of data generated by modern cities. This has created numerous new challenges for city information management [1], [2] and software architecture [3], [4], as well as led to the creation of several research initiatives (e.g. European Smart Cities project[1], IBM Smarter Planet Initiative[2], MIT Smart City Group[3], and URBEM[4]). Stakeholders need to be able to make informed decisions based on analyses of domain experts. This is accomplished using Distributed Analytical Environments (DAE). DAEs can be considered an instance of Software-defined Elastic Systems for Big Data Analytics [5]. Such a DAE relies on dynamic analytical service compositions which are created based on specific questions from stakeholders to provide the desired results. However, these compositions cannot always be executed due to compliance constraints between service providers. These constraints lead to satisfiability problems of service compositions resulting in the inability to answer stakeholders' questions. In this paper we propose a framework to overcome this limitation by enabling dynamic service migrations and by doing so delivering a strong quality improvement for constrained DAEs.

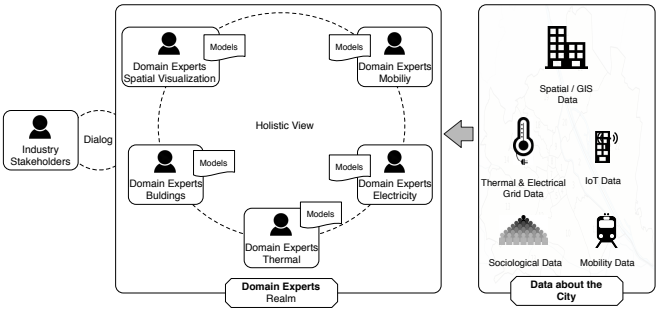


Fig. 1: URBEM Overview

The remainder of this paper is structured as follows: In Section II we present the scenario and outline the specific problem as well as requirements. Section III introduces the NOMADS framework as an approach to address these problems. This is followed by a validation and short evaluation of our approach in Section IV, as well as a discussion of related work in Section V. The paper concludes in VI followed by an outlook on future research.

II. Scenario

In this section we introduce a motivating scenario based on the problems tackled in the URBEM initiative. Industry stakeholders try to make informed decisions based on massive amounts of data provided by a Smart City. In URBEM, they aim to achieve this by utilizing analysis of specific aspects of the city, which are provided by domain experts. These domain experts rely on complex models, which interact in a highly dynamic fashion depending on the specific aim of the industry stakeholders. An overview can be seen in Figure 1.

Consider the following case of urban planning: The stakeholders want to know the impact photovoltaics would have on the energy grid if they would be installed on every housing area in a specific district. In order to answer this question a plethora of different data needs to run through different models of various domains experts in the areas of building-physics, energy grids and spatial visualization, forming a DAE. Figure 2 shows the analytical process underlying such a DAE.

From a technical perspective the models provided by domain experts, as well as the data, are exposed as abstract

[1]http://smart-cities.eu/
[2]http://www.ibm.com/smarterplanet/
[3]http://cities.media.mit.edu/
[4]http://urbem.tuwien.ac.at/

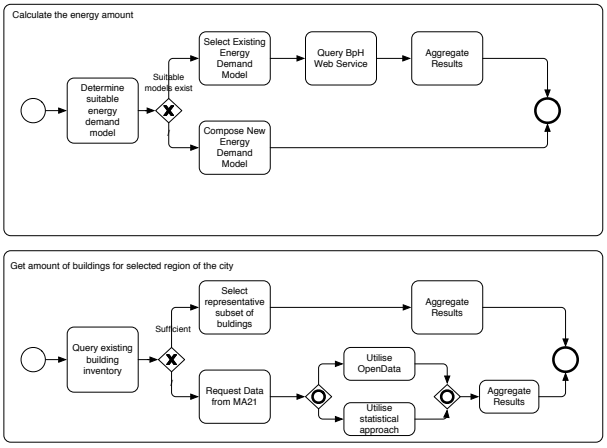


Fig. 2: Example of an analytical process in URBEM

services, each of these services providing different capabilities. There are services providing data about the spatial aspects of the city, about the thermal and electrical grids, services that compute the energy demands of buildings, services that provide mobility models and many more. These abstract services are dynamically composed depending on the specific question of the stakeholders. Each of these abstract services can have different specific implementations called concrete services which again originate from multiple providers. In URBEM providers include companies like energy providers and public transportation services, municipalities and city administration, as well as third party solution providers. Due to the variety of data including business critical sensitive information, as well as certain regulatory requirements with numerous compliance aspects there are strict data exchange constraints between these providers, resulting in a constrained DAE. These constraints usually apply to sensitive information (e.g., detailed household energy usage, medical data, etc.), but do not apply to results of analyses that process sensitive data and produce insights or aggregated results that cannot be used to infer the input data from their results.

A. Problem Description

The first elementary characteristic of constrained DAEs is the fact that only if a concrete service composition is possible the respective question of a stakeholder can be answered, making satisfiability an essential quality metric. The major factor for determining satisfiability are the constraints between providers of specific services.

These constraints can be *bidirectional* as well as *unidirectional*, for example since the liberation of the Austrian energy market the regulatory agency prohibits data exchange from grid infrastructure operators to energy providers. However an exchange in the opposite direction is possible. This also is important for a migration of a service. Even though the grid operator is not allowed to exchange data with the energy provider, it would be possible for the consuming service of the provider to migrate to infrastructure controlled by the grid operator and successfully produce results while respecting all constraints.

The second elementary aspect of constrained DAEs is the highly dynamic nature of the service composition. Since specific compositions of services are triggered by stakeholders' questions they are not known a priori. This leads to the need for a dynamic mechanism to move concrete services between providers. However, the migration of certain services might not be possible or feasible. An example for this case are certain sensitive data services or services dealing with large amounts of data.

Figure 3 shows an overview of the problem. A specific question of a stakeholder leads to a service composition that is necessary to answer this question, as depicted at the top of the figure. Due to the provider constraints it is however not possible to satisfy this question with the current deployment of concrete services. This leads to a satisfiability problem of the DAE and the inability to answer the question, therefore significantly impacting the quality. However with the ability to migrate concrete services on demand the constraints can be satisfied and the desired results can be produced.

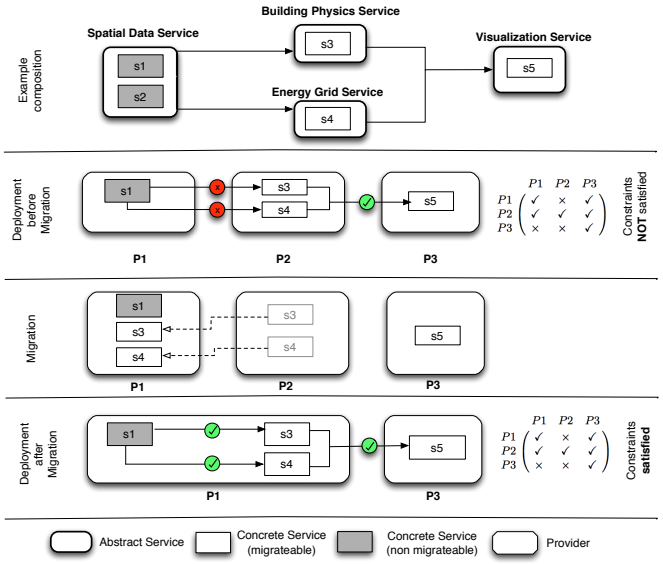


Fig. 3: Composition Scenario with Exchange Constraints and Service Migration

We can therefore state the following essential requirements in context of the outlined problem.

- Constrained Distributed Analytical Environments depend on the satisfiability of concrete service compositions under consideration of constraints. Service migration is an essential factor to enable this satisfiability.
- Due to the dynamic nature of the service composition triggered by specific stakeholders, which cannot be known a priori, a dynamic mechanism allowing service mobility is required.

III. NOMADS

In this section we introduce the Nomads framework as an enabling platform for executing complex DAEs. We start by

formalizing a system model followed by a detailed framework architecture description.

A. System Model

This section introduces the system model considered in this paper. Table I summarizes the model symbols, and provides a brief example with reference to the scenario of Section II. $\mathcal{P}(X)$ denotes the power set of a given set X, and $M[m,n]$ denotes the entry in row m and column n of a matrix M.

Symbol	Description	Example
A	Set of abstract services	$A = \{a_1, ..., a_4\}$
S	Set of concrete services	$S = \{s_1, ..., s_8\}$
P	Set of service providers	$P = \{p_1, ..., p_4\}$
$s : A \to \mathcal{P}(S)$	Maps abstract to concrete services	$s : a_1 \mapsto \{s_1, s_2\}, ...$
$o : S \to P$	Provider from which a concrete service originates	$o : s_1 \mapsto p_1, s_2 \mapsto p_1, ...$
$D \subseteq A \times A$	Data dependencies between pairs of abstract services	$D = \{(a_1, a_3), (a_2, a_3), (a_3, a_4)\}$
$E \subseteq \{0,1\}^{\lvert P\rvert \times \lvert P\rvert}$	Data exchange constraint matrix: $E[x,y] = 1$ if provider p_x can exchange data with provider p_y	$E = \begin{pmatrix} 1 & 0 & \cdots & 1 \\ 1 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 1 \end{pmatrix}$
$I = [A \to S \times P]$	Set of possible runtime instantiations (selection of concrete services and executing providers)	(all instantiations which satisfy the constraints in E, see example below)
$i \in I$	Runtime instantiation	$i : a_1 \mapsto (s_1, p_1), a_2 \mapsto (s_3, p_2), a_3 \mapsto (s_6, p_4), a_4 \mapsto (s_8, p_2)$
$m_i : S \to P$	Services to be migrated to different provider (for instantiation $i \in I$)	$m_i : s_8 \mapsto p_2$
$\pi : S \mapsto \{0,1\}$	Migration policy (is a service allowed to migrate)	$\pi : s_1 \mapsto 0, s_2 \mapsto 1 \ldots$

TABLE I: Model for Compositions with Constraints and Service Migration

Basic Model. A service composition in our problem domain consists of a multitude of abstract services (A), whose functionality is implemented by one or more concrete services (S). The function $s : A \to \mathcal{P}(S)$ maps abstract to concrete services. The set of providers collaborating within the composition is denoted as P, and each service $s \in S$ originates from one provider $o(s) \in P$. D denotes the set of pairwise data dependencies between abstract services. The abstract services in A (as nodes) and the dependencies D (as edges) span up a directed acyclic graph (DAG), which defines the execution flow of the service composition.

Constraints. The core motivation of this work is the fact that providers have constraints concerning data exchange, which are expressed in our model using a matrix representation E. An example of such a constraint matrix can be seen in Figure 4.

Migration and Instantiation. In order to instantiate the composition defined so far in this section, we need to 1) select concrete services, and 2) determine the providers, which are needed to execute the code of each service. That is, a valid instantiation $i \in I$ determines for each abstract service the concrete service and executing provider, expressed by the mapping $A \to S \times P$. Note that, due to data exchange constraints in E, the instantiation may require to migrate the code of some services from one provider to another. Given an instantiation $i \in I$, any concrete service $s \in S$, which is supposed to be executed by a provider $p \in P$ but in fact originates from some other provider, i.e., $o(s) \neq p$, needs to be migrated to p for execution (see definition of function m_i in Equation 1). Further details concerning the motivation and necessity of service migration follow in Section III-B.

$$\begin{array}{l} P_1 \quad P_2 \quad \ldots \quad P_n \\ P_1 \begin{pmatrix} 1 & E[P_1,P_2] & \ldots & E[P_1,P_n] \\ E[P_2,P_1] & 1 & \ldots & E[P_2,P_n] \\ \vdots & \vdots & \vdots & \vdots \\ E[P_n,P_1] & E[P_n,P_2] & \ldots & 1 \end{pmatrix} \\ P_2 \\ \vdots \\ P_n \end{array}$$

Fig. 4: Data exchange constraints matrix between providers

$$\begin{array}{l} \forall i \in I : m_i = \\ \{(s,p) \mid \exists a \in A : i(a) = (s,p) \wedge o(s) \neq p\} \end{array} \tag{1}$$

B. Problem Formulation

Based on the system model introduced in Section III-A we now provide a detailed formulation of the problem studied in this work.

Given the model of a service composition (A, S, P, D) and the matrix E, we seek for a valid instantiation $i \in I$ such that all constraints in E are satisfied (see Equation 2).

$$\begin{array}{l} \forall (a_x, a_y) \in D, i(a_x) = (s_x, p_x), \\ i(a_y) = (s_y, p_y) : E[p_x, p_y] = 1 \end{array} \tag{2}$$

The problem is that, without service migration, it may be infeasible to find a valid instantiation under the information exchange constraints defined in E (see Section III-A). Hence, the possibility of migrating services is the central assumption of our approach. Assuming two abstract services $a_x, a_y \in A$ connected via a data dependency, the concrete service s_y for a_y (i.e., $s_y = i(a_y)$) needs to be migrated to the provider of a_x (denoted p_x) if there is a constraint between the providers of the two services and s_y is allowed to migrate (i.e., $\pi(s_y) = 1$ (see Equation 3).

$$\begin{array}{l} \exists (a_x, a_y) \in D, s_y = i(a_y), \\ p_x = o(i(a_x)) : E[p_x, o(s_y)] = 0 \\ \quad \wedge \ E[o(s_y), p_x] = 1 \ \wedge \ \pi(s_y) = 1 \Longrightarrow (s_y, p_x) \in m_i \end{array} \tag{3}$$

Finding a valid instantiation i under the conditions in Equations 2 and 3 is a hard computational problem. A complexity analysis of the problem is out of scope for this paper, our focus here is to provide a framework for finding valid instantiations and performing the necessary service migrations. Details of our approach are discussed in Section III-C.

C. Framework Architecture

In this section, we outline the architecture of our framework to address the need for service mobility as discussed in Section III-B. Every service composition to be instantiated is executed within a network of containers to allow for necessary constraint enforcement and enable the novel service migration mechanism. Each provider who is part of a composition instantiation has an on-premise deployment of a Nomads container instance to handle constraints as well as service migrations. Additionally there exist a number of Service containers able to execute local and transfered services. A graphical overview of the Nomads container as well as the Service container in the context of a provider is shown in Figure 5.

The coordination between all Nomads containers is handled via a distributed consistent key value store. In our prototypical implementation of the framework we use etcd[5] as key value store which relies on the Raft consensus algorithm [6]. Each Nomads container, in the container network, registers via a unique token per container network. Nomads Container can find and interact with each other via this unique token. If a new provider joins the container network the Nomads container in the provider deployment can register via this unique token. This not only allows us to dynamically extend the Nomads container network if new providers are being added it also enables multiple independent Nomads container networks by using different unique tokens.

Stakeholder questions that need to be answered by a composition instantiation can arrive at any request router (RR) in the container network. The RR then hands over the request to the constraint manager (CM) component to determine if the potential instantiation is feasible based on previous requests. If no information about prior executions is available, the CM invokes the service migration manager (SMM) to find a valid instantiation in coordination with all known partner containers. The container SMMs elect one master SMM to perform the search for a valid instantiation. The SMM components share their provider's information exchange constraints with the elected master SMM. The master SMM then searches for a valid instantiation relying on the modularly designed Solution Component (SC) within the SMM.

The SC can utilize different pluggable algorithms to find valid instantiations, for demonstration purposes we implemented a very basic algorithm. It uses a depth-first search to find a possible solution, which is sufficient in the context of this paper since we focus on finding valid solutions for previously infeasible composition instances. First the number of Abstract Services is determined, which can vary depending on the service composition. Then for each Abstract Service a possible Concrete Service and provider combination is assigned. This assignment respects possible migration policies. If the assignment is valid according to the Data Constraints the Concrete Service provider pair is fixed and the algorithm recurses over the remaining Abstract Services. This traverses the search space until a possible solution is found.

The SC component can be extended to utilize more sophisticated optimizations like genetic algorithms. This also allows to incorporate multiple objectives for optimizations enabling for example, not just finding an instantiation but the optimal one, or one with minimal migrations etc. When a suitable instantiation was found, the respective service migration managers request all necessary service migrations from their partner containers to initiate a valid instantiation. The migrations are performed by moving Service Containers between providers. The Nomads framework offers a pluggable mechanism to implement different migration approaches. In the current version of the prototype, services are relocated using application container migration, providing an implementation-agnostic way for transferring service code, suitable for stateless services. Control is subsequently returned to the CM, which forwards the request to the appropriate local or transferred service instances to answer the stakeholder's question. An exemplary sequence diagram illustrating request handling is shown in figure 6.

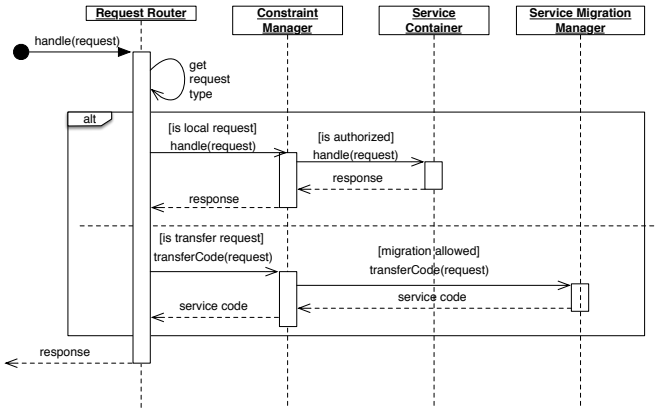


Fig. 6: Nomads Request Sequence Diagram

All service requests performed during the execution of a composition instantiation are intercepted by the RR to enforce all modeled constraints and ensure the sandboxed execution of transferred services on "foreign" premises. The CM component is integrated with the SeCoS (Secure Collaboration in service-based Systems) framework [7], but also allows the utilization of different constraint checking mechanisms.

IV. Validation

For validation purposes we create three CoreOS[6] based clusters representing different providers from our URBEM scenario, each of them consisting of three CoreOS hosts. On each of these hosts we deploy different Docker[7] based service containers representing various concrete services. Each of these concrete services represents an specific instance of an abstract service relevant to satisfy the example process illustrated in figure 2. Additionally to these service containers we deploy service containers representing random services with varying loads and communication patterns to simulate an environment closer to a real world setting. In each of these clusters we further deploy one Nomads container. In each of these clusters there is a private Docker Registry present.

[5]https://github.com/coreos/etcd

[6]https://coreos.com/

[7]https://www.docker.com/

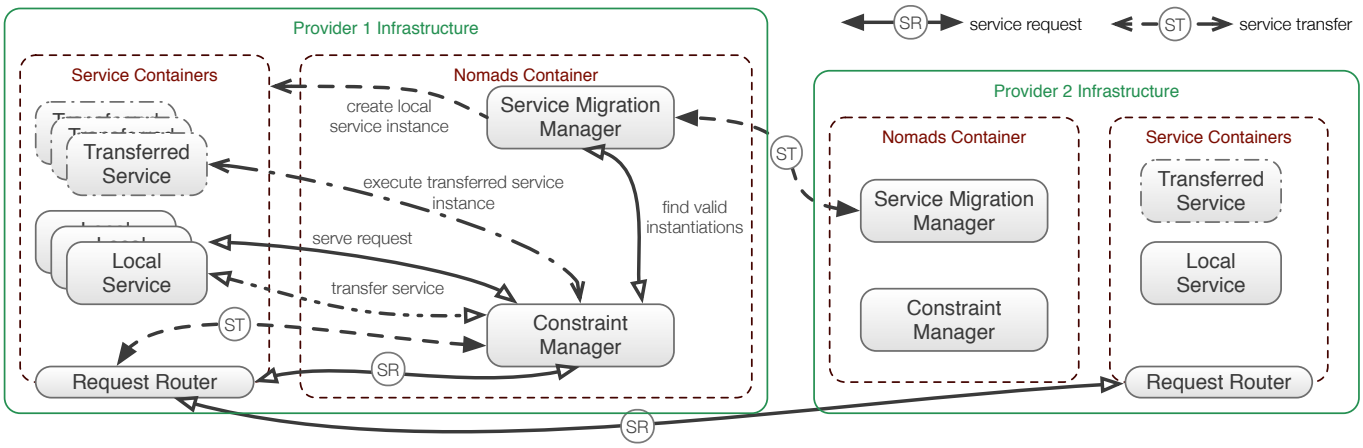


Fig. 5: Nomads Architecture

To demonstrate the actual migration of Service containers we transfer Docker images between the private registries and start them accordingly in their new location. This sample deployment is illustrated in figure 7.

We then generate an exemplary exchange constraint matrix ensuring that several service containers need to be transfered in order to satisfy the execution of our example process. Based on these setting we execute the example process and show that the necessary service migrations can be performed in order to satisfy the exchange constraint and successfully execute the process. The framework sample implementation as well as all scripts to reproduce the validation environment can be found on Github[8].

A. *Evaluation*

Additionally to our validation we perform an evaluation to test the claim that our framework increases the satisfiability of a DAE. For the evaluation, we generate all directed acyclic graphs representing an interaction of up to 7 Abstract Services, which represents the current maximum of Abstract Services in URBEM. This amounts to 2^{21} different graphs representing service compositions to answer possible stakeholder questions. For these graphs we then generate a random set of Concrete Services with random deployments distributed among the 6 providers in the URBEM scenario. Based on this we create a data constraint matrix representing the Interaction Constraints between the providers.

We start with a theoretical optimum that allows every provider to exchange data with every other provider to determine the theoretical optimal baseline. To show the impact of data exchange constraints on satisfiability of composition instantiations, we increase the number of active constraints in 10% increments up to a maximum of 90%, leading to a scenario where only 10% of all possible provider interactions are allowed. For each constraint percentage and graph we determine the number of possible solutions without and with migrations.

[8]https://github.com/jomis/nomads

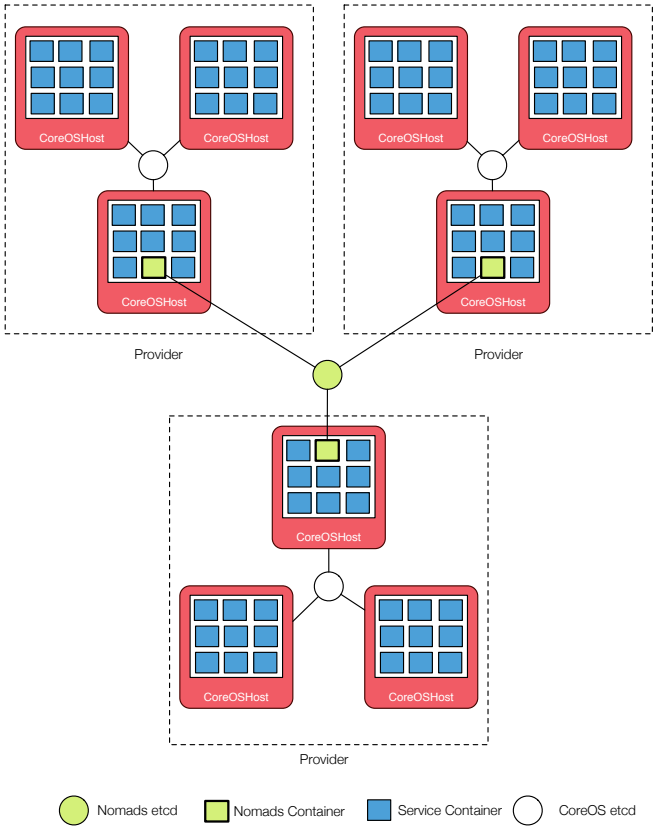
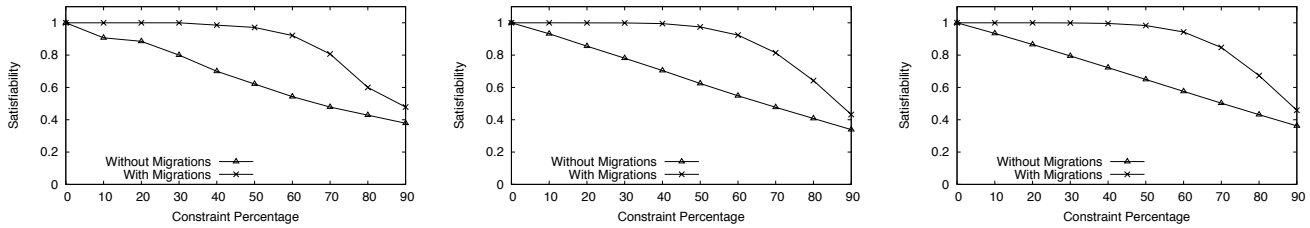


Fig. 7: Deployment for validation purposes

Figure 8 shows the results of our evaluation for $|A|$ $3-7$. We see that with the increase of Abstract Services the amount of possible solutions and therefore also the minimal possible satisfiability increases. This also leads to a more linear decrease of satisfiability both with and without migrations. In conclusion Figures 8a-8c clearly illustrate that our approach with service migrations significantly improves satisfiability of composition instantiations.



(a) Evaluation Results with $|A| = 3$ (b) Evaluation Results with $|A| = 5$ (c) Evaluation Results with $|A| = 7$

Fig. 8: Evaluation Results

V. Related Work

Due to the holistic nature of compliance constrained Distributed Analytical Environments relevant related research areas cover the following topic like (i) compliance and constraint based service compositions, (ii) constraint satisfaction and enforcement in Role Based Access Control (RBAC) systems, and (iii) service and code mobility mechanisms that can be used to deal with constraints in terms of compliance or availability.

Daniel et al. present challenges in SOA-based compliance governance [8], by defining research goals in compliance governance and a compliance management life cycle. One of the goals, which is related to our scenario, is data outsourcing, where privacy constraints are enforced by combining data fragmentation with encryption. In addition to the definition of the basic concepts of compliance, [9] proposes an approach that supports data in both service choreography modeling and analysis. The authors present a model that uses data-aware interactions as the basic event. Based on this model, choreographies can be analyzed with the focus on data compliance, by avoiding state space explosion. Next to compliance another important aspect in service compositions are constraints in general, therefore Zhao et al. present a service composition model based on constraints [10], where requirements are defined as a group of constraints for an abstract service workflow. On top of the defined constraints a concrete service workflow can be generated by binding an activity with an appropriate service in terms of constraint satisfaction. Although the authors present the overall approach, they do not consider the workflow verification and validation at run time. Instead of just defining the overall model for creating constraint-aware service compositions, [11] proposes a constraint-aware service composition method based on two concepts, service intension and service extension. The authors use a graph-based search algorithm to generate all feasible solutions for a general service composition problem. Aggarwal et al. [12] present a constraint driven web service composition approach in the METEOR-S framework, which enables the composition of web services, based on both business and process constraints. The general idea of the authors is to transform the overall service composition problem in a general constraint satisfaction problem. Our problem domain of data exchange constraints also relates to the field of quality attributes and service level agreements (SLAs), where constraints have been intensively studied, for instance in the recent work by Ivanovic et al. [13].

Another related field is the area of access constraints, including Role Based Access Control (RBAC), in the context of web services. Hummer et al. [7] propose *SeCoS*, a framework for model-driven definition of RBAC constraints in service-based business processes. The authors present a runtime enforcement mechanism which intercepts service invocations and therefore prevents the actual invocation in case of a policy violation. As the proposed approach is generic and not limited to RBAC constraints, our framework builds on SeCoS and uses it as the foundation for constraint enforcement. Memon et al. [14] present the SECTISSIMO framework, which aims at modeling constraints in security-critical services. Faravelon et al. [15] propose access restrictions in service compositions based on Computational Tree Logic. In contrast to our work, none of the aforementioned works considers service mobility as a solution to resolve data access and data exchange constraints. Since there are several techniques available that can be facilitated to deal with constraints in service compositions, we focus on the concept of service migration. Amoretti et al. [16] propose an approach that facilitates a code mobility mechanism in the cloud. Based on this mechanism, services can be replicated to provide a highly dynamic platform and increase the overall service availability. Rao et al. [17] provide an extensive survey of automated web service composition methods, together with Sirin et al. [18] this provides a good point of departure for potential composition implementations. Next to a framework for service migration, [19] discusses a cost model and a genetic decision algorithm that addresses the tradeoff on both service selection and migration in terms of costs, to find a optimal service migration solution. As the mobility of code plays an important role in the overall service migration process, Carzaniga et al. [20], [21] provide a study of code mobility paradigms. The authors classify mobile systems into three categories: remote evaluation, code on demand, and mobile agent. Based on these categories the authors discuss abstractions that are related, to those in traditional architectural styles. The work In addition to the aforementioned categories, [22] describes mobile code paradigms with regard to network security vulnerabilities. Following these definitions and paradigms there are several centralized [23], [24] and decentralized [25], [26] approaches available to implement code mobility. Especially the approach proposed in [25], where Arden et al. describe a decentralized computing platform for running mobile code, based on explicit policies for confidentiality and integrity, will be further investigated in the context of our scenario and possibly applied as one of the migration techniques.

In this paper we focus on the specific challenges of holistic aspects of smart city analytical environments. The approaches discussed above are orthogonal to our framework and can be considered to implement framework aspects.

VI. Conclusion

In this paper we presented Nomads a framework for service mobility in Distributed Analytical Environments. We described the URBEM scenario as an example of a constrained Distributed Analytical Environment (DAE). Based on this scenario we outlined the main problem and requirements of aforementioned DAEs and delivered a comprehensive problem formalization. We introduced Nomads to address this problem, described its architecture as well as a solution finding algorithm and concluded with an validation and evaluation that clearly showed that the framework is feasible and that we could significantly increase satisfiability of constrained DAEs.

The problem of service mobility is a core issue in DAEs, and we anticipate that it will gain increased momentum, also in the broader service research community and considering upcoming trends like Big Data [27] or Microservices [28]. Our initial findings and results are promising, and we are currently exploring various directions to further optimize our approach. As part of our future work, we are currently studying advanced techniques towards minimization of migrations by adding new solution finding algorithms, for instance heuristics based on genetic algorithm. Additionally, we investigate the utility of Nomads in scenarios with less coordination, for example in the context of decentralized service choreography.

References

[1] M. Naphade, G. Banavar, C. Harrison, J. Paraszczak, and R. Morris, "Smarter Cities and Their Innovation Challenges," *Computer*, vol. 44, no. 6, pp. 32–39, Jun. 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5875937

[2] M. Kehoe and P. Nesbitt, "Front cover Smarter Cities Series : A Foundation for Understanding IBM Smarter Cities."

[3] W. M. da Silva, A. Alvaro, G. H. Tomas, R. A. Afonso, K. L. Dias, and V. C. Garcia, "Smart cities software architectures: a survey," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, 2013, pp. 1722–1727.

[4] F. Li, M. Vögler, S. Sehic, S. Qanbari, S. Nastic, H.-L. Truong, and S. Dustdar, "Web-scale service delivery for smart cities," *Internet Computing, IEEE*, vol. 17, no. 4, pp. 78–83, July 2013.

[5] H.-L. Truong and S. Dustdar, "Principles of software-defined elastic systems for big data analytics," in *International Workshop on Software Defined Systems in conjunction with IC2E*. IEEE, 2014.

[6] P. Agrawal, "Raft: A recursive algorithm for fault tolerance." in *ICPP*, 1985, pp. 814–821.

[7] W. Hummer, P. Gaubatz, M. Strembeck, U. Zdun, and S. Dustdar, "Enforcement of entailment constraints in distributed service-based business processes," *Information and Software Technology*, vol. 55, no. 11, pp. 1884–1903, Nov. 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950584913001006

[8] F. Daniel, F. Casati, V. D'Andrea, E. Mulo, U. Zdun, S. Dustdar, S. Strauch, D. Schumm, F. Leymann, S. Sebahi, F. de Marchi, and M.-S. Hacid, "Business Compliance Governance in Service-Oriented Architectures," in *International Conference on Advanced Information Networking and Applications*. IEEE, 2009, pp. 113–120. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=5076188

[9] H. N. Nguyen, P. Poizat, and F. Zaidi, "Automatic skeleton generation for data-aware service choreographies," in *24th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2013, pp. 320–329. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=6698885

[10] H. Zhao and H. Tong, "A Dynamic Service Composition Model Based on Constraints," in *6th Int. Conf. on Grid and Cooperative Computing (GCC)*, 2007, pp. 659–662. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4293843

[11] P. Wang, Z. Ding, C. Jiang, and M. Zhou, "Constraint-Aware Approach to Web Service Composition," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 6, pp. 770–784, Jun. 2014. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6637082

[12] R. Aggarwal, K. Verma, J. Miller, and W. Milnor, "Constraint Driven Web Service Composition in METEOR-S," 2004.

[13] D. Ivanović, M. Carro, and M. V. Hermenegildo, "A constraint-based approach to quality assurance in service choreographies," in *Service-Oriented Computing*. Springer, 2012, pp. 252–267.

[14] M. Memon, M. Hafner, and R. Breu, "Sectissimo: a platform-independent framework for security services," in *Modeling Security Workshop at MODELS*, 2008.

[15] A. Faravelon, S. Chollet, C. Verdier *et al.*, "Configuring private data management as access restrictions: from design to enforcement," in *Service-Oriented Computing*. Springer, 2012, pp. 344–358.

[16] M. Amoretti, M. C. Laghi, F. Tassoni, and F. Zanichelli, "Service migration within the cloud: Code mobility in SP2A," in *2010 International Conference on High Performance Computing & Simulation*. IEEE, Jun. 2010, pp. 196–202. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=5547130

[17] J. Rao and X. Su, "A survey of automated web service composition methods," in *Semantic Web Services and Web Process Composition*. Springer, 2005, pp. 43–54.

[18] E. Sirin, J. Hendler, and B. Parsia, "Semi-automatic composition of web services using semantic descriptions," in *1st Workshop on Web Services: Modeling, Architecture and Infrastructure*, 2003, pp. 17–24.

[19] W. Hao, I.-L. Yen, and B. Thuraisingham, "Dynamic Service and Data Migration in the Clouds," in *2009 33rd Annual IEEE International Computer Software and Applications Conference*, vol. 2. IEEE, 2009, pp. 134–139. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5254135

[20] A. Carzaniga, G. P. Picco, and G. Vigna, "Designing distributed applications with mobile code paradigms," in *19th International Conference on Software Engineering (ICSE)*. ACM Press, May 1997, pp. 22–32. [Online]. Available: http://dl.acm.org/citation.cfm?id=253228.253236

[21] ——, "Is Code Still Moving Around? Looking Back at a Decade of Code Mobility," in *29th International Conference on Software Engineering (ICSE'07 Companion)*. IEEE, May 2007, pp. 9–20. [Online]. Available: http://dl.acm.org/citation.cfm?id=1248821.1248922

[22] R. Brooks, "Mobile code paradigms and security issues," *IEEE Internet Computing*, vol. 8, no. 3, pp. 54–59, May 2004. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1297274

[23] D. Chandra, C. Fensch, W. Hong, L. Wang, E. Yardimci, and M. Franz, "Code generation at the proxy: An infrastructure-based approach to ubiquitous mobile code," in *ECOOP Workshop on Object-Orientation and Operating Systems*, 2002.

[24] H. Lufei and W. Shi, "Fractal: A mobile code based framework for dynamic application protocol adaptation in pervasive computing," in *19th IEEE International Parallel and Distributed Processing Symposium*, 2005.

[25] O. Arden, M. D. George, J. Liu, K. Vikram, A. Askarov, and A. C. Myers, "Sharing Mobile Code Securely with Information Flow Control," in *2012 IEEE Symposium on Security and Privacy*. IEEE, May 2012, pp. 191–205. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6234413

[26] P. Liu and M. Lewis, "Mobile code enabled web services," in *IEEE International Conference on Web Services (ICWS)*, July 2005, pp. 167–174 vol.1.

[27] S. Shekhar, V. Gunturi, M. R. Evans, and K. Yang, "Spatial big-data challenges intersecting mobility and cloud computing," in *Proceedings of the Eleventh ACM International Workshop on Data Engineering for Wireless and Mobile Access*. ACM, 2012, pp. 1–6.

[28] J. Lewis and M. Fowler. (2014, Mar.) Microservices. [Online]. Available: http://martinfowler.com/articles/microservices.html