Cooperative Service Composition

Nikolay Mehandjiev[1], Freddy Lécué[2], Martin Carpenter[1], and Fethi A. Rabhi[3]

[1] The University of Manchester Centre for Service Research, Manchester, UK
initial.lastname@manchester.ac.uk
[2] IBM Research, Smarter Cities Technology Centre, Dublin, Ireland
freddy.lecue@ie.ibm.com
[3] The University of New South Wales, Sydney, Australia
f.rabhi@unsw.edu.au

Abstract. Traditional service composition approaches are top-down(using domain knowledge to break-down the desired functionality), or bottom-up (using planning techniques). The former rely on available problem decomposition knowledge, whilst the latter rely on the availability of a known set of services, otherwise automatic composition has been considered impossible. We address this by proposing a third approach: Cooperative Service Composition (CSC),inspired by the way organisations come together in consortia to deliver services. CSC considers each service provider as proactive in service composition, and provides a semantics-based mechanism allowing innovative service compositions to emerge as result of providers' interactions. The key challenges we resolve are how to determine if a contribution brings the composition closer to its goal, and how to limit the number of possible solutions. In this paper we describe the approach and the solutions to the two key challenges, and demonstrate their application to the composition of financial web services.

Keywords: service composition, semantic services, software agents.

1 Introduction

Service-oriented software development often focuses on composing services to fulfil a set of user requirements. An intuitive view to service composition would perceive it as an activity which aims to satisfy the need for a (non-existing) service by bringing together existing ones. This integration activity can be done manually, yet automating it makes it more in tune with the vision of composing services at the point of need [1], and takes service-oriented computing beyond component-based software engineering.

There is a wide variety of approaches and methods for service composition [2], yet most of them fit into one of two "camps":

- top-down approaches, which break-down the desired functionality into smaller units using domain knowledge and templates, or
- bottom-up approaches which use planning to construct a composition without a template, only by using knowledge of available services and their interfaces.

Top-down approaches [3–6] use formalised problem decomposition knowledge, breaking down desired functionality into simpler units, and then seeking services for

J. Ralyté et al. (Eds.): CAiSE 2012, LNCS 7328, pp. 111–126, 2012.

each such unit. For example, [6] shows how we can select a set of services which fit in terms of input and output data types. These approaches are efficient and preferred where the decomposition knowledge exists. However, they only find solutions prescribed by the decomposition knowledge, often missing the chance for innovation.

The bottom-up approaches rely on the availability of a fixed set of well-specified services. In principle these are quite inefficient yet many improvements in terms of search heuristics have been developed, and also many hybrid approaches which attempt to optimise the split between top-down and bottom-up approaches (*e.g.* HTN [7]).

There are however situations, where we do not have a fixed set of known services nor decomposition knowledge about the desired functionality, and innovative compositions are desired. In these circumstances neither of the two mainstream approaches is suitable, and here we propose a third, novel approach called *cooperative service composition*.

Our approach, detailed in Section 4, is inspired by the manner in which human organisations form consortia to respond to market demands and provide innovative services. It implies pro-active roles for the service composer and for the service providers, represented by autonomous software entities, called agents. These are pre-programmed to behave according to the business interests of their organisations, and are able to reason over the semantic specifications of requirements and candidate services.

In our approach, innovative service compositions emerge as a result of cooperation between service providers, responding to requests by service composers. There are two key challenges we had to resolve in achieving this idea:

1. how to determine if a contribution brings the composition closer to its goal, and
2. how to limit the number of possible solutions, thus avoiding combinatorial explosion of complexity.

In this paper we describe the approach and the solutions we developed for the two key challenges enabling the approach, in Sections 4 and 5. This is preceeded by the introduction of a motivating example in Section 2, and by the description of semantic annotations of services, and techniques for calculating semantic distances we use to guide our approach in Section 3. We then proceed to apply the approach to our detailed example in Section 6, comparing with top-down approaches.

2 Motivating Example

This scenario illustrates the value of cooperative service composition: Carl is a fund manager, preparing a new equity (stock) fund focusing on the Far East. He needs to conduct a number of visits to companies of interest in China and Hong Kong, and he should filter target companies from a large number of candidates. For this he needs a software service which, given the details of a company, will produce a portfolio of financial and legal information about the company, including the following:

- volatility of company share price over the last 3 years;
- “abnormal returns” from the company stock over the last 3 years;
- company announcements, translated into English if necessary;
- news in English regarding the company over the last three months.

The desired functionality is illustrated in Figure 1. To formalise the concepts involved in Carl's request and thus allow semantic annotation of our services, we have customised the SUMO Financial Ontology[1]. Extracts from the ontology are included in the T-box presented in Figure 2.

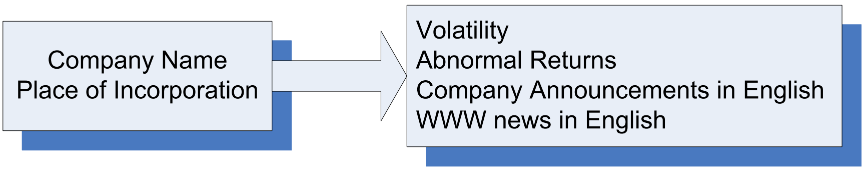

Fig. 1. Desired functionality in terms of inputs and outputs

Given a formalised ontology of domain concepts, the desired functionality of the service Carl is seeking can be specified within a service broker or a service search engine, attempting to find an appropriate service to directly deliver this functionality. If this fails, we can proceed to service composition, attempting to find services which deliver parts of the desired functionality, and to bring them together [8]. The service broker may have a description of how this problem can be decomposed into sub-problems, in which case it will use it to decompose the functionality into tasks providing (a) volatility and abnormal returns information based on stock price movements; (b) the company announcements in English; and (c) the news about the company in English), and attempt to find at least one candidate service for each task. This is the conventional top-down approach [3], which allows efficient search, yet it misses innovative solutions to the problem, for example one service providing all the information we need in Chinese, and then we translate into English with another service.

This is avoided by the alternative approach which uses program synthesis and AI planning techniques [9]. Given a start state St_{start} and an end state St_{end}, it employs reasoning over the pre- and post-conditions of available services, trying to create a plan of putting them together to satisfy the required functionality. It will use the Chinese company information service since its inputs fit the available inputs of company name and place of incorporation, and will proceed to look for a service which can bridge the two states St_c(all information in Chinese) and St_e(all information in English. This approach is inefficient, although many heuristic-based improvements exist, and also hybrids between the top-down and bottom-up approaches [7]. The bigger problem is relying on the broker knowing all available services, with their functionality fully defined and fixed for the duration of the composition.

The approach we propose here also works bottom-up, yet it relies on the composition being created by the service providers in a cooperative fashion. In our example, the service broker will announce the need for the service including descriptions of start and end states on a specialist noticeboard for software services in the financial sector. Software service providers will see this opportunity, and bid to contribute to a solution as a part of a bigger consortium. Provider A, for example, will bid to provide the company

[1] `http://www.ontologyportal.org/`

$$
\begin{aligned}
&Corporation \sqsubseteq Organization \sqsubseteq CognitiveAgent \sqsubseteq SentientAgent\\
&SentientAgent \sqsubseteq Agent \sqsubseteq Entity\\
&StockMarket \sqsubseteq Organization\\
&StockMarketTransaction \equiv FinancialTransaction\\
&\qquad \sqcap \forall hasPatient.Stock\\
&\qquad \sqcap \forall islocated.StockMarket\\
&Stock \equiv FinancialAsset \sqcap \exists issuesBy.Corporation\\
&\qquad \sqcap \forall issuesBy.Corporation\\
&\qquad \sqcap \exists listedOn.Organization \sqcap \forall listedOn.Organization\\
&\qquad \sqcap \exists stockSymbol.SymbolString \sqcap \forall stockSymbol.SymbolString\\
&StockSymbol \equiv \forall hasStock.Stock \sqcap \exists stockMarket.StockMarket\\
&\qquad \sqcap \forall stockMarket.StockMarket\\
&CompanyName \equiv \forall names.Corporation \sqcap \exists names.Corporation\\
&PartialInformation \equiv FactualText \sqcap \forall hasPart.Corparation\\
&CompanyAnnoucement \equiv FactualText \sqcap \forall authors.Corporation\\
&CompanyNews \equiv FactualText\\
&\qquad \sqcap \forall containsInformation.PartialInformation\\
&Corporation \equiv \exists hasAnnouncement4Company.CompanyAnnoucement\\
&\qquad \sqcap \exists hasNews4Company.CompanyNews\\
&TimeSeries \equiv TimeDependentQuality \sqcap \exists hasTime.Time\\
&\qquad \sqcap \exists hasStockSymbol.StockSymbol\\
&\qquad \sqcap \exists hasTimeSeriesPerStock.Value\\
&EODPrices \equiv TimeSerie \sqcap \forall hasRange.ClosingPrice\\
&StockTimeSeries \equiv TimeSeries \sqcap \exists hasStockSymbol.StockSymbol\\
&\qquad \sqcap \forall hasStockSymbol.StockSymbol
\end{aligned}
$$

Fig. 2. Part of an $\mathcal{ALE}$ Terminology in the Finance Domain of SUMO

information in Chinese, whilst Provider B will bid to translate from Chinese to English. This allows for circumstances where services have just appeared on the marketplace and are still unknown to the service broker, or where the service providers are willing to change their service to address a lucrative business opportunity. In our case Provider Aś original service may not provide news relating to the company, and they may be willing to sub-contract such a service and modify their offering to suit the opportunity.

The key to successful operation of this approach, in common with the planning approach, is the availability of a guiding metric indicating if a service is usefully contributing to the eventual solution or not. This is the first main issue that we address. The other main issue is how to trim the number of possible partial solutions, avoiding the unnecessary creation of millions of consortia with only marginal difference between their offerings. These two issues will be explored further in the remainder of this paper.

3 Preliminaries

To support the cooperative assembly of composite services by software agents, we need formal knowledge (semantic annotation) about the inputs and outputs of these services as well as about their functionality.

3.1 Semantically Annotating Services, Links and Compositions

To allow automated reasoning, we need to annotate software services with formal semantic descriptions of their functionality (or goals), inputs, outputs, pre-conditions, and post-conditions. These services are then known as *Semantic web services* [10].

In principle, the formal semantic specifications of semantic web services are based on Description Logics (DL) [11]. Within the domain of information processing, we can focus on three parameters - functionality, inputs and outputs, since within this domain, a pre-condition can usually be simplified to the availability of all the needed inputs, and the post-condition is usually the availability of all or some of the declared outputs.

The functionality, input and output parameters of services are described according to a common ontology or Terminology $\mathcal{T}$ (e.g., Fig.2), where the OWL-S profile [12] or SA-WSDL [13] can be used to describe them through semantic annotations[2].

Semantic links [6] between input and output parameters of services can be then defined, based on semantic similarities between an output of a service $s_1(Out_{s_1})$ and the input of a subsequent service $s_2(In_{s_2})$, where both are DL concept descriptions (e.g., Fig.2). Its objective is to measure how services could fit together in a semantic context. However, reaching a composition model where services are *semantically* linked is far from trivial, with more services required to fill the missing gap. We use "feeder" F and "sink" S to refer to pairs of concepts describing a desired transformation by a "yet-to-be determined" set of services, "feeder" serves as an input to this transformation which produces "sink" as a result. In this work, the gap between F and S is called *semantic gap*, referring to both semantic concept descriptions and services which are missing to properly compose services and "plug" that gap.

In this paper we use semantic gaps $g_{i,j}$ between concepts F and S to guide the cooperative composition process. In the extreme initial case the semantic gaps covers the whole of the desired composite service ($F = In_{start}$ and $S = Out_{end}$). In the general case, though, the semantic gap will be between the output of a service Out_{s_i} and the input of another service In_{s_j}.

$$g_{i,j} \doteq \langle s_i, Sim_{\mathcal{T}}(F, S), s_j \rangle \quad (1)$$

where the *feeder* $F = Out_{s_i}$ and the *sink* $S = In_{s_j}$ are DL-based concept descriptions.

Given a terminology $\mathcal{T}$, [15] and [16] value the range of $Sim_{\mathcal{T}}$ along five matching types: i) *Exact* i.e., $F \equiv S$, ii) *PlugIn* i.e., $F \sqsubseteq S$, iii) *Subsume* i.e., $S \sqsubseteq F$, iv) *Intersection* i.e., $\neg(F \sqcap S \sqsubseteq \bot)$ and v) *Disjoint* i.e., $F \sqcap S \sqsubseteq \bot$ (i.e., inconsistent gap).

The valuation of semantic gaps is important to judge how two services could be close or far in term of descriptions they manipulate. To this end, the matching function $Sim_{\mathcal{T}}$ of (1) enables, at design time, finding semantic compatibilities (i.e., Exact, PlugIn, Subsume, Intersection), incompatibilities (i.e., Disjoint) among independently defined service descriptions, and thus can be used to define the size ("quality") of gaps between services. The process model of web service composition and its semantic gaps is specified by a directed graph which has web service specifications s_i or semantic gaps $g_{i,j}$ (description dissimilarities, serving as placeholders for new composite

[2] Distributed ontologies [14] for achieving semantic annotation are not considered here but are largely independent of the problem addressed in this work.

services) as its edges, and the input and output concepts of services as its nodes. The start node(s) of the graph will be the input concept(s) In_{start}, whereas the end nodes of the graph will be the output concept(s) Out_{end}.

Semantics gaps in composition should be filled with appropriate services to bridge different data descriptions, thus we suggest to compute the information contained in S but not in F, using Concept Abduction (Definition 3 in [17]). We adapt it from [18] as follows.

Definition 1. ***(Concept Abduction)***
Let $\mathcal{L}$ be a DL, F, S be two concepts in $\mathcal{L}$, and $\mathcal{T}$ be a set of axioms in $\mathcal{L}$. A Concept Abduction Problem, denoted as $S \backslash F$ consists in finding a concept $H \in \mathcal{L}$ such that $\mathcal{T} \not\models F \sqcap H \equiv \bot$, and $\mathcal{T} \models F \sqcap H \sqsubseteq S$.

In case a semantic gap (1) is not valued by a Disjoint or Exact matching, Definition 1 is applied to obtain $S \backslash F$. This information refers to the *Missing Description* required but not provided by F to resolve a gap with S.

3.2 Quality of Semantic Gaps

In this section we present the quality model used to evaluate different consortia and to also guide contributions by individual service providers. The model, inspired from [6], evaluates compositions based on both non functional (QoS attributes such as price, response time, etc. [19]) and semantic quality, based on the quality of semantic gaps. In more detail two generic quality criteria for a semantic gap (1) are considered: its i) *Common Description* rate (Definition 2), and ii) *Matching Quality*.

We compute *Common Description* between *feeder* F and *sink* S concepts using Concept Abduction as follows:

Definition 2. ***(Common Description rate of a Semantic Gap)***
Given a semantic gap $g_{i,j}$ between F and S, the Common Description rate $q_{cd} \in (0, 1]$ provides one possible measure for the degree of similarity between F and S and the quality of the semantic gap $g_{i,j}$. This rate is computed using the following expression:

$$q_{cd}(g_{i,j}) = q_{cd}(F, S) = \frac{|lcs(F, S)|}{|S \backslash F| \ + \ |lcs(F, S)|} \tag{2}$$

This criterion estimates the proportion of descriptions which is well specified by "feeder" *F in* "sink" *S.*

The expressions in between "|" refer to the size of concept descriptions ([20] p.17) i.e., $|\top|$, $|\bot|$, $|A|$, $|\neg A|$ and $|\exists r|$ is 1; $|C \sqcap D| \doteq |C| + |D|$; $|\forall r.C|$ and $|\exists r.C|$ is $1 + |C|$.

Definition 3. ***(Matching Quality of a Semantic Gap)***
The Matching Quality q_m of a semantic gap $g_{i,j}$ is a value in $(0, 1]$ defined by $Sim_{\mathcal{T}}(i, j)$ i.e., either 1 (Exact), $\frac{3}{4}$ (PlugIn), $\frac{1}{2}$ (Subsume) or $\frac{1}{4}$ (Intersection).

At composition level, Common Description rate measures the average similarity between all corresponding pairs (F, S) which create semantic gaps in the composition.

The matching quality of a composition c estimates the overall matching quality of its semantic gaps. Contrary to the common description rate, this criteron aims at easily distinguishing and identifying between very good and very bad matching quality.

3.3 Quality of Service Compositions

We present definitions for comparing and ranking different compositions along the common description rate and matching quality dimension. The rules for aggregating quality values (Table 1) for any concrete composition c are driven by them. In more details the approach for computing semantic quality of c is adapted from the application-driven heuristics of [6], while the computation of its non functional QoS is similar to [21].

Definition 4. ***(Common Description rate of a Composition)***
The Common Description rate of a composition c measures the average degree of similarity between all corresponding pairs (F, S) which create semantic gaps in c.

The Common Description rate Q_{cd} of both a sequential and AND-Branching composition is defined as the average of its semantic gaps' common description rate $q_{cd}(g_{i,j})$. The common description rate of an OR-Branching composition is a sum of $q_{cd}(g_{i,j})$ weighted by $p_{g_{i,j}}$ i.e., the probability that semantic gap $g_{i,j}$ be chosen at run time. Such probabilities are initialized by the composition designer, and then eventually updated considering the information obtained by monitoring the workflow executions.

Definition 5. ***(Matching Quality of a Composition)***
The matching quality of a composition estimates the overall matching quality of its semantic gaps. Contrary to the common description rate, this criteron aims at easily distinguishing and identifying between very good and very bad matching quality.

The matching quality Q_m of a sequential and AND-Branching composition is defined as a product of $q_m(g_{i,j})$. All different (non empty) matching qualities involved in such compositions require to be considered together in such a (non-linear) aggregation function to make sure that compositions that contains semantic gaps with low or high matching quality will be more easily identified, and then pruned for the set of potential solutions. The matching quality of an OR-Branching composition is defined as its common description rate by replacing $q_{cd}(g_{i,j})$ with $q_m(g_{i,j})$.

Details for computing **Execution Price** Q_{pr} and **Response Time** Q_t can be found in Table 1, and further explained in [21].

Using Table 1, the quality vector of any concrete composition can be defined by:

$$Q(c) \doteq (Q_{cd}(c), Q_m(c), Q_t(c), Q_{pr}(c)) \tag{3}$$

The adopted quality model has a limited number of criteria (for the sake of illustration), yet (3) is extensible, allowing new functional and non-functional criteria such as reputation, availability, reliability, etc.

Table 1. Quality Aggregation Rules for Semantic Web Service Composition

Composition Construct	Quality Criterion			
	Functional		Non Functional	
	Q_{cd}	Q_m	Q_t	Q_{pr}
Sequential/ AND- Branching	$\frac{1}{\lvert g_{i,j}\rvert}\sum_{g_{i,j}} q_{cd}(g_{i,j})$	$\prod_{g_{i,j}} q_m(g_{i,j})$	$\sum_{s_i} q_t(s_i)$ / $\max_s q_t(s)$	$\sum_{s_i} q_{pr}(s_i)$
OR-Branching	$\sum_{g_{i,j}} q_{cd}(g_{i,j}).p_{g_{i,j}}$	$\sum_{g_{i,j}} q_m(g_{i,j}).p_{g_{i,j}}$	$\sum_{s_i} q_t(s_i).p_{s_i}$	$\sum_{s_i} q_{pr}(s_i).p_{s_i}$

4 Our Approach to Emergent Service Composition

4.1 Outline of the Overall Approach

Software agents representing service providers observe a number of noticeboards of interest. When a description of a requested composite service appears, each agent will check if they can meaningfully contribute to providing a partial solution, and record this partial solution onto the noticeboard. Using our example from Section 2, AgentB representing a translator service s_b from Chinese to English will record a partial solution $St_c \xrightarrow{S_b} St_{end}$ onto the noticeboard.

Another agent (AgentA) will notice that they can now extend the new partial solution into a full solution by providing the service s_a to link $St_{start} \xrightarrow{s_a} St_c$, and apply to join the consortium behind the partial solution (in our case AgentB only).

Multiple partial solutions are allowed addressing the same business opportunity. Some of them will be elaborated into complete solutions, which will be internally consistent and deliver the requested functionality. The evaluation of all complete solutions is conducted by the service composer agent, using the quality vector from (3) in Section 3. A consortium will also apply quality of service composition metrics to decide between a number of potential applicants, this is described in detail in Section 5.

Initially, we focus on how we specify each of the states within the system, and how, using this state representation, each agent can determine if one of their services can meaningfully contribute to a partial solution on the noticeboard.

4.2 State Description

The system we propose targets the construction of composite information processing services, where a set of service components are "wired" together to deliver a set of information items as outputs for each set of information items provided as inputs.

Each component service within such a system, and the system itself, can be specified as transforming an input set of information items $In \equiv \{i_1, i_2, \cdots, i_n\}$ into an output set of information items $Out \equiv \{o_1, o_2, \cdots, o_n\}$. Each information item i_i and o_i is defined by a concept from a terminology $\mathcal{T}$, shared by all agents.

We consider a partial solution ps to be defined by two states - its input state St_{in} and its output state St_{out}: $ps = \langle St_{in}, St_{out}\rangle$

Because of the information processing nature of the domain we are investigating, we consider a state to be defined by the set of information items available at this state. Therefore $St_{in} \equiv In_{ps}$ whilst $St_{out} \equiv Out_{ps}$.

The required composite service is also defined by its two states St_{start} and St_{end}. When $St_{in} = St_{start}$ and $St_{out} = St_{end}$, we have a full solution.

4.3 Concept Similarity Metric

Each bid to join a partial solution changes one of the states defining a partial solution, either providing further processing to bring St_{out} closer to the target St_{end}, or some pre-processing to bring St_{in} closer to the in information available at St_{start}.

In the general case, each agent will apply their knowledge of the problem domain and the overall declared goal of the target composition to determine when their contribution is meaningful. However, within the information processing domain, we can formulate a more specific heuristic metric to help both the proposers and the accepting consortia judge if a contribution is meaningful (or "constructive"). To derive this heuristic, we use the semantic distance between the concepts defining two information items (semantic gap) using the concept of Common Description Rate as defined in (2). Recall that in this interpretation, the "feeder" F and "sink" S describe a desired transformation by a "yet to be determined" set of services.

4.4 Heuristics for Joining a Partial Solution

The heuristic described here is used both by the service provider agents when they decide which service from several they should propose as an extension to a partial solution, and by the consortium behind a partial solution when they decide which extension bid to accept from several alternatives.

For given partial solution $ps = \langle St_{in}, St_{out} \rangle$ and a target solution $G = \langle St_{start}, St_{end} \rangle$, an agent will apply to join the consortium and extend the partial solution if their service s can reduce one of the existing concept gaps $q_{cd}(g_{i,j})$.

In formal terms we have two cases of reducing a concept similarity value:

Case 1: Reducing distance from the "input" side:

$$(F \in St_{start}) \wedge (S \in St_{in}) \wedge (Out_S = S) \\ \wedge (q_{cd}(F, In_s) < q_{cd}(F, S)) \tag{4}$$

Case 2: Reducing the distance from the "output" side::

$$(F \in St_{out}) \wedge (S \in St_{end}) \wedge (In_s = F) \\ \wedge (q_{cd}(Out_s, S) < q_{cd}(F, S)) \tag{5}$$

For two alternative services provided by the agent, it will choose to put forward the service which minimises the respective q_{cd}. We presume the services are alternative, *i.e.* they extend the partial solution from the same point along the same direction.

The same heuristic will be used by the consortium behind a partial solution when they have to evaluate several alternative bids by providers who offer their services to extend the existing partial solution.

4.5 Semantic Quality of Partial Solutions

Based on the composition quality metric we defined in Section 3.2, any compositions (partial or complete) can be compared along two dimensions i.e., non functional and semantic quality. Therefore the quality of compositions in Table 1 can be compared by analysing their Q_{cd}, Q_m, Q_{pr} and Q_t elements. For instance $Q(c_1) > q(c_2)$ iff the quality of composition c_1 is better than quality of c_2 along each and every quality criteria. In case of conflicts e.g., the value of Q_{cd} and Q_m is better for c_1 but worse for values of Q_{pr} and Q_t, we compare a weighted average (with a weight of $\frac{1}{4}$) of their normalised components.

5 Reducing the Number of Partial Solutions Considered

Without any measures to alleviate the problem, the self-interested agents representing service providers will attempt to propose their services every time when there is an input or an output matching. Since alternative partial solutions are in principle allowed by the approach, this may quickly lead to a combinatorial explosion regarding the number of partial solutions held at a given noticeboard.

The following simple heuristics are designed to ensure this is controlled as far as possible. They represent a suggested procedural framework within which the composition may proceed in a controlled manner.

5.1 Service Providers Appraise Their Bids

A service provider will consider it possible to place a bid for extending a partial solution $ps = \langle St_{in}, St_{out} \rangle$ if the inputs or the outputs of one of its services can extend the partial solution (i.e. $In_s \in St_{out} \vee Out_s \in St_{in}$). The only services considered for this will be those with matching functional classification, *i.e.* those which fit to the declared goal for the service composition on the noticeboard.

If a provider recognises more than one service as "fitting" the conditions, they will conduct an internal evaluation using the same rule which the consortium will apply to evaluate competing offers. This means they will select the service which achieves the smallest Common Description Rate q_{cd}.

A service provider will have a limited visibility of all the parameters of the partial solution on the noticeboard, so they would not be able to calculate and use the full quality model for the partial composition (3).

5.2 Consortia behind Partial Solutions Evaluate Competitive Bids

A consortium will choose a single provider from a number of alternative providers offering to extend a solution in the same direction (i.e. providing the same output concept), using two metrics:

1. Minimising q_{cd}. This will ensure the consortium picks the "most effective offer" between the different service providers.
2. The overall service quality for the partial solution. This criteria will also be used by the service broker when evaluating from different complete solutions, so the consortium should extend reasonable effort to maximise this metric.

6 Example of Applying the Approach to Financial Services

In the example below we demonstrate how our approach can be used to support the scenario from Section 2.

As a first step we calculate the Common Description Rates $q_{cd}(F, S)$ for the example from the $\mathcal{ALE}$ Terminology in Figure 2. The results are as follows:

Table 2. Computation of Common Description (The lower the better; "-" means no compatible concept under $\sqsubseteq_{\mathcal{T}}$)

	CompanyName	StockSymbol	EODPRices	AR	Volatility	CompanyAn.	CompanyNews	CompanyAnEn
CompanyName	0	0.23	0.41	0.69	0.95	0.64	0.71	0.72
StockSymbol	0.12	0	0.35	0.55	0.90	0.65	0.79	0.66
EODPRices	0.33	0.31	0	0.41	0.67	0.85	0.81	0.89
AR	0.85	0.41	0.62	0	0.71	-	-	-
Volatility	0.75	0.55	0.55	0.88	0	0.95	0.91	0.97
CompanyAn.	0.55	0.67	-.88	-	0.91	0	0.12	0.11
CompanyNews	0.76	0.63	0.76	-	0.92	0.21	0	0.09
CompanyAnEn	0.66	0.60	0.89	-	0.88	0.09	0.11	0

Please note that rows are the Sink (S) concepts, whilst the columns are the Feeder (F) concepts, so $q_{cd}(CompanyName, StockSymbol) = 0.12$.

Services in the System. In addition we have a number of services aligned with the domain of financial information provision. The service owners are monitoring the relevant financial service requests noticeboards. Figure 3 represents these services (in bold and italics) as a network organised around the concepts they process. Please note this is for visualisation only, and the network does not exist, since the knolwdge about it is distributed amongst service providers.

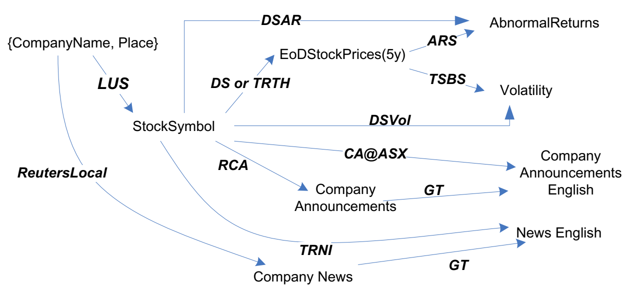

Fig. 3. A visualisation of all financial services participating in our example, this knowledge is not available to any single agent in the system but is distributed amongst service provider agents

Sequence of Contributions. Carl's service requestor agent will formulate the following noticeboard request:

$$St_{start} \equiv \{companyName, StockMarket\}$$
$$St_{end} \equiv \{AbnormalReturns, Volatility, CompanyAnEn, CompanyNews\}$$

AgentA, representing a stock market provider, will consider the provision of one of two services for calculating Abnormal Returns (AR):

1. ARS, where $In_{ARS} = EODPrices$ and $Out_{ARS} = AR$, or
2. $DSAR$, where $In_{DSAR} = StockSymbol$ and $Out_{DSAR} = AR$

AgentA will then calculate $q_{cd}^{ARS}(CompanyName, EODPrices) = 0.33$ whilst $q_{cd}^{DSAR}(CompanyName, StockSymbol) = 0.12$. Therefore AgentA will propose $DSAR$ to the noticeboard, and will form the first partial solution $StockSymbol \xrightarrow{DSAR} AR$. Note that for brevity we have not enumerated the information items which form part of the start and end states yet have not been used in this step.

AgentA also has a service $DSVol$ which calculates $Volatility$ using $StockSymbol$ as an input ($In_{DSVol} = StockSymbol$ and $Out_{DSVol} = Volatility$. This service is appropriate (matching inputs and outputs), and also it has no "internal competition" within AgentA. It will thus be submitted to the noticeboard as a bid for extending the current partial solution. However, the noticeboard has also received another overlapping bid from AgentB, where its service $TSBS$ can take $EODPrices$ and calculate $Volatility$ ($In_{TSBS} = EODPrices$ and $Out_{TSBS} = Volatility$).

The consortium behind the existing partial solution will compare

1. $q_{cd}^{DSVol}(CompanyName, StockSymbol) = 0.12$ and
2. $q_{cd}^{TSBS}(CompanyName, EODPrices) = 0.33$

The choice to extend the partial solution will therefore be Option 1, ie. $DSVol$. We now have a new partial solution illustrated in Fig. 4-(a).

AgentC will now see this partial solution where $StockSymbol \in St_{input}$, and since it has a stock lookup service LUS, where $In_{LUS} = CompanyName$ and $Out_{LUS} = StockSymbol$, it offers it to the consortium behind the partial solution. This service is in a competition with FT Lookup Service $FTLUS$ and Reuters LookUP Service $RLUS$, all three taking exactly the same parameters. The consortium will make the choice between these three, based on the overall quality of the partial solutions formed by each of these candidate services. Considering Sections 3.2 and 4.5, we have Q_{cd} and Q_m equal amongst all three candidate services, so the comparison will be based on differences in price and speed (Q_{pr} and Q_t). The new service will be added to the existing partial solution in a sequence, therefore from Table 1 we have the formulae for $Q_{pr} = \sum_{s_i} q_{pr}(s_i)$ and $Q_t = \sum_{s_i} q_t(s_i)$. The three candidate services can therefore be compared directly in terms of quality and price, and any trade-offs between them considered in terms of weighting factors of these two parameters.

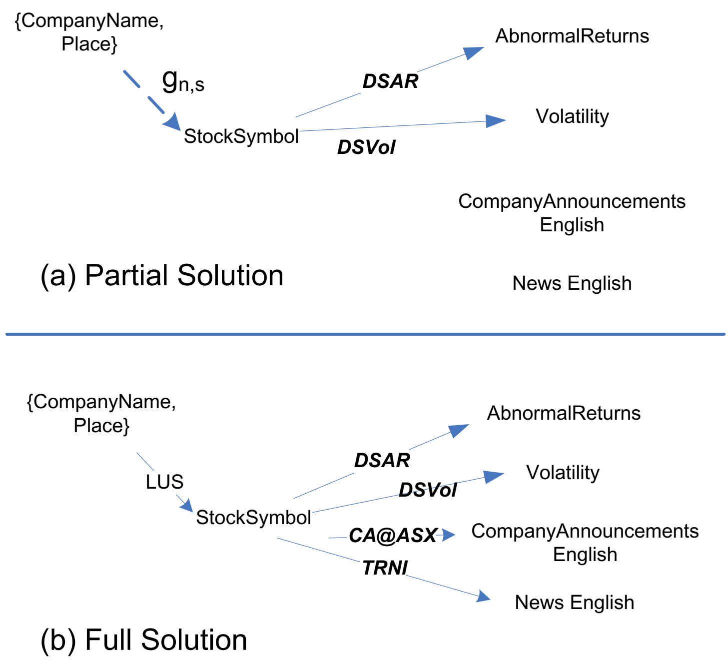

Fig. 4. Two stages of the solution formation

For the purpose of this example we can presume that LUS has the best combinaton of quality and price, and that it will be added to the partial solution on the noticeboard. The solution is still not complete, though, because we have no service providing $CompanyAnnouncementsEnglish$ and $NewsEnglish$. $TRNI$ uses $StockSymbol$ to deliver $NewsEnglish$, so the consortium will prefer it to ReutersLocal where the news are not necessarily in English. Similarly, $CA@ASX$ takes $StockSymbol$ again, and delivers $CompanyAnnouncementsEnglish$, so it will be preferred to the combination of RCA and GT (GoogleTranslate) services.

We therefore obtain the full solution depicted in Figure 4-(b). This solution will be proposed to the service requestor agent, any competing solutions will be evaluated using the composition quality metrics from Table 1.

A top-down solution to the same problem, driven by a centralised decomposition knowledge, would depend on the way in which this knowledge is structured, rather than on the ways in which the available services could fit best together as in the example above. For example, if the knowledge decomposes the problem along functional lines into a sub-solution for stock market information, a sub-solution for company announcements and a sub-solution for company news. If the centralised decomposition knowledge does not have a further breakdown indicating the use of a stock symbol lookup service for each of these sub-solutions, it will fail to find a solution within the currently available set of services. But even if such knowledge is available, the top-down approach will not be able to identify the synergy of having only one stock symbol lookup service, which feeds its output into all other information providing services.

7 Related Research

The role of service providers is normally not in the focus of researchers in service composition, apart from the work on quality assurances and monitoring. However, representing providers through software agents allows innovative solutions to the process of assembling robust and efficient composite services.

For example, having software agents allows us to use agent negotiation, an effective way of addressing the complex issues associated with automated service composition [22]. Negotiation processes range from simple one-shot interactions to handling counter proposals across different processes and facilitating agent-based coalition formation. The subject of coalition-formation is explored in [23] and agent-based coalition formation for service composition is discussed in [24].

There are previously proposed systems where agent-based service providers play an active role in service composition to achieve robust composition on a semantic level [25, 26]. Both papers consider fixed centralised knowledge about the composition structure, and employ agents to negotiate how to turn all semantic links into robust ones.

An application of agents to create emergent compositions of software services is considered in [27], a PhD thesis focused on using agents for emergent team formation in the domain of automotive manufacturing. The work points out some major issues in applying such ideas in the domain of software services, results which have inspired our work in addressing these issues using semantic models and composition mechanisms.

8 Discussion and Conclusion

Our decentralised approach to *cooperative service composition* is driven by service providers. It avoids the need for centrally available problem decomposition knowledge, instead relying on providers' knowledge of their services and on semantic measures of similarity to guide the composition process. It creates potential for innovative compositions, and is suitable for operation in an open domain, where the list of available services and service providers can change dynamically.

The approach is inherently less efficient than centralised top-down decomposition, and its computational complexity is potentially as high as brute-force planning approaches. However, we use semantic similarity metrics to limit the number of partial solutions considered and to reduce the time necessary for the approach.

At present, the approach is tuned to the domain of information processing, which lends itself to semantic reasoning with inputs and outputs of services. In the future we will endevour to extend the approach to include reasoning with state information which is not reduced to availability of inputs and outputs.

The other directions of expanding our work are:

– We will explore reasoning with semantic information about domain structure rather than simply specialisation-based abduction as in the current paper.
– At present agents are only allowed to bid if the semantic links they introduce are exact match, in our future work we will relax this limitations to increase the variety of services eligible for inclusion in our partial solutions, and to exploit the concept of functional quality of semantic links(*c.f.* [6]) and their robustness.

– We will introduce a more general distance model which will allow contributions to gaps “in the middle” of a partial solution. The quality model is capable of covering this, yet at present we are only extending partial solution “at one end”.

In terms of strategies for reducing the numbers of partial solutions, the current restriction to only choosing the best candidate for a given extension may be sub-optimal in terms of the complete solutions thus generated. We will implement a simulator based on the approach so that we can explore the tradeoff between number of candidate solutions allowed, the overall quality of the solutions obtained and the time taken to obtain a solution. We will also explore the effectiveness of alternative control strategies for reducing the set of partial solutions.

References

1. Bennett, K., Munro, M., Xu, J., Gold, N., Layzell, P., Mehandjiev, N., Budgen, D., Brereton, P.: Prototype implementations of an architectural model for service-based flexible software. In: Hawaii International Conference on System Sciences, vol. 3, p. 76b (2002)
2. Papazoglou, M.P., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: A research roadmap. International Journal of Cooperative Information Systems 17(2), 223–255 (2008)
3. Wu, D., Parsia, B., Sirin, E., Hendler, J., Nau, D.S.: Automating DAML-S Web Services Composition Using SHOP2. In: Fensel, D., Sycara, K., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870, pp. 195–210. Springer, Heidelberg (2003)
4. Wielinga, B., Schreiber, G.: Configuration-design problem solving. IEEE Expert: Intelligent Systems and Their Applications 12(2), 49–56 (1997)
5. Motta, E.: Parametric Design Problem Solving - Reusable Components For Knowledge Modelling Case Studies. IOS Press (1999)
6. Lécué, F., Mehandjiev, N.: Seeking quality of web service composition in a semantic dimension. IEEE Trans. Knowl. Data Eng. 23(6), 942–959 (2011)
7. Erol, K., Hendler, J., Nau, D.S.: Htn planning: complexity and expressivity. In: Proceedings of the Twelfth National Conference on Artificial intelligence, AAAI 1994, vol. 2, pp. 1123–1128. American Association for Artificial Intelligence, Menlo Park (1994)
8. ten Teije, A., van Harmelen, F., Wielinga, B.: Configuration of Web Services as Parametric Design. In: Motta, E., Shadbolt, N.R., Stutt, A., Gibbins, N. (eds.) EKAW 2004. LNCS (LNAI), vol. 3257, pp. 321–336. Springer, Heidelberg (2004)
9. McIlraith, S.A., Son, T.C.: Adapting golog for composition of semantic web services. In: KR, pp. 482–496 (2002)
10. Sycara, K.P., Paolucci, M., Ankolekar, A., Srinivasan, N.: Automated discovery, interaction and composition of semantic web services. J. Web Sem. 1(1), 27–46 (2003)
11. Baader, F., Nutt, W.: The Description Logic Handbook: Theory, Implementation, and Applications (2003)
12. Ankolenkar, A., Paolucci, M., Srinivasan, N., Sycara, K.: The OWL-S coalition, OWL-S 1.1. Technical report (2004)
13. Kopecký, J., Vitvar, T., Bournez, C., Farrell, J.: Sawsdl: Semantic annotations for WSDL and XML schema. IEEE Internet Computing 11(6), 60–67 (2007)
14. Euzenat, J.: Semantic precision and recall for ontology alignment evaluation. In: IJCAI, pp. 348–353 (2007)

15. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic Matching of Web Services Capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 333–347. Springer, Heidelberg (2002)
16. Li, L., Horrocks, I.: A software framework for matchmaking based on semantic web technology. In: WWW, pp. 331–339 (2003)
17. Noia, T.D., Sciascio, E.D., Donini, F.M., Mongiello, M.: Abductive matchmaking using description logics. In: IJCAI, pp. 337–342 (2003)
18. Lécué, F., Delteil, A., Léger, A.: Applying abduction in semantic web service composition. In: ICWS, pp. 94–101 (2007)
19. O'Sullivan, J., Edmond, D., ter Hofstede, A.H.M.: What's in a service? Distributed and Parallel Databases 12(2/3), 117–133 (2002)
20. Küsters, R.: Non-Standard Inferences in Description Logics. LNCS (LNAI), vol. 2100. Springer, Heidelberg (2001)
21. Cardoso, J., Sheth, A.P., Miller, J.A., Arnold, J., Kochut, K.: Quality of service for workflows and web service processes. J. Web Sem. 1(3), 281–308 (2004)
22. Ben Hassine, A., Matsubara, S., Ishida, T.: A Constraint-Based Approach to Horizontal Web Service Composition. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 130–143. Springer, Heidelberg (2006)
23. Shehory, O., Kraus, S.: Methods for task allocation via agent coalition formation. Artif. Intell. 101(1-2), 165–200 (1998)
24. Muller, I., Kowalczyk, R., Braun, P.: Towards agent-based coalition formation for service composition. In: IAT 2006: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology, pp. 73–80. IEEE Computer Society, Washington, DC (2006)
25. Lecue, F., Wajid, U., Mehandjiev, N.: Negotiating robustness in semantic web service composition. In: Seventh IEEE European Conference on Web Services, ECOWS 2009, pp. 75–84 (November 2009)
26. Mehandjiev, N., Lécué, F., Wajid, U.: Provider-Composer Negotiations for Semantic Robustness in Service Compositions. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 205–220. Springer, Heidelberg (2009)
27. Carpenter, M.: Cooperative team formation using distributed decomposition knowledge. PhD thesis, Manchester Business School (2010)