A QoS-aware Service Composition Approach based on Semantic Annotations and Integer Programming

I. INTRODUCTION

Service Oriented Architecture (SOA) is a widely adopted paradigm for developing distributed applications by leveraging on capabilities offered by distributed heterogeneous systems while easing their future extensions and adaptations. Service-oriented specifications and tools help designers and developers in describing and publishing software capabilities as services and potential clients in discovering, locating and invoking target capabilities.

By leveraging on service abstractions, it is possible to provide added-value services by combining existing and "basic" services into a novel "composite" service. Composite services can be defined by designers using some process specification languages, such as BPEL (OASIS, 2007) and BPMN (OMG, 2011). Process specifications can then be executed by workflow engines.

A main goal of ongoing research activities in industry and academic communities is to provide models and mechanisms for automating service composition and selection at run-time. The need for such automation can be characterized according to different facets: on one side the need of distinguishing and automating as much as possible "glue" and general-purpose tasks with respect to domain-specific application logic; on the other side the need of postponing decisions as close as possible to execution time, in order to take into proper account dynamic factors such as application-related context information and non-functional attributes, including Quality of Service (QoS) properties.

Typically, QoS-aware service composition has mostly been tackled by distinguishing two consecutive steps (Agarwal et al., 2005):

- A first step (service composition) aiming at specifying a composite service as a flow of invocation of "abstract services", via user-driven specification or automatic composition approaches. The output is given in the form of "abstract plans" and binding to concrete endpoints is not specified.
- A second step (service selection) aiming at selecting service endpoints among available candidates for services in a given abstract plan. Selection is performed by adopting algorithms optimizing a QoS-based utility function for a given composition plan.

A main drawback of this two-layered approach is that most available implementations lead to sub-optimal solutions in terms of QoS. This is mainly due to the fact that functional and non-functional problems are solved separately. To limit the computation time of solving algorithms, the Service Composition step typically returns a sub-set of existing solutions.. As a consequence, QoS optimization is typically not performed over the whole set of available solutions.

An alternative approach, aiming at finding the best available solution, consists in taking into account QoS non-functional requirements together with functional ones, in order to drive the specification of the service invocation flow. Such observation is also discussed and grounded with examples in (Jiang et al., 2010) and (Yoo et al., 2008).

Some recent works have begun adopting this approach, as discussed in the following section. However, a main limitation of these works consists in the fact that they model functional constraints by relying on syntactical and structural properties of service descriptions. As a consequence, structural and syntactical differences typically existing across different, even similar, service interfaces may make the service composition computation infeasible.

To solve this issue, in this paper we propose a semantic-driven QoS-aware dynamic composition approach based on an Integer Linear Programming (ILP) model (Salkin and Mathur, 1989). The proposed approach, called SEQOIA, is based on the adoption of an ontology-based service profile model that represents structural and semantic properties of service descriptions. By adopting the ILP technique, the service composition problem can be expressed as a set of functional and non-functional constraints and an objective function built on top of a reference domain representing functional, structural and semantic properties of registered service profiles. Semantic properties are exploited to build equivalence relationships between data types of different service interfaces.

The paper is structured as follows: Section II discusses related work and Section III discusses main principles driving this work; Section IV describes in detail the SEQOIA approach and Section V reports results of testing activities for evaluating the performance of the proposed technique and making a comparison with an alternative service composition and selection approach. In Section VI we conclude the paper with insights into future work.

II. RELATED WORK

As mentioned in the previous section, a widely adopted approach consists in distinguishing the QoS-aware service composition problem in two separate sub-problems: a functional problem (service composition) and a non-functional problem (service selection).

Most works focusing on the service composition problem can be distinguished into methods based on workflow techniques and methods based on AI planning techniques. Hereafter we briefly mention some representative approaches and refer the reader to existing surveys (Dustdar and Schreiner, 2005; Rao and Su, 2004) for more details.

In workflow-based composition techniques the service composition problem is defined by means of workflow models. Examples are eFlow (Casati et al, 2000) and the model proposed by (Zhao and Tong, 2007). In AI planning techniques, service composition is modeled as a planning problem.

A planning problem can be represented by a five tuple (S, S0, G, A, Γ), where S is the set of all possible states of the world, $SO \subset S$ denotes the initial state of the world, $G \subset S$ denotes the goal state of the world, A is the set of actions the planner can perform in attempting to change one state to another state in the world, and the translation relation $\Gamma \subseteq S \times A \times S$ defines the precondition and effects for the execution of each action. When applying these concepts to the service composition problem, A represents the set of available services, S0 and G are the initial states and the goal states provided by the composite service requester. Γ further denotes the state change function of each service. Examples of AI-Planning approaches for service composition have been proposed by Zheng and Yan (2008), Oh et al. (2008), and Hewett et al. (2009).

QoS-based service selection assumes the existence of a predefined work plan describing a flow of invocations to abstract services. Its objective is to select for each abstract service the service endpoint meeting some local or global QoS constraints. The approach proposed by Zeng et al. (2004) includes two service selection approaches based on Multiple Criteria Decision Making: a local optimization and a global planning technique based on Integer Programming. Yu et al. (2006) model the service selection problem in two ways: as a multi-dimension multi-choice knapsack problem (MMKP) by adopting a combinatorial model and as a multi-constrained optimal path (MCOP) problem by adopting a graph model. In both ways, a user-defined utility function is specified to optimize application-specific objectives. Alternatively, the work in (Canfora et al., 2007) proposes an approach based on genetic algorithms, which, although slower than Integer Programming approaches, has the advantage of representing also non-linear constraints.

As discussed also by Jiang et al. (2010), two-layered approaches for QoS-aware service composition may often lead to sub-optimal solutions, as the composition plan is defined by taking into account only functional constraints.

With the term "QoS-aware dynamic service composition" we refer to those approaches that take into account both QoS and functional constraints to drive the specification of the optimal service invocation flow. An example of existing approaches for QoS-aware composition is the technique proposed by Jiang et al. (2010), implementing a graph exploration algorithm enhanced with QoS computation; the proposed technique is characterized by the use of an inverted index table to represent the graph and a counting mechanism to judge whether a service is enabled. Yoo et al. (2008) present a technique adopting an Integer Linear Programming model representing non-functional objectives and constraints together with functional constraints representing syntactic matching of Web Services features. More recently, Xu et al. (2011) proposed an algorithm for QoS-based semantic service composition for finding the shortest sequence composition with QoS guarantees. The proposed approach aims at coping with scalability requirements and its performance has been tested against a population of up to 15.000 services.

A main limitation of most existing works consists in the fact that they model functional constraints relying only on syntactical and structural properties of service descriptions. As a consequence, such approaches can work well if data to be delivered in services' input and output messages are specified in XML Schemas by using the same syntactic and structural properties. For example, let's assume that a target composite service can be realized as a sequence of service invocations, e.g., service A and service B, where output data of service A are needed to invoke service B. In this case, the above-mentioned approaches work well if a data element in the output message of service A exactly matches the corresponding data element declaration in service B input message. This condition implies that service A and service B description files should at least refer to the same namespace. Moreover, while this condition could apply quite well for elements "globally" declared in a given namespace, extending its use to locally declared elements is not straightforward and can lead to errors in the execution phase.

As discussed by Tan et al. (2010) and Weber (2007) existing solutions weakly address requirements of real business scenarios. While authors of both works agree on the assumption that functional capabilities are available as services or executable sub-processes can be seen as realistic, they have different positions with respect to semantic-based services description; while Weber considers semantic-based description as a building block for service composition, Tan et al. (2010) proposes a data-driven approach for service composition based on syntactic-based data definition schemas and Petri Nets to meet business needs. However, none of these two works aims at addressing QoS-based composition issues.

III. MAIN DRIVING PRINCIPLES

Hereafter we discuss main principles driving the design of the SEQOIA approach.

A. *Message and data-driven approach*

We rely on the definition of a service as an operation defined by a couple of input and output XML messages. In this sense, a service interface can be modeled as an XML document processor. As discussed by Vogels (2003), *"the XML document is the Web service's keystone because it contains all the application specific information that a service consumer sends to the service for processing".* XML Schema is the reference grammar used for describing syntactic and structural rules of an XML document. The XML Schema description of messages is typically included in an interface description document based on the Web Services Description Language (WSDL). Nonetheless, such interaction model based on XML document processing is typically applied also to represent adapters in the Enterprise Application Integration domain. With the term Enterprise Application Integration (EAI) we refer to the integration of heterogeneous applications and systems via message-based communication enabling the carrying out of enterprise processes. Typically, EAI infrastructures rely on a huge set of specific inbound and outbound adapters used for interacting with the external systems and for converting proprietary message formats. An EAI adapter interaction typically requires an input XML message and optionally produces a resulting output XML message. Thus, the simplest form of an adapter interface is a specific input and output schema for each adapter (Böhm et al., 2008). We thus leverage on an XML message-driven approach in order to support different types of "service-oriented paradigm implementations", including Web Services, as well as EAI adapters.

B. *Lightweight semantic description.*

As mentioned above, XML Schema is the reference language for service and adapter interface description. XML Schema provides specific support for syntactic and structural rules representation.

In real-life contexts, services offering an operation with the same meaning, typically present similar, but not identical interfaces. As an example, one of world's largest services registry, Seekda[i] offers description of several services providing weather information for a given location, but each service has a different service interface. This fact may seriously compromise feasibility of dynamic service composition for the following reasons: a service invocations chain can be broken at specific points if output data of a given service do not correspond with the syntax and structure of a following service, even if they imply the same concepts (e.g. weather information for a given location); analogously, reuse and substitution of services implementations may be compromised. Nonetheless, also elements with the same name but embedded in a different context (i.e. a message, a part in a message) can have a different meaning. Global declaration in a schema for a common namespace can help in establishing equivalence among elements in different contexts. However, such assumption would depend on XML Schema design practices, which are not standardized.

To overcome these issues, data declaration in message description document can be enhanced with references to concepts in an ontology. OWL-S profile (Martin et al. 2005), WSMO capabilities (Roman et Al. 2005) or SAWSDL (Farrell and Lausen, 2007) annotations could be used to model these references. Typically, semantic descriptions can be used to model information, functional, non-functional, as well as behavioral aspects of services to automate service discovery, brokering, composition and invocation. However, adoption of semantic annotations is still limited in business scenarios at it requires much human effort. As our scope is on message-based and data driven service interactions, we chose to minimize the need on semantic-based service description to information and functional aspects. We thus decided to adopt SAWSDL specifications. As discussed by Kopecky et al. (2007), semantic SAWSDL-based annotations at the level of schema and interfaces components in a WSDL document allow to represent information and functional aspects. Moreover, SAWSDL is a W3C recommendation and it offers simple means for associating services description elements with semantic concepts.

C. *Adaptation to dynamic user requirements and QoS policies*

Users may pose different requirements with respect to QoS optimization in service composition. While in principle the desired goal would be to calculate the service composition providing the optimal QoS value over the totality of registered services in as minimal time as possible, we recognize that providing users with a wider set of options for specifying the optimization problem could better cope with real practices. Such options include:

- Specifying global and local constraints on single QoS dimensions.
- Specifying given constraints as mandatory or optional in a composition problem.
- Running the QoS-aware service composition request to replace a part of an existing composite service. For instance, this might be required when the QoS of a chosen service instance is decreased or because new service providers are available.

IV. SEQOIA APPROACH

Main aspects of the proposed SEQOIA approach, namely message-oriented and data-driven model and lightweight semantic description, are supported by the adoption of a service profile model describing functional and information properties of registered services. The adopted service profile model offers primitives for modeling structural and semantic properties of service interfaces by using an ontology-based language, independently from specific service interface description languages (such as WSDL and XML Schema). QoS properties of registered service endpoints are modeled by adopting a general-purpose and extensible Quality Model, inspired by the widely adopted approach proposed by Zeng et al. (2004).

Based on these models, the problem of QoS-aware service composition consists in constructing a graph of interconnected service profiles, where edges represent functional and information properties of service endpoints, which are selected among available candidates to optimize the overall QoS. As described in the following sections, based on the service profile model, we first build a STRIPS-like model for representing entities of the domain for the service composition problem. We chose to adopt an Integer Programming technique (Salkin and Mathur, 1989) to solve the QoS-aware composition problem for the following reasons: as discussed in (Oh et al., 2006) and (Vossen et al., 1999) ILP is a widely adopted approach for solving AI planning problem with good performance profile and it inherently allows to model non-functional constraints and optimization problems. Moreover several Integer Programming solvers exist (e.g. IBM C-PLEX[ii], Gurobi[iii], LPSolve[iv]), and their use is facilitated also by upper modeling languages and interfaces (e.g. OptimJ[v]).

A. *Service Profile Model*

The Service Profile Model adopted in SEQOIA defines modeling primitives for representing structural, semantic and non-functional properties of service interfaces. The model is defined in terms of: Service Entities, representing main functional and structural information of service interfaces and Qualifying Attributes, which can extend the model with additional properties, such as non-functional or domain-specific attributes.

Fig. 1 shows main entities and properties of the service profile model:

- Service Entities represent the basic constructs of a functional profile in terms of Operations, Messages and Atoms.
- An Operation defines the functional capability of a service in terms of input-output message pairs.
- A Message is an XML document.

- An Atom represents an XML node carrying data values in an input or output Message. In well-formed messages, atoms are XML attributes or simple type elements.

Atoms, Messages and Operations are characterized by a Qualified Name (has_qname attribute). An operation may be provided by one or more endpoints (has_endpoint attribute). Atoms are the key entities in our model. Each atom is so characterized by a set of properties:

- Syntactic properties: properties relying on basic rules of the chosen formatting language (XML), such as the local or qualified name ;
- Structural properties, i.e. the hierarchical structure of the message, representing a sort of "context" for a leaf node.
- Semantic properties: the concept representing the meaning implied by the XML node.

Further optional properties (i.e. Qualifying properties) can be added to each input-output message pair to describe the non-functional aspects of the operations performed by the service upon invocation (e.g., Quality of Service properties) or domain-specific attributes (e.g., geo-referenced information).

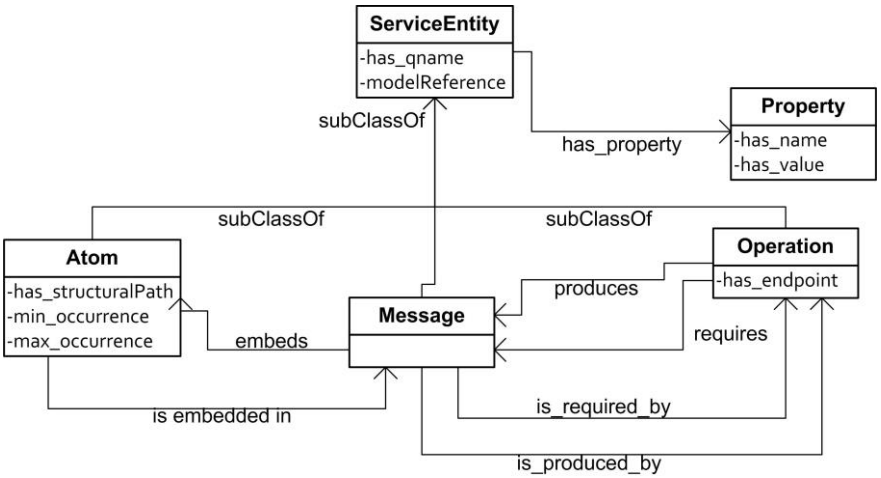

Figure 1. Service Profile Model

1) Structural Properties

Structural properties specify how data are embedded in a message:

- The min_occurrence and max_occurrence attributes specify the allowed instance multiplicity range.
- The type attribute specifies the atom data type (referring to XML Schema pre-defined types).
- The structural_path attribute characterizes the position of the atom in the XML document tree by representing the chain of parent elements up to the document root and parents' content models. More details can be found in (Paganelli et al., 2010).

2) Semantic Properties

Semantic properties link service profile entities, especially atoms, with concepts defined in a given shared domain representation. Semantic properties allow to wrap structural-based atom data with the intended meaning. At the level of semantic reasoning, interpretation and relations among atom data may be built. This may help in:

- decoupling client and service providers interface, as differences in message structure and syntax may be reconciled at the level of semantics;
- defining composite services as sequences of service invocations as it is possible to decide whether instance data values obtained in a response message provided by service X may be coherently used to fill the request message for invoking service Y.

The problem of defining to which extent elements of different messages are "equivalent" (in the sense that data values carried out in one element can be used to fill an element in a different message) typically implies to take into account syntactical, structural and/or semantic properties of atom data.

Ontologies can be effectively used as a mediation language bridging across different XML namespaces and messages declarations, for they can be used to explicitly convey the meaning associated to atoms. This view matches well with the goal motivating the specification of the Semantic Annotation for WSDL (SAWSDL) standard (Kopecky et al., 2007).

SAWSDL assumes that semantic concepts can be identified via URIs and it is thus independent of any specific ontology technology. The SAWSDL specification defines extension attributes that can be applied to elements both in WSDL and in XML Schema definitions to annotate interfaces, operations, messages and messages' elements and types. SAWSDL extension attributes may be distinguished into two types: "model references" pointing to semantic concepts and "schema mappings" specifying transformations between messages' XML data structure and the associated semantic model.

The SAWSDL annotation "modelReference" is translated in a "hasModelReference" semantic property in the service profile model. According to this approach, atom data may thus be considered "semantically equivalent" if they refer to the same ontology concept. At present atom data can be related with "0" or "1" semantic equivalence values.

More granular similarity values could be derived by reasoning over the ontologies referred to via SAWSDL annotations, as proposed in (Ngan et al., 2006). However, to the purpose of this work, we do not take into account semantic similarity values, for they do not provide clear hints for effectively supporting automatic composition and invocation. In other words, we chose to rely on an exact semantic equivalence in order to enable the transfer of a data value from an atom in a message to another atom in a different message. Moreover, type-safe conversion from one atom to another might be performed by applying the transformation rules encoded in the SAWSDL model "Schema mapping" annotations. Partial semantic matching would not be exploitable in such an automated process without some design-time assumptions made by the software designer.

B. Quality Model

We adopted a web service quality model based on a set of Quality of Service criteria that could be applied to a wide range of services. Nonetheless, the model can be extended with further criteria without altering the proposed service composition technique.

Leveraging on existing works (Zeng et al., 2004; Canfora et al., 2005), we considered the following five generic quality criteria for elementary services:

- Execution cost: is the fee that a service requester has to pay for invoking an operation.
- Duration: is the delay measured at client side between request delivery and response reception.
- Reputation: is a measure of trustworthiness of a service. It mainly depends on end user's experiences.
- Reliability: is the probability that a request is correctly responded within a maximum expected timeframe.

- Availability: is the probability that the service is accessible.

Possible techniques for measuring and calculating such QoS values for a given service may be found in (Zeng et al., 2004). In the context of QoS optimization, we may distinguish positive and negative criteria. Positive criteria, such as reputation, reliability and availability, have an ascending metric: the higher the value, the higher the quality. Negative criteria (e.g. duration and cost) have a descending metric: the higher the value, the lower the quality.

C. Overall functional model of SEQOIA

The overall functional model of the proposed approach for QoS-aware service composition is depicted in Fig. 2. The ILP-based QoS-aware Service Composition component represents the functional block that applies an ILP-based technique to solve a composition problem with a QoS optimization objective and, optionally QoS constraints. The Service Registry maintains the functional profile of registered service endpoints. This profile entails semantic and structural properties of service interfaces, as briefly described above. Semantic properties can make references to concepts in ontologies. Non-functional QoS properties handled by the system are represented according to the Quality Model described in subsection B. The QoS Manager offers storage, processing, search and delivery features for QoS measurements associated with registered endpoints.

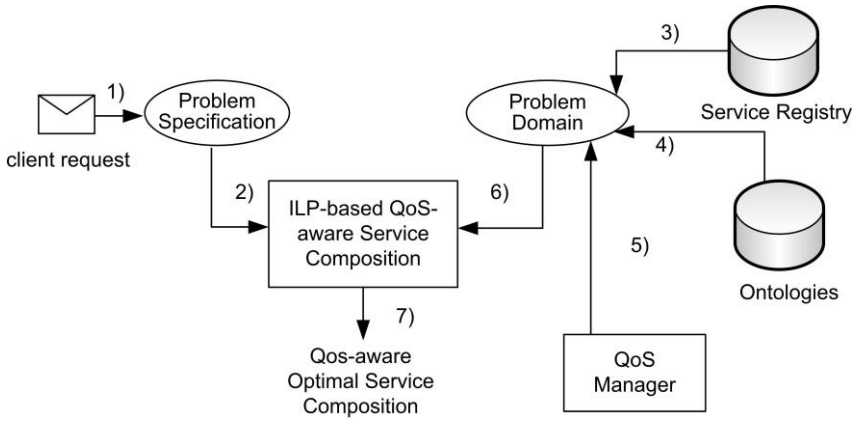

Figure 2. Functional model of the SEQOIA approach

The reference functional flow is hereafter described. The client sends a request message containing some input data values and the specification of the expected data, representing respectively the inputs and goals of the service composition problem (step 1 in Fig. 2). The composition problem is forwarded to the ILP-based QoS-aware service composition component (step 2). First, a representation of the problem domain is built based on knowledge extracted from profile models made available by the Service Registry (step 3) and from related ontologies (step 4), and on the adopted Quality Model and related QoS measurements collected by the QoS Manager (step 5). Based on the given problem domain specification (step 6), the component attempts to retrieve the optimal QoS-aware service composition flow (step 7).

D. STRIPS-like Problem Domain

First, we build a STRIPS-like model for representing the problem domain. STRIPS operators have preconditions, add-effects, and delete-effects, all of which are conjuncts of propositions, and have parameters that can be instantiated to objects in the world. Preconditions have to be valid immediately before the operator is applied, add effect and delete effects are the sets of literals

added / deleted to / from the world state after the operator ends. An instantiated operator (in this case, added to a plan) is called action.

More precisely, a STRIPS planning operator is a four-tuple o = (Name(o), Preconditions(o), AddEffects(o), DeleteEffects(o)).

- Name(o), the name of the operator, is a syntactic expression of the form $n(X_1,\ldots, X_k)$, in which n is a symbol called an operator symbol, $X_{1,\ldots,}X_k$ are the variable symbols that are used in o, and n is unique.
- Preconditions(o) is a finite set of literals whose variables are taken from the set $(X_{1,\ldots,}X_k)$,
- AddEffects(o) and DeleteEffects(o) are also a set of literals.

The STRIPS-like models is built based on the information handled by the Service Registry according to the ontology-based service profile described above. We defined three types of operators:

- Functional operators, representing services operations. Each operation is characterized by an input and an output message. For each operation, input messages are modeled as preconditions and output messages represent the add effects.
- Structural operators, representing structural containment relationships among messages and atom data. In this case, two operator types may be distinguished: the "extract" operator that extracts atom data (add effects) from the containing message (preconditions) and the opposite operator "embed_into" that links atom data (preconditions) to a message (add effect).
- Semantic operators, expressing relations between atoms and semantic concepts (e.g., equivalence) and across semantic concepts (e.g., subsumption).

E. ILP-based modeling

An (integer) linear programming problem instance is defined by a set of (integer) variables, an objective function (a linear function of the variables), and a set of linear constraints (linear inequalities and equalities) on the variables.

Following the approach proposed by Bylander (1994), we translated the STRIPS-like problem domain into an ILP-based specifications, by defining proper variables and linear constraints to represent the above-mentioned operators.

Sets of linear constraints are expressed in order to progressively build a chain of functional profiles, by properly representing the following service invocation condition: a functional profile can be added to a chain of functional profiles, if all the inputs required by that profile are contained in the profile outputs already in the chain. Semantic properties can be exploited in order to connect different XML nodes referring to the same concept or to concepts related by semantic relationships (i.e. subsumption).

In order to minimize the set of information required to specify the service composition problem in the ILP model, hereafter we make the following assumptions: each atom data has a semantic annotation pointing to a concept in an ontology; we collapse functional and structural operators into a unique operator (called "service operator") that take concepts as inputs and provide corresponding output concepts. This simplified reference domain is thus defined as follows:

- W is the set of available service operators;
- S is the set of available semantic operators;
- $O = W \bigcup S$, is the set of all available operators o;

- C is the set of concepts;
- $C_{In} \subseteq C$, is the set of concepts that are used as input in operator in O;
- $C_{Out} \subseteq C$, is the set of parameters that are used as output in any operator in O;
- $C_{Input} \subseteq C$, is the set of concepts that are initially given in O by a client;
- $C_{Goal} \subseteq C$, is the set of concepts that are goal in O;
- $O_{C_{input}} \subseteq W, \forall c \in C$, is the set of operators in O that have concept c as input;
- $O_{C_{output}} \subseteq W, \forall c \in C$, is the set of operators in O that have concept c as output;

Moreover, we model the target composite service as a sequence of stage, where each stage contains operators that are independent and can be executed in parallel.

- Stage (s): $1 \leqslant s \leqslant S$, where S is the maximal number of stages for web service composition.

To model the functional problem, we defined four binary variables.

First, we modeled three binary variables that allow to locate the specific location (stage) of each web service in the graph of related functional profiles, as shown in (1).

$$x_{c,s}^{input}, x_{c,s}^{output}, x_{c,s}^{available} \in \{0,1\}, \forall c \in C, s \in 0,...,S \quad (1)$$

More specifically, at each stage *s*, x^{input} assumes value 1 if concept *c* is used as input for an operator *o* invoked in this stage, 0 otherwise. Analogously, x^{output} assumes value 1 if the concept *c* is an output of an operator *o* invoked in this stage. If a concept *c* is known at a given stage, but not used either as input or as output, $x^{available}$ has value 1, 0 if it is not known yet.

We defined a fourth binary variable, $y_{o,s,}$ associated with each operator *o* that can be invoked at a stage *s*. More specifically, $y_{w,s}$ takes value 1 if *o* is selected for stage s, 0 otherwise, as shown in (2).

$$y_{o,s} \in \{0,1\}, \forall o \in O, s \in 1,...,S \quad (2)$$

Analogously to what proposed in (Bylander, 1994), the functional chain of operators may be built by applying the following constraints:

- (3) and (4) express initial conditions at stage 0 (only input concepts are provided):

$$x_{c,s}^{output} = 1, x_{c,s}^{input} = x_{c,s}^{available} = 0, \forall c \in C_{input}, s = 0 \quad (3)$$

$$x_{c,s}^{output}, x_{c,s}^{input}, x_{c,s}^{available} = 0, \forall c \notin C_{input}, s = 0 \quad (4)$$

- (5) represents the target concepts to be found

$$x_{c,s}^{output} + x_{c,s}^{input} + x_{c,s}^{available} \geq 1, \forall c \in C_{goal}, s = S \quad (5)$$

- Expressions (6) - (10) model service invocation preconditions and effects;

$$\sum_{o \in O_c^{output} \subset O_c^{input}} y_{o,s} \geq x_{c,s}^{output}, \forall c \in C, s \in 1,....,S \quad (6)$$

$$y_{o,s} \leq x_{c,s}^{output}, \forall o \in O_c^{output} \subset O_c^{input}, \forall c \in C, s \in 1,....,S \quad (7)$$

$$\sum_{o \in O_c^{input}} y_{o,s} \geq x_{c,s}^{input}, \forall c \in C, s \in 1,....,S \quad (8)$$

$$y_{o,s} \leq x_{c,s}^{input}, \forall o \in O_c^{input}, \forall c \in C, s \in 1,....,S \quad (9)$$

$$x_{c,s}^{input} + x_{c,s}^{available} \leq x_{p,s-1}^{input} + x_{p,sl-1}^{output} + x_{p,s-1}^{available}, \forall c \in C, s \in 1,....,S \quad (10)$$

QoS criteria of a composite service can be calculated by applying proper aggregation functions over QoS criteria of elementary services (Zeng et al., 2004). For each QoS variable, Table I shows the aggregation functions we adopted.

The execution cost of a composite service is the sum of costs of all elementary services. The overall reputation is computed as the average value of reputation over all elementary services. Reliability and availability are computed as the sum of the natural logarithm value of reliability and availability of elementary services, respectively. As explained in (Zeng et al., 2004), logarithm is applied to linearize a non-linear aggregation function.

The duration of a composite service is computed as the sum of the value of a *d*max variable computed for each stage. This variable represents the highest value of duration across services in a given stage *s*, as expressed below:

$$d\max_s \geq valueDuration_w * y_{w,s}, w \in W, s \in S \quad (11)$$

the value of total duration at each stage s is given by the maximum value $d\max_s$, and thus the total duration of a composite service is computed as the sum of *d*max values over all stages.

TABLE I. AGGREGATION FUNCTIONS FOR THE QOS COMPUTATION FOR A COMPOSITE SERVICE

Criteria	Aggregation function
Cost	$\cos t = \sum_{w \in W} \sum_{s \in S} c_w * y_{w,s}$
Reputation	$reputation = \sum_{w \in W} \sum_{s \in S} rep_w * y_{w,s}$
Reliability	$reliability = \sum_{w \in W} \sum_{s \in S} rel_w * y_{w,s}$
Availability	$availability = \sum_{w \in W} \sum_{s \in S} ava_w * y_{w,s}$
Duration	$duration = \sum_{s \in S} d\max_s$

Finally, we conclude with the objective function that allows to select web service that minimize the overall QoS in a composition problem, as follows:

$$Minimize \sum_{w \in W} \sum_{s \in S} s_w \cdot y_{w,s} \tag{12}$$

where s_w is the sum of all values of quality attributes:

$$s_w \equiv \cos t_w * W_1 + rep_w * W_2 + rel_w * W_3 + dur_w * W_4 + ava_w * W_5, \forall w \in W \tag{13}$$

and $W_i \in [0,1]$ and $\sum_{j=1}^{5} W_i = 1,$

In the utility function definition, all QoS attributes are weighted according to their relevance. As different QoS criteria typically have values in different domain ranges, QoS values are also normalized by applying proper scaling functions.

A set of optional constraints may be added in order to express possible user preferences on QoS values:

$$\sum_{j \in A} \sum_{i \in S_j} c_{i,j} * y_{i,j} \leq M, M > 0 \tag{14}$$

where M is the maximum cost allowed by the user.

Our model allows to express upper bound values (as in the example above). Constraints representing lower bound values (e.g. cost ≥ L) cannot be added in this model, as they might not be correctly satisfied with this problem formulation. As a matter of fact, ILP-solving tools could simply add to the solution further services, even if not useful to the functional problem, to increase the overall value of a QoS attribute over a given lower bound threshold.

V. IMPLEMENTATION DETAILS

In order to realize the functional model depicted in Fig. 1 and consequently validate the proposed approach for service composition, we developed a "Proof of Concept". The PoC has been implemented by leveraging on the capabilities provided by a middleware infrastructure, called Service and Application Integration (SAI) middleware, implemented in our research laboratory (Parlanti et al., 2010).

The Service and Application Integration (SAI) middleware is a service-oriented infrastructure conceived as a set of configurable components to be variably assembled into different system deployments to best fit domain-specific integration needs. This implies that concrete middleware deployments are not required to instantiate all the components envisaged by the logical system architecture.

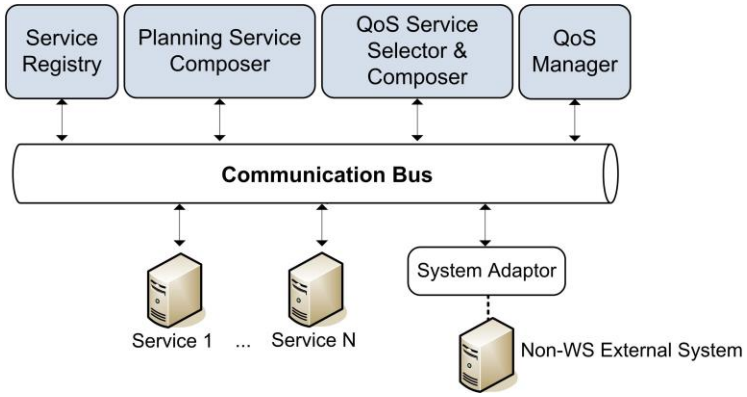

Figure 3. Proof of Concept Architecture

Hereafter we provide a brief description of the SAI middleware by focusing on components more strictly related to the PoC implementation, as shown in Fig.3:

A *Communication Bus* handles the communication among SAI components by offering a generic interface for message handling capabilities.

The *Service Registry* maintains the service profiles of registered web services and offers the APIs for service registration and lookup. In the registration phase, the Service Registry parses the service interface descriptions to generate an internal representation according to the Service Profile Model. At present, the system can interpret WSDL service description as well as XML-Schema based message service descriptions. Semantic annotations for data atoms can be provided by SAWSDL annotations or by providing a SAI-custom XML document. The logic for WSDL and XML Schema parsing and the generation of structural properties is based on the XML Schema Object Model (XSOM) (XSOM, 2007) since it is the only general-purpose Java schema parser which we are currently aware of. Registration is not restricted only to providers compliant with WS-based interfaces. External systems with non WS-interfaces may also be accessed via customized Adaptor components exposing XML message-based interfaces. The Service Profile Model is expressed using the Web Ontology Language (OWL, 2004). Registered service profiles are maintained in a knowledge-base that has been implemented using JENA[vi], an open source Java-based semantic framework. We adopted the Jena TDB triple store storage system for functional profiles persistence.

The *QoS Manager* allows to store and retrieve QoS measurements associated to the registered service endpoints.

The *Planning Service Composer* includes two main components: a STRIPS domain builder parsing the functional profiles maintained in the SAI service registry for creating corresponding the corresponding STRIPS operators; a service composition engine based on a refactoring of PL-PLAN, an open source Java library implementing the GraphPlan algorithm (Blum and Furst, 1997). More details on these components can be found in (Paganelli et al., 2010).

The QoS *Service Selector and Composer* implements two main modules:

1. a QoS-aware Service Selector, which takes as input an abstract plan (e.g. a plan generated by AI Planning Service Composer) and creates a concrete plan by selecting those services that optimize the overall QoS.
2. a QoS-aware service Composer, which implements the SEQOIA composition approach described above.

Both features have been implemented by leveraging on ILP-solving capabilities provided by the LP-Solve library (LP-Solve URL). The Planning Service Composer and QoS-aware Service Selector can be combined to provide a two-layered implementation of

QoS composition and selection approach. These components have thus been used as reference implementation of an alternative approach to be compared with SEQOIA in performance evaluation tests, as discussed in the following Section.

VI. SEQOIA APPROACH EVALUATION

In this Section we describe testing activities that we carried out to evaluate the performance of the proposed SEQOIA technique and make a comparison with an alternative solution.

A. Two-layered alternative approach

We adopted an alternative solution implementing a two-layered approach, i.e. a technique handling service composition and QoS-aware selection as two separate steps. As depicted in Fig. 4, the alternative solution is made of the following two components, deployed in the Proof of Concept:

- a Planning Service Composer, based on the GraphPlan algorithm;
- a QoS-aware Service Selector, implementing a service selection algorithm based on the ILP technique.

The functional flow of this alternative approach, named PLAN-ILP, is shown in Fig.4. Given a problem specification in a client request, the GraphPlan-based algorithm compiles the problem into a structure called planning graph. The planning graph is extended one level at a time until a solution is find. When a solution is found, the Graphplan terminates its search and returns the shortest possible sequence of service invocations necessary to achieve the objective conditions (abstract plan). Then, the Planning Service Composer selects those endpoints among available candidates that optimize the overall QoS, and produces as output the resulting concrete plan.

The service selection problem has been mapped into an ILP formulation by following an approach analogous to the one proposed by Zeng et al. (2004). We describe the implemented ILP-based formulation in Appendix I.

For the sake of conciseness, hereafter we will use the term "SEQOIA" to refer to the QoS-aware composition approach proposed in this work and the term "PLAN_ILP" to refer to the two-layered alternative solution.

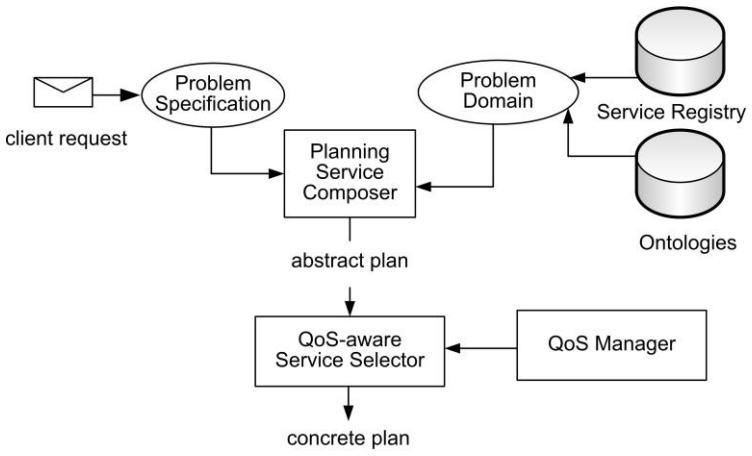

Figure 4. PLAN-ILP approach

B. *Test methodology*

We defined a set of test cases to measure the computational time needed by both approaches to compute a solution for a given composition problem. Test cases were defined by varying the number of registered services and the depth and number of services in the expected solutions. We defined two types of composition problems, each with a known type of solution:

- *Invocation sequences*: the expected composite service is specified as a sequential flow of service invocations. An invocation sequence is composed by a sequence of stages, each containing one service to be invoked. We defined three composition problems, by varying the number of elementary services in the expected solution (i.e. solutions 4s, 7s and 10s made of 4, 7, 10 services, respectively). For these expected solutions, the number of stages (solution depth) has the same value of service cardinality (i.e. solutions 4s, 7s, and 10 s are made of 4, 7, 10 stages respectively). An example (solution 4s) is shown in Fig. 5.
- *Invocation plans*: the expected solution is a flow of sequential and parallel invocation paths. In this case, each stage contains one or more services that can be invoked in parallel. Also in this case we defined three problems by varying the number of elementary services in the expected solution (i.e. solutions 4p, 7p and 10p made of 4, 7, 10 services, with a depth of 3, 5, and 7 stages, respectively). An example (solution 4p) is shown in Fig. 6.

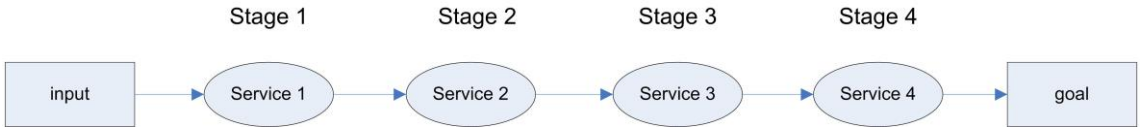

Figure 5. Example of an invocation sequence solution

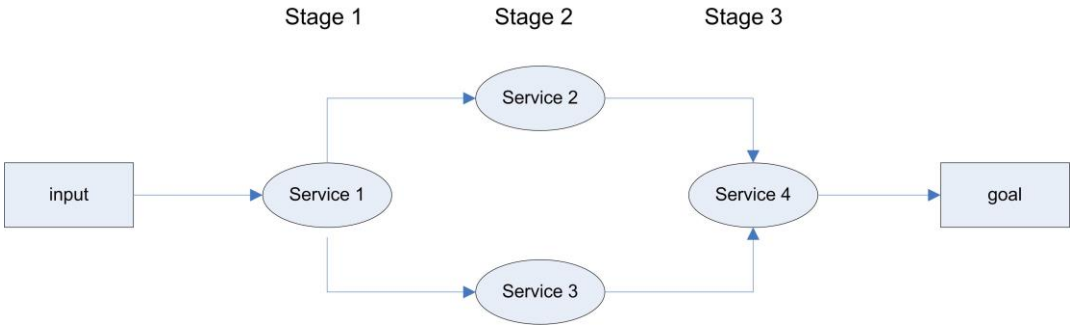

Figure 6. Example of an invocation plan solution

Test cases possible configurations are shown in Table II.

TABLE II. TEST CASES CONFIGURATIONS

Service Registry Population (number of registered services)	Number of Concepts	Composition Problem Invocation Sequence	Composition Problem Invocation Flow
20	200	4s, 7s, 10s	4p, 7p, 10p
40	400	4s, 7s, 10s	4p, 7p, 10p

70	600	4s, 7s, 10s	4p, 7p, 10p
100	800	4s, 7s, 10s	4p, 7p, 10p

The testing environment was built by using JUnit 4.7 testing framework and the Eclipse 3.6 Helios development environment. Tests were run on a PC equipped with an Intel Core 2 Duo processor with at 2.4 GHz operating frequency and 4GB DDR2 RAM.

C. Test results

Each test case was executed 10 times and performance values were computed as the trimmed mean on the actually measured values. Testing results are shown in Table III.

TABLE III. QOS-AWARE SERVICE COMPUTATION TIME FOR PLAN_ILP AND SEQOIA TECHNIQUES

testSet (registered services cardinality)	**Technique**		**Solution Type**					
			4s	**7s**	**10s**	**4p**	**7p**	**10p**
20	**PLAN_ILP**	composition	78,3	221	589,7	58,6	169,8	413,8
		QoS selection	6,1	6	13,1	6	8,3	15,2
		TOTAL	84,4	227	602,8	64,7	178,1	429
	SEQOIA		110,2	144,2	191,6	137,7	143,9	180,2
40	**PLAN_ILP**	composition	193,1	379,8	702,1	122,8	265,6	475,2
		QoS selection	7,1	8,9	13,4	6,1	11,5	15,4
		TOTAL	200,2	388,7	715,5	128,9	277,1	490,6
	SEQOIA		235,1	321,6	400,7	279	299,8	370,2
70	**PLAN_ILP**	composition	180,6	370,1	858,7	199,5	392,9	622,1
		QoS selection	8,4	12,5	15,5	7,3	15,1	16,7
		TOTAL	189	382.6	874,2	206,8	408	638,8
	SEQOIA		406,7	539,3	649,2	484,5	467	611,7
100	**PLAN_ILP**	composition	320,2	618,2	1020,8	280,1	469,3	797,1
		QoS selection	10,9	12,6	17,2	8,4	15,9	17,2
		TOTAL	331,1	630,8	1034	288,5	485,2	814,3
	SEQOIA		519,2	715,2	918,1	568,5	690,4	891

As shown in Fig. 7, the computation time measured for "SEQOIA" technique grows almost linearly with the number of registered services, since the ILP algorithm should search a solution on an increasingly large set of registered services. For the sake of conciseness Fig. 7 shows measured results for 10s and 10p composition problems.

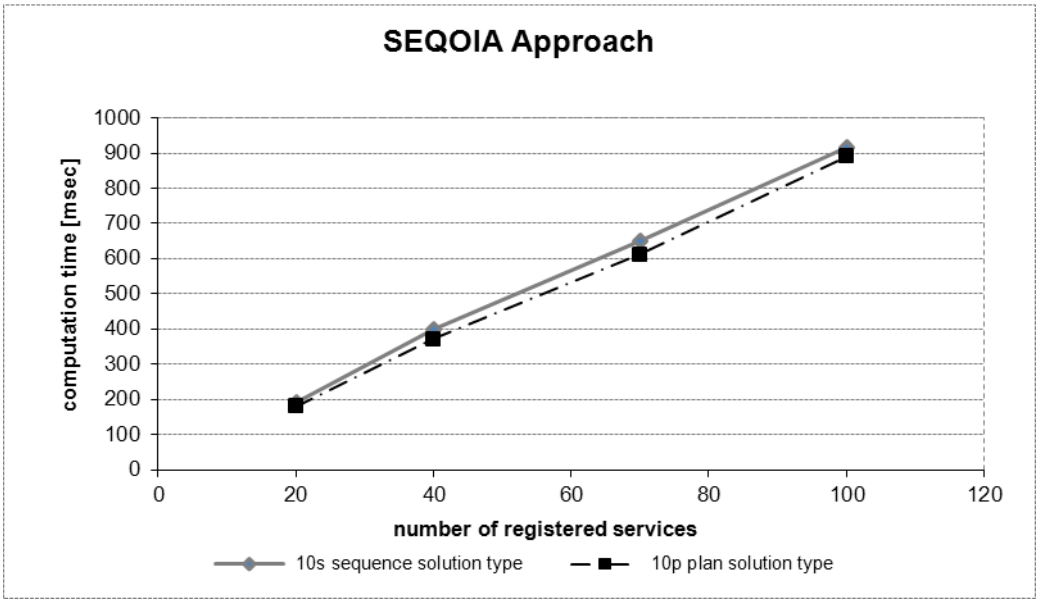

Figure 7. Computation time for SEQOIA approach

Moreover, measured computation time for invocation plan solutions (i.e., 10p) increases more slowly than for invocation sequence solutions (i.e. 10s). This is due to the fact that for a given number of services, invocation plans have typically fewer stages than invocation sequences and, consequently, fewer linear constraints in the ILP problem.

Analogously, the computation time measured for the PLAN_ILP technique increases both in the composition and selection steps with the number of registered services and the number of stages in the expected solution (as shown in Table III). With specific regard to the composition step (see Fig. 8), which is computationally more demanding, such behavior is due to the fact that for a greater number of registered services the GraphPlan algorithm must search for a solution in an increasingly large area, while for a greater number of stages in the expected solution, the GraphPlan algorithm performs an iterative search expanding one level at a time until a solution is found. Therefore, if the expected solution includes more stages, longer time is needed to find this solution.

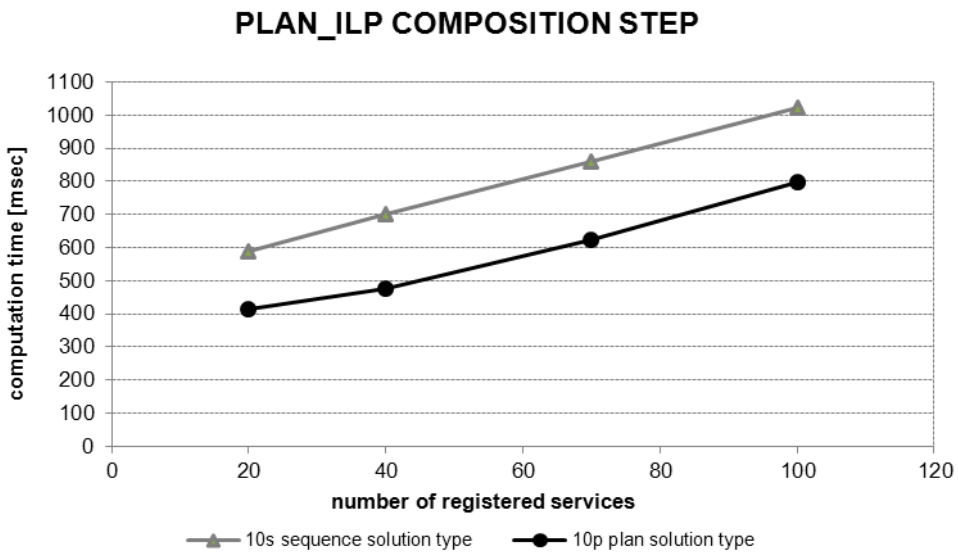

Figure 8. Computation time for the PLAN_ILP composition step

The obtained results confirm an expected behavior for both techniques. More interestingly, by comparing the results obtained for the two optimization techniques, we may draw the following considerations:

- if the number of registered services is small, the SEQOIA technique shows better performance than PLAN_ILP both for invocation sequences and plan solution types, as shown respectively in Fig.9 and Fig.10. This advantage progressively decreases as the number of registered services increases. This advantage decreases to zero more quickly for invocation plan solution types.

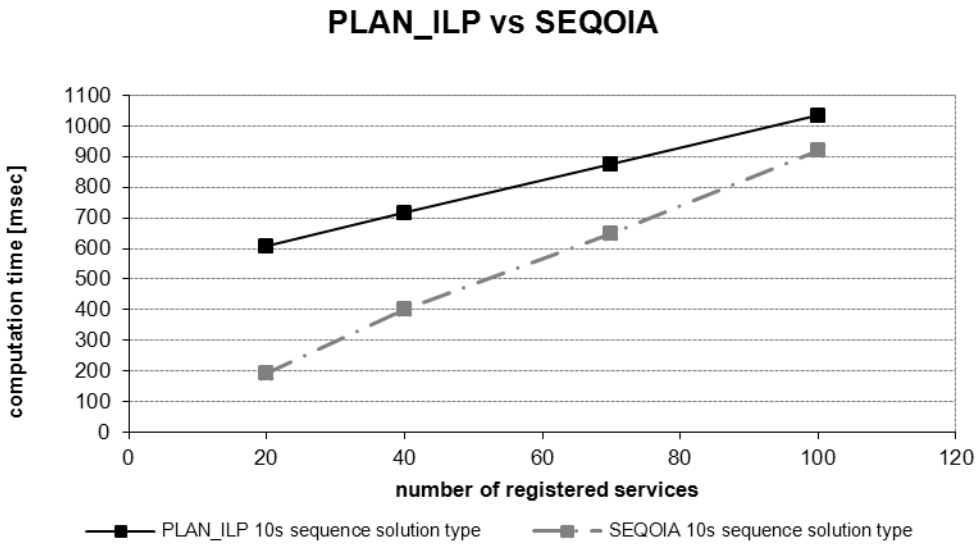

Figure 9. PLAN_ILP vs SEQOIA technique. Computation time measured for invocation sequence solutions

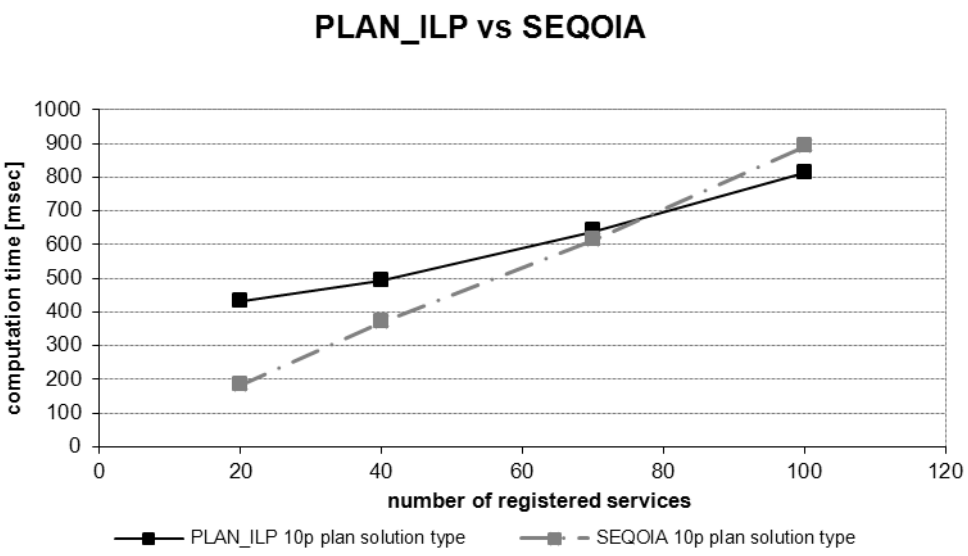

Figure 10. PLAN_ILP vs SEQOIA technique. Computation time measured for invocation plan solutions

- if the number of stages in the expected solution is small, the PLAN_ILP technique shows better performance than SEQOIA, (see Fig.11 and Fig.12). As the solution depth increases, the SEQOIA technique shows better performance. This advantage progressively decreases with the number of registered services, as already mentioned above.

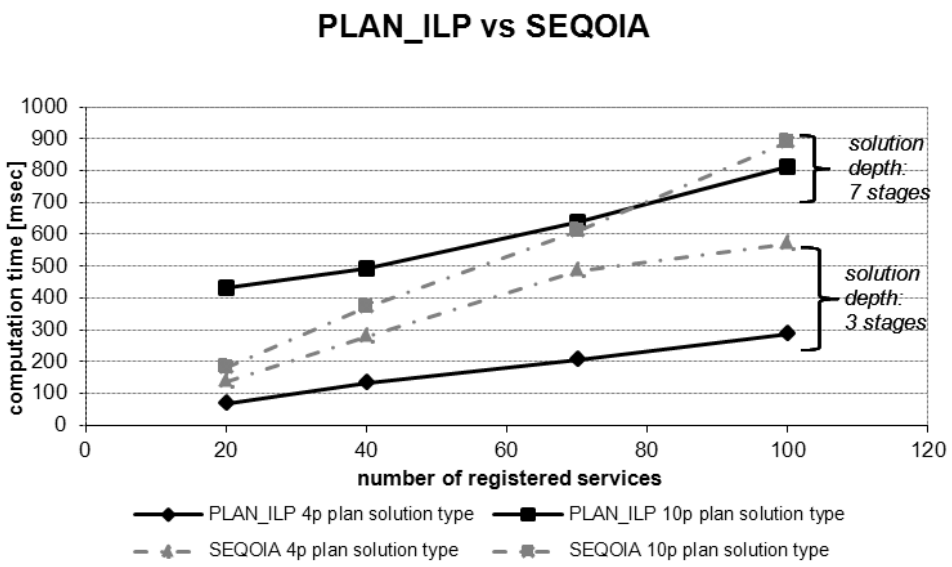

Figure 11. PLAN_ILP vs SEQOIA technique. Computation time versus number of services

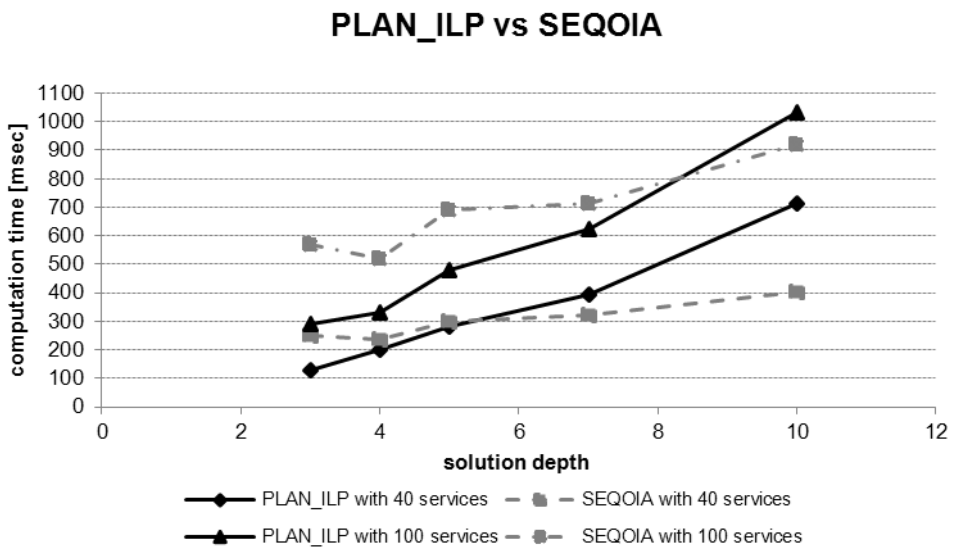

Figure 12. PLAN_ILP vs SEQOIA technique. Computation time versus solution depth

Testing results have shown that the SEQOIA approach performs better than the alternative PLAN-ILP technique when the number of candidate services is limited and for an increasing solution depth. Techniques for service clustering (Nayak et al., 2007; Platzer et al., 2009) could be applied on the registered service population in order to widen the range of SEQOIA profitable exploitation.

In the perspective of real-life adoption of service composition techniques, we expect these two techniques to provide complementary capabilities. As a matter of fact, we implemented both techniques in our Proof of Concept, as we think that such configuration can help in flexibly satisfying possible different requirements posed by end-users. For instance, user requirements could range from the specification of a whole optimal composition solution, to the re-planning of a specific subset of a service composition, or to the mere substitution of previously selected service endpoints with novel ones improving the overall QoS and/or complying with local or global QoS constraints while not changing the service invocation flow.

VII. CONCLUSIONS

In this paper we presented SEQOIA, an approach for QoS-aware dynamic service composition leveraging on Integer Linear Programming model and on the expressiveness of an ontology-based service profile model, representing structural and semantic properties of service interface descriptions.

The proposed approach leverages on a message and data-driven model of services interfaces and lightweight semantic annotation for building a STRIPS-like representation of the composition problem domain. We applied the ILP programming technique to solve the composition problem expressed as a set of functional and non-functional linear constraints and an objective function for maximizing the global QoS.

With respect to existing approaches, SEQOIA leverages on semantic annotations in order to make service composition feasible coping with syntactic and structural differences typically existing across different, even similar, service implementations. We adopted the ILP programming technique as it allows to model both functional and non-functional constraints and several solving libraries are available.

The Proof of Concept implementation of SEQOIA leverages on the service brokering, registration and QoS management capabilities provided by a service-oriented middleware implemented in our research laboratory. To ease the adoption of SEQOIA in real enterprise scenarios, the system relies on an XML-based message model of services interfaces and it does not strictly require the use of WSDL.

The PoC includes also the implementation of a dynamic service composition engine based on the GraphPlan algorithm and a QoS-aware selection algorithm based on the ILP technique. This configuration was used for performing testing activities and compare the SEQOIA approach with an alternative solution adopting state of the art composition and selection techniques.

Testing results have shown that the SEQOIA approach performs better than the alternative PLAN-ILP technique when the number of candidate services is limited and for an increasing solution depth. This behavior was expected, as SEQOIA guarantees to find the service composition providing the optimal QoS value, while the alternative approach does not provide this guarantee as it handles separately the specification of the functional service composition flow and the QoS-based service selection step.

In the perspective of real-life adoption of service composition techniques, we think that a "one-size fits all approach" is not applicable, while we recognize that providing users with a wider set of options could better cope with real practices. To this purpose, the Proof of Concept is enhanced with two techniques for service composition and QoS selection, whose capabilities are complementary to the ones provided by SEQOIA. In this direction, we plan to investigate possible alternative optimization techniques, such as meta-heuristics (e.g. genetic algorithms), in order to maintain good performance with an increasing number of registered services.

We also plan to enrich the service brokering and composition infrastructure with a composition execution engine and extend the use of selection and composition technique for run-time service execution adaptation (e.g. to handle possible exceptions and QoS variations).

REFERENCES

(Agarwal et al., 2005) Agarwal, V., Chafle, G., Dasgupta, K., Karnik, N., Kumar, A., Mittal, S. and Srivastava, B. (2005), "Synthy: A system for end to end composition of web services". *Web Semantics Journal*, pp. 311-339, December.

(Blum and Furst, 1197) Blum, A. and Furst, M. (1997), "Fast Planning Through Planning Graph Analysis", Artificial Intelligence, pp. 90, 281-300.

(Bylander, 1194) Bylander, T. (1994), "The Computational Complexity of Propositional STRIPS Planning", Artificial Intelligence, pp. 69, 165-204.

(Canfora et al., 2005) Canfora, G., Penta, M.D., Esposito, R. and Villani, M.L. (2005), "An approach for qos-aware service composition based on genetic algorithms". In Proceedings of the 2005 conference on Genetic and evolutionary computation (GECCO '05), Hans-Georg Beyer (Ed.). ACM, New York, NY, USA, pp. 1069-1075.

(Casati et al., 2000) Casati, F., Ilnicki, S. and Jin, L. (2000), "Adaptive and dynamic service composition in EFlow", in Proceedings of 12th International Conference on Advanced Information Systems Engineering (CAiSE), Springer Verlag.

(Dustdar and Schreiner, 2005) Dustdar, S. and Schreiner, W. (2005), "A survey on web services composition", International Journal of Web and Grid Services, vol. 1, no.1, pp.1-30.

(Farrell and Lausen, 2007) Farrell, J. and Lausen, H. (2007), Semantic Annotations for WSDL and XML Schema. W3C Recommendation, http://www.w3.org/TR/sawsdl/28.

(Hewett et al., 2009) Hewett, R., Kijsanayothin, P. and Nguyen, B. (2009), "Scalable Optimized Composition of Web Services with Complexity Analysis", in Proceedings of the 2009 IEEE international Conference on Web Services. ICWS. IEEE Computer Society, Washington, DC, pp. 389-396, July 06 - 10.

(Jiang et al., 2010) Jiang, W., Zhang, C., Huang, Z., Chen, M., Hu, S. and Liu, Z. (2010), "QSynth: A Tool for QoS-aware Automatic Service Composition". 2010 IEEE International Conference on web Services (ICWS) pp.42-49, 5-10 July.

(Kopecky et al., 2007) Kopecky, J., Vitvar, T., Bournez, C. and Farrell, J. (2007), "SAWSDL: Semantic Annotations for WSDL and XML Schema," Internet Computing, IEEE , vol.11, no.6, pp.60-67, November-December.

(Martin et al., 2005) Martin, D., Burstein, M., Mcdermott, D., Mcilraith, S., Paolucci, M., Sycara, K., Mcguinness, D., Sirin, E. and Srinivasan, N. (2005), "Bringing Semantics to Web Services: The OWLS Approach". In: Cardoso, J., Sheth, A.P. (eds.) SWSWPC 2004. LNCS, vol. 3387, pp. 26–42. Springer, Heidelberg.

(Nayak et al., 2007) Nayak, R. and Lee, B. (2007), "Web Service Discovery with additional Semantics and Clustering". In Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence (WI '07), 2007. IEEE Computer Society, Washington, DC, USA, 555-558. DOI=10.1109/WI.2007.112, http://dx.doi.org/10.1109/WI.2007.112.

(OASIS, 2007) OASIS, Web Services Business Process Execution Language Version 2.0., OASIS Standard.

(Oh et al., 2008) Oh, S.C., Lee, D. and Kumara, S.R.T. (2008), "Effective Web Service Composition in Diverse and Large-Scale Service Networks", IEEE Transactions on Services Computing, vol. 1 No. 1, January-March.

(OMG, 2011). Object Management Group, Business Process Model and Notation (BPMN) Specification 2.0. OMG Document Number: formal/2011-01-03, Standard document URL: http://www.omg.org/spec/BPMN/2.0

(Paganelli et al., 2010) Paganelli, F., Parlanti, D. and Giuli, D. (2010), "Message-based Service Brokering and Dynamic Composition in the SAI Middleware", in the Proc. of IEEE International Conference on Service Computing (SCC 2010), Miami, Florida.

(Parlanti et al., 2010) Parlanti, D., Paganelli, F. and Giuli, D. (2010), "A Service-Oriented Approach for Network-Centric Data Integration and its Application to Maritime Surveillance", IEEE Systems Journal, vol. 5 No. 2, date of publication: 29 November 2010, doi: 10.1109/JSYST.2010.2090610

(Platzer et al., 2009) Platzer, C., Rosenberg, F. and Dustdar, S. (2009), "Web service clustering using multidimensional angles as proximity measures". *ACM Trans. Internet Technol.* 9, 3, Article 11 (July 2009), 26 pages. DOI=10.1145/1552291.1552294 http://doi.acm.org/10.1145/1552291.1552294.

(Rao and Su, 2004) Rao, J. and Su, X. (2004), "A Survey of Automated Web Service Composition Methods". In Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition, SWSWPC 2004, San Diego, California, USA, July 6th.

(Roman et al., 2005) Roman, D., Keller, U., Lausen, H., De Bruijn, J., Lara, R., Stollberg, M., Polleres, A., Feier, C., Bussler, C. and Fensel, D. (2000), "Web Service Modeling Ontology", *Applied Ontology*, 1(1): pp. 77 - 106, 200.

(Salkin and Mathur, 1989) Salkin, H.M. and Mathur, K. (1989), "Foundations of Integer Programming", North-Holland.

(Xu et al., 2011) Xu, Bin, Luo, Sen, Yan, Yixin, Sun, Kewu (2011), "Towards efficiency of QoS-driven semantic web service composition for large-scale service-oriented systems", *Service Oriented Computing and Applications*, Springer London, pp. 1-13.

(Yoo et al., 2008) Yoo, J.W., Kumara, S.R.T., Lee, D. and Oh, S.C. (2008), "A web Service Composition Framework Using Integer Programming with Non-functional Objectives and Constraints". In the Proc. of 10th E-Commerce Technology Conference and the Fifth IEEE Conference on Enterprise Computing, E-Commerce and E-Services,CEC/EEE 2008: pp. 347-350.

(Yu et al., 2007) Yu, T., Zhang, Y. and Lin, K.J. (2007), "Efficient algorithms for web services selection with end-to-end qos constraints". ACM Transactions on theWeb, pp. 1-6.

(Zeng et al., 2004) Zeng, L., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J. and Chang, H. (2004), "QoS-Aware Middleware for web Services Composition", IEEE transactions on Software Engineering, 30(5), pp. 311-327.

(Zhao and Tong, 2007) Zhao, H. and Tong, H. (2007), "A Dynamic Service Composition Model Based on Constraints" In the Proc. of the Sixth International Conference on Grid and Cooperative Computing, pp.659-662, 16-18 August.

(Zheng and Yan, 2008) Zheng, X. and Yan, Y. (2008), "An Efficient Syntactic Web Service Composition Algorithm Based on the Planning Graph Model", In Proceedings of the 2008 IEEE international Conference on Web Services IEEE Computer Society, pp. 691-699.

(Weber, 2007) Weber, I. (2007), "Requirements for implementing business process models through composition of semantic web services", In I-ESA'07: Proceedings of the 3rd International Conference Interoperability for Enterprise Software and Applications, pp. 3-14, Funchal, Madeira, March.

(W3C XML, 2008) XML Schema Object Model (XSOM). Available at: https://xsom.dev.java.net/.

(W3C OWL,2009) Web Ontology Language, W3C recommendation. Available at: http://www.w3.org/TR/owl-guide/.

APPENDIX I – QOS SELECTION TECHNIQUE IN THE PLAN-ILP APPROACH

This technique for QoS-based service selection assumes the existence of a predefined work plan describing a flow of invocations to abstract services (or tasks). Its objective is to select for each abstract service the service endpoint meeting some local or global QoS constraints and contributing to the optimization of the overall QoS value.

First, for every web service endpoint s_{ij} that can be used to execute a task (or abstract service) t_j, we include in the IP problem an integer variable y_{ij}, such that by convention y_{ij} is 1 if service s_{ij} is selected for executing task t_j, 0 otherwise. For each task t_j, there is a set of web services s_j that can be assigned (allocated) to it. However, for each task t_j, we should only select one web service to execute this task. Given that y_{ij} denotes the selection of web service s_{ij} to execute task t_j, we introduce the following allocation constraint that must be satisfied on each task j:

$$\sum_{i \in S_j} y_{i,j} = 1, \forall j \in A \tag{15}$$

NOTICE: this is the author's version of a work that was accepted for publication in *International Journal of Web Information Systems*. Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in: F. Paganelli, T. Ambra, D. Parlanti, (2012) "A QoS-aware service composition approach based on semantic annotations and integer programming", Int. Journal of Web Information Systems, Vol. 8 Iss: 3, pp.296 – 321 http://dx.doi.org/10.1108/17440081211258187

where A is the set of tasks in the specific composite service.

QoS criteria of a composite service can be calculated by applying aggregation functions similar to those proposed in (Zeng et al. 2004). For each QoS variable, Table IV shows the aggregation functions we adopted.

TABLE IV. AGGREGATION FUNCTIONS FOR QOS COMPUTATION

Criteria	Aggregation function
Cost	$\cos t = \sum_{j \in A} \sum_{i \in S_j} c_{i,j} * y_{i,j}$
Reputation	$reputation = \sum_{j \in A} \sum_{i \in S_j} rep_{i,j} * y_{i,j}$
Reliability	$reliability = \sum_{j \in A} \sum_{i \in S_j} rel_{i,j} * y_{i,j}$
Availability	$availability = \sum_{j \in A} \sum_{i \in S_j} ava_{i,j} * y_{i,j}$
Duration	$\sum_{j \in sp} e_j \le duration, \forall sp \in ep$

While the constraints introduced for cost, durability, reliability and availability attributes are those already studied for the SEQOIA approach, for the duration we define new constraints and variables in our ILP problem as follows:

1. x_j: start time of task j;
2. e_j: duration of task j;
3. t_j ->t_k: denote the fact that task t_k is a direct successor of task t_j.

The introduction of these variables allows to define the following constraints associated to the variable duration.

The first constraint indicates that the execution duration of a given task t_j is equal to the execution duration of the web service selected to execute task t_j.

$$\sum_{i \in S_j} e_{i,j} * y_{i,j} = e_j, \forall j \in A \tag{16}$$

The second constraint indicates that if task t_k is a direct successor of task t_j, then the execution of t_k must start after task t_j has been completed.

$$x_k - (e_j + x_j) \ge 0, \forall t_j -> t_k \tag{17}$$

NOTICE: this is the author's version of a work that was accepted for publication in *International Journal of Web Information Systems*. Changes resulting from the publishing process, such as peer review, editing, corrections, structural formatting, and other quality control mechanisms may not be reflected in this document. Changes may have been made to this work since it was submitted for publication. A definitive version was subsequently published in: F. Paganelli, T. Ambra, D. Parlanti, (2012) "A QoS-aware service composition approach based on semantic annotations and integer programming", Int. Journal of Web Information Systems, Vol. 8 Iss: 3, pp.296 – 321 http://dx.doi.org/10.1108/17440081211258187

Hence the duration of a composite service is computed as the sum of the value of a e_j variable computed for each task. This variable represents the value of duration selected to execute task t_j across services in a given task j.

Finally, we conclude with the objective function that allows to select web service that maximize the overall QoS in a composition problem, as follows:

$$Max[dur * W_1 + \cos t * W_2 + ava * W_3 + rel * W_4 + rep * W_5] \quad (18)$$

and $W_i \in [0,1]$, $\sum_{j=1}^{5} W_i = 1$.

A set of optional constraints may be added in order to express possible user preferences on QoS values:

$$\sum_{j \in A} \sum_{i \in S_j} c_{i,j} * y_{i,j} \leq M, M > 0 \quad (19)$$

where M is the maximum cost allowed by the user.

---

[i] http://webservices.seekda.com/
[ii] http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/
[iii] www.gurobi.com
[iv] lpsolve.sourceforge.net
[v] www.ateji.com/optimj/index.html
[vi] http://jena.sourceforge.net/documentation.html