A Survey Web Services Composition

by

`A survey on web services composition',
Schahram Dustdar* and Wolfgang Schreiner,
Int. J. Web and Grid Services, Vol. 1, No. 1, 2005

2. The need for web services composition

Composite service

- If the implementation of a web service's business login involves the invocation of other web services, it is necessary to combine the functionality of several web services, referred as a composite service (Alonso et al., 2004).
- The process of developing a composite service in turn is called service composition.
- Service composition can be either performed by composing elementary or composite services.
- Composite services in turn are recursively defined as an aggregation (Khalaf and Leymann, 2003) of elementary and composite services.

The need for web services composition

- When composing web services, the business logic of the client is implemented by several services.
- This is analogous to workflow management, where the application logic is realised by composing autonomous applications.
- This allows the definition of increasingly complex applications by progressively aggregating components at higher levels of abstraction.
- A client invoking a composite service can itself be exposed as a web service.

The need for web services composition

- Programming languages focus on APIs rather than on the actual business logic.
- Different approaches and the need for workflow modelling have finally led to the development of the business process execution language for web services BPEL4WS (Doulkeridis et al., 2003).
- A composition model and language
  - to specify the services involved in the composition,
  - a development environment with a graphical user interface to drag and drop web service components and
  - a run-time environment to execute the business logic can be identified as the three main elements of a web services composition middleware.

The need for web services composition

- Workflow Management Systems (WfMS) are highly flexible and generic but on the other hand require the components to be aware of the WFMS API.

The need for web services composition

- Alonso et al. (2004) describe six different dimensions of service composition models.
- A **component model** can make different assumptions of what a component is and what it is not.
- The advantage of a model making very basic assumptions, for example components only have to exchange messages via XML, is a more general model, while it has to deal with much more heterogeneity of the components.

The need for web services composition

- An **orchestration model** defines abstractions and languages to define the order in which and the conditions under which web services are invoked.
- Orchestration models use process-modelling languages, such as UML activity diagrams,
  - Petri-nets, state-charts, rule-based orchestration, activity hierarchies, and π-calculus.
- **Data access models** define how data is specified and exchanged between components.
- The **service selection model** deals with static and dynamic binding that is how a web service is selected as a component, statically at design-time or dynamically during run-time.

The need for web services composition

- The Business Process Execution Language for Web Services (BPEL4WS) standard proposed by IBM (Virdell, 2003), Microsoft and BEA (Andrews et al., 2003) is in many cases related to former standards, such as WSFL or XLANG. BPEL descriptions are XML documents, which describe the roles involved in the message exchange, supported port types and orchestration, and correlation information as aspects of a process.
- BPEL4WS is a service composition model (Wohed et al., 2003), which supports both, composition and coordination protocols.
- It also consists of an activity-based component model, an orchestration model that allows the definition of structured activities, XML schema data types, a service selection model and a mechanism for exception, event and compensation handling.

3. Managing a film crew: a case study

- According to Dustdar and Treiber (2004) the case study of managing a film crew has been selected, in order to explain the different concepts and frameworks introduced in various papers, with a consistent reference scenario that runs through the whole paper.
- Basically, a film crew consists of one or more film directors, external experts and crew-members.
- Each of these teams provides particular services and work together in a loosely coupled way, providing their expertise on demand, while they depend on the services provided by other film teams.

3. Managing a film crew: a case study

- A film director must coordinate the film teams, is responsible for the budget, and must enable communication among different teams.
- Since different phases in the film production place different requirements on the arrangement of the teams, each phase requires flexible composition and configuration of the film teams.
- Directors hire film crews and external experts who provide own services and capabilities necessary for shooting a film.
- External experts offer their expertise on several topics, such as physics, health, or law to make the film and special effects more realistic. Crew-members may act as stuntmen for example.

4. Composition issues

- Web service composition (Benatallah et al., 2001, 2002) is a very complex and challenging task, as will be seen when discussing different composition approaches in greater detail.
- Before performing web service composition, some basics to enable service composition have to be performed.
- Six different issues that have a large impact on service composition have been identified.
- The following sub-sections do not claim completeness of composition issues. The most important issues may be explained in a few words.

4. Composition issues

- The concepts, some standardisation efforts and related research work as well have been introduced.
  - Sun Microsystems, for example, has proposed standards for coordination (Terai et al., 2003),
  - transaction, and context and has put them together to one comprising standard called WS-CAF (Web Services Composite Application Framework) (Bunting et al, 2003b).
- Coordination, transaction, context and conversation modelling is discussed. When composing services composite service execution must be considered.
- Centralised and distributed service execution has been differentiated, and frameworks that fall into these categories have also been discussed.

4.1 Coordination

- When it comes to composing web services and building complex software systems, it is likely that interaction requires coordination of sequences of operations, to ensure correctness and consistency.
- New protocols and abstractions are needed and this is exactly the case of coordination.
- In order to provide modelling abstractions and simplify web service development, different standardisation efforts, such as WS-Coordination (Cabrera et al., 2002) by IBM or WS-CF by Sun (Bunting et al, 2003a) have been taken.

4.2 Transaction

- To add guarantee to the interactions, it is of considerable importance to add a transaction (Papazoglou, 2003) protocol to the coordination framework in order to provide short-duration transactions, called atomic transactions, and long running business activities as well.
- In case of long running activities, it is not always possible to ensure ACID (atomicity, consistency, integrity, and durability) properties of web service transactions.
- Very important to name are the WS-Transaction (Cabrera et al., 2001; Freund and Storey, 2002a, 2002b) standard, proposed by IBM and WS-TXM (Bunting et al, 2003c) suggested by Sun.
- WS-Transaction builds upon the WS-Coordination framework mentioned above, since it defines protocols for centralised and peer-to-peer transactions, both of which require the existence of coordination.

4.3 Context

- The term context is a very vague one, since many different definitions can be found in literature.
- In terms of web services, context maybe inferred as information utilised by the web service to adjust execution and output to provide the client with a customised and personalised behaviour (Keidl and Kemper, 2004a; Álvarez et al., 2003).
- Context is extensible with new types of information at any time, without any changes to the underlying infrastructure.
- Context may contain information such as a consumer's name, address, and current location, the type of client device, including hard- and software that the consumer is using, or all kinds of preferences regarding the communication.
- The WS-Context (Bunting et al., 2003d) standard, which is part of the WS-CAF standard proposed by Sun, specifies context, context sharing, and context management.

4.4 Conversation modelling

- Benatallah et al. (2004b) introduce a framework that builds on currently existing standards to support developers in defining service models and richer web service abstractions.
- The authors identify several abstractions, which can be classified as completion or activation abstractions.
- Completion abstractions contain compensating operations in case an effect of a so-called forward operation has to be cancelled and resource-locking operations that lock resources for a client.
- Activation abstractions describe implicit and explicit transitions between states.

4.4 Conversation modelling

- An important type of implicit transitions is timed transitions, which occur automatically after some time.
- To model transitions, multiple properties are used: activation properties describe a transition's triggering features, transaction properties specify a transition's effect on the client state, and locking properties reserve certain resources for the requester for a given time.
- The conversation model facilitates
  - service discovery and dynamic binding,
  - service composition model validation,
  - service composition skeleton generation,
  - analysis of compositions and conversations and conversation model generation.

4.4 Conversation modelling

- In Bultan et al. (2003), the authors propose a specification for designing and analysing service composition. Peers, modelled as Mealy machines, communicate through asynchronous messaging and maintain a queue for each incoming messages.
- Related work in web service conversation modelling include the WSCL (Web Services Conversation Language) specification, the WSCI (Web Service Choreography Interface) specification, and also WS-Coordination and WS-Transaction, although they do not provide means to model business conversations, but rather propose specific conversations that can be used to coordinate interacting parties and provide middleware properties (Benatallah et al., 2004a).

4.5 Execution monitoring

- We distinguish between centralised and distributed execution of composite web services.
- Centralised execution is similar to the client-server paradigm.
- In this case, the server is the central scheduler that controls the execution of the components of the composite web service.
- The e-flow platform developed by HP, described in the previous section, works with a centralised-scheduler (Sun et al., 2003).

4.5 Execution monitoring

- The distributed paradigm in contrast expects the participating web services to share their execution context.
- Each of the hosts running a web service has its own coordinator, which has to collaborate with the coordinators of the other hosts, to guarantee a correct ordered execution of the services.
- SELF-SERV (Sheng et al., 2002) developed by the University of New South Wales uses such a distributed execution system.
- A hybrid form of the distributed and centralised paradigms may be coordinator that controls not only one but a set of web services.

4.6 Infrastructure

- Ran (2003) suggests another model for discovering web services based on QoS constraints by extending the basic web services model discussed in Section 1.
- 48% of the UDDI entries are unusable. Pointers are missing, broken or contain inaccurate information. Service discovery in UDDI is limited to functional requirements only.
- The approach taken in Ran (2003) does not touch the existing web service model but extends it by adding another entity called the ‘QoS certifier’.
- The process of publishing, finding, and binding to a web service remains the same. But it adds to the role of the QoS certifier to verify the service provider’s QoS claims before the registration of a service in the UDDI registry can take place.
- In the new model, a service provider also has to provide QoS information, and not just the functional aspects.
- The certifier then may approve the claim of the provider, or may down grade it.

4.6 Infrastructure

- When a service consumer looks for a service, it may add some QoS constraints to the functional requirements, and thus, enforce a much finer search.
- Since there may be many services that meet the functional requirements of the user, the QoS constraints may help to filter the proper services.
- If no service is found, the user may also consider trade-offs by reducing the QoS constraints.

4.6 Infrastructure

- In order to realise the proposed extension of the web service model, it is necessary to extend the UDDI data structure by an additional element called ‘qualityInformation’.
- The structure provides the description of different QoS aspects, such as availability, reliability or performance.
- Like the ‘bindingTemplate’ element it refers to the ‘tModel’ defined in the UDDI registry as references to QoS taxonomies.
- If we now consider a simple SOAP request to a UDDI registry, then it may now, related to the new model, contain QoS parameters to find the proper services that meet these requirements:

```
<SOAP-ENV:Envelope>
        <SOAP-ENV:Body>
                <find service>
                ...
                        <qualityInformation>
                        <availability>0.9</availability>
                        </qualityInformation>
                </find service>
        </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The registry may respond with service references matching the search criteria:

```
<SOAP-ENV:Envelope>
<SOAP-ENV:Body>
        <serviceList>
                <serviceInfos>
                <serviceInfo serviceKey="…">
                        <qualityInformation>
                        <availability>0.99</availability>
                        </qualityInformation>
                </serviceInfo>
                <serviceInfo serviceKey="…">
                        <qualityInformation>
                        <availability>0.91</availability>
                        </qualityInformation>
                </serviceInfo>
                </serviceInfos>
        </serviceList>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

4.6 Infrastructure

- In Ran (2003) the author organises different QoS aspects in runtime related QoS, such as
  - scalability, capacity, performance (response time, latency, throughput), reliability, availability, flexibility, exception handling, and accuracy;
  - transaction support related QoS, like regulatory, supported standard, stability, cost, and completeness;
  - and security related QoS, for example authentication, authorisation, confidentiality, trace ability and audit ability, data encryption and non-repudiation.

4.6 Infrastructure

- P2P systems bring more availability, scalability, and extensibility (Wang et al., 2003). Peer-Serv is a framework that is composed of several providers, brokers and requestors to support web services in a P2P environment (Wang et al., 2003).
- Service providers and service requestors have the same role as the service providers and the service consumers described in the web service model.
- Service brokers are responsible for maintaining the status of the peers. A service broker maintains providers and requestors as well to form a community.
- Several communities may then interact with each other over their brokers. Peer-Serv uses P2P technology for service execution as well as for service publishing and service querying.

5. Composition approaches

- Five categories of composition strategies have identified here. Static and dynamic composition strategies concern the time when web services are composed. They are equivalent to design-time and run-time composition.
- Model driven service composition, business rule driven service composition, and declarative composition have been dealt with. Another topic concerns automated (Narayanan and McIlraith, 2002) and manual service composition.
- A survey of ontology-based composition and semantic web services composition (Rao and Su, 2004; Agarwal et al., 2003) as counter-part to manual composition techniques such as provided by BPEL has been provided. This section closes with an overview of context based service discovery and composition.

5.1 Static vs. dynamic service composition

- Static composition takes place during design-time when the architecture and the design of the software system are planned.
- The components to be used are chosen, linked together, and finally compiled and deployed.
- This may work fine as long as the web service environment – business partners, and service components does not, or only rarely, change.
- Microsoft Biztalk and Bea WebLogic are examples of static composition engines (Sun et al., 2003).

5.1 Static vs. dynamic service composition

- If other businesses provide newer services, or the old services are replaced by other ones, inconsistencies might be caused.
- In that case, it is unavoidable to change the software architecture and bind to other services or, in the worst case, even change the process definition and redesigns the system.
- In this case, static composition may be too restrictive and components should automatically adapt to unpredictable changes (Sun et al., 2003).
- e-flow from HP or the StarWSCoP, both described later in this section implement a dynamic composition engine.

5.1 Static vs. dynamic service composition

- The service environment is a highly flexible and dynamic environment.
- New services become available on a daily basis and the number of service providers is constantly growing.
- Ideally, service processes should be able to transparently adapt to environment changes, and to adapt to customer requirements with minimal user intervention.
- HP has developed a system called e-flow for specifying, enacting and monitoring composite e-services (Casati and Shan, 2001).
- HP defines e-services as the means by which an enterprise offers its products, services, resources, and know-how via the internet. E-services may be used by persons (business to consumer), enterprises (business to business), and electronic devices and is thus, a broader term than e-commerce or e-business.

5.2 Model driven service composition

- The paper (Orriens et al., 2003a) introduces the approach of Model Driven Service Composition, which is based on dynamic service composition discussed in the previous section, since it should facilitate the management and development of dynamic service compositions.
- UML (Unified Modelling Language) is used to provide a high level of abstraction, and to enable direct mapping to other standards, such as BPEL4WS.
- The OCL (Object Constraint Language) is used to express business rules and to describe the process flow.

5.2 Model driven service composition

- Business rules can be used to structure and schedule service composition, and to describe service selection and service bindings.
- The paper identifies two main use cases: the process of service composition development, which may be sub-divided into the four phases of service definition, scheduling, construction, and execution.
- In order to enable representation of all possible service compositions, the paper introduces an information model – an abstract meta-model – that models components and relationships between the components.
- For that purpose, service composition elements and service composition rules are defined.

5.3 Declarative service composition

- The approach followed by enTish (Ambroszkiewicz, 2003) is somewhat different from typical composition platforms.
- Services are typically created on the fly to realise client requests.
- Anyway, most frameworks are based on the assumption that first the business process has to be created. For enTish, a different architecture is needed, since client requests are expressed in a declarative way using formal languages.
- The declarative approach consists of two phases:
  - The first phase takes an initial situation and the desired goal as starting point, and constructs generic plans to reach the goal.
  - The latter one chooses one generic plan, discovers appropriate services, and builds a workflow out of them.

5.3 Declarative service composition

- The first phase is realised using
  - PDDL (Planning Domain Definition Language) and estimated-regression planning as used in XSRL (XML Web-services Request Language), which must provide machine-readable semantics and specify the abstract service behaviour.
- The second phase may be realised by using existing process modelling languages, such as BPEL.

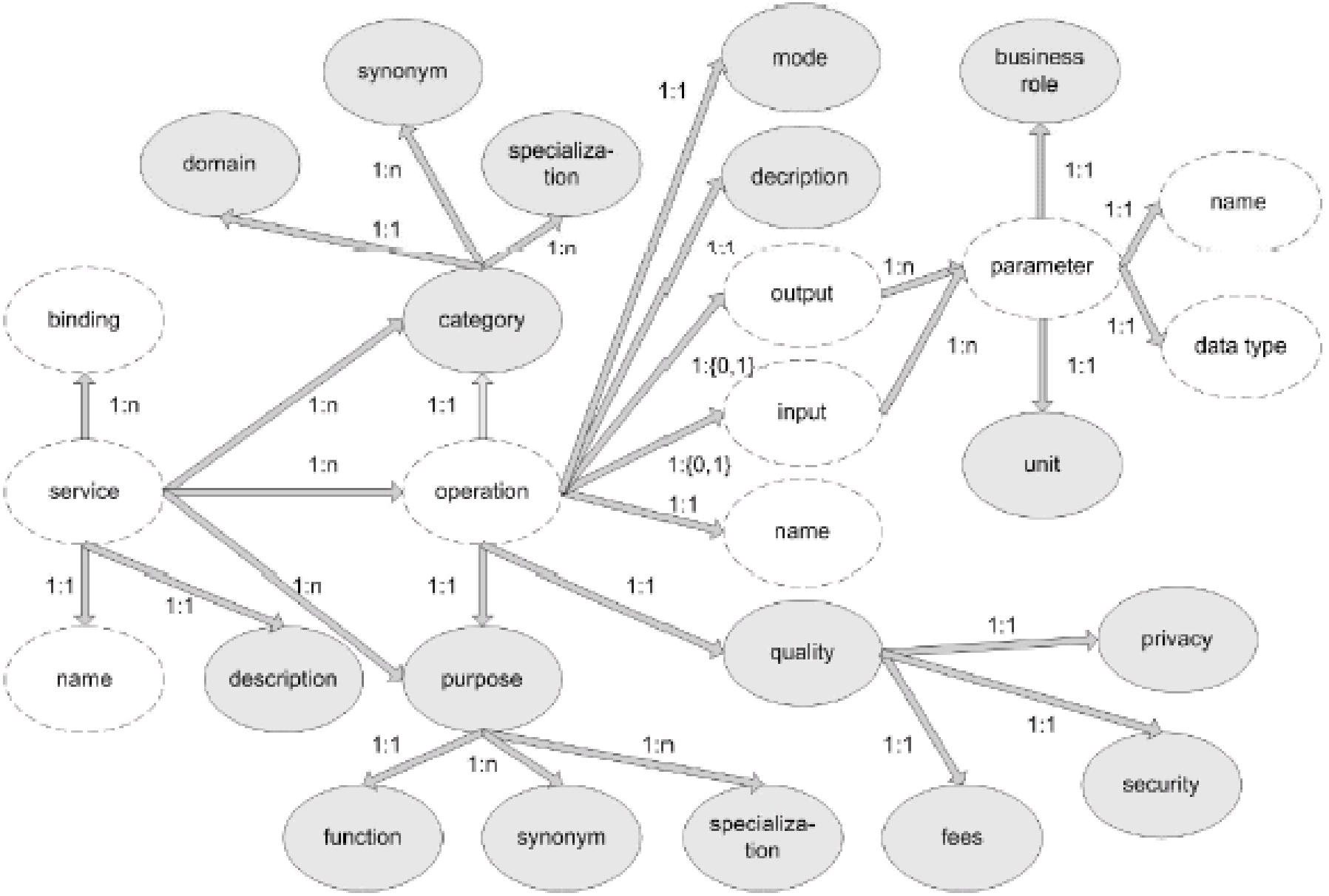
5.4 Automated vs. manual web services composition

- BPEL4WS is used as a manual composition model.
- In this sub-section automated service composition is dealt with in greater detail.
- In contrast to manually describing service composition, there is also a lot of effort done in the direction of automated or ontology based service composition (Tosic et al., 2002b; Fileto et al., 2003).
- The most frequently used definition of ontology in the literature on semantic web services (Motta et al., 2003; Medjahed et al., 2003; Horrocks and Tessaris, 2002) is that an ontology is a collection of web services that share the same domain of interest and describe how web services can be described and accessed.

5.4 Automated vs. manual web services composition

- Ontology languages, such as RDF, DAML-S (DARPA Agent Markup Language for Web services), DAML+OIL, or OWL (Web Ontology Language) have formal specifications and, thus, can be queried and provide the means to define sophisticated class properties.
- A web service dealing with film equipment may belong to a film-production-ontology for example.
- DAML-S provides the mechanisms to organise web services into ontology.

Ontology based description of web services



5.5 Context based Web Service Discovery and Composition

- Services become available through different channels, meaning that e-services may be accessed using different devices, such as PCs, palmtops, cell-phones or TV sets, and also different network technologies and protocols (Baresi et al., 2003).
- The goal should be to deliver the same service via web, SMS, or call centres, for example.
- The Multi-channel Adaptive Information Systems project MAIS proposed in Baresi et al. (2003) aims to create a platform, a methodology and design tools for the development of distributed information systems, based on e-services.
- Baresi et al. (2003) defines a modelling framework for adaptive information systems separating the application and technological levels and providing formalised contracts between the Service Provider and the Service Consumer.

Web services composition models

Framework	*Composition strategy*	*Execution monitor*	*Dynamic conversation selection*	*QoS Modelling*	*WSDL Language extension*	*Transaction support*	*Graph support*
E-flow	Dynamic composition	Yes	Yes	No	No	No	Yes
MAIS	Context based	No	No	Yes	Yes	No	No
MOEM	Context based	No	No	No	No	No	Yes
SELF-SERV	Declarative composition	No	No	No	No	No	Yes
OntoMat-Service	Semantic composition	No	No	No	Yes	No	No
SHOP2	Semantic composition	No	No	No	Yes	No	No
WebTransact	Dynamic composition	No	No	No	Yes	Yes	No
StarWSCoP	Dynamic composition	Yes	No	Yes	Yes	No	No