Dagstuhl seminar on Service Oriented Computing

Service design and development

Group report by Barbara Pernici, Politecnico di Milano

Abstract

This paper reports on the discussions on service design and development which were held during the Dagstuhl seminar on Service Oriented Computing held at Dagstuhl in November 2005. The group discussed service design and development considering modeling and methodological issues, different perspectives on the development life cycle, and set up a research agenda on these issues.

1. Introduction

When service design and development is discussed, and important aspect is to clarified which of the several possible service design goals, and in particular:_

- *Integration* (EAI): in this case the goal is to integrate different information systems, to create new cooperative applications. In this case it is important to define the granularity at which services are considered, and how existing systems are wrapped to offer their functionalities in a cooperative information system through a service-oriented approach (Papazoglou 2006).
- *Redesign* (e.g., mobile, multi-channel): the goal is to modify existing functionalities in order to offer them in a variable environment setting. For instance, allowing the client to interact with the system from a variety of different devices and from different places. To provide the requested functionality, redesign has to take into consideration quality of service parameters and their variability, dynamic selection and binding of services.
- *New added-value services*: new services are created as a composition of existing services. Composition allows reusing services in several contexts and for different applications. Composition may be performed following a fixed process schema, or designing the composition structure dynamically.

As shown Figure 1, the service provisioning infrastructure considers services in different layers. Starting from existing legacy systems, wrapped as components and provided through a service infrastructure, business level services can be offered, combined in business processes and distributed.

The issue of differences between component-based and service-based approaches was discussed. A service is a functionality that a component offers, while a component is a software system which can be deployed (service is already deployed, component is like a package). A service can also be seen as an access point to a component. The issue of

composition has been studied also within component literature, and in particular with respect to quality of service aspects, such as for instance availability.
Another distinction which was underlined is that a service is seen as a business thing, while a component as a software thing (e.g., EJBs).

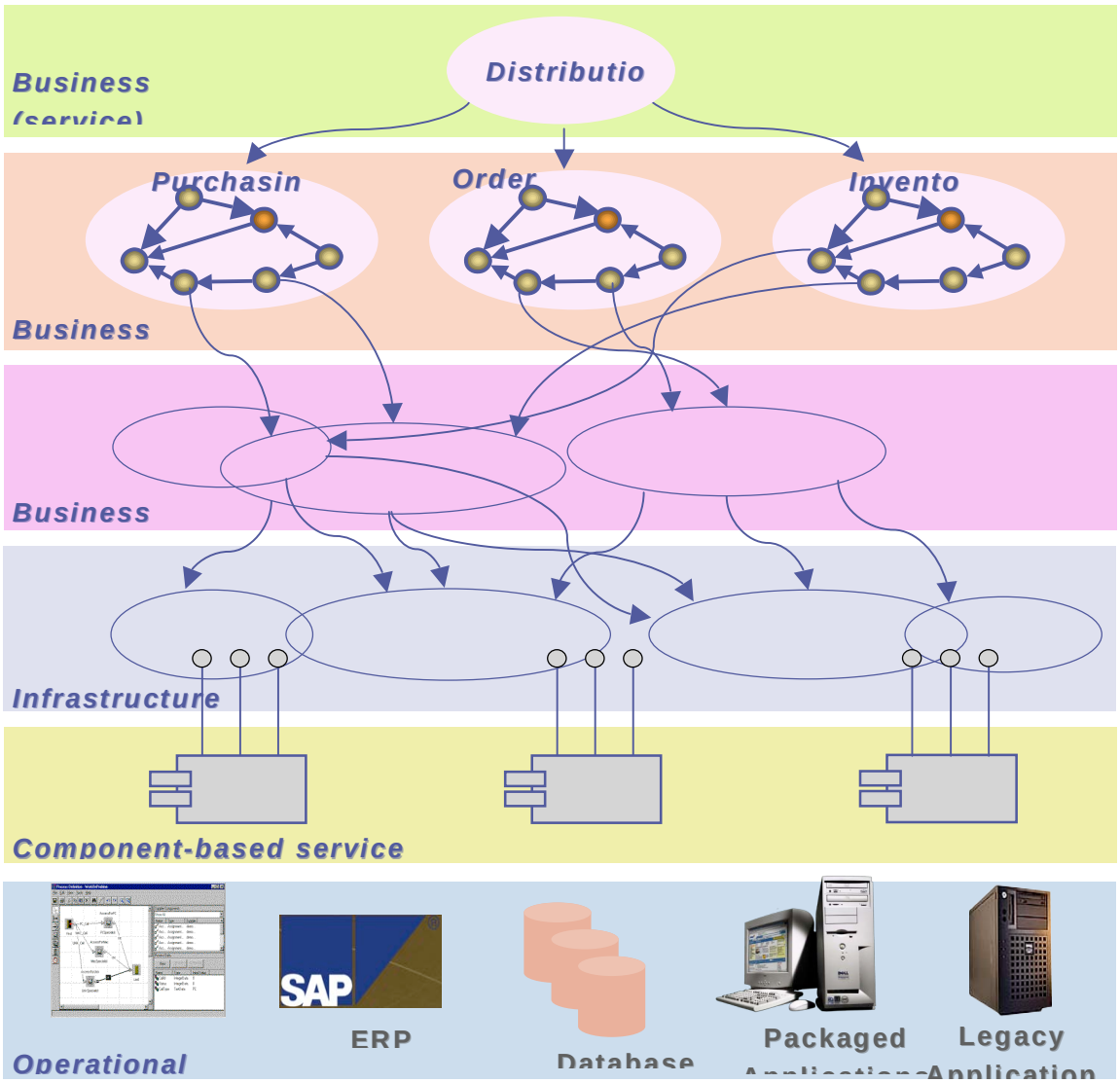

Figure 1 - Service provisioning and legacy systems (Papazoglou 2006)

2. Design perspectives

2.1 Client and provider perspectives

The discussion in the introduction presents services mainly from the point of view of the providers, which are offering their information systems functionality through a service-oriented infrastructure. However, the group agreed during the discussions that in the design process the perspective of the client of the service is an essential element to correctly design the service, independently on how it is provided (new service, wrapping of an existing application or composed service). In fact, only the client can design the business process for his purpose, with his own requirements.

Some authors have proposed a goal driven approach to elicitate requirements for services to be provided to clients and to design their functionality and deployment (Kaabi et al. 2004)

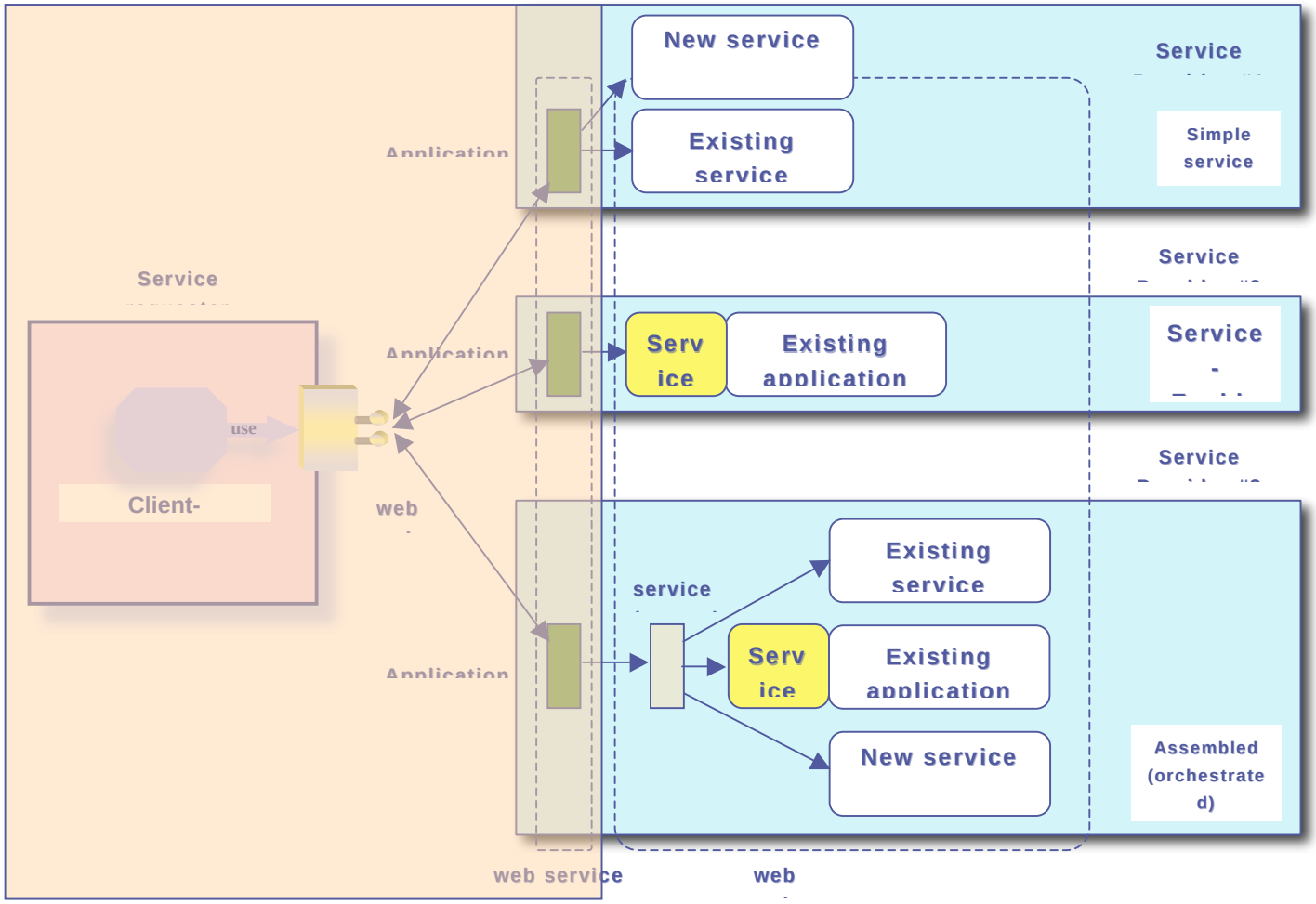

Figure 2 - Providing services to clients (modified from Papazoglou 2006)

A possible direction to build services for a wide community is to study requirements of reference clients or groups of clients, including current clients or future clients. Standardization and certification criteria can help developing services which may be used in different contexts, possibly with a customization of services for personalizing to a client, with limited costs and well defined functionality.
Economic aspects have to considered to pursue this approach, to make service provisioning possible in a competitive market and the possibility of providing add-ons with respect to standardized services possible, to differentiate products of different producers.

3. Models

The need for models with a rich set of elements as a basis for the design process emerged as a clear prerequisite for design methodologies.
Models should include a variety of aspects, and in particular:

- business services
- Composition and coordination
- Interaction between clients and providers
- How to model a service and its QoS properties
- Transactional properties
- Self description (meta-data)

For each model element a modeling language should be defined. The discussion focused on existing background in the area, and which are the languages and models which can be a basis for modeling web services, including for instance the WS-stack, UML2, BPMN, EPC, Aris, and so on. No clear and unique background emerged from the discussion.

Among the aspects to be considered, the representation of interaction among organizations was discussed. Boundaries between organization are relevant in the ability for a client to state his requirements, either with request languages or with a goal oriented approach, specifying his goals, and in the possibility of modeling processes across organizational boundaries.

3. Methodologies

3.1 Starting points

Clear starting points emerging from the discussion, and represented in Section 1, are the representation both from the service requestor and the service provider points of view, including both functional and non-functional aspects.
The discussion was centered on the specific point of how a service public view should be. The common black box approach presents some limitations which are difficult to overcome, and this consideration paves the way to a grey box approach, although a black box view would be preferrable since it is simpler. A grey box view becomes necessary

when services have an observable state, on which external protocols defining possible conversations can be defined, similarly to what is done in the hardware design area.

3.2 Strategies

Different aspects related to design strategies where examined. A meet-in-the-middle approach to design emerged as preferrable. The granularity of the service has to be defined, in particular when considering legacy systems specifying services using a fine granularity can be an excruciating task. A possible solution to some design problems has been proposed using reference models and ontologies.

Horizontal and vertical integration (process integration or within same company) where discussed.

One of the problems considered is where to start from in the design process, and the question arises if the analysis should be limited to e-service design. An approach starting from business services, looking also at existing processes) seems appropriate.

Part of the discussion debated at length on how domain-specific approaches should be followed, to the extreme that all development should be basically domain-specific. This issue also emerged when discussing some of the phases of the development life cycle illustrated below.

A need for a governance strategy was indicated.

3.3 Life cycles

A number of life cycles were discussed and proposed. They originate from the different points of view discussed above, and in particular distinguishing between the provider's view, when the goal of a provider is to develop new services, and the requestor's and provider's views, in the case the provider has to provide a service dynamically adapting the characteristics to a client request.

3.3.1 Provider's view: developing and providing a new service

The main issue in this case is how to transform a system into a service. The existence of legacy systems and the definition of the granularity of the services are particularly relevant here.
Vice versa, a different approach is needed when designing new services from platform-independent services into runtime artifacts, going to abstract to concrete services.

An important issue related to development is the ability to manage changes.

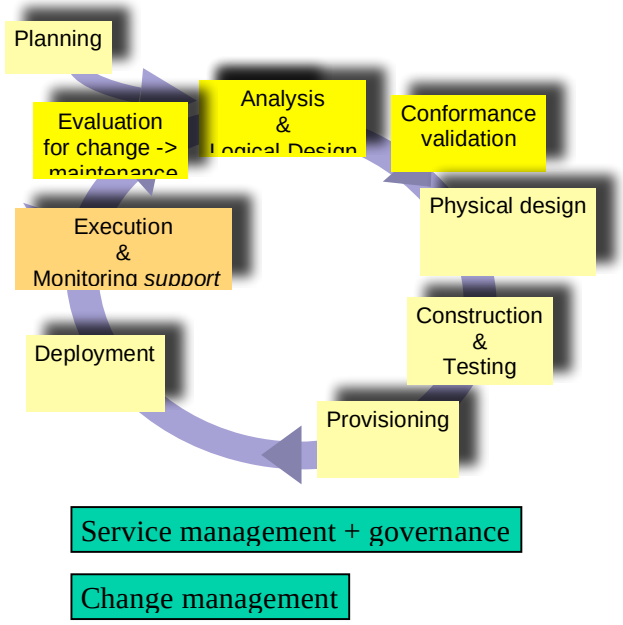

Figure 3 - Development of services

The phases illustrated in
Figure 3 include the following:

- *Analysis*: Proposed alternatives are to define goals to guide the composition of existing services and to base design (or redesign) on QoS requirements. The granularity of services should be defined in this phase, adopting criteria such as cohesion and coupling.

- *Logical design:* it should define what the services are (the focus here is mainly on micro-services rather than complex business services) and in particular:
    - interface
    - composition of services
    - selection of components
    - coordination design (and possibly distributed orchestration)
    - interaction paradigms
    - adaptivity
    - exceptions and self-healing WS

  Domain-specific editors for composition, for instance to generate BPEL code have been proposed. The issue of domain-specific development was debated and strongly supported during the meeting by some of the participants of the group.

- *Physical design*: there was some debate on whether physical design should be an issue specific of service design. Relevant topics are optimization, aimed at improving performance, and selective reuse.

- *Deployment*: for the deployment phase deployment criteria should be specified during the design of services.

- *Monitoring:* the issue here is to design the right level of information for monitoring, providing selective visibility of QoS for a given service (depending on consumer, context, costs involved, ….)

- *Testing*: testing should be performed both on functional and non-functional characteristics of services. A contract with the clients should be provided as a basis. Atomic services (i.e. services providing a single function) should be tested. A language for specifying testing should be designed.

3.3.2 Requestor's view

A requestor's view focuses on the selection and negotiation of services before invocation, according to the SOA approach.
After discovery and selection, either the service can be invoked and therefore integrated in the application invoking it, or some adaptation or customization performed either on the requestor's or on the provider's side.

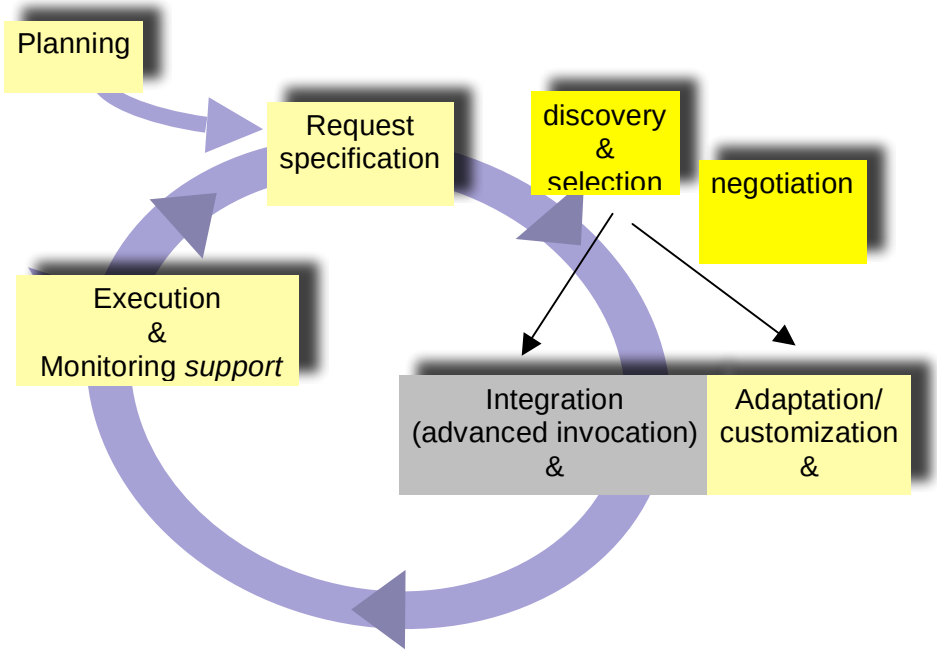

Figure 4 - Service request from the requestor point of view

3.3.3 Provisioning for a given request

Symmetrically to the perspective given above, providers may adapt a service for a given request. In this case, as shown in Figure 5, the main relevant phases are negotiation and adaptation, followed by the other phases described above for the development of new services.

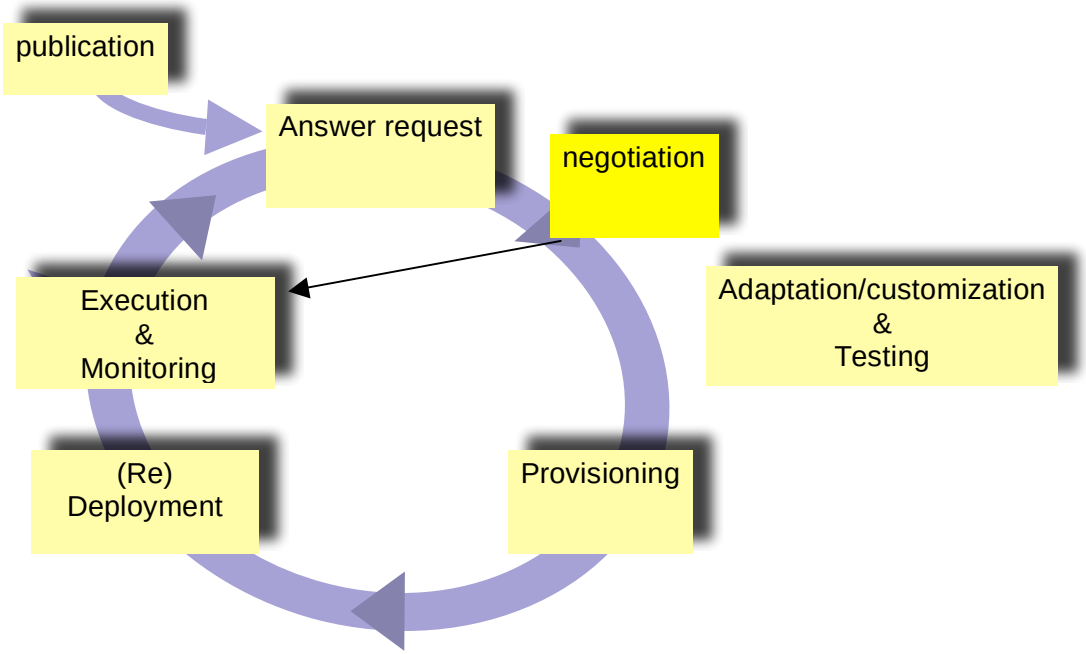

Figure 5 - Service request from the provider point of view

4. Research issues

A number of open research issues where discussed during the seminar concerning the models and requirements for service development.

- Research on models should focus on the following issues:
  - Policies (QoS), behavior, …
  - Service design patterns (WF, interaction, …)
  - Business rules
  - Relationships among models
  - Service hierarchies
  - Self-description (also for data; visibility of properties for grey box approaches)
  - Historical information
  - Protocols and states
  - traceability among models
  - Negotiation and contracts representation (differences with agents area)
  - Exceptions and recovery
- The following general requirements need investigation:
  - Adaptive services (design for), autonomic
  - Context awareness
  - Hierarchical approaches, multiple services until final requestor's goal is reached (services in the middle)
  - Data heterogeneity (also semantic)
  - Traceability
  - Management of life cycle

- Rich request languages
- Reference models: repositories, criteria for conformance

Requirements for design environments have also been discussed:

- Support for integrating legacy systems
  - Integration
  - Transformation into service
- Process modeling (integration of processes – vertical and horizontal)
- Change management support
- Testing
- Deployment
  - criteria (including adaptivity features, monitoring support)
  - Migration of services

5. References

- B. Pernici (ed.) Mobile Information Systems. Infrastructure and Design for Flexibility and Adaptivity, Springer, 2006
- MAIS web site: http://www.mais-project.it
- M. Papazoglou, W.-J. van den Heuvel, Service-Oriented Design and Development Methodology, Int. J. on Web Engineering and technology, 2006
- D. Florian, B. Pernici, Insights into Web Service Orchestration and Choreography," accepted for the special issue of International Journal of E-business Research (IJEBR) on Web Services-Based E-Business Systems, Jan. 2006
- C. Francalanci, S. Grega, P. Losi, A. Maurino, S. Modafferi, B. Pernici, C. Raibulet, F. Tisato. "The MAIS approach to web service design". EMMSAD 2005 Workshop Proceedings, Porto, June 2005. Invited for publication on Adavanced Topics in Database Research - Vol. 5, Idea Group Publ.
- D. Ardagna, B. Pernici. "Global and Local QoS Guarantee in Web Service Selection". Workshop on Business Processes and Services, Sept. 2005.
- S. Modafferi, B. Benatallah, F. Casati, B. Pernici. "A Methodology For Designing And Managing Context-Aware Workflows". In Proceedings of IFIP International Working Conference MOBIS, Dec. 2005.
- R.S. Kaabi, C. Souveyet, C. Rolland, Eliciting service composition in a goal driven manner, ICSOC, 2004