Requirements-Based Composition of Tailored Decision Support Systems

Jonas Kirchhoff(✉), Christoph Weskamp, and Gregor Engels

Software Innovation Lab, Paderborn University, Paderborn, Germany
{jonas.kirchhoff,christoph.weskamp,engels}@upb.de

Abstract. Corporate decision makers have individual requirements for decision support influenced by business goals, regulatory restrictions or access to resources such as data. Ideally, decision makers could quickly create tailored decision support systems (DSS) themselves which optimally address their individual requirements for decision support. Although service-oriented architectures have been proposed for DSS customization, they are primarily targeting trained software developers and cannot immediately be adapted by decision makers or domain experts with little to no software development knowledge. In this paper, we therefore motivate an assisted process-based service composition approach which can be used by non-developers to create tailored DSS. For assistance during service composition, we contribute a meta-model for the formalization of both decision support requirements and functionality of decision support services. Models created according to the meta-model can be used to detect mismatches between a decision maker's requirements for decision support and services selected in the service composition representing a DSS. Furthermore, the formalizations may even be used for automated service composition given a decision maker's decision support requirements. We demonstrate the expressiveness of our meta-model in the domain of regional energy distribution network planning.

Keywords: Service-oriented DSS · End-user programming · Automated service composition · Decision support system generator

1 Introduction

Business environments are increasingly volatile, uncertain, complex and ambiguous (VUCA). As a result, decision makers must consider many influencing factors with frequent, unpredictable change and unknown cause-effect relationships when making a decision [2,14]. More and more decision makers therefore rely

Partially supported by the North Rhine Westphalian Ministry of Economic Affairs, Innovation, Digitalisation and Energy (MWIDE) through grant 005-2011-0022 and the European Regional Development Fund (ERDF) through grant EFRE-0801186.

Published by Springer Nature Switzerland AG 2022
R. Bernhaupt et al. (Eds.): HCSE 2022, LNCS 13482, pp. 150–162, 2022.
https://doi.org/10.1007/978-3-031-14785-2_10

on interactive computer-based decision support systems (DSS) to assist them in the identification of optimal decisions [18].

As observed during an interdisciplinary research project with industry partners for decision support in the domain of regional energy distribution network planning [10], decision makers traditionally turn to established DSS developers to obtain an "off-the-shelf DSS" for their decision problem. Due to the fact that these DSS frequently come with limited to no customization capabilities, the decision support provided by an off-the-shelf DSS must immediately align with the requirements for decision support of an individual decision maker. Requirements for decision support however are situational in the sense that they vary based on the context in which an individual decision maker operates. For instance, in the domain of regional energy distribution network planning, decision makers can only leverage a cross-sectoral planning approach when managing distribution networks for multiple energy sectors. Decision makers may furthermore prioritize metrics such as network reliability and network reinforcements costs differently, or have different access to resources such as data or time available to identify an optimal decision. Similar situations can be observed in the domain of business model development [8] and supply chain management [4,21]. With respect to these situational factors, it is unlikely that the decision support provided by a non-customizable DSS fully addresses the individual requirements for decision support of a concrete decision maker.

A misalignment of DSS functionality and requirements for decision support leaves decision makers with two alternatives: The first alternative is to use the DSS anyway, thereby potentially basing decisions on suboptimal recommendations computed by the DSS. Suboptimal decisions however endanger the success of a company and – in case the company manages critical infrastructure such as a regional energy distribution network – might even negatively affect society as a whole. The second alternative is working with the DSS developer to extend the decision support provided by the DSS according to the situational requirements of the decision maker. This however is a cost- and time-intensive undertake, which again limits competitiveness in VUCA business environments.

In order to ensure the quick availability of optimal decision support, decision makers should ideally be able to create tailored DSS by themselves. Figure 1 shows how this could work: After selecting an application domain (here: energy distribution network planning), a decision maker can specify their requirements for decision support, e.g., to minimize investment costs (CAPEX) of an electricity network with 2.5 h available to identify an optimal decision. Next, the decision maker can model their decision making process as a business process using the *Business Process Model and Notation* (BPMN) [17]. BPMN seems suitable due to its widespread use and familiarity among many stakeholders [1,13], regardless of their background. Each activity of the decision support process corresponds to an interoperable decision support service provided by a DSS developer. In the example shown in Fig. 1, a demand forecast is generated before the associated network is optimized. Data can be produced at runtime by the decision maker (e.g., `currentDemand`) or as a result of service execution (`electricityDemand`). Out-

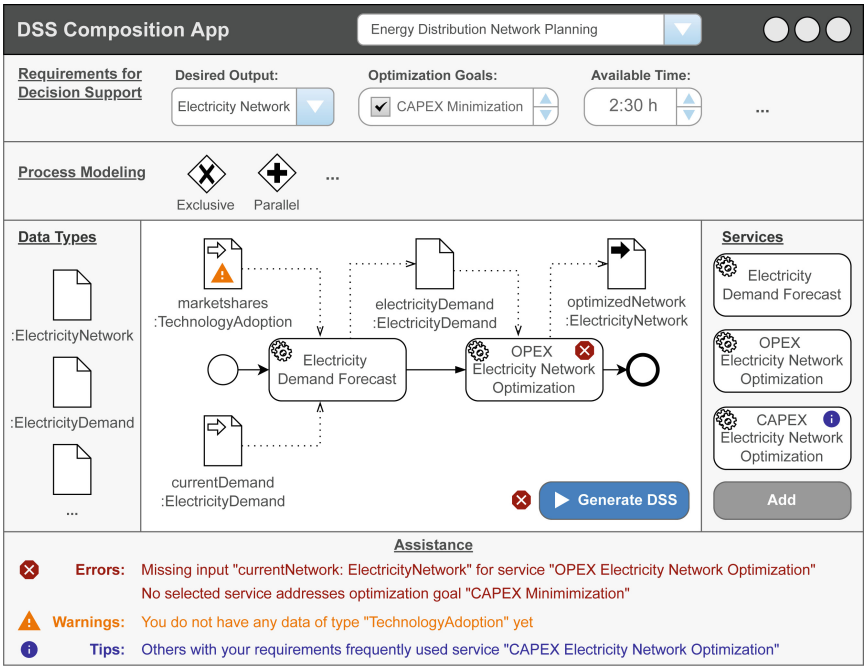

Fig. 1. Mockup for a *DSS composition application*

put data (`optimizedNetwork`) will be returned to the decision maker. An assistance highlights errors, warnings and tips regarding the composition. These can regard service interfaces (e.g., missing or unsuitable input data types), requirements for decision support (e.g., optimizing for a wrong metric), or resource availability (e.g., having no suitable data instances to be used at runtime). After the service composition is completed, the corresponding DSS can be generated and executed. During execution, the decision maker can interact with the DSS as usual. In the background, a workflow engine invokes services with the data provided by the decision maker or generated by other services as described by the underlying process model.

The suitability of BPMN for service composition in service-oriented architectures has already been proven with process-driven applications [20]. This includes assistance in the form of dataflow validation between application services based on service interfaces (e.g., [11] for DSS). In this paper, we therefore focus on laying the foundations for assisting decision makers in selecting services during process composition that align with their decision support requirements. For this purpose, we contribute a meta-model for the formalization of both decision support requirements and service functionality in the context of a software platform for DSS generation shown in Fig. 1. Models created according to the meta-model can be used to detect mismatches between a decision maker's requirements for decision support and services selected by decision makers in the composition

representing a DSS. The formalizations may even be used for automated service composition given a decision maker's requirements for decision support.

The remainder of the paper is structured as follows: We first discuss related work with a focus on service-oriented DSS and service composition in Sect. 2 to support explanations throughout the paper. In Sect. 3, we present requirements for the meta-model which is subsequently explained in Sect. 4. We demonstrate our model in Sect. 5 and conclude the paper with a summary and outlook on future work in Sect. 6.

2 Research Background

In this section, we first present background information on service-oriented DSS (Sect. 2.1) which supports the subsequent explanations throughout the paper. Afterwards, we discuss existing work for automated service composition in service-oriented DSS (Sect. 2.2) which also utilizes formalizations for required/provided service functionality and therefore relates to our approach.

2.1 Service-Oriented DSS

Demirkan and Delen [5] define a *service-oriented DSS* as a DSS with a service-oriented architecture which supports the cross-enterprise design, development, identification, and consumption of reusable services. There are multiple motivations to choose a service-oriented architecture for a DSS. While Demirkan and Delen mention agility in response to change in the context of definition, Figueira et al. [7] mention maintainability in the context of a Software-as-a-Service deployment model, and Horita et al. [9] as well as Stănescu et al. [19] mention the need for cloud platforms' scalability to handle big data sources – an aspect which also plays an important role in Demirkan and Delen's paper. In this context, it is comprehensible that many works (e.g., [9,19,22]) primarily focus on replacement of DSS data sources by decision makers. This is however not sufficient with respect to our project experience presented in the previous Sect. 1, which demonstrates that even computation methods and other decision support functionality must be exchanged based on the situational requirements of decision makers. However, service-oriented architectures are designed for trained software engineers and cannot be immediately adapted by non-developers which prevents the quick availability of an individualized DSS. The same applies for DSS generators without a service-oriented architecture which use mathematical modeling [3] or software frameworks [4] to abstract from concrete programming tasks – skills that decision makers usually do not display [18].

2.2 Automated Service Composition

Automated service composition may be used to account for the lack of programming knowledge by decision makers. Approaches using automated service composition in a DSS context include the work by Kwon [12], Dzemydienė et

al. [6], and Mustafin et al. [16]. They are based on using a similar formalization for both service functionality and a decision maker's requirements for decision support. Based on these formalizations, a service composition can be identified which matches the requirements.

The approach by Mustafin et al. [16] comes closest to the meta-model for requirements and functionality formalization presented in this paper. The approach is based on the OWL-S [15] specification to formalize service functionality (and requirements for decision support). A provided/required service profile includes information about input and output (process) parameters as well as preconditions and side-effects. Non-functional service characteristics such as resource consumption can be specified as service parameters (cf. [15, Fig. 2] and associated explanation). However, this approach lacks some specificity: With respect to functional decision support service characteristics/requirements, it is not possible to specify optimization goals, i.e., which metrics should be minimized and maximized, or which computation method is used. The latter is for instance relevant in energy distribution network planning to be conformant with regulatory constraints. With respect to non-functional service characteristics/requirements, the approach does not provide any guidelines how for instance the consumption of monetary or temporal resources can be specified as a service parameter. These shortcomings will become more evident throughout the next section in which we explain the requirements for the formalization of provided and required decision support service functionality in more detail.

3 Formalization Requirements

In the following, we describe characteristics of decision making which are prerequisites for the formalization of decision support requirements and decision support services. The requirements are derived from the aforementioned research project for decision support in the domain of energy distribution network planning with partners from academia and industry [10].

FR1 – ***Data Constraints.*** Decision support software, independent of whether it is provided in form of a holistic DSS or a modular decision support service, has certain data requirements which must be fulfilled by the decision maker using the software. An individual decision maker however has only limited data access. It is possible to differentiate between constraints on data types (e.g., the required input is an electricity network) and data instances (e.g., the electricity network provided at runtime may not exceed a certain size).

FR2 – ***Method Constraints.*** A method in this context refers to how the value of a metric or the overall decision recommendation is computed. For instance, when assessing the reliability of an electricity network, there might be multiple standards on how this reliability is computed. Regulatory constraints however may only allow a specific standard to be used. If a decision should be optimized with respect to one or multiple metrics such as the investment costs for reinforcement of an electricity network, the considered metrics need to be documented with an associated goal, i.e., minimization or maximization.

FR3 – ***Decision Constraints.*** Especially in presence of decisions which should optimize a metric, it is useful to define constraints on other metrics of the provided decision recommendation. For instance, when minimizing investment costs during electricity network reinforcement, a lower bound for network reliability should be provided from a decision maker's perspective to avoid the recommendation of doing nothing to avoid any investment costs.
FR4 – ***Resource Constraints.*** In addition to data, other limited resources such as money or time are consumed to compute a decision recommendation.
FR5 – ***Execution Constraints.*** The execution mode of a decision support service may also decide whether or not a service can be included in a tailored DSS. For instance, for time-critical decisions, it is important that the selected services have a high availability so that the decision support is actually available if needed. Additionally, the place of service execution, i.e., on-premises or remote, can be relevant when dealing with sensitive data such as clients' energy demands.
FR6 – ***Domain Ontology.*** An ontology is needed to document which entities are present in the application domain under consideration. Furthermore, parameterizing the formalization with respect to an ontology of the application domain makes it extensible to support multiple application domains.

4 Formalization Meta-Model

Our meta-model to address the requirements defined in the preceding Sect. 3 is documented as a UML class diagram and split over multiple packages which are shown in Figs. 2 to 5 and subsequently explained.

The `Types`**-package** shown in Fig. 2 allows the definition of entity types with respect to their `Attribute`s which may be of qualitative or quantitative nature. A `QuantitativeAttribute` can optionally specify a `minValue`, `maxValue` and a `measurementUnit`. A `QualitativeAttribute` may specify a `range` of valid values. `SingleValueQualitativeAttribute` and `MultiValueQualitativeAttribute` document how many of these values may be specified for an `Instance` of the `Type`. `Instance`s of a `Type` are described using the `Instances`**-package**. The `Attribute`-values of an `Instance` are defined via fulfilled `Assertion`s. There is one subclass of `Assertion` for each of the `Attribute`-subclasses. The explanation of the `Types`- and `Instances`-packages is deliberately kept short as they are not useful on their own. Instead, they serve as a base for the other packages.

The `Data`**-package** shown in Fig. 3 defines `DataType`s and associated `DataInstance`s of an application domain. An example for a `DataType` in the example domain of electricity distribution network planning is an *Electricity Network* with a `QuantitativeDataAttribute` *Investment Costs* and a `MultiValueQualitativeAttribute` *Reliability Design* specifying which types of network assets such as "cables" or "transformers" may fail without impacting network functionality. A `DataType` can be expressed by a nonzero amount of `DataFormat`s. A `DataInstance` is a certain data document available at runtime, e.g., a concrete file in a specific `format` describing a concrete *Electricity Network* such as *"my-network.en"*. The `DataInstance` furthermore documents

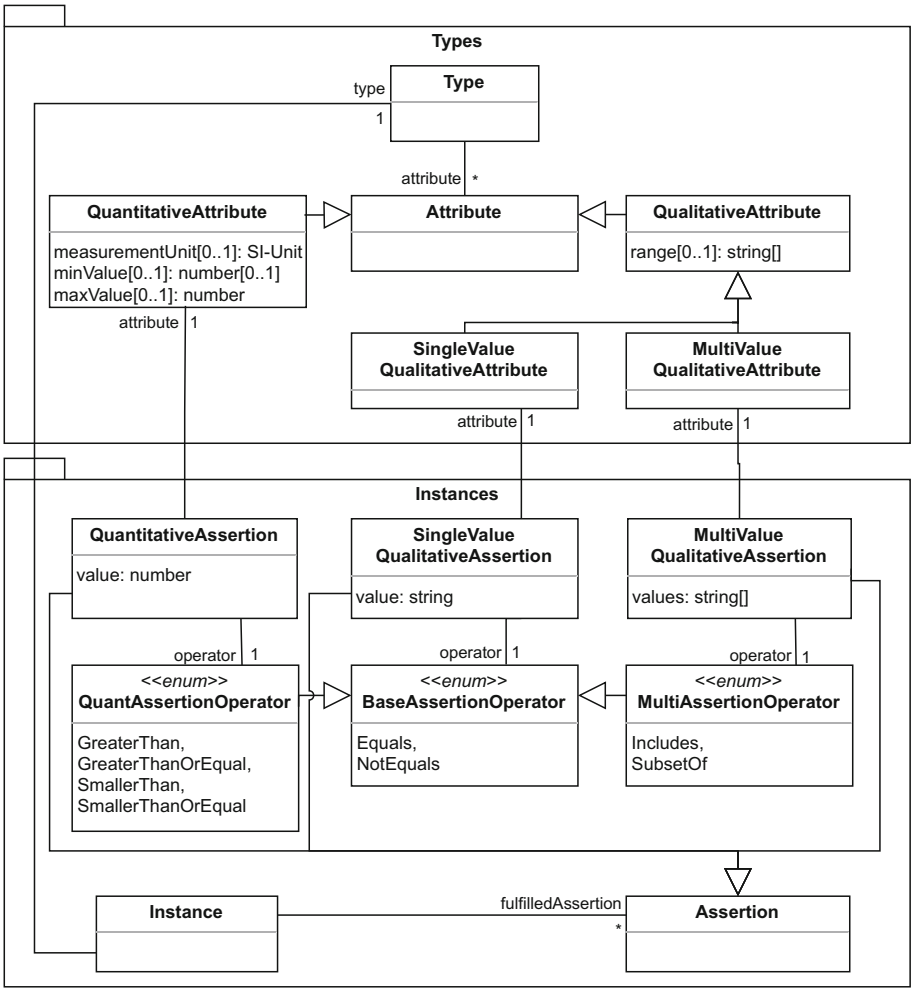

Fig. 2. Meta-model for types and instances

`Attribute`-values via `Assertion`s, e.g., that the *Investment Costs* equal 3.14 million Euro.

Information on how data was or can be computed is documented using the `Computation Methods` package also shown in Fig. 3. Again, the `Types`- and `Instances`-package enable the definition of `ComputationMethodType`s and `ComputationMethodInstance`s respectively. An example for a `ComputationMethodType` would be *Electricity Network Optimization*. Associated `ComputationMethodInstance`s could be optimization approaches such as *Mathematical Exact Network Optimization* or *Heuristic Network Optimization*. In case of optimization, the optimization target is documented as an `Objective`. The `Objective` documents the `QuantitativeDataAttribute` which is optimized

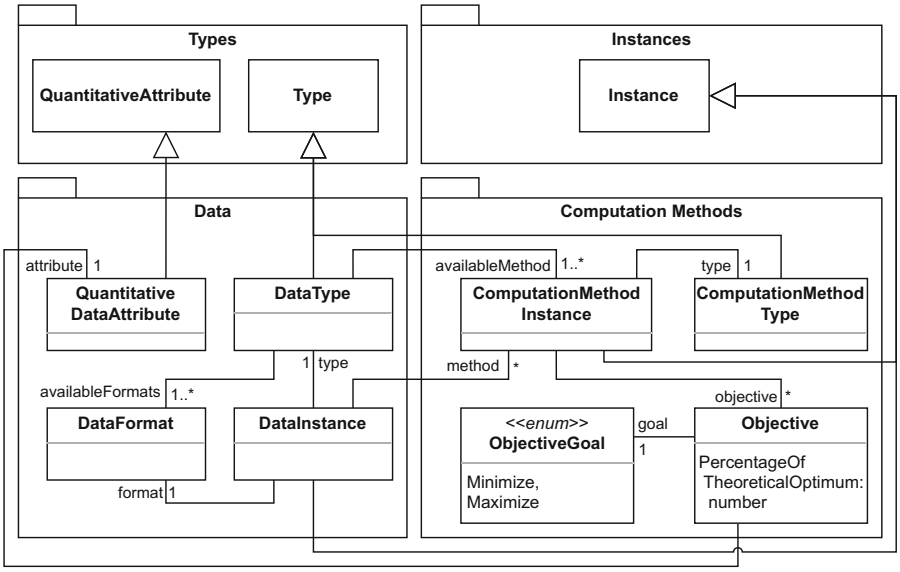

Fig. 3. Meta-model for data and computation methods

and the `goal` of the optimization, e.g., *Minimize Investment Costs* in case of an *Electricity Network*. Furthermore, the gap to a theoretical optimum can be specified, e.g., to document that a heuristic approach only achieves 80% of the theoretically achievable optimum. The available `ComputationMethodInstance`s are documented for each `DataType`. Each `DataInstance` documents the actually used `ComputationMethodInstance`. The design choice to document the used computation method at data-level instead of service-level is that decision makers may reuse data computed during a previous interaction with a tailored DSS. In this case, the information about how the data was computed should not get lost.

The `Service Interface`-package shown in Fig. 4 describes the `input` and `output` of a `Decision Support Service` as a `Slot`. Each `Slot` specifies the `data` it consumes/produces with an associated `DataQuantity` documenting the minimum and maximum quantity of `DataInstance`s that are consumed/produced. Constraints on the consumed and produced data as well as the used computation method for the produced data can be annotated using the previously discussed packages of Fig. 3. In the context of electricity network planning, a `Decision Support Service` might be a service encapsulating a concrete linear optimization model which minimizes investment costs given the current state of the network and expected future electricity demands.

The execution of a `Decision Support Service` such as *Remote* or *On-Premises* is specified using the `Service Execution`**-package** (cf. Fig. 4). Execution consumes resources as described using the `Resource Consumption`**-package**. A `ResourceConsumption` is specified as a combi-

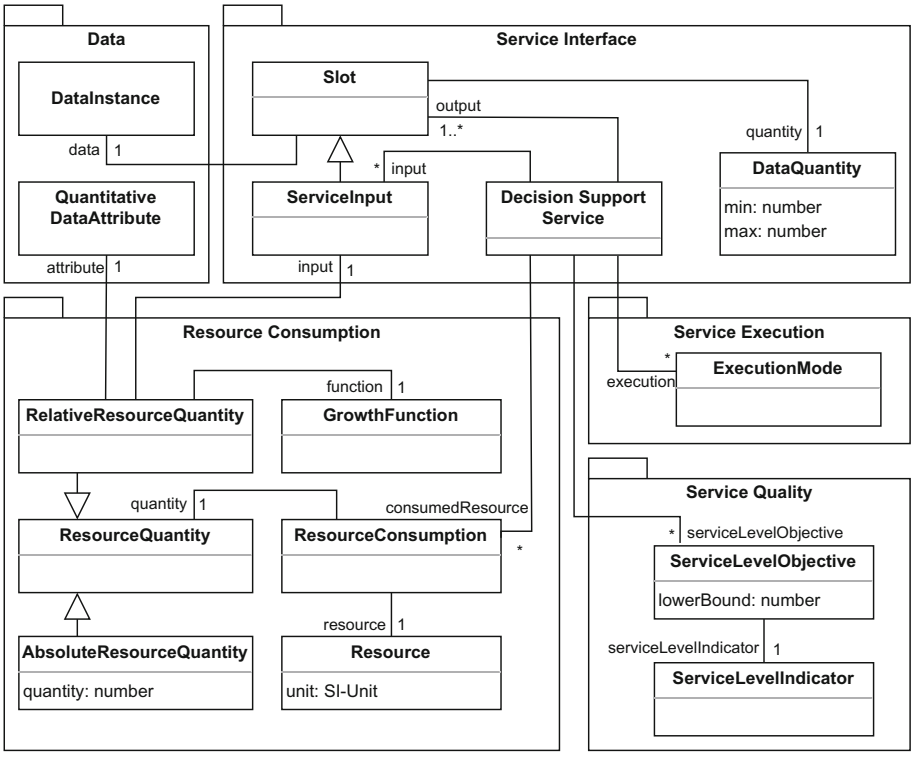

Fig. 4. Meta-model for the formalization of decision support services

nation of a `Resource` such as *time in seconds* or *money in Euro* and a `ResourceQuantity`. The quantity of the resource can either be specified "statically" as an `AbsoluteResourceQuantity` or "dynamically" as a `RelativeResourceQuantity`, i.e., relative to a `QuantitativeDataAttribute` of a `ServiceInput` given a mathematical `GrowthFunction`. For instance, the previously mentioned service for electricity network optimization using a mathematical optimization model may specify a runtime which is *exponential* in the *size* of the *electricity network* provided as input.

Lastly, a `Decision Support Service` exhibits certain quality characteristics as described by the `Service Quality-package` (cf. Fig. 4). Each quality characteristic is described using a `ServiceLevelObjective` which specifies a `lowerBound` with respect to a `ServiceLevelIndicator`. For instance, a service may specify an *availability of 99% or higher*.

Another package provides the capabilities to model decision makers' requirements for decision support. It builds upon the previously explained packages by introducing a `DecisionMaker` class to document desired output and available input `DataInstance`s, available `Resource`s and acceptable service qualities. Due

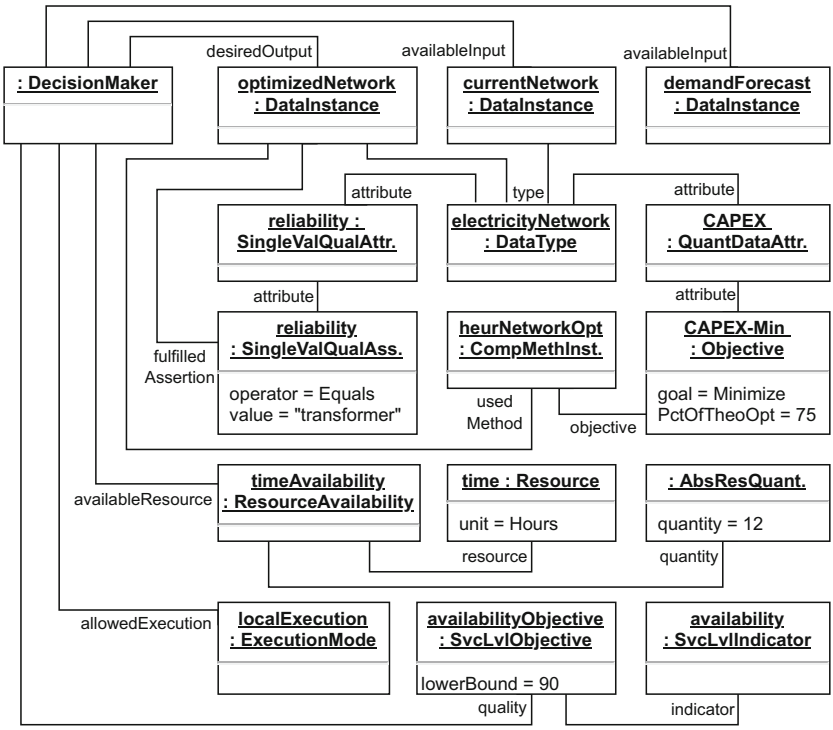

Fig. 5. Example model for decision support requirements

to its simplicity, we do not explain the package here but use it for the demonstration in the upcoming section.

5 Demonstration and Discussion

We instantiated our meta-model in the domain of energy distribution network planning [10] to demonstrate its applicability. An excerpt documenting a decision maker's requirements for electricity network planning is given by the UML object diagram depicted in Fig. 5: The desired output of a decision maker is an electricity network which minimizes investment costs (CAPEX) while ensuring network reliability with respect to transformers. For this purpose, the decision maker can provide the current electricity network and a demand forecast (whose data type is not shown in the diagram for simplification purposes). The decision maker prefers the use of a heuristic optimization method which guarantees at least 75% of the theoretical optimum. The execution may take up to 12 h and can only be performed on local hardware with 90% service availability.

Other excerpts of the case study were given as textual examples throughout the previous explanations of meta-model packages. Our case study confirmed that the meta-model addresses the formalization requirements presented

in Sect. 3. The `Data`-package is a key enabler to address FR1 – *Data Constraints* as it allows the documentation of produced and consumed data in the `Service`-package as well as the documentation of data available to and desired by a decision maker. The assertion on data instances also addresses FR3 – *Decision Constraints*. Requirement FR2 – *Method Constraints* is addressed by the `Computation Methods`-package. Due to the relation between computation methods and data depicted in Fig. 3, decision makers can also include the computation method as a requirements for decision support when specifying their desired output. However, we found that the meta-model should be adapted so that decision makers can specify multiple allowed computation methods. Requirement FR4 – *Resource Constraints* is addressed by the `Resource Consumption`-package in the context of services and by documenting a decision maker's resource availability. Packages `Service Execution` and `Service Quality` address FR5 – *Execution Constraints*. Lastly, FR6 – *Domain Ontology* is mainly enabled by the `Types`- and `Instances`-package as well as their subclasses and the overall instantiation of the meta-model classes for a given application domain.

6 Summary and Future Work

Decision makers require the quick availability of tailored decision support systems that optimally assist them in their individual decision making process. We proposed that, ideally, decision makers should create these tailored DSS themselves based on a service-oriented DSS architecture. However, since decision makers often have no software development knowledge, they either need to be assisted during the composition of decision support services, or service composition must happen automatically based on requirements for decision support specified by a decision maker. Both approaches require a formalization of decision support functionality requested by decision makers and provided by decision support services in order to match those two together. In this paper, we therefore contributed a meta-model for the formalization of decision support functionality. It can be used both for the formalization of decision support service functionality and decision makers' requirements for decision support. The meta-model extends on existing approaches by increasing expressiveness with respect to functional and non-functional decision support characteristics. In particular, these extensions include the method and metrics used for optimization as well as service-level objectives. We demonstrated the expressiveness of our meta-model by applying it to an example in the application domain of regional energy distribution network planning.

As stated before, the presented meta-model is only the foundation for matching decision support requirements and functionality. Naturally, a next step in future work is to develop a validation algorithm which checks whether a service composition described according to our meta-model addresses all (or some) of the modeled requirements for decision support of an individual decision maker. On the one hand, this approach can be used for validating that a service composition assembled by a decision maker addresses all requirements for decision

support (cf. assistance in Fig. 1). On the other hand, given a formalization of requirements for decision support, the approach can be used to match decision support services and service compositions in context of an automated service composition approach. Before implementing and utilizing this validation algorithm, we may try to merge our meta-model with OWL-S to leverage existing service matching and composition functionality based on the W3C recommendation. The mapping onto OWL-S would also immediately give us an XML-based concrete syntax, which otherwise would need to be defined before the models can be algorithmically processed.

References

1. Awad, A., Decker, G., Lohmann, N.: Diagnosing and repairing data anomalies in process models. In: Rinderle-Ma, S., Sadiq, S., Leymann, F. (eds.) BPM 2009. LNBIP, vol. 43, pp. 5–16. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12186-9_2
2. Bennett, N., Lemoine, G.J.: What a difference a word makes: understanding threats to performance in a VUCA world. Bus. Horiz. **57**(3), 311–317 (2014)
3. Bhargava, H., Sridhar, S., Herrick, C.: Beyond spreadsheets: tools for building decision support systems. Computer **32**(3), 31–39 (1999)
4. Brodsky, A., Al-Nory, M., Nash, H.: SC-CoJava: a service composition language to unify simulation and optimization of supply chains. In: Dolk, D., Granat, J. (eds.) Modeling for Decision Support in Network-Based Services. LNBIP, vol. 42, pp. 118–142. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27612-5_6
5. Demirkan, H., Delen, D.: Leveraging the capabilities of service-oriented decision support systems: putting analytics and big data in cloud. Decis. Support Syst. **55**(1), 412–421 (2013)
6. Dzemydienė, D., Maskeliūnas, S., Miliauskas, A., Naujikienė, R., Dzemydaitė, G.: E-service composition for decision support, based on monitoring of contamination processes and analysis of water resource data. Technol. Econ. Dev. Econ. **21**(6), 869–884 (2015)
7. Figueira, G., Amorim, P., Guimarães, L., Amorim-Lopes, M., Neves-Moreira, F., Almada-Lobo, B.: A decision support system for the operational production planning and scheduling of an integrated pulp and paper mill. Comput. Chem. Eng. **77**, 85–104 (2015)
8. Gottschalk, S., Yigitbas, E., Nowosad, A., Engels, G.: Situation- and domain-specific composition and enactment of business model development methods. In: Ardito, L., Jedlitschka, A., Morisio, M., Torchiano, M. (eds.) PROFES 2021. LNCS, vol. 13126, pp. 103–118. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-91452-3_7
9. Horita, F.E., de Albuquerque, J.P., Marchezini, V., Mendiondo, E.M.: Bridging the gap between decision-making and emerging big data sources: an application of a model-based framework to disaster management in Brazil. Decis. Support Syst. **97**, 12–22 (2017)
10. Kirchhoff, J., Burmeister, S.C., Weskamp, C., Engels, G.: Towards a decision support system for cross-sectoral energy distribution network planning. In: Breitner, M.H., et al. (eds) Energy Informatics and Electro Mobility ICT (2021)

11. Kirchhoff, J., Gottschalk, S., Engels, G.: Detecting data incompatibilities in process-driven decision support systems. In: Business Modeling and Software Design. Springer (2022, to appear)
12. Kwon, O.B.: Meta web service: building web-based open decision support system based on web services. Expert Syst. Appl. **24**(4), 375–389 (2003)
13. Lohmann, N., Nyolt, M.: Artifact-centric modeling using BPMN. In: Pallis, G., Jmaiel, M., Charfi, A., Graupner, S., Karabulut, Y., Guinea, S., Rosenberg, F., Sheng, Q.Z., Pautasso, C., Ben Mokhtar, S. (eds.) ICSOC 2011. LNCS, vol. 7221, pp. 54–65. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-31875-7_7
14. Mack, O., Khare, A.: Perspectives on a VUCA world. In: Mack, O., Khare, A., Krämer, A., Burgartz, T. (eds.) Managing in a VUCA World, pp. 3–19. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-16889-0_1
15. Martin, D., et al.: OWL-S: semantic markup for web services. W3C Member Submission (2004). https://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/
16. Mustafin, N., Kopylov, P., Ponomarev, A.: Knowledge-based automated service composition for decision support systems configuration. In: Silhavy, R., Silhavy, P., Prokopova, Z. (eds.) CoMeSySo 2021. LNNS, vol. 231, pp. 780–788. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-90321-3_63
17. Object Management Group: Business process model and notation (2014). https://www.omg.org/spec/BPMN/. Accessed 21 June 2022
18. Savić, D.A., Bicik, J., Morley, M.S.: A DSS generator for multiobjective optimisation of spreadsheet-based models. Environ. Modell. Softw. **26**(5), 551–561 (2011)
19. Stănescu, I.A., Ştefan, A., Filip, F.G.: Cloud-based decision support ecosystem for renewable energy providers. In: Camarinha-Matos, L.M., Baldissera, T.A., Di Orio, G., Marques, F. (eds.) DoCEIS 2015. IAICT, vol. 450, pp. 405–412. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16766-4_43
20. Stiehl, V.: Definition of process-driven applications. In: Process-Driven Applications with BPMN, pp. 13–41. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-07218-0_2
21. Weskamp, C., Koberstein, A., Schwartz, F., Suhl, L., Voß, S.: A two-stage stochastic programming approach for identifying optimal postponement strategies in supply chains with uncertain demand. Omega **83**, 123–138 (2019)
22. Zhang, C., Zhao, T., Li, W.: The framework of a geospatial semantic web-based spatial decision support system for digital earth. Int. J. Digit. Earth **3**(2), 111–134 (2010)