Chapter 1

Semantic Web Service Coordination

Matthias Klusch

Semantic service coordination aims at the coherent and efficient discovery, composition, negotiation, and execution of semantic Web services in a given environment and application context. What makes coordination of services in the semantic Web different from its counterpart in the Web is its far more advanced degree of automation through means of logic based reasoning on heterogeneous service and data semantics.

In this chapter, we only focus on approaches to semantic discovery and composition planning of semantic Web services, and briefly comment on their interrelationships and selected open problems of both fields. For reasons of space limitations, the set of presented examples is representative but not exhaustive.

1.1 Semantic Service Discovery

Semantic service discovery is the process of locating existing Web services based on the description of their functional and non-functional semantics. Discovery scenarios typically occur when one is trying to reuse an existing piece of functionality (represented as a Web service) in building new or enhanced business processes. Both service oriented computing and the semantic Web envision intelligent agents to proactively pursue this task on behalf of their clients.
Service discovery can be performed in different ways depending on the service description framework, on means of service selection, and on its coordination through assisted mediation or in a peer-to-peer fashion. In general, any semantic service discovery framework needs to have the following components ([34]).

- Service description: Formal means to describe the functional and non-functional semantics of Web services.

- Service selection: Reasoning mechanisms for service matching, that is the pairwise comparison of service descriptions in terms of their semantic relevance to the query, and ranking of the results based on partially or totally ordered degrees of matching and preferences.
- Discovery architecture: Environmental assumptions on (centralized, decentralized) network topology, service information storage (e.g. distribution of services, ontologies, registries) and location mechanisms, and functionality of agents involved (e.g. service requester, provider, middle agents).

In the following, we survey existing approaches to semantic service selection and discovery architectures.

1.1.1 Classification of SWS matchmakers

Semantic service matching determines whether the semantics of a desired service (or goal) conform to that of an advertised service. This is at the very core of any semantic service discovery framework. Current approaches to semantic service matching can be classified according to

- what kinds and parts of service semantics are considered for matching, and
- how matching is actually be performed in terms of non-logic based or logic based reasoning on given service semantics or a hybrid combination of both, within or partly outside the respective service description framework (cf. figure 1.1).

Non-logic, logic, and hybrid semantic service matching The majority of SWS matchmakers performs logic based semantic service matching. That is, they are keeping with the original idea of the semantic Web to determine semantic relations between resources including services based on logical inferencing on their annotations grounded in description logics (DL) and/or rules (cf. chapter 3). In fact, the set of logic based SWS matchmakers for OWL-S and WSML still outnumbers the complete set of non-logic based or hybrid semantic matchmakers available for any SWS description format. Non-logic based semantic matchmaker do not perform any logic based reasoning but compute the degree of semantic matching of given pairs of abstract service descriptions based on, for example, syntactic similarity measurement, structured graph matching, or numeric concept distance computations over given ontologies.

Service profile and process model matching Most SWS matchmakers perform service profile rather than service process model matching. Service profile matching (so called "black-box" service matching) determines the semantic correspondence between services based on the description of their profiles. The profile of a service

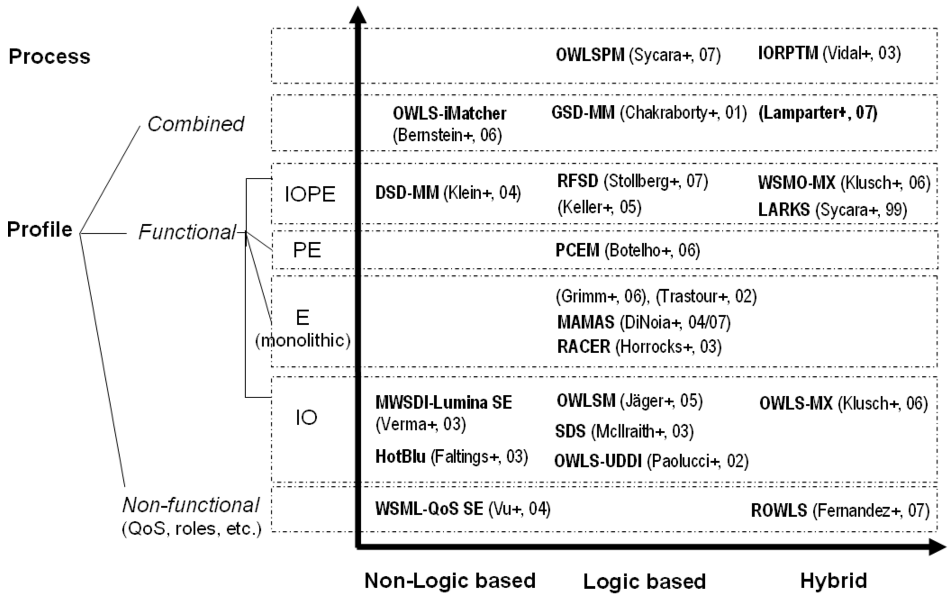

Figure 1.1: Categories of existing SWS matchmakers.

describes what it actually does in terms of its signature, that is its input and output (IO), as well as preconditions (P) and effects or postconditions (E), and non-functional aspects such as the relevant business category, name, quality, privacy and pricing rules of the service. We classify additional context information for service matching such as the organisational (social or domain) roles, or geographic location of service requesters and providers in their interaction as non-functional. Service process-oriented matching (so-called "glass-box" service matching) determines the extent to which the desired operational behavior of a given service in terms of its process control and data flow matches with that of another service. Like with service profile matching, we can distinguish between non-logic based, logic based and hybrid semantic process matching approaches depending on whether automated reasoning on operational semantics specified in some certain logic or process algebraic language (e.g. CCS, π-calculus) is performed, or not. An overview of relevant approaches to process mining for process discovery is given in (van der Aalst & Weijters, 2004)[104].

Supported SWS description formats Each of the implemented stand-alone SWS matchmakers shown in figure 1.1 supports only one of many existing SWS description formats (cf. chapter 3) as follows. This list is representative but not exhaustive.

- **OWL-S** matchmakers: Logic based semantic matchmakers for OWL-S services are the OWLSM (Jäger et al., 2005)[42] and OWLS-UDDI (Paolucci et al., 2002)[75] focussing on service input/output (IO) matching, and the PCEM (Botelho et al., 2006)[16] that converts given OWL-S services to PDDL actions for PROLOG based matching of preconditions and effects. Further OWL-S matchmakers are, the hybrid service IO matchmaker OWLS-MX (Klusch et al., 2006)[53], the hybrid non-functional profile matchmaker ROWLS (Fernandez et al., 2006)[31], and the non-logic based service IOPE matchmaker OWLS-iMatcher (Bernstein & Kiefer, 2006)[13]. An approach to logic based OWL-S process model verification is in (Vaculin & Sycara, 2007)[103], an approach to the matching of process dependency graphs based on syntactic similarity measurements while (Bae et al., 2006)[10] present an approach to the matching of OWL-S process dependency graphs based on syntactic similarity measurements, and Bansal and Vidal (2003)[11] propose a hybrid matchmaker that recursively compares the DAML-S process model dependency graphs.

- **WSML** matchmakers: Implemented approaches to WSML service discovery include the hybrid semantic IOPE matchmaker WSMO-MX (Kaufer & Klusch, 2006)[46], and the non-logic based search engine of the WSMO studio for non-functional (QoS based) WSML service discovery in P2P networks (Vu et al., 2006)[106]. Other approaches to logic based WSML service IOPE matchmaking are presented in (Keller et al., 2005; Stollberg et al., 2006)[45, 97], though it is unclear to what extent they have been implemented.

- **WSDL-S/SAWSDL** matchmakers: The METEOR-S WSDI discovery infrastructure (Verma et al., 2004)[105] and the UDDI based search component Lumina[1] are the only tool support of searching for SAWSDL services so far. While searching with Lumina is keyword based, the MWSDI discovery of SAWSDL services relies on non-logic based matching means.

- **Monolithic DL based** matchmakers: Only very few matchmaker are agnostic to the above mentioned structured SWS description formats without conversion by accepting monolithic descriptions of services in terms of a single service concept written in a given DL. In this case, semantic matching directly corresponds to DL inferencing, that is, semantic service matching is done exclusively within the logic theory such as performed by RACER (Li & Horrocks, 2003)[59], MaMaS[2](Di Noia et al., 2004; 2007)[26, 27], and in (Grimm et al., 2006)[35]. Recently, an implemented approach to matching of monolithic service descriptions in OWL-DL extended with (non-functional) pricing policies modeled in DL-safe SWRL rules according to given preferences using SPARQL queries to a service repository is presented in (Lam-

[1] lsdis.cs.uga.edu/projects/meteor-s/downloads/Lumina/
[2] sisinflab.poliba.it/MAMAS-tng/

parter & Ankolekar, 2007)[56].

- **Others**: Non-logic based service IOPE profile matchmakers for other structured service description formats are the DSD matchmaker (Klein & König-Ries, 2004)[49] for DIANE services, the numeric service IO type matching based HotBlu matchmaker (Constantinescu & Faltings, 2003)[23], and the hybrid service IOPE matchmaker LARKS for services in an equally named format (Sycara et al., 2002)[99].

In the following, we discuss each category of semantic Web service matching together with selected representative examples of the above mentioned SWS matchmakers in more detail. This is complemented by a classification of existing service discovery architectures in which these matchmakers can be used in principle, or explicitly have been designed for. As stand-alone implementations, each of them classifies, in principle, as centralized service discovery system, though a few of them have been also tested for, or were originally developed for decentralized P2P service retrieval systems like the OWLS-MX and the OWLS-UDDI matchmaker, respectively, the WSMO-QoS search engine and the DReggie/GSD matchmaker.[3]

1.1.2 Logic based semantic service profile matching

As mentioned above, logic based semantic service matchmakers perform purely logical reasoning on service semantics. The majority of such SWS matchmakers focus on comparing the formal profile semantics of a given pair of services. The concepts and/or rules used to define these semantics are specified in ontologies considered as first-order or rule-based background theories with a shared minimal vocabulary. Different ontologies of service providers and service requester have to be appropriately matched or aligned either at design time, or at runtime as part of the matching process.

Matching degrees

The degree of logic based matching of a given pair of semantic service profiles can be determined either (a) exclusively within the considered theory by means of logic reasoning, or (b) by a combination of logical inferences within the theory and algorithmic processing outside the theory. Prominent logic based matching degrees are exact, plugin, subsumes, and disjoint which are defined differently depending on the parts of service semantics and kind of logic theory used to compute them. For example, a software specification S plugs into (plug-in matches with) another specification R if the effect of S is more specific than that of R, and vice versa

[3] For reasons of readability, the implemented (stand-alone) SWS matchmakers shown in figure 1.1 each representing ako central discovery system per se are not again listed in figure 1.2, and vice versa, that is, those matchmaking approaches being inherent part of the functionality of each node of decentralized discovery systems (but not available as stand-alone matchmaker) are not listed in figure 1.1.

for the preconditions of S and R (Zaremski & Wing, 1996)[109]. That is called a post plug-in match, if it is restricted to their effects. Unfortunately, this original notion of plug-in matching has been adopted quite differently by most logic based SWS matchmakers for both monolithic and structured service descriptions.

Monolithic service matching

Matching of monolithic DL based service descriptions (cf. chapter 3) is performed exclusively within the considered theory by classical means of DL reasoning.That is, each service concept describing the effect of corresponding Web services in a description logic gets terminologically compared against a given query concept written in the same logic over a shared (matchmaker) ontology. This kind of logic based service effect matching is simple but agnostic to any structure imposed by other SWS formats like OWL-S or WSML.

For example, the post plug-in match of the effect of a service S with that of a service request R is defined as the DL entailment of concept subsumption of S by R over given knowledge base kb extended by the axioms of S and R ($kb \cup S \cup R \models S \sqsubseteq R$). That is, in every possible world or valid interpretation I of kb, the service provider's set S^I of (possible) service instances (represented by the monolithic description of the effect of S to the state space) is fully contained in the set R^I of instances acceptable to the requester ($S^I \subseteq R^I$). This assures the requester that each offered service instance is covered by her more generic request, hence S is definitely relevant, regardless of how unspecified issues in R are resolved.

In contrast, a logical service subsumes match ($kb \cup S \cup R \models R \sqsubseteq S$) assures the requester that her acceptable service instances are also acceptable to the provider, while for an intersection match the satisfiability of the conjunction of S and R (i.e., there exists an interpretation I of $kb \cup S \cup R$ such that $S^I \cap R^I \neq \emptyset$) identifies their compatibility with some underspecified constraints to agree upon. The latter is also called a potential match. An accessible account of logic based matching filters under possible world semantics over the universe of concrete services (service instances) is given in (Grimm, 2007)[34].

The complexity of matching monolithic DL based SWS descriptions is equal to the combined DL complexity. Post plugin matching of service concepts in SHIQO$^+$ (with transitive non-primitive roles) has been shown to be undecidable (Baader et al., 2005)[9] but decidable for OWL-DL, WSML-DL and DL-safe SWRL. Examples of monolithic service matchmakers are MAMAS (Dinoia et al., 2004)[26, 27] and RACER (Li & Horrocks, 2003) for service concepts in OWL and DAML+OIL. Notably, they determine the post plugin matching degree inverse to its original definition in (Zaremski & Wing, 1996)[109].

In MAMAS (Colucci et al., 2005; DiNoia et al., 2007)[22, 27], non-standard explanation services, that are abduction, a form of commonsense reasoning, and contraction, a typical belief revision operation, are devised as non-monotonic inferences for monolithic DL based service matching. In particular, concept contraction computes an explanation concept G of why a request concept R is not compatible with service concept S, that is, $S \sqcap R$ is not satisfiable (no intersection or partial match), that is $(S \sqcap R) \sqsubseteq \perp$. For this purpose, it keeps the least specific concept expression K of concept R such that K is still compatible with S, i.e. $\neg(K \sqcap S) \sqsubseteq \perp$. The remaining set G of constraints of R represents the desired explanation of mismatch.

If $S \sqcap R$ is satisfiable (potential match), concept abduction computes a concept expression K representing what is underspecified in service S (which constraints are *missing* in S) to completely satisfy a request R. That is, it determines a minimal explanation concept K for a failed concept subsumption $S \sqsubseteq (K \sqsubseteq) R$ ($S \sqcap K$ unsatisfied and $K \sqsubseteq R$). Both cases of approximated matching (partial, potential) are NP-hard for the simple description logic ALN. However, research in this direction has just begun and is, in part, related to research on non-monotonic reasoning with semantic Web (rule) languages.

Service specification matching

Service specification or profile PE matching determines the logic based semantic relation between service preconditions and effects. For example, the original notion of plug-in matching of two software components S, R requires that the logic based definition of the effect or postcondition of S logically implies that of R, while the precondition of S is more general than that of R (Zaremski & Wing, 1996)[109]. In other words, a logic based semantic plug-in match of service advertisement S with service request R requires (in every model of given knowledge base kb) the service effect to be more specific, and its precondition more general than requested. Depending on the SWS description framework (cf. chapter 3), the specification of preconditions and effects ranges from, for example, decidable def-Horn (DLP), WSML-DL, OWL-DL to undecidable SWRL, KIF and F-Logic(LP).

For example, the logic based service PE matchmaker PCEM (cf. chapter 10) exploits tuProlog for exact matching of service preconditions or effects (checking if there is a possibly empty variable substitution such that, when applied to one or both propositions, this results into two equal expressions), or domain specific inference rules (for computing subPartOf relations) represented in Prolog (cf. chapter 10).

Other examples are the IOPE matchmakers. The hybrid semantic WSML matchmaker WSMO-MX (Kaufer & Klusch, 2006)[46] is checking approximated query containment over finite service instance bases for WSML service constraints in undecidable F-Logic(LP) using OntoBroker. The IOPE matchmaker RFSD (Stollberg et al., 2007)[97] uses the VAMPIRE theorem prover for matching pairs of preconditions and effects in FOL, while the hybrid IOPE matchmaker LARKS (Sycara et al., 2002)[99] performs polynomial theta-subsumption checking of preconditions and postconditions in def-Horn for this purpose. There are no non-logic based or hybrid semantic service profile PE matchmaker available yet.

Service signature and IOPE matching

Logic based semantic service signature or profile IO matching is the stateless matching of declarative data semantics of service input and output parameters by a combination of logical inferences within the theory and algorithmic processing outside the theory. For example, the logic based plug-in matching of state based service specifications can be adopted to the plugin matching of stateless service signatures: Service S is expected to return more specific output data whose logically defined semantics is equivalent or subsumed by those of the desired output in request R, and requires more generic input data than requested in R.
More concrete, the signature of S plugs into the signature of request R iff $\forall$ $\textsc{in}_S$ $\exists$ $\textsc{in}_R$: $\textsc{in}_S \dot{\geq} \textsc{in}_R \wedge \forall$ $\textsc{out}_R$ $\exists$ $\textsc{out}_S$: $\textsc{out}_S \in \text{LSC}(\textsc{out}_R)$, with $LSC(C)$ the set of least specific concepts (direct children) C' of C, i.e. C' is a immediate subconcept of C in the shared (matchmaker) ontology. The quantified constraint that S may require less input than specified in R guarantees at a minimum that S is, in principle, executable with the input provided by the user in R iff the corresponding input concept definitions are equivalently mapped to WSDL input messages and corresponding service signature data types.
Examples of SWS matchmakers that perform logic based semantic signature matching only are the OWLSM (Jäger et al., 2005)[42] and the OWLS-UDDI (Paolucci et al., 2002)[75]. Though the latter determines signature plug-in matching inverse to the original definition and restricted to the output. (Keller et al., 2005) and (Stollberg et al., 2007) propose approaches to logic based semantic IOPE matching of Web services. In general, logic based matching of stateless service descriptions with I/O concepts and conjunctive constraints on their relationship specified in SHOIN has been proven decidable though intractable (Hull et al., 2006)[39]. This indicates the respective decidability of IOPE matching for OWL-S (with OWL-DL) and WSML (with WSML-DL).

1.1.3 Non-logic based semantic profile matching

As mentioned above, non-logic based SWS matchmaker do not perform any logical inferencing on service semantics. Instead, they compute the degree of semantic matching of given pairs of service descriptions based on, for example, syntactic similarity measurement, structured graph matching, or numeric concept distance computations over given ontologies. There is a wide range of means of text similarity metrics from information retrieval, approximated pattern discovery, and data clustering from data mining, or ranked keyword, and structured XML search with XQuery, XIRQL or TeXQuery [36, 5]. In this sense, non-logic based semantic service matching means exploit semantics that are implicit in, for example, patterns, subgraphs, or relative frequencies of terms used in the service descriptions, rather than declarative IOPE semantics explicitly specified in the considered logic.
One example is the OWLS-iMatcher (Bernstein & Kiefer, 2006)[13] which imprecisely queries a set of OWL-S service profiles that are stored as RDF graphs in a

RDF database with an extension of RDQL, called iRDQL, based on four (token and edit based) syntactic similarity metrics from information retrieval. The imprecise querying of RDF resources with similarity joins bases on TFIDF and the Levenshtein metric. The results are ranked according to the numerical scores of these syntactic similarity measurements, and a user-defined threshold.
The DSD matchmaker (Klein & König-Ries, 2004)[49, 55] performs, in essence, graph matching over pairs of state based service descriptions in the object oriented service description language DSD (with variables and declarative object sets) without any logic based semantics. The matching process determines what assignment of IOPE variables is necessary such that the state based service offer is included in the set (of service instances) defined by the request, and returns a numeric (fuzzy) degree of DSD service matching.

1.1.4 Hybrid semantic profile matching

Syntactic matching techniques are first class candidates for the development of hybrid semantic service profile matching solutions that combine means of both crisp logic based and non-logic based semantic matching where each alone would fail. Indeed, first experimental results of evaluating the performance of both non-logic based and hybrid semantic service matchmakers (OWLS-MX, OWLS-iMatcher) show that crisp logic based semantic service selection can be significantly outperformed by the former under certain conditions.
LARKS (Sycara et al., 2002)[99] has been the first hybrid semantic IOPE matchmaker while OWLS-MX (Klusch et al., 2006)[53] was the first hybrid semantic IO matchmaker for OWL-S services. OWLS-MX complements crisp logic based reasoning with approximate reasoning by use of selected token based IR similarity metrics for services.
WSMO-MX (Kaufer & Klusch, 2006)[46] is the first hybrid semantic matchmaker for services written in an LP extension of WSML-Rule, called WSML-MX. The hybrid service matching scheme of WSMO-MX is a combination of ideas of hybrid semantic matching of the OWLS-MX, the object-oriented graph matching of the DSD-MM, and the concept of intentional matching of services proposed in (Keller et al., 2005). WSMO-MX synthesizes means of both logic programming and approximate reasoning, and applies different filters to retrieve services that are relevant to a given query with respect to strictly ordered degrees of hybrid semantic matching. These degrees are recursively computed by aggregated valuations of (a) ontology based type matching, (b) logical constraint matching in F-logic, (c) relation matching, and (d) syntactic similarity measurement as well. Evaluation of WSMO-MX is ongoing work.
It is not yet known, however, what kind of approximative (hybrid) service matching will scale best to the size of the Web in practice, if at all. Research in this direction is in perfect line with the just recent call in (van Harmelen and Fensel, 2007)[30] for a general shift in semantic Web research towards scalable, approximative rather than strict logic based reasoning.

1.1.5 Logic based semantic process matching

Semantic matching of service process models, in general, is very uncommon, and not intended by the designers of current SWS description formats. Besides, the semantics of process models in OWL-S or WSML have not been formally defined yet, while neither SAWSDL nor monolithic service descriptions offer any process model. This problem can be partly solved by intuitively rewriting the process model descriptions in an appropriate logic with automated proof system and respective analysis tool support.

For example, in (Vaculin & Sycara, 2007)[103], OWL-S service process models are mapped into (intuitively) equivalent logical Promela statements that are then efficiently evaluated by the SPIN model checker. [4] This allows to verify the correctness of a given service process model in terms of consistency and liveness properties of an advertised service like the Delivery process always executes after the Buy process. The results of such service process model checking can be exploited for limited process oriented OWL-S service selection; this is a topic of ongoing research.

Alternatively, the matching of process models of OWL-S services that are grounded in WSDL can be reduced to the matching of corresponding orchestrations in BPEL. As mentioned in chapter 3, the OWL-S process model captures a common subset of workflow features that can be intuitively mapped to BPEL which offers an all-inclusive superset of such features (e.g. structured process activities in BPEL like Assignment, Fault Handler, Terminate are not available in OWL-S) [8]. Though BPEL has been given no formal semantics either yet, there are a few approaches to fill this gap based on Petri nets (Lohmann, 2007)[63] and abstract state machines (Fahland & Reisig, 2005)[29] that allow to formally verify liveness properties of BPEL orchestrations [66].

1.1.6 Non-logic based and hybrid semantic process model matching

Non-logic based business process matching can be applied to appropriately transformed pairs of SWS process models. For example, an approach to the matching of process dependency graphs based on syntactic similarity measurements is presented in (Bae et al., 2006)[10]. Bansal and Vidal (2003)[11] propose a hybrid matchmaker (IO-RPTM) that recursively compares the DAML-S process model dependency graphs based on given workflow operations and logical match between IO parameter concepts of connected (sub-)service nodes of the process graphs. On the other hand, means of functional service process matching can be exploited to search for a set of relevant subservices of a single composite service.

[4] A model checker verifies if a given system (service process) model satisfies a desirable property. If the property does not hold, it returns a counter-example of an execution where the property fails.

1.1.7 Semantic service discovery architectures

Existing SWS discovery architectures and systems in the literature can be broadly categorized as centralized and decentralized by the way they handle service information storage and location in the considered service network (Aktas et al., 2006; Grimm, 2007)[4, 34]. A classification of implemented SWS discovery systems is given in figure 1.2.
Centralized service discovery systems rely on one single, possibly replicated, global directory service (repository, registry) maintained by a distinguished so called super-peer or middle agent like matchmaker, broker or mediator agent (Klusch & Sycara, 2001)[52]. Contrary, decentralized service discovery systems rely on distributing service storage information over several peers in a structured, unstructured or hybrid P2P network.

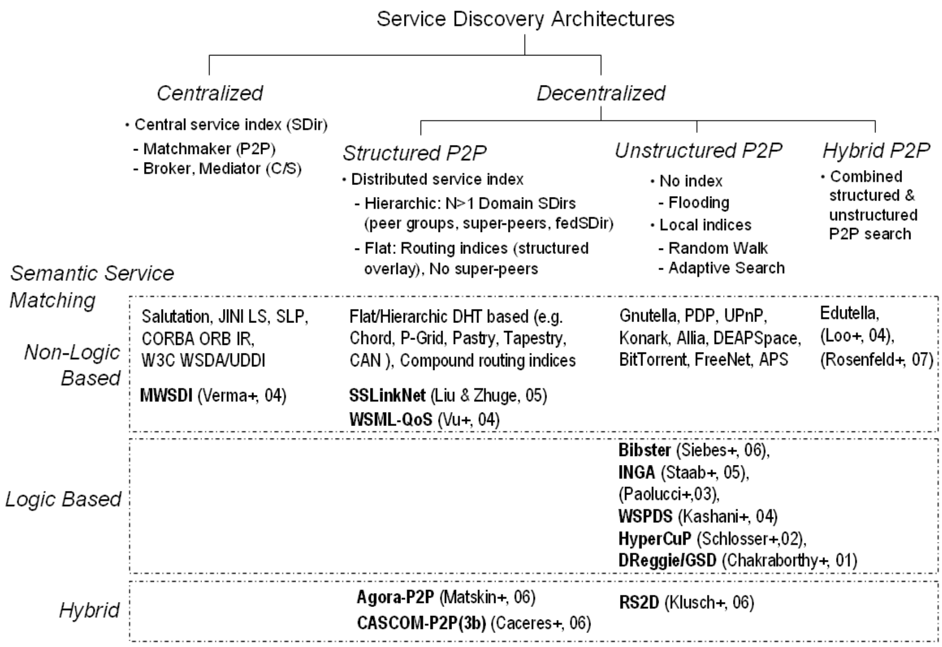

Figure 1.2: Categories of SWS discovery architectures and systems.

Semantic service discovery systems can be further classified with respect to the kind of semantic service matching means used by the intelligent agents in the network. For example, the exact keyword based service location mechanisms of all contemporary P2P systems like JINI, SLP, Gnutella flooding, and DHT (distributed hash table) can be complemented or replaced by sophisticated logic based semantic matching means to improve the quality of the search result.
As mentioned above, due to its generic functionality, any service matchmaker (cf.

figure 1.1) can be used in arbitrary discovery architectures and systems. In the extremes, a matchmaker can either serve as a central service directory (index) or look-up service, or can be integrated into each peer of an unstructured P2P service network to support an informed adaptive service search like in RS2D (Basters & Klusch, 2006)[12]. In fact, a few means of semantic service matching were originally developed for decentralized semantic P2P service retrieval in different applications.

Centralized semantic P2P service discovery

In centralized semantic P2P service systems, a dedicated central service directory or matchmaker returns a list of providers of semantically relevant services to the requester. Contrary to centralized client-server middleware or brokering, the requester then directly interacts with selected providers for service provision (Klusch & Sycara, 2001)[52]. The advantage of such centralized discovery architectures is a fast resource or service lookup time, though the central look-up server or registry like in JINI or the CORBA ORB interface registry is a single point of failure that can be only partially mitigated by replication and caching strategies.

An application of centralized P2P service discovery is the Napster music file sharing system, and the SETI@home system that is exploiting a vast set of distributed computional resources world wide to search for extraterrestrial signals. From the SWS discovery perspective, each of the above mentioned stand-alone SWS matchmakers, in principle, realizes a centralized logic based semantic service discovery system by itself. For example, the SCALLOPS e-health service coordination system uses the hybrid semantic matchmaker OWLS-MX as a central matchmaker for the selection of relevant e-health services in a medical emergency assistance application. The same matchmaker is distributed to each peer of an unstructured P2P network for decentralized OWL-S service discovery (Basters & Klusch, 2006)[12].

MWSDI (Verma et al., 2004)[105] is a centralized semantic P2P service system with non-logic based semantic service signature matching. Each peer in the system maintains one domain specific WSDL-S (SAWSDL) service registry and respective ontologies; multiple peers can form a domain oriented group. However, a distinguished central gateway or super-peer provides a global registries ontology (GRO) that maintains the complete taxonomy of all domain registries, the mappings between WSDL-S service I/O message types and concepts from shared domain ontologies in the system, associates registries to them, and serves as central look-up service for all peers. This central super-peer is replicated in form of so called auxiliary peers for reasons of scalability. For service location, any client peer (user) selects the relevant domain registries via the central GRO at the super-peer which then performs non-logic based semantic matching (structural XMLS graph matching, NGram based syntactic similarity, synonyms/hyponyms/hypernyms in the GRO) of service input and output concepts with those of the desired service. However, it would be hard to build the GRO, and difficult for the user to query the GRO without knowing its details in advance.

Decentralized semantic P2P service discovery

Decentralized semantic service discovery systems rely on service information storage and location mechanisms that are distributed over all peers in structured, unstructured or hybrid P2P networks.

Structured semantic P2P service systems Structured P2P systems have no central directory server but a significant amount of structure of the network topology (overlay) which is tightly controlled. Resources are placed neither at random peers nor in one central directory but at specified locations for efficient querying. In other words, the service index of the system is distributed to all peers according to a given structured overlay enforcing a deterministic content distribution which can be used for routing point queries.
Prominent examples of structured P2P systems are those with flat DHT based resource distribution and location mechanism like Chord rings (Stoica+, 2001), Pastry (Rowstron+, 2001), Tapestry (Zhao+, 2001), CAN (Ratnasamy+, 2001), P-Grid and P2PAlvis (Aberer et al., 2006), and structured hierarchic P2P systems. Flat DHT based systems allow to route queries with certain keys to particular peers containing the desired data. But to provide this functionality all new content in the network has to be published at the peer responsible for the respective key, if new data on a peer arrives, or a new peer joins the network.
In structured hierarchical or N-super-peer P2P systems (N>1), peers are organized in N domain oriented groups with possibly heterogeneous service location mechanisms (e.g hierarchic DHT, that is, one group with Chord ring overlay, another one with P-Grid overlay, etc.). Each group is represented by one super-peer hosting the group/domain service index. The set of super-peers, in turn, can be hierarchically structured with federated service directories in a super-peer top level overlay of the network. Peers within a group query its super-peer which interacts with other super-peers to route the query to relevant peer groups for response. The functionality of a super-peer of one peer group is not necessarily fixed, but, in case of node failure, transferable to a new peer of that group. Typically JXTA, a collection of P2P protocols, is used to realize super-peer based P2P systems, though it does not enforce such architectures.
Examples of decentralized SWS discovery in structured P2P networks are (Vu et al., 2006)[106],WSPDS (Kashani et al., 2004)[44], SSLinkNet (Liu & Zhuge, 2005)[60], CASCOM-P2P$_{3b}$ (Caceres et al., 2006)[17] and Agora-P2P (Küngas et al., 2006)[54, 61]. SSLinkNet, Agora-P2P, and (Vu et al., 2006) combine keyword based service discovery in the underlying Chord ring, respectively, P-Grid system with semantic service profile matching. The CASCOM and Agora-P2P systems are demonstrated for semantic OWL-S (DAML-S) service discovery, while SSLinkNet and WSPDS perform a P2P search for conventional Web services.
In the SSLinkNet (Liu & Zhuge, 2005)[60], a Chord ring based search is complemented by forwarding the same Web service request by the identified peers to relevant neighbors based on a given semantic service link network. The semantic

links between services are determined by non-logic based semantic service matching, and are used to derive semantic relationships between service provider peers based on heuristic rules.

Similarly, the AGORA-P2P system (Küngas et al., 2006)[54, 61] uses a Chord ring as the underlying infrastructure for a distributed storage of information about OWL-S services over peers. Service input and output concept names are syntactically hashed as literals to unique integer keys such that peers holding the same key are offering services with equal literals in a circular key space. A service request is characterized as a syntactic multi-key query against this Chord ring. Both systems, SSLinkNet and AGORA-P2P, do not cope with the known problem of efficiently preserving the stability of Chord rings in dynamic environments.

The generic CASCOM semantic service coordination architecture has been instantiated in terms of a hierarchic structured P2P network with N interacting super-peers each hosting a domain service registry that make up a federated Web service directory. Each peer within a group can complement a keyword based preselection of OWL-S services in their super-peer domain registries with a more complex semantic matching by a selected hybrid or logic based semantic OWL-S matchmaker (ROWL-S, PCEM or OWLS-MX) on demand. Both, the simple service discovery agent and SWS matchmaking module are integrated into each peer (cf. chapter 10).

Service discovery in structured P2P networks can provide search guarantees, in the sense of total service recall in the network, while simultaneously minimizing messaging overhead. However, this challenge has not been fully explored for unstructured P2P networks yet.

Unstructured semantic P2P service systems In unstructured P2P systems, peers initially have no index nor any precise control over the network topology (overlay) or file placement based on any knowledge of the topology. That is, they do not rely on any structured network overlay for query routing as they have no inherent restrictions on the type of service discovery they can perform.

For example, resources in unstructured P2P systems like Gnutella or Morpheus are located by means of network flooding: Each peer broadcasts a given query in BFS manner to all neighbour peers within a certain radius (TTL) until a service is found, or the given query TTL is zero. Such network flooding is extremely resilient to network dynamics (peers entering and leaving the system), but generates high network traffic.

This problem can be mitigated by a Random Walk search where each peer builds a local index about available services of its direct neighbour peers over time and randomly forwards a query to one of them in DFS manner until the service is found[5] as well as replication and caching strategies based on, for example, access frequencies and popularity of services (Lu et al., 2002)[65]. Approaches to informed proba-

[5] This is valid in case the length of the random walk is equal to the number of peers flooded with bounded TTL or hops).

bilistic adaptive P2P search like in APS (Tsoumakos & Roussopoulos, 2005)[102] improve on such random walks based on estimations over dynamically observed service location information stored in the local indices of peers. In contrast to the structured P2P search, this only provides probabilistic search guarantees, that is incomplete recall.

In any case, the majority of unstructured P2P service systems only performs keyword based service matching and does not exploit any qualitative results from logic based or hybrid semantic service matching to improve the quality of an informed search. In fact, only a few system are available for logic based or hybrid semantic Web service retrieval such as DReggie/GSD (Chakraborthy et al., 2002; Chen et al., 2001)[20, 21], HyperCuP (Schlosser et al., 2003)[89], Sem-WSPDS (Kashabi et al., 2004)[44], (Paolucci et al., 2003)[76], Bibster (Haase et al., 2006)[37], INGA (Löser et al., 2005)[62], and RS2D (Basters & Klusch, 2006)[12]. These systems differ in the way of how peers perform flooding or adaptive query routing based on evolving local knowledge about the semantic overlay, that is knowledge about the semantic relationships between distributed services and ontologies in unstructured P2P networks. Besides, all existing system implementations, except INGA and Bibster, perform semantic service IO profile matching for OWL-S (DAML-S), while HyperCuP peers dynamically build a semantic overlay based on monolithic service concepts.

For example, Paolucci et al. (2003)[76] propose the discovery of relevant DAML-S services in unstructured P2P networks based on both the Gnutella P2P discovery process and a complementary logic based service matching process (OWLS-UDDI matchmaker) over the returned answer set. However, the broadcast or flooding based search in unstructured P2P networks like Gnutella is known to suffer from traffic and load balancing problems.

Though Bibster and INGA have not been explicitly designed for SWS discovery, they could be used for this purpose. In INGA (Löser et al., 2005)[62], peers dynamically adapt the network topology, driven by the dynamically observed history of successful or semantically similar queries, and a dynamic shortcut selection strategy, which forwards queries to a community of peers that are likely to best answer given queries. The observed results are used by each peer for maintaining a bounded local (recommender) index storing semantically labelled topic specific routing shortcuts (that connect peers sharing similar interests).

Similarly, in Bibster (Haase et al., 2006)[37] peers have prior knowledge about a fixed semantic overlay network that is initially built by means of a special first round advertisement and local caching policy. Each peer only stores those advertisements that are semantically close to at least one of their own services, and then selects for given queries only those two neighbours with top ranked expertise according to the semantic overlay it knows in prior. Further, prior knowledge about other peers ontologies as well as their mapping to local ontologies is assumed. This is similar to the ontology based service query routing in HyperCuP (Schlosser et al., 2003)[89].

In RS2D (Basters & Klusch, 2006)[12], contrary to Bibster and DReggie/GSD, the peers perform an adaptive probabilistic risk-driven search for relevant OWL-S services without any fixed or prior knowledge about the semantic overlay. Each peer uses an integrated OWLS-MX matchmaker for hybrid semantic IO matching of local services with given query, and dynamically learns the average query-answer behaviour of its direct neighbours in the network. The decision to whom to forward a given semantic service request is then driven by the estimated mixed individual Bayes' conditional risk of routing failure in terms of both semantic loss and high communication costs. Peers are dynamically maintaining their local service (matchmaker) ontology based on observations of the results which, in particular, renders RS2D independent from the use of any fixed global ontology for semantic annotation like in DReggie/GSD.

Semantic hybrid P2P service systems Hybrid P2P search infrastructures combine both structured and unstructured location mechanisms. For example, Edutella combines a super-peer network with routing indices and an efficient broadcast. In (Loo et al., 2004)[64] a flat DHT approach is used to locate rare items, and flooding techniques are used for searching highly replicated items. A similar approach of hybrid P2P query routing that adaptively switches between different kinds of structured and unstructured search together with preliminary experimental results are reported in (Rosenfeld et al., 2007)[88]. However, there are no hybrid P2P systems for semantic service discovery available yet.

Despite recent advances in the converging technologies of semantic Web and P2P computing (Staab & Stuckenschmidt, 2006)[96], the scalability of semantic service discovery in structured, unstructured or hybrid P2P networks such as those for real-time mobile ad-hoc network applications is one major open problem. Research in this direction has just started. Preliminary solutions to this challenge vary in the expressivity of semantic service description, and the complexity of semantic matching means ranging from computationally heavy SWS matchmakers like OWLS-MX in SCALLOPS and CASCOM, to those with a streamlined DL reasoner such as Krhype (Kleemann & Sinner, 2007)[48] suitable for thin clients on mobile devices in IASON (Furbach et al., 2007)[32]. An example analysis of semantic service discovery architectures for realizing a mobile e-health application is given in [19].

1.2 Semantic Service Composition Planning

Semantic Web service composition is the act of taking several semantically annotated component services, and bundling them together to meet the needs of a given customer. Automating this process is desirable to improve speed and efficiency of customer response, and, in the semantic Web, supported by the formal grounding of service and data annotations in logics.

1.2.1 Web service composition

In general, Web service composition is similar to the composition of workflows such that existing techniques for workflow pattern generation, composition, and management can be partially reused for this purpose (Henocque & Kleiner, 2007)[38]. Typically, the user has to specify an abstract workflow of the required composite Web service including both the set of nodes (desired services) and the control and data flow between these nodes of the workflow network. The concrete services instantiating these nodes are bound at runtime according to the abstract node descriptions, also called "search recipes" (Casati & Shan, 2001)[18]. In particular, the mainstream approach to composition is to have a single entity responsible for manually scripting such workflows (orchestration and choreography) between WSDL services of different business partners in BPEL (Papazoglou, 2007; Alonso et al., 2003)[77, 1]. This is largely motivated by industry to work for service composition in legally contracted business partner coalitions - in which there is, unlike in open service environment, only very limited need for automated service composition planning, if at all. Besides, neither WSDL nor BPEL or any other workflow languages like UML2 or YAWL have formal semantics which would allow for an automated logic based composition.
In fact, the majority of existing composition planners for semantic Web services draws its inspiration from the vast literature on logic based AI planning (Peer, 2005)[78]. In the following, we focus on these approaches to SWS composition, and comment on the interleaving of service composition planning with discovery, and distributed plan execution. Please note that, the set of presented examples of SWS composition planners is representative but not exhaustive.

1.2.2 AI planning based Web service composition

The service composition problem roughly corresponds to the state based planning problem (I, A, G) in AI to devise a sound, complete, and executable plan which satisfies a given goal state G by executing a sequence of services as actions in A from a given initial world state I. Classical AI planning focuses on the description of services as deterministic state transition (actions) with preconditions, and state altering (physical) effects that are applicable to states based on the evaluation of preconditions and yield new states where the effects are valid. Further, classical planning is performed under the assumption of closed world with complete, fully observable initial states.
The goal and all logic based semantic service concepts (IO parameter values, preconditions and effects) defined in a formal ontology (domain or background theory) and outside are converted to one declarative (FOL) planning domain and problem description that serves a given logic based AI planner as input. In particular, service outputs are encoded as special non-state altering knowledge effects, and inputs as special preconditions. The standard language for this purpose is PDDL (Planning Domain Description Language) but alternative representation

formalisms are, for example, the situation calculus [68], linear logic [86], high-level logic programming languages based on this calculus like GOLOG [67], Petri nets, or HTN planning tasks and methods[93].
However, as pointed out in (Srivastava & Koehler, 2003)[95], the naive adoption of classical AI planning for service compositions has severe limits. In particular, they are insufficient for planning under uncertainty in open service environments where (a) the initial state is incomplete, and (b) actions may have several possible (conditional) outcomes and effects that are modeled in the domain but not deterministically known at planning time, or unknown outcomes at all that can be determined only at run-time. We survey implemented functional and process level composition planner for semantic Web services that rely on either classical planning or planning under uncertainty in the following.

1.2.3 Classification of SWS composition planners

In general, any AI planning framework for semantic Web service composition can be characterized by

- the representation of the planning domain and problem to allow for automated reasoning on actions and states,
- the planning method applied to solve the given composition problem in the domain, and
- the service semantics that are used for this purpose.

We can classify existing semantic Web service composition planners according to the latter two criteria, which yields the following classes.

- Dynamic or static SWS composition planners depending on whether the plan generation and execution are inherently interleaved in the sense that actions can be executed at planning time, or not.
- Functional level or process level SWS composition planners depending on whether the plan generation relies on service profile (data flow/IOPE) semantics only, or process model semantics in addition (data and control flown) [57].

Figure 1.3 shows the respective classification of existing SWS composition planners.

Static and dynamic composition

In summary, the majority of SWS composition planners such as GOAL (Pfalzgraf, 2006)[80], MetaComp (Botelho et al., 2007; cf. chapter 11), PLCP (Pistore et al., 2005), RPCLM-SCP (Lecue & Leger, 2006)[57] and AGORA-SCP (Rao et al., 2006)[86] are static classical planners. Approaches to dynamic composition planning with different degrees of interleaving plan generation and execution are rare. Unlike the static case, restricted dynamic composition planners allow the execution of information gathering but no world state altering services, hence are capable of planning under uncertainty about action outcomes at planning time. Examples of such composition planners are SHOP2 (Sirin et al., 2002) [91, 93], GOLOG-SCP (McIlraith et al., 2002)[67] and OWLS-XPlan1 (Klusch et al., 2005)[50].

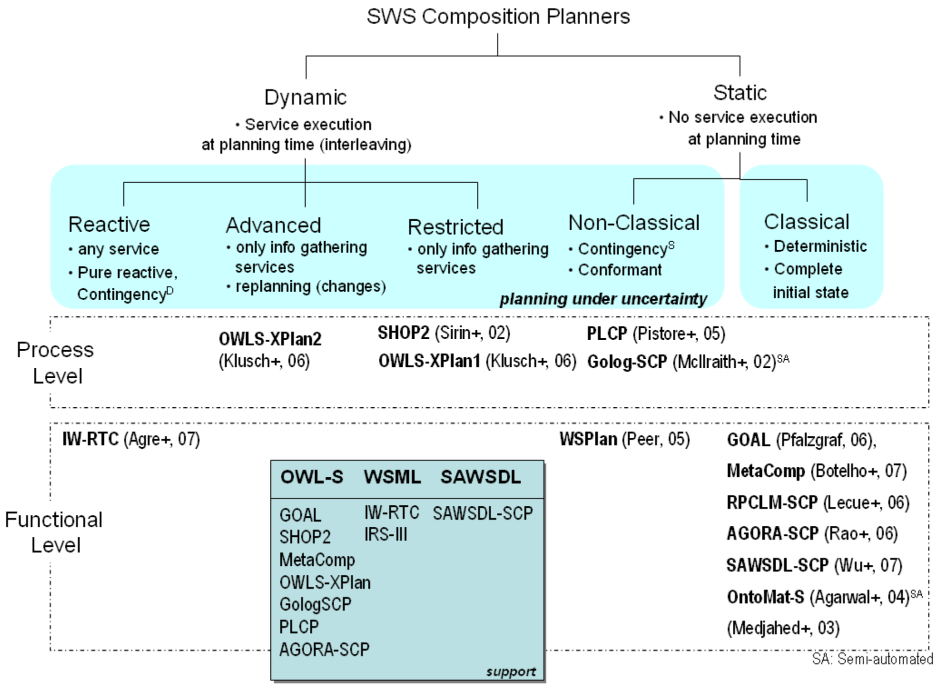

Figure 1.3: Classes of semantic Web service composition planners.

Advanced and reactive dynamic composition planners in stochastic domains even take non-deterministic world state changes into account during planning. While advanced dynamic planners like OWLS-XPlan2 (Klusch & Renner, 2006)[51] are capable of heuristic replanning subject to partially observed (but not caused) state changes that affect the current plan at planning time, their reactive counterparts like INFRAWEBS-RTC (Agre & Marinova, 2007)[3] fully interleave their plan generation and execution in the fashion of dynamic contingency and real-time planning.

Functional and process level composition

As shown in figure 1.3, most SWS composition planners perform functional level or service profile based composition (FLC) planning. FLC planning considers services as atomic or composite black-box actions which functionality can solely be described in terms of their inputs, outputs, preconditions, and effects, and which can be executed in a simple request-response without interaction patterns. Examples of FLC planners are GOAL (Pfalzgraf, 2006)[80], SAWSDL-SCP (Wu et al., 2007)[107] and OntoMat-S (Agarwal et al., 2004)[2].
Process level composition (PLC) planning extends FLC planning in the sense that it also the internal complex behavior of existing services into account. Prominent examples are SHOP2 (Sirin et al., 2004)[93], PLCP (Giunchiglia & Traverso, 1999; Pistore et al., 2001, 2005) [81, 83] and OWLS-XPlan (Klusch et al., 2005, 2006)[50, 51]. Both kinds of composition planning exploit semantic profile or process matching means that is either inherent to the AI planning mechanism, or provided by a connected stand-alone matchmaker.

Support of SWS description frameworks

Remarkably, most implemented semantic Web service composition planners support OWL-S like GOAL, OWLS-XPlan, SHOP2, GologSCP and MetaComp, while there is considerably less support of the standard SAWSDL and WSML available to date. In fact, the SAWSDL-SCP planner (Wu et al., 2006)[107] is the only one for SAWSDL, while the IW-RTC planner (Agre & Marinova, 2007)[3] is, apart from the semi-automated orchestration of WSML services in IRS-III, the only fully automated FLC planner for WSML yet.
Most composition planner feature an integrated conversion of semantic Web services, goals and ontologies into the internally used format of the planning domain and problem description, though a few others like the framework WSPlan (Peer, 2005)[79] for static PDDL based planning under uncertainty, and the recursive, progression based causal-link matrix composition planner RPCLM-SCP (Lecue & Leger, 2006)[57] do not.
In the following, we discuss each category and selected examples of SWS composition planners in more detail.

1.2.4 Functional level composition planners

Intuitively, FLC planning generates a sequence of semantic Web services based on their profiles that exact or plug-in matches with the desired (goal) service. In particular, existing services S_i, S_{i+1} are chained in this plan such that the output of S_i matches with the input of S_{i+1}, while the preconditions of S_{i+1} are satisfied in the world state after execution of S_i. Depending on the considered SWS description format (cf. chapter 3), different approaches to logic based, non-logic based or hybrid semantic service profile IOPE matching are available for this purpose (cf. figure 1.1).

In order to automatically search for a solution to the composition problem, FLC planners can exploit different AI planning techniques with inherent logic based semantic profile IOPE or PE matching like WSPlan (Peer, 2005), respectively, MetaComp (Botelho et al., 2006). The recursive forward-search planner GOAL (Pfalzgraf, 2006) as well as the SAWSDL-SCP (Wu et al., 2007) for apply non-logic based semantic profile IO matching of OWL-S, respectively, SAWSDL services.
In AGORA-SCP (Rao et al., 2006), theorem proving with hybrid semantic profile IO matching is performed for OWL-S service composition: Both services and a request (theorem) are described in linear logic, related to classical FOL, while the SNARK theorem prover is used to prove that the request can be deduced from the set of services. The service composition plan then is extracted from the constructive proof.
The FLC planner in (Medjahed, 2003)[69] uses proprietary composability rules for generating all possible plans of hybrid semantic profile IO matching services in a specific description format (CSSL). From these plans the requester has to select the one of best quality (QoS).

1.2.5 Process level SWS composition planners

Though FLC planning methods can address conditional outputs and effects of composite services with dynamic planning under uncertainty, considering services as black-boxes does not allow them to take the internal complex service behaviour into account at planning time. Such behavior is usually described as subservice interactions by means of control constructs including conditional and iterative steps. This is the domain of process level composition (PLC) planning that extends FLC planning in the aforementioned sense.
However, only few approaches to process level composition planning for semantic Web services exist to date. For example, orchestration of WSML services in IRS-III (Domingue et al., 2005)[28] synthesizes abstract state machines to compose individual services in a given process flow defined in OCML[6]. Though, the functionality of the WSMX orchestration unit has not been completely defined yet.
Other automated PLC planners of OWL-S services exploit different AI planning techniques such as

- HTN (Hierarchical Task Network) planning of OWL-S process models converted to HTN methods like in SHOP2 (Sirin et al., 2004)[93],
- Neo-classical GRAPHPLAN based planning mixed with HTN planning of OWL-S services converted to PDDL in OWLS-XPlan (Klusch et al., 2005, 2006)[50, 51],
- Value based synthesis of OWL-S process models in a given plan template of situation calculus based GOLOG programs (McIlraith et al., 2002)[67, 68],

[6] kmi.open.ac.uk/projects/ocml/

- Planning as model checking of OWL-S process models converted to equivalent state transition systems (STS) in the PLCP (Giunchiglia & Traverso, 1999; Pistore et al., 2001, 2005) [81, 83].

In the following, we discuss each class of static and dynamic SWS composition planners together with selected examples, if available.

1.2.6 Static semantic Web service composition planners

The class of static AI planning based composition covers approaches to both classical and non-classical planning under uncertainty.

Static classical planning

As mentioned above, classical AI planners perform (off-line) planning under the assumption of a closed, perfect world with deterministic actions and a complete initial state of a fully observable domain at design time. For example, Graphplan is a prominent classical AI planning algorithm that first performs a reachability analysis by constructing a plan graph, and then performs logic based goal regression within this graph to find a plan that satisfies the goal. Classical AI planners are static since their plan generation and execution is strictly decoupled.

Examples of static classical SWS composition planners

One example of a static classical SWS composition planner is GOAL (Pfalzgraf, 2006)[80] developed in the SmartWeb project. GOAL composes extended OWL-S services by means of a classical recursive forward-search (Ghallab et al., 2004)[33]. Both, the initial state and the goal state are derived from the semantic representation of the user's question (goal) obtained by a multimodal dialogue system in SmartWeb. At each stage of the planning process the set of services which input parameters are applicable to the current state is determined by signature (IO) matching through polynomial subgraph isomorphism checking (Messmer, 1995)[70]: The instance patterns of input parameters are matched against the graph representation of the state, and a service is applied to a plan state (simulated world state) by merging the instance patterns of its output parameters with the state. As a result, GOAL does not exploit any logical concept reasoning but structural service I/O graph matching to compose services. If plan generation fails, GOAL detects non-matching paths within instance patterns and consequently produces a clarification request (ako information gathering service) conveyed to the user by the dialogue system; on response by the user the planning process is restarted in total. Static service composition in the AGORA-SCP service composition system (Rao et al., 2006)[86] relies on linear logic (LL) theorem proving. The profiles of available DAML-S services are translated in to a set of LL axioms, and the service request is formulated as a LL theorem to be proven over this set. In case of success,

the composition plan can be extracted from the proof, transformed to a DAML-S process model and executed as a BPEL script. The AGORA planner is the only approach to decentralized composition planning in structured P2P networks (Küngas & Matskin, 2006)[54].
An example of a static classical SWS composition planner based on a special logic based PDDL planner is MetaComp (Botelho et al., 2006) which we describe in detail in chapter 11.

Static planning under uncertainty

In general, work on planning under uncertainty in AI can be classified according to (a) the representation of uncertainty, that is whether uncertainty is modeled strictly logically, using disjunctions, or is modeled numerically (e.g. with probabilities), and (b) observability assumptions, that is whether the uncertain outcomes of actions are not observable via sensing actions (conformant planning); partially or fully observable via sensing actions (conditional or contingency planning) [24]. As mentioned above, we can have uncertainty in the initial states and in the outcome of action execution. Since the observation associated to a given state is not unique, it is also possible to model noisy sensing and lack of information. Information on action outcomes or state changes that affect the plan can be gathered either at planning time (dynamic) or thereafter (static) for replanning purposes.

Static conditional or contingency planning. Static conditional or contingency planner like Cassandra (Pryor & Collins, 1996) and DTPOP (Peot 1998) devise a plan that accounts for each possible contingency that may arise in the planning domain. This corresponds to an optimal Markov policy in the POMDP framework for planning under uncertainty with probabilities, costs and rewards over a finite horizon (Puterman, 1994). The contingency planner anticipates unexpected or uncertain outcomes of actions and events by means of planned sensing actions, and attempts to establish the goals for each different outcome of these actions through conditional branching of the plan in advance [7]. The plan execution is driven by the outcome of the integrated sensing subplans for conditional plan branches, and decoupled from its generation which classifies these planners as static.

Static conformant planning. Conformant planner like the Conformant-FF (Hoffmann and Brafman, 2007), Buridan (Kushmerick et al., 1995), and UDTPOP (Peot 1998) perform contingency planning without sensing actions. The problem of conformat planning to search for the best unconditional sequence of actions under uncertainty of intial state and action outcome can be formalized as fully

[7]Examples of decision criteria according to which contingency branches are inserted in the (conventional) plan, and what the branch conditions should be at these points, are the maximum probability of failure, and the maximum expected future reward (utility) as a function of, for example, time and resource consumption. Uncertainty is often characterized by probability distributions over the possible values of planning domain predicates.

non-observable MDP, as a particular case of POMDP, with a search space pruned by ignoring state observations in contingency planning. For example, conformant Graphplan planning (CGP) [94] expresses the uncertainty in the initial state as a set of completely specified possible worlds, and generates a plan graph for each of these possible worlds in parallel. For actions with uncertain outcomes the number of possible worlds is multiplied by the number of possible outcomes of the action. It then performs a regression (backward) search on them for a plan that satisfies the goal in all possible worlds which ensures that the plan can be executed without any sensory actions. Conformant planner are static in the sense that no action is executed at planning time.

Examples of static SWS composition planners under uncertainty

The PLCP (Pistore et al., 2005)[82, 83] performs static PLC planning under uncertainty for OWL-S services. OWL-S service signatures and process models together with a given goal are converted to non-deterministic and partially observable state transition systems which are composed by a model checking based planner (MBP)[81] to a new STS which implements the desired composed service. This STS eventually gets transformed to an executable service composition plan (in BPEL) with possible conditional and iterative behaviors. No action is executed at planning time, and uncertainty is resolved by sensing actions during plan execution.
An example of static FLC planning under uncertainty is the WSPlan framework (Peer, 2005)[79] which provides the user with the option to plug in his own PDDL based planner and to statically interleave planning (under uncertainty) with plan execution. Static interleaving refers to the cycle of plan generation, plan execution, and replanning based on the result of the executed sensing subplans (in the fashion of static conditional planning) until a sequential plan without sensing actions is generated that satifies the goal. There are no static classical PLC planner for semantic Web services with deterministic (sequential) process models of composite services only available.

1.2.7 Dynamic SWS composition planners

The class of dynamic AI planning based composition covers approaches to restricted, advanced and reactive dynamic planning under uncertainty.

Restricted dynamic planning

Dynamic planning methods allow agents to inherently interleave plan generation and execution. In restricted dynamic planning, action execution at planning time is restricted to information gathering (book-keeping callbacks) about uncertain action outcomes. These special actions add new knowledge in form of ground facts to the partial observable initial state under the known IRP (Invocation and Reasonable Persistence) assumption[67] to ensure conflict avoidance[8]. Like in classical planning, however, world state altering services with physical effects (in opposite to knowledge effects of service outputs) are only simulated in local planning states and never get executed at planning time.

Examples of restricted dynamic SWS composition planners

Prominent examples of restricted dynamic composition planners are SHOP2, and OWLS-XPlan1 (Klusch et al., 2005)[50] for OWL-S services of which we describe the latter in detail in chapter 11. SHOP2 (Sirin et al., 2003, 2004)[91, 92] converts given OWL-S service process models into HTN methods and applies HTN planning interleaved with execution of information gathering actions to compose a sequence of services that satisfies the given task. Mapping any OWL-S process model to a situation calculus based GOLOG program, the authors prove that the plans produced are correct in the sense that they are equivalent to the action sequences found in situation calculus.

Advanced dynamic planning

Advanced dynamic planning methods allow in addition to react on arbitrary changes in the world state that may affect the current plan already during planning such as in OWLS-XPlan2. This is in contrast to static planning under uncertainty where sensing subplans of a plan are executed at run time only. However, in both restricted and advanced dynamic planning the interleaved execution of planning with world state altering services is prohibited to prevent obvious problems of planning inconsistencies and conflicts.

Examples of advanced dynamic SWS composition planners

To the best of our knowledge, OWLS-XPlan2 (Klusch & Renner, 2006)[51] still is the only one implemented example of an advanced dynamic composition planner. OWLS-XPlan2 will be described in chapter 11.

Reactive dynamic planning

Finally, reactive dynamic planning like in Brooks's subsumption architecture, RETE based production rule planners, and the symbolic model checking based planner SyPEM (Bertoli et al., 2004)[14] allows the execution of arbitrary actions at planning time. Pure reactive planner produce a set of condition-action (if-then)

[8] The IRP assumption states that (a) the information gathered by invoking the service once cannot be changed by external or subsequent actions, and (b) remains the same for repeating the same call during planning. That is, the incremental execution of callbacks would have the same effect when executing in prior to planning for extending the initial state which, in essence, closes the world for planning.

or reaction rules for every possible situation that may be encountered, whether or not the circumstances that would lead to it can be envisaged or predicted. The inherently interleaved planning and execution is driven through the evaluation of state conditions at every single plan step to select the relevant if-then reaction rule and the immediate execution of the respective, possibly world state altering action; This cycle is repeated until the goal is hopefuly reached.
A variant of reactive dynamic planning is dynamic contingency planning like in XII (Golden, Etzioni & Weld, 1994) and SAGE (Knoblock, 1995). In this case, a plan that is specified up to the information-gathering steps gets executed to that stage, and, once the information has been gathered, the rest of the plan is constructed. Interleaving planning and execution this way has the advantage that it is not necessary to plan for contingencies that do not actually arise. In contrast to pure reactive planners, reasoning is only performed at branch points predicted to be possible or likely.
In any case, reactive dynamic planning comes at the possible cost of plan optimality, and even plan existence, that is suboptimality and dead-end action planning or failure (Olawsky & Gini, 1990). The related ramification problem[9] is usually addressed either by restrictive assumptions on the nature of service effects on previous planning states (Bertoli et al., 2004)[14] in safely explorable domains (Koenig and Simmons 1995; 1998; Koenig 2001), or by integrated belief revision (TMS) in the planners knowledge base at severe computational costs.

Examples of reactive dynamic SWS composition planners

One example of an implemented reactive dynamic composition planner is the real-time composition planner IW-RTC (Agre & Marinova, 2007) developed in the European research project INFRAWEBS. It successively composes pairs of keyword based IO matching services, executes them and proceeds with planning until the given goal is reached. Unfortunately, the authors do not provide any detailed description of the composition and matching process nor complexity analysis.

Problems of SWS composition planning under uncertainty

One problem with adopting planning under uncertainty for service composition is that the execution of information gathering (book keeping) or even world state altering services at design or planning time might not be charge free, if granted by providers at all. That is, the planning agent might produce significant costs for its users even without any return value in case of plan generation or execution failure. Another problem is the known insufficient scalability of conditional or conformant planning methods to planning domains at Web scale or business application environments with potentially hundreds of thousands of services and

[9]The problem of ensuring the consistency of the planners knowledge base and the reachability of the original goal in spite of (highly frequent) world state altering service execution during plan generation.

vast instance bases. Research on exploiting conditional or conformant planning methods for semantic Web service composition has just started.

1.2.8 FLC planning of monolithic DL based services

Research on AI based FLC planning with monolithic DL based descriptions of services has just started. Intuitively, the corresponding AI planning (plan existence) problem for the composition of such services is as follows. Given an acyclic TBox T describing the domain or background theory in a DL, ABoxes S and G which interpretations I (consistent wrt T) over infinite sets of individual (object) names are describing, respectively the initial and goal state, and a set A of operators describing deterministic, parameterized actions α which precondition and effects are specified in the same DL and transform given interpretations of concepts and roles in T ($I \rightarrow_{\alpha}^{T} I'$), is there a sequence of actions (consistent with T) [10] obtained by instantiating operators with individuals which transforms S into G?
It has been shown in (Baader et al., 2005)[9] that the standard reasoning problems on actions, that are executability [11] and projection[12], are decidable for description logics between ALC and ALCOIQ. Furthermore, it has been shown in (Milicic, 2007)[71] only recently that the plan existence problem for such actions in ALCOIQ is co-NEXPTIME decidable for finite sets of individuals used to instantiate the actions, while it is known to be PSPACE-complete for propositional STRIPS-style actions. In addition, the extended plan existence problem with infinitely countable set of individuals was proven undecidable, as it is for Datalog STRIPS actions, for actions specified in ALC_U with universal role U for assertions over the whole domain by reduction to the halting problem of deterministic Turing machines. However, there is no implemented composition planner for monolithic DL based services available to date.

1.3 Interrelations

Though semantic service discovery and composition planning are active fields of research by themselves, they are mainly treated separatedly in the literature. In the following, we discuss the principled relationships between them.

1.3.1 Discovery and composition planning

What if the search for relevant existing services fails? In this case, service matchmaker agents may attempt to compose services together to satisfy the given service request. In fact, functional IOPE or process oriented semantic service matching

[10] An action is consistent with TBox T, if for every model I of T there exists I' s.t. $I \rightarrow_{\alpha}^{T} I'$.

[11] Action executability is equal to the satisfaction of action preconditions in given world states: $I \models pre_1$, $\forall i, 1 \leq i \leq n, I'.I \rightarrow_{\alpha_1 \ldots \alpha_i}^{T} I' : I' \models pre_{i+1}$.

[12] Satisfaction of assertion ϕ as a consequence or conjunctive effect of applying actions to a given state: For all models I of S and T,$I'.I \rightarrow_{\alpha_1 \ldots \alpha_n}^{T} I' : I' \models \phi$

is inherent part of the functionality of FLC or PLC planners that is either integrated into the planner itself or outsourced to respective matchmakers with which the planner interacts on demand; though most existing SWS matchmakers are used as stand alone tools for service discovery only (cf. figure 1.1).
From the view of composition planning, semantic service discovery is of importance for the following reasons.

- Discovery means provide the complete set of initially available services as a prerequisite of composition planning. This set of services together with related ontologies forms the basis of the initial and goal state to be specified in the planning domain description format used by the planner.
- Selection of semantically relevant (e.g. equivalent or plug-in matching) services that are also execution compatible after or even during (re-)planning on demand.

In other words, semantic service matching can be used by the composition planning tool to intially set and prune the search space of potential services by selecting relevant services in prior to, interleaved with, or after planning (re-planning). However, there is no agreed strategy for a-priori pruning the set of potential services accessible to the planner for composing. Heuristic pre-filtering of services can be performed, for example, against non-functional criteria such as observed quality of service, relevant application and business domains, and user and organisational roles like in ROWLS (Fernandez et al., 2006)[31]. To enable fast replanning in case of detected (temporarily) unavailability of planned services, or optimization of the plan quality, the composition planner can also exploit respectively precomputed lists of semantically equivalent or plug-in services for each service sorted delivered by a matchmaker.
Likewisely, from the view of semantic service discovery, the composition of complex services is of importance if none of the registered services satisfies the given request. In this case, the matchmaker agent can delegate its task to a composition planner for successfully generating a composite service for the given query.

Examples There are only a few implemented approaches that explicitly interleave semantic matching with composition planning.
In (Lecue et al., 2007)[58], logic based service matching is extended with concept abduction to provide explanations of mismatches between pairs of service profiles that are iteratively used as constructive feedback during composition planning and replanning when searching for alternative services to bridge identified semantic gaps between considered IOPE profiles of services in the current plan step. A similar abduction based matchmaking approach is presented in [26]. This scenario of explicitly interleaved discovery and composition has been implemented and tested in a non-public France Telecom research project.
In (Küsters et al., 2007)[55], the functional level composition of services specified in the DIANE service description language DSD is explicitly integrated with a DSD

matchmaker module that matches service requests asking for multiple connected effects of configurable services. By using a value propagation mechanism and a cut of possible (not actual) parameter value fillings for service descriptions that cover multiple effects the authors avoid exponential complexity for determining an optimal configuration of plug-in matching service advertisements used for a composition.
In (Binder et al., 2004)[15], the syntactic functional level service composition is based on partial matching of numerically encoded service IO data types in a service directory. Unfortunately, the justification of the proposed numeric codings for matching services appears questionable, though it was shown to efficiently work for certain applications.
The composition planner OWLS-XPlan2 (Klusch et al., 2006) [87] integrates IOPE matching and calls the component OWLS-MXP of the matchmaker OWLS-MX 1.1 to check the compatibility of input/output data types of sequenced services at each plan step. This ensures the principled executablity of the generated sequential plan at the service grounding level.
The UMBC interactive OWL-S service composer (Sirin et al., 2004)[92] uses the OWLS-UDDI matchmaker to help users filter and select relevant services while building the composition plan. At each plan step, the composer provides the user with advertised services that plug-in or exact match with the current service selection yielding an incremental backward IO chaining of services in the plan.
The Agora-P2P service composition system (Küngas & Matskin, 2006)[54] is the only approach to decentralized SWS composition planning. It uses a Chord ring to publish and locate OWL-S service descriptions keyword based while linear logic theorem proving and logic based semantic service IO matching is applied to compose (and therefore search for relevant subservices of) the desired service.

1.3.2 Composition planning and execution

The semantic compatibility of subsequent services in a plan does not guarantee their correct execution in concrete terms on the grounding level. A plan is called correct, if it produces a state that satifies the given goal (Lecue & Leger, 2006)[57]. The principled plan executability, also called execution composability of a plan requires its data flow to be ensured during plan execution on the service grounding level (Medjahed et al., 2003)[69]. This can be verified through complete (XMLS) message data type checking of semantically matching I/O parameters of every pair of subsequent services involved in the plan. For example, OWLS-XPlan2 calls a special matchmaker module that checks plan execution compatibility at each plan step during planning.
The consistent, central or decentral plan execution can be achieved by means of classical (distributed) transaction theory and systems. An advanced and implemented approach to distributed SWS composition plan execution is presented, for example, in chapter 12 of this book (Semantic Web Service Execution) and [73]. However, the availability of non-local services that are not owned by the planning

agent can be, in principle, refused by autonomous service providers without any prior commitment at any time. This calls for effective replanning based on alternative semantic matching services delivered by the matchmaker to the composition planner prior to, or during planning such as in OWLS-XPlan2.

1.3.3 Negotiation

Services may not be for free but pay per use. In particular, requester agents might be charged for every single invocation of services at discovery or planning time. Besides, the service pricing is often private which makes it hard, if not infeasible, for any search or composition agent to determine the total expenses of coordinated service value provision to its user.
Standard solution is to negotiate service level agreements and contracting of relevant services based on non-functional service parameters such as QoS, pricing, and reputation between service requester and provider agents involved (Yan et al., 2007) [108]. Usually, such negotiation takes place after service discovery depending on service configurations and user preferences, followed by contracting (Preist, 2007)[84]. Most existing semantic Web service frameworks offer slots for non-functional provenance information as part of their service description.
However, the problem of how to dynamically interleave composition (re-)planning and negotiation remains open. Related work draw upon means of parallel auctioning (Preist et al., 2003)[85], and coalition forming (Müller et al., 2006)[74] of planning agents in different competitive settings.

1.4 Open problems

The research field of semantic Web service coordination is in its infancies. Hence, it comes at no surprise that there are many open problems of both semantic service discovery and composition planning that call for intensive further investigation in the domain. Some major open problems of semantic service discovery are the following.

- *Approximated matching.* How to deal with uncertain, vague or incomplete information about the functionality of available services and user preferences for service discovery? Fuzzy, probability, and possibility theory are first class candidates for the design of approximated (hybrid) semantic service matching algorithms to solve this problem. In particular, efficient reasoners for respective extensions of semantic Web (rule) languages like probabilistic pOWL, fuzzyOWL, or pDatalog can be applied to reason upon semantic service annotations under uncertainty and with preferences.

  However, there are no such semantic service matchmakers available yet. Apart from the first hybrid matchmakers for OWL-S and WSML services, OWLS-MX and WSMO-MX, the same holds for the integrated use of means of statistical analysis from data mining or information retrieval for approximative matching of semantic service descriptions.

- *Scalability.* How to reasonably trade off the leveraging of expensive logic based service discovery means with practical requirements of resource bounded, just-in-time and light-weight service discovery in mobile ad-hoc or unstructured P2P service networks? What kind of approximated and/or adaptive semantic service discovery techniques scale best for what environment (network, user contxt, services distribution, etc) and application at hand? The required very large scale, comparative performance experiments under practical real-world conditions have not been conducted yet.

- *Adaptive discovery.* How to leverage semantic service discovery by means of machine learning and human-agent interaction? Though a variety of adaptive personal recommender and user interface agents have been developed in the field, none of the currently implemented semantic Web service matchmakers is capable of flexibly adapting to its changing user, network, and application environment.

- *Privacy.* How to protect the privacy of individual user profile data that are explicit or implicit in service requests submitted to a central matchmaker, or relevant service providers? Approaches to privacy preserving semantic Web service discovery are still very rare, and research in this direction appears somewhat stagnant. Amongst the most powerful solutions proposed are the Rei language for annotating OWL-S services with privacy and authorization policies [25, 43], and the information flow analysis based checking of the privacy preservation of sequential OWL-S service plans [40, 41]. However, nothing is known about the scalability of these solutions in practice yet.

- *Lack of tool support and test collections.* Current easy to use tool support of semantic Web service discovery is still lagging behind the theoretical advancements, though there are differences to what extent this is valid for what service description framework (cf. figure 1.1). In particular, there is no official test collection for evaluating the retrieval performance of service discovery approaches (matchmakers, search engines) for the standard SAWSDL and WSML, while there are two publicly available for OWL-S (OWLS-TC2, SWS-TC). There are no solutions for the integrated matching of different services that are specified in different languages like SAWSDL, OWL-S and WSML. Relevant work on refactoring OWL-S and WSML to the standard SAWSDL is ongoing.

Some major challenges of research and development in the domain of semantic Web service composition planning are as follows.

- Scalable and resource efficient approaches to service composition planning under uncertainty and their use in real-world applications of the Web 3.0 and in intelligent pervasive service applications of the so called "Internet of

Things" that is envisioned to interlink all kinds of computing devices without limit on the global scale.

- Efficient means of distributed composition planning of semantic Web services in peer-to-peer and grid computing environments.
- Easy to use tools for the common user to support discovery, negotiation, composition and execution semantic Web services in one framework for different semantic Web service formats like the standard SAWSDL, and non-standards like OWL-S, WSML, and SWSL.
- Interleaving of service composition planning with negotiation in competitive settings.

1.5 Conclusions

This chapter provided a brief romp through the fields of SWS discovery and composition planning. We classified existing approaches, discussed representative examples and commented on the interrelationships between both service coordination activities. Despite fast paced research and development in the past years world wide, SWS technology still is commonly considered immature with many open theoretical and practical problems as mentioned above. However, its current convergence with Web 2.0 towards a service Web 3.0 in an envisioned Internet of Things helds promise to effectively revolutionize computing applications for our everday life.

Bibliography

[1] G. Alonso, F. Casati, H. Kuno, V. Machiraju: Web Services. Springer, 2003

[2] S. Agarwal, S. Handschuh, S. Staab: Annotation, composition and invocation of semantic web services. *Web Semantics*, 2, 2004.

[3] G. Agre, Z. Marinova: An INFRAWEBS Approach to Dynamic Composition of Semantic Web Services. *Cybernetics and Information Technologies (CIT)*, 7(1), 2007.

[4] M.S. Aktas, G. Fox, M. Pierce: Managing Dynamic Metadata as Context. Proceedings of Intl. Conference on Computational Science and Engineering (ICCSE), Istanbul, 2005

[5] S. Amer-Yahia, C. Botev, J. Shanmugasundaram: TeXQuery: A Full-Text Search Extension to XQuery. Proceedings of the World-Wide-Web Conference WWW 2004, 2004.

[6] A. Ankolekar, M. Paolucci, K. Sycara: Spinning the OWL-S Process Model - Toward the Verification of the OWL-S Process Models. Proceedingsof International Semantic Web Services Workshop (SWSW), 2004

[7] I.B. Arpinar, B. Aleman-Meza, R. Zhang, A. Maduko: Ontology-driven web services composition platform. Proc.of IEEE International Conference on E-Commerce Technology CEC, San Diego, USA, IEEE Press, 2004.

[8] M.A. Aslam, S. Auer, J. Shen: From BPEL4WS Process Model to Full OWL-S Ontology. Proceedings of 2nd European COnference on Semantic Web Services ESWC, Buda, Montenegro, 2006.

[9] F. Baader, C. Lutz, M. Milicic, U. Sattler, F. Wolter: Integrating Description Logics and action formalisms: First results. Proc. 20th National Conference on Artificial Intelligence (AAAI), Pittsburgh, USA, AAAI Press, 2005

[10] J. Bae, L. Liu, J. Caverlee, W.B. Rouse: Process Mining, Discovery, and Integration using Distance Measures. Proceedings of International COnference on Web Services ICWS, 2006.

[11] S. Bansal, J. Vidal: Matchmaking of Web Services Based on the DAMLS Service Model. Proc. International Joint Conference on Autonomous Agents and Multiagent Systems AAMAS, 2003.

[12] U. Basters and M. Klusch: RS2D: Fast Adaptive Search for Semantic Web Services in Unstructured P2P Networks. Proceedings 5th Intl. Semantic Web Conference (ISWC), Athens, USA, Lecture Notes in Computer Science (LNCS), 4273:87-100, Springer, 2006.

[13] A. Bernstein, C. Kiefer: Imprecise RDQL: Towards Generic Retrieval in Ontologies Using Similarity Joins. Proceedings ACM Symposium on Applied Computing, Dijon, France, ACM Press, 2006.

[14] P. Bertoli, A. Cimatti, P. Traverso: Interleaving Execution and Planning for Nondeterministic, Partially Observable Domains. Proceedings of European Conference on Artificial Intelligence (ECAI), 2004.

[15] W. Binder, I. Constantinescu, B. Faltings, K. Haller, C. Tuerker: A Multi-Agent System for the Reliable Execution of Automatically Composed Ad-hoc Processes. Proceedings of the 2nd European Workshop on Multi-Agent Systems (EUMAS), Barcelona, Spain, 2004.

[16] L. Botelho, A. Fernandez, B. Fries, M. Klusch, L. Pereira, T. Santos, P. Pais, M. Vasirani: Service Discovery. In M. Schumacher, H. Helin (Eds.): CASCOM - Intelligent Service Coordination in the Semantic Web. Chapter 10. Birkh"auser Verlag, Springer, 2008.

[17] C. Caceres, A. Fernandez, H. Helin, O. Keller, M. Klusch: Context-aware Service Coordination for Mobile Users. Proceedings IST eHealth Conference, 2006.

[18] F. Casati, M.C. Shan: Dynamic and Adaptive Composition of E-services. *Information Systems*, 6(3), 2001.

[19] CASCOM Project Deliverable D3.2: Conceptual Architecture Design. September 2005.www.ist-cascom.org

[20] D. Chakraborty, F. Perich, S. Avancha, A. Joshi: DReggie: Semantic Service Discovery for M-Commerce Applications. Proceedings of the International Workshop on Reliable and Secure Applications in Mobile Environment, 2001.

[21] H. Chen, A. Joshi, and T. Finin: Dynamic service discovery for mobile computing: Intelligent agents meet JINI in the aether. 4(4):343-354, 2001.

[22] S. Colucci, T.C. Di Noia, E. Di Sciascio, F.M. Donini, M. Mongiello: Concept Abduction and Contraction for Semantic-based Discovery of Matches and Negotiation Spaces in an E-Marketplace. *Electronic Commerce Research and Applications*, 4(4):345361, 2005.

[23] I. Constantinescu, B. Faltings: Efficient matchmaking and directory services Proceedings of IEEE Conference on Web Intelligence WI, 2003.

[24] R. Dearden, N. Meuleauy, S. Ramakrishnany, D.E. Smith, R. Washington: Incremental Contingency Planning. Proc. of ICAPS-03 Workshop on Planning under Uncertainty, Trento, Italy, 2003.

[25] G. Denker, L. Kagal, T. Finin, M. Paolucci, K. Sycara: Security For DAML Web Services: Annotation and Matchmaking. Proceedings of the Second International Semantic Web Conference (ISWC 2003), USA, 2003.

[26] T. Di Noia, E.D. Sciascio, F.M. Donini, M. Mogiello: A System for Principled Matchmaking in an Electronic Marketplace. *Electronic Commerce*, 2004.

[27] T. Di Noia, E. Di Sciascio, F.M. Donini: Semantic Matchmaking as Non-Monotonic Reasoning: A Description Logic Approach. *Artificial Intelligence Research (JAIR)*, 29:269–307, 2007.

[28] J. Domingue, S. Galizia, L. Cabral: Choreography in IRS-III: Coping with Heterogeneous Interaction Patterns in Web Services. Proc. International Semantic Web Conference, LNAI, Springer, 2005.

[29] D. Fahland, W. Reisig: ASM-based semantics for BPEL: The negative Control Flow. Proceedings of the 12th International Workshop on Abstract State Machines (ASM'05), 2005.

[30] D. Fensel, F. van Harmelen: Unifying reasoning and search to Web scale. *IEEE Internet Computing*, March/April 2007.

[31] A. Fernandez, M. Vasirani, C. Caceres, S. Ossowski: A role-based support mechanism for service description and discovery. In: huang et al. (eds.), Service-Oriented Computing: Agents, Semantics, and Engineering. LNCS 4504, Springer, 2006.

[32] U. Furbach, M. Maron, K. Read: Location based informationsystems. Künstliche Intelligenz, 3/07, BöttcherIT, 2007.

[33] M. Ghallab, D. Nau, P. Traverso: Automated planning. Elsevier, 2004.

[34] S. Grimm: Discovery - Identifying relevant services. In [98], 2007.

[35] S. Grimm, B. Motik, C. Preist: Matching semantic service descriptions with local closed-world reasoning. Proc. European Semantic Web Conference (ESWC), Springer, LNCS, 2006.

[36] L. Guo, F. Shao, C. Botev, J. Shanmugasundaram: XRANK: Ranked Keyword Search over XML Documents. Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data, San Diego, USA, 2003.

[37] P. Haase, R. Siebes, F. van Harmelen: Expertise-based Peer selection in Peer-to-Peer Networks. *Knowledge and Information Systems*, Springer, 2006

[38] L. Henoque, M. Kleiner: Composition - Combining Web Service Functionality in Composite Orchestrations. Chapter 9 in [98], 2007.

[39] D. Hull, U. Sattler, E. Zolin, R. Stevens, A. Bovykin, I. Horrocks: Deciding semantic matching of stateless services. Proc. 21st National Conference on Artificial Intelligence (AAAI), AAAI Press, 2006

[40] D. Hutter, M. Klusch, M. Volkamer: Information Flow Analysis Based Security Checking of Health Service Composition Plans. Proceedings of the 1st European Conference on eHealth, Fribourg, Switzerland, 2006.

[41] D. Hutter, M. Volkamer, M. Klusch, A. Gerber: Provably Secure Execution of Composed Semantic Web Services. Proccedings of the 1st International Workshop on Privacy and Security in Agent-based Collaborative Environments (PSACE 2006), Hakodate, Japan, 2006.

[42] M.C. Jäger, G. Rojec-Goldmann, C. Liebetruth, G. Mühl, K. Geihs: Ranked Matching for Service Descriptions Using OWL-S. Proceedings of 14. GI/VDE Fachtagung Kommunikation in Verteilten Systemen KiVS, Kaiserslautern, 2005

[43] L. Kagal, T. Finin, M. Paolucci, N. Srinivasan, K. Sycara, G. Denker: Authorization and Privacy for Semantic Web Services. *IEEE Intelligent Systems*, July/August, 2004.

[44] F. B. Kashani, C.C. Shen, C. Shahabi: SWPDS: Web service peer-to-per discovery service. Proceedings of Intl. Conference on Internet Computing, 2004.

[45] U. Keller, R. Lara, H. Lausen, A. Polleres, D. Fensel: Automatic Location of Services. Proceedings of the 2nd European Semantic Web Conference (ESWC), Heraklion, Crete, LNCS 3532, Springer, 2005.

[46] F. Kaufer and M. Klusch: Hybrid Semantic Web Service Matching with WSMO-MX. Proc. 4th IEEE European Conference on Web Services (ECOWS), Zurich, Switzerland, IEEE CS Press, 2006

[47] F. Kaufer and M. Klusch: Performance of Hybrid WSML Service Matching with WSMO-MX: Preliminary Results. Proc. First Intl. Joint ISWC Workshop SMR2 2007 on Service Matchmaking and Resource Retrieval in the Semantic Web, Busan, Korea, 2007.

[48] T. Kleemann, A. Sinner: Description logic based matchmaking on mobile devices. Proceedgins of 1st Workshop on Knowledge Engineering and Software Engineering (KESE 2005), 2005.

[49] M. Klein, B. König-Ries: Coupled Signature and Specification Matching for Automatic Service Binding. European Conference on Web Services (ECOWS 2004), Erfurt, 2004.

[50] M. Klusch, A. Gerber, M. Schmidt: Semantic Web Service Composition Planning with OWLS-XPlan. Proc. 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web, Arlington VA, USA, AAAI Press, 2005.

[51] M. Klusch, K-U. Renner: Dynamic Re-Planning of Composite OWL-S Services. Proc. 1st IEEE Workshop on Semantic Web Service Composition, Hongkong, China, IEEE CS Press, 2006.

[52] M. Klusch, K. Sycara: Brokering and Matchmaking for Coordination of Agent Societies: A Survey. In: Coordination of Internet Agents, A. Omicini et al. (eds.), Springer

[53] M. Klusch, B. Fries, K. Sycara: Automated Semantic Web Service Discovery with OWLS-MX. Proc. 5th Intl. Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), Hakodate, Japan, ACM Press, 2006

[54] P. Küngas, M. Matskin: Semantic Web Service Composition through a P2P-Based Multi-Agent Environment. Proc. of the Fourth International Workshop on Agents and Peer-to-Peer Computing (in conjunction with AAMAS 2005), Utrecht, Netherlands, LNCS 4118, 2006.

[55] U. Küster, B. König-Ries, M. Stern, M. Klein: DIANE: An Integrated Approach to Automated Service Discovery, Matchmaking and Composition. Proceedings of the World Wide Web COnference WWW, Banff, Canada, ACM Press, 2007.

[56] s. Lamparter, A. Ankolekar: Automated Selection of Configurable Web Services. 8. Internationale Tagung Wirtschaftsinformatik. Universittsverlag Karlsruhe, Karlsruhe, Germany, March 2007.

[57] F. Lecue, A. Leger: Semantic Web service composition through a matchmaking of domain. Proc. of 4th IEEE European Conference on Web Services (ECWS), Zurich, 2006.

[58] F. Lecue, A. Delteil, A. Leger: Applying Abduction in Semantic Web Service Composition. Proceedings of IEEE International Conference on Web Services (ICWS 2007), 2007.

[59] L. Li, I. Horrocks: A software framework for matchmaking based on semantic Web technology. Proceedings of the world wide Web conference (WWW), Budapest, 2003.

[60] J. Liu, H. Zhuge: A Semantic-Link-Based Infrastructure for Web Service. Proc. of the International World Wide Web Conference, 2005.

[61] S. Liu, P. Küngas, M. Matskin: Agent-Based Web Service Composition with JADE and JXTA. Proc. of Intl Conference on Semantic Web and Web Services (SWWS), Las Vegas, USA, 2006.

[62] A. Löser, C. Tempich, B. Quilitz, W.-T. Balke, S. Staab, W. Nejdl: Searching Dynamic Communities with Personal Indexes. Proceedings of Internatioanl Semantic Web Conference, 2005.

[63] N. Lohmann: A Feature-Complete Petri Net Semantics for WS-BPEL 2.0. Proceedings of the Workshop on Formal Approaches to Business Processes and Web Services (FABPWS'07), 2007.

[64] B.T. Loo, R. Huebsch, I. Stoica, J.M. Hellerstein: The Case for a Hybrid P2P Search Infrastructure. Proceedings of rd Intl Workshop on P2P Systems (IPTPS), USA, Springer, LNCS, 2004.

[65] Q. Lu, P. Cao, E. Cohen, K. Li, S. Shenker: Search and Replication in Unstructured Peer-to-Peer Networks. Procceedings of ACM 6th ACM International Conference on Supercomputing ICS, New York, USA, 2002.

[66] A. Martens: Analyzing Web Service based Business Processes. Proceedings of Workshop on Fundamental Approaches to Software Engineering FASE, 2005.

[67] S. McIllraith, T.C. Son: Adapting Golog for composition of semantic Web services. Proc. International Conference on Knowledge Representation and Reasoning KRR, Toulouse, France, 2002.

[68] S. Narayanan, S. McIllraith: Simulation, verification and automated composition of web services. Proc. of 11th International COnference on the World Wide Web (WWW), Hawaii, 2002.

[69] B. Medjahed, A. Bouguettyaya, A.K. Elmagarmid: Composing Web services on the semantic Web. *Very Large Data Bases (VLDB)*, 12(4), 2003.

[70] B.T. Messmer: New approaches on graph matching. PhD Thesis, University of Bern, Switzerland, 1995.

[71] M. Milicic: Planning in Action Formalisms based on DLS: First Results. Proceedings of the Intl Workshop on Description Logics, 2007.

[72] D.S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, Z. Xu: Peer-to-peer computing. Technical Report HPL-2002-57, Hewlett-Packard, 2002.

[73] T. Möller, H. Schuldt, A. Gerber, M. Klusch: Next Generation Applications in Healthcare Digital Libraries using Semantic Service Composition and Coordination. *Health Informatics*, 12 (2):107-119, SAGE publications, 2006.

[74] I. Müller, R. Kowalczyk, P. Braun: Towards Agent-Based Coalition Formation for Service Composition. Proceedings of the IEEE International Conference on Intelligent Agent Technology, Washington, USA, 2006.

[75] M. Paolucci, T. Kawamura, T.R. Payne, K. Sycara: Semantic Matching of Web Services Capabilities. Proceedings of the 1st International Semantic Web Conference (ISWC2002), 2002.

[76] M. Paolucci, K. Sycara, T. Nishimara, N. Srinivasan: Using DAML-S for P2P Discovery. Proc. of International Conference on Web Services, Erfurt, Germany, 2003.

[77] M. Papazoglou: Web Services: Principles and Technology. Pearson - Prentice Hall, September 2007.

[78] J. Peer: Web Service Composition as AI Planning: A Survey. Technical Report, University of St. Gallen, Switzerland, 2005. Available at elektra.mcm.unisg.ch/pbwsc/docs/pfwsc.pdf

[79] J. Peer: A POP-Based Replanning Agent for Automatic Web Service Composition. Proceedings of the 2nd European Semantic Web Conference (ESWC), Heraklion, Crete, LNCS 3532, Springer, 2005.

[80] A. Pfalzgraf: Ein robustes System zur automatischen Komposition semantischer Web Services in SmartWeb. Master Thesis, University of the Saarland, Saarbrücken, Germany, Juni 2006.

[81] M. Pistore, P. Traverso: Planning as model checking for extended goals in non-deterministic domains. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI-01), 2001.

[82] M. Pistore, P. Roberti, P. Traverso: Process-Level Composition of Executable Web Services: On-the-fly Versus Once-for-all Composition Proceedings of the 2nd European Semantic Web Conference (ESWC), Heraklion, Crete, LNCS 3532, Springer, 2005.

[83] M. Pistore, P. Traverso, P. Bertoli, A. Marconi: Automated synthesis of composite BPEL4WS web services. Proceedings of the 2005 IEEE International Conference on Web Services, Orlando, USA, IEEE Press, 2005.

[84] C. Preist: Semantic Web Services - Goals and Vision. Chapter 6 in [98], 2007.

[85] C. Preist, C. Bartolini, A. Byde: Agent-based service composition through simultaneous negotiation in forward and reverse auctions. Proceedings of the 4th ACM Conference on Electronic Commerce, San Diego, California, USA, 2003.

[86] J. Rao, P. Kuengas, M. Matskin: Composition of Semantic Web services using Linear Logic theorem proving. *Information Systems*, 31, 2006.

[87] K.-U. Renner, P. Kapahnke, B. Blankenburg, M. Klusch: OWLS-XPlan 2.0 - A Dynamic OWL-S Service Composition Planner. BMB+F project SCALLOPS, Internal Project Report, DFKI Saarbrücken, Germany, 2007. www.dfki.de/ klusch/owls-xplan2-report-2007.pdf

[88] A. Rosenfeld, C. Goldman, G. Kaminka, S. Kraus: An Agent Architecture for Hybrid P2P Free-Text Search. Proceedings of 11th Intl Workshop on COoperative Information Agents (CIA), Delft, Springer, LNAI 4676, 2007.

[89] M. Schlosser, M. Sintek, S. Decker, W. Nejdl: A Scalable and Ontology-based P2P Infrastructure for Semantic Web Services. Proceedings of 2nd IEEE Intl Conference on Peer-to-Peer Computing (P2P), Linkoping, Sweden, 2003

[90] B. Schnizler, D. Neumann, D. Veit, C. Weinhardt: Trading Grid Services - A Multi-attribute Combinatorial Approach. *European Journal of Operational Research*, 2006.

[91] E. Sirin, J. Hendler, B. Parsia: Semi-automatic Composition of Web Services using Semantic Descriptions. Proceedings of Intl Workshop on Web Services: Modeling, Architecture and Infrastructure workshop in conjunction with ICEIS conference, 2002.

[92] E. Sirin, B. Parsia, J. Hendler: Filtering and Selecting Semantic Web Services with Interactive Composition Techniques. *IEEE Intelligent Systems*, July/August, 2004.

[93] E. Sirin, B. Parsia, D. Wu, J. Hendler, D. Nau: HTN planning for Web service composition using SHOP2. *Web Semantics*, 1(4), Elsevier, 2004.

[94] D.E. Smith, D.S. Weld: Conformant Graphplan. Proc. of 15th AAAI Conference on on AI, Pittsburgh, USA, 1998.

[95] B. Srivastava, J. Koehler: Web Service Composition: Current Solutions and Open Problems. Proceedings of the ICAPS 2003 Workshop on Planning for Web Services, 2003.

[96] S. Staab, H. Stuckenschmidt (eds.): Semantic Web and Peer-to-Peer. Springer, 2006.

[97] M. Stollberg, U. Keller, H. Lausen, S. Heymans: Two-phase web service discovery based on rich functional descriptions. Proceedings of European Semantic Web Conference, Buda, Montenegro, LNCS, Springer, 2007.

[98] R. Studer, S. Grimm, A. Abecker (eds.): Semantic Web Services. Concepts, Technologies, and Applications. Springer, 2007.

[99] K. Sycara, M. Klusch, S. Widoff, J. Lu: LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace. *Autonomous Agents and Multi-Agent Systems*, 5(2):173 - 204, Kluwer Academic, 2002.

[100] D. Trastour, C. Bartolini, C. Priest: Semantic Web Support for the Business-to-Business E-Commerce Lifecycle. Proceedings of the International World Wide Web Conference (WWW), 2002.

[101] P. Traverso, M. Pistore: Automated Composition of Semantic Web Services into Executable Processes. Int Semantic Web Conference, LNCS 3298, Springer, 2004.

[102] D. Tsoumakos, N. Roussopoulos: Adaptive Probabilistic Search (APS) for Peer-to-Peer Networks. Proc. Int. IEEE Conference on P2P Computing, 2003.

[103] R. Vaculin, K. Sycara: Towards automatic mediation of OWL-S process models. IEEE International Conference on Web Services (ICWS 2007), 2007.

[104] W.M.P. van der Aalst, A.J.M.M. Weijters: Process mining: a research agenda. *Computers in Industry*, 53, 2004.

[105] K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar, J. Miller: METEORS WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services. *Information Technology and Management*, Special Issue on Universal Global Integration, Vol. 6, No. 1, 2005.

[106] L.H. Vu, M. Hauswirth, F. Porto, K. Aberer: A Search Engine for QoS-enabled Discovery of Semantic Web Services. Ecole Politechnique Federal de Lausanne, LSIR-REPORT-2006-002, Switzerland, 2006. Also available in the Special Issue of the International Journal of Business Process Integration and Management (IJBPIM) (2006).

[107] Z. Wu, K. Gomadam, A. Ranabahu, A. Sheth, J. Miller: Automatic Composition of Semantic Web Services using Process Mediation. Proceedings of the 9th Intl. Conf. on Enterprise Information Systems ICES 2007, Funchal, Portugal, 2007.

[108] J. Yan, R. Kowalczyk, J. Lin, M.B. Chhetri, S.K.Goh, J. Zhang: Autonomous service level agreement negotiation for service composition provision. *Future Generation Computing Systems*, 23(6), Elsevier, 2007.

[109] A.M. Zaremski, J.M. Wing: Specification Matching of Software Components. *ACM Transactions on Software Engineering and Methodology*, 6(4), 1997.