Service-oriented Resource Management

Jürgen Dorn and Hannes Werthner,
Institute of Software Technology and Interactive Systems, Vienna University of Technology,
Favoritenstrasse 9-11, A-1040 Wien, Austria,
email: {juergen.dorn|hannes.wertner}@ec.tuwien.ac.at}

Abstract

Service Management controls how business services are developed and delivered to customers. Business services must be flexible negotiable between customer and provider. Temporal attributes are important and in most cases resources are involved and for customer satisfaction as well as for efficient usage an optimization of resources is required. Often business services are aggregated from services from different organizations. To enable efficient construction, basic services and the information content should be standardized as far as possible without constraining the flexibility.

This paper presents an approach and a prototypical implementation to search, reserve and book resources based on Web services. Groups of resources are managed by an organization's information system which contains a database for storing the expected usage of the resources and some business logic expressed by means of explicit business rules. A distributed resource management system uses the Web services to build up a complex service for customers asking for different resources. This architecture is applicable in many domains. As a showcase, we use the organization of facilities for a scientific conference.

1. Introduction

Service Science, Service Management and Service Engineering are new paradigms for addressing the IT requirements for the growing service sector of our global economy [1]. In contrast to tangible products, e.g., a car, the definition and consumption of business services involves the customer. Service "interfaces" as well as their specific value are defined weaker in the sense that they can be more easily adapted to different customer expectations. This implies that the management of business services is more complex.

One of the key challenges in this context is the alignment of business services with the IT infrastructure [2]. Service-oriented architectures promise a more flexible usage of Internet as well as real world resources; thus, they may also support business services in a more flexible way [3].

Since software encapsulated or expressed by Web services can be located and bound dynamically, a real world resource may also be exposed dynamically by a Web service. An organization possessing one or more such resources may publish them to offer these resources over the Internet. The booking of such a resource is one business service offered by this organization. An information system in the background has to check the availability of resources and the creditability of customers. Furthermore, the system will compute the price of this resource dependent on the customer's expectation. Thus, the business service is defined collaboratively between provider and customer.

A great challenge is to transform automated or semi-automated business services into Web services at the IT-layer. One should note that the notion of service refers to different definitions: in management science a service is defined as a business economic activity, offered by one party to another to achieve a certain benefit, and "generated" by business processes [4]. In information systems a service is a complex (or simple) task executed (within) an organization on behalf of a customer [5]. And in computer science a Web services refers to programmable, self-describing, encapsulated, and loosely coupled functions accessed and invoked over the Internet [6].

Our research question is now, whether a Web service designed according to some still to be defined criteria may be a business service, or the other way round, may we implement a business service by a Web service? Usually, a business service is seen as a service independent of IT, involving customer contact and is "only" supported by IT. We see however, that already many services are offered in the Internet implemented only by IT. Thus, we may distinguish e-business services from ordinary business services where still human interaction is involved. Another reason to neglect the implementation of business services by Web services is the argument that business services are offered by combining services from different

1530-1605/08 $25.00 © 2008 IEEE

organizations and Web services are implemented in one organization. If we use virtual organizations as an intermediary then we may deploy the Web service at this virtual organization and use recursively Web services of involved organizations. The third argument against our simplified assumption is that business services are complex entities and Web services should be small encapsulated functions. The requirement that Web services should be "small" stems from the objective to design them reusable. If we define large complex software artifacts the probability that they are reusable is small [7], [8]. The common approach to make complex software reusable is to use generators that transform a number of small functions/components to larger systems [9]. In the case of Web services two approaches are usually proposed:

- Combining Web services by processes specified in a descriptive process language (e.g. BPMN) where different steps in the process are Web services and
- Composing Web services by advanced planning techniques derived from AI-planning.

In our research we address the challenge of transforming a complex business service into a number of Web services in both aspects [10]. In the following, we consider the composition of complex services from simple Web services as addressed on a formal level in [11]. Web services aggregating services from different organizations are assigned to a kind of virtual enterprise.

A virtual organization or a virtual enterprise has certain characteristics:

- cooperation of independent enterprises for complex service production,
- coordination by information technology,
- electronic communication,
- a shared management of customers and resources and
- a single access point.

In the following we recapitulate requirements on service-oriented business structures, present conference management as a showcase and present then our approach with examples from the showcase. Finally we conclude.

2. Requirements

One of the general requirements which is also the motivation of a service-oriented architecture (SOA) is the flexibility in service generation. Dependent of customer records and preferences, on the actual availability of resources the actual service is to be configured. Moreover, the approach shall be flexible in enabling further providers and resources to be included on-the-fly. This results in an extensibility requirement that also includes new kinds of services.

A further requirement is a controlled transparency of service offers. In principle, customers are interested under which conditions a service is offered. Thus the transparent display of business objectives and rules could be used to show customers the conditions. Since we consider to offer different customers different conditions it may be necessary to hide certain regulations if this would otherwise annoy customers that may not use these conditions.

An important requirement is trust. If a business is conducted electronically it is important to know who is doing business with whom. We have to represent organizations and their representatives that are allowed to perform certain actions. On one side we need identity management (mainly with authentication mechanisms, certificates and digital signatures), clear regulations who may perform which action (authorization of activities) and encryption for securing the communication.

The previously mentioned requirements are coming from a business perspective. Looking at the IT development in organizations, the reuse of artifacts (not only software components, but also designs and other software related artifacts) is the most prominent expectation into SOAs. To fulfill these expectations design principles known from the research on software reuse have to be applied. It is well known that small software components with few simple interfaces are usually easier to be reused. On the other side, if we can reuse larger components, then the efficiency gain is of course higher. Important for the reuse of software are the interfaces to the components and that it is not required to explain the inside of components (the information hiding principle).

It seems to be a good practice to model real objects of our application domain also as software components (e.g. a resource or a room as derived object). For reusing components, it is important to have a common understanding what these objects are. A resource, for example, is a very general concept and used in the resource description framework (RDF) for any information in the Internet. Considering manufacturing management, a resource is something totally different. If we offer a Web service for booking a resource, it may be derived from the verb what the semantic of a resource is, but it is unclear if we announce a Web service designed for searching a resource. Also input and output arguments of Web services must be understandable.

It is not always possible to encapsulate real objects by Web services. For example, if we integrate the paying (i.e. money transfer between two organizations) we will encapsulate an activity by one or more Web services. For such activities a fail-safe behavior and transaction management are important. Thus, components should be deployed so that after their execution a safe state is achieved since we do not know how clients would react on failures.

We have identified a number of requirements that are derived from two perspectives: from the business view and the IT view. Either we have to define Web services by considering all these requirements or we need a transformation from business services down to IT services. In principle, both approaches can be used in parallel.

3. Showcase

In our showcase, we describe a conference management system that has to manage such resources as meet sing rooms. If these resources are offered via Web services, the system is able to organize or to compose an entire conference schedule using these resources by querying these Web services. The conference organization is then a virtual organization [12] using services from other organizations.

In the implementation we cover several resources such as the TechGate Vienna (TGV) and the Austria Center Vienna (ACV) which offer facilities for large conferences and smaller meetings. Both venues are close together and so that both venues can be used together. The TGV has one large conference room for 300 people, a foyer, a sky lobby, and six rooms with a capacity of up to 60 persons. Moreover, there are additional rooms owned by companies in the TGV. For example, the research institute ec3 has four rooms that may also be used by external customers. The ACV has more rooms, a room with a capacity for up to 4,600 persons and is more expensive in most cases.

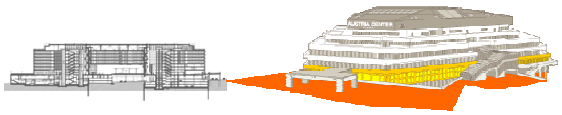

Figure 1: TGV and ACV Buildings

In the following we consider the problem of planning which rooms to use for a conference that has to be organized. Given is an expected number of participants, a financial budget, quality standards and a structure of the conference. The structure describes that there are sessions which will be visited by all participants and that there may be parallel sessions with some expected number of visitors. Constraint based modeling and satisfaction [13] is applied to specify and to determine such desired services/products. A rough schedule is the result demanding, for example, for Monday morning for two hours a large conference room for up to 300 people and afterwards five rooms with a capacity of 60 persons.

There exist considerable research and development work in the area of distributed scheduling and planning (e.g. [14]). We will sketch our planning algorithm, but the focus of this paper is not on this algorithmic problem but on the service-oriented design and deployment of such an application.

4. Architecture

The software architecture of our system is distributed over several organizations. Resources may be grouped logically to clusters (resource groups) such as resources existing in a certain building or owned by a company. This grouping may also be defined hierarchically. In any case, there is a resource management system for each identified group. In our example, these may be resource groups for the two buildings TGV and ACV and a subgroup of rooms at ec3.

Each cluster maintains its own database containing descriptions of its resources, a schedule with reservations and bookings and decision rules describing rules for accepting reservations and bookings. An Intelligent Resource Reasoner (IRR) applies the rules given for a certain resource cluster to the schedules in order to answer queries posted to the respective resource management system. Different types of queries can be posted to the resource management system by means of Web service calls. A simple query may ask whether a resource with certain characteristics exists in the cluster. Part of this query may be a temporal interval during which the resource shall be available. A second Web service is used to reserve a previously identified resource. Such a reservation has a certain time limit after which the reservation will be cancelled if not booked with another Web service. Further Web services exist for cancellation and modifications of bookings.

These different resource management systems offer Web service interfaces. A planner queries these clusters in order to create the respective conference schedule using the different resources. The planner on its turn receives a requirement specification containing the number of resources with specified characteristics. The relationship between these resources may be constrained further. For example, the locations must be close together or the sum of all costs must be smaller than a certain amount.

The following figure illustrates the described architecture. The Web service interface is illustrated by small arrow boxes, representing services for offering (o), reservation (r), booking (b), cancellation (c) and modification (m). Since the architecture is generic, other Web services can be included.

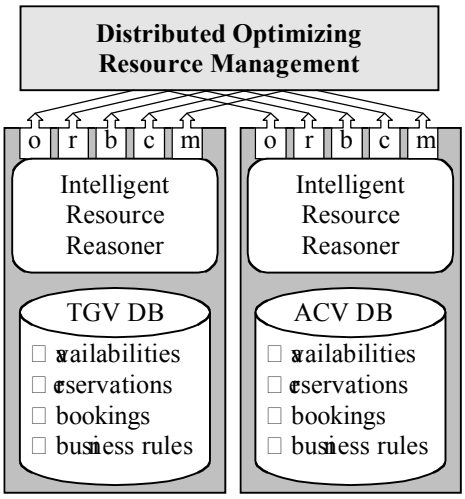

Figure 2: Architecture

The resource management system cannot only be used by a program such as the proposed planner. Another possibility is to call the Web services from an HTTP client with a typical Web form or from a mobile phone with a WAP client. These clients are used typically from companies at TGV to reserve and book rooms for their daily work. Thus, if we need a short term meeting room, we may also go through the information transformed into a simple Web form and select a room. Furthermore, a service to notify members of a meeting is considered.

5. Resource Booking Business Process

In the following we present flow of control as well as information flow for a single resource management system. The upper lane in the following diagram shows activities of a client (e.g. the distributed optimizing resource management system) and the lower lane the activities of the resource management system.

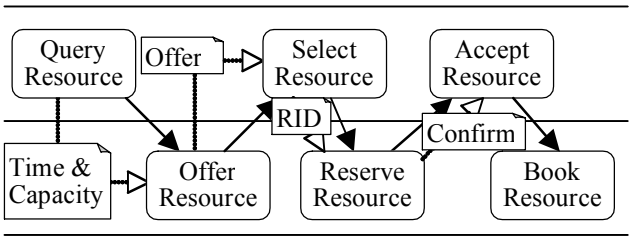

Figure 3: Simplified BPMN – process

The typical booking process consists of the following steps. First a query is sent to a resource management system to find an available resource r_i in a certain time interval. This interval is called *query interval* and we denote it by i_q. The required resource may have required characteristics such as a capacity which may be constrained by attributes $a_1, \ldots a_n$. For example, we may ask whether the sky lobby is available on May, 16th for the whole day. Or we may ask for a meeting room with a capacity for 40 persons for two hours in the next week. The resource management system will answer in the first case with yes/no and in the second case it will supply a list of possible two-hour-intervals in the next week for each room having approximately place for 40 peoples. The proposed interval is called *usage interval* and should be during the query interval. By i_u we denote this interval and the starting and ending time point are abbreviated by t_s and t_e, respectively. In the following figure, the time when the query was issued is t_q and the time of the offer is t_o. The objective of our approach is to make the difference $t_o - t_q = i_r$ (*response interval*) as small as possible. This function is part of the overall optimization function. The resource management system applies a planning and optimization algorithm sketched later.

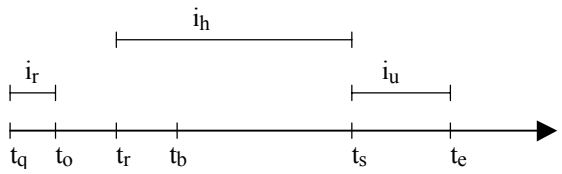

Figure 4: Temporal constraints of process

Based on this offer, we can reserve a room. This reservation is necessary if we are not certain whether we really need the room, otherwise we could book immediately. Reservation is also required if we want to book a number of rooms together. This booking should be seen as one transaction that succeeds or fails completely. If we cannot book all of the required resources, the booking may be senseless for us. Therefore, we would first reserve all rooms and book them afterwards. The room owner is, however, interested to minimize the number of reserved rooms. He will specify rules how long a reservation will be valid. In the figure t_r specifies the time point of the reservation. There are no constraints on the difference $t_r - t_o$. However, the possibility that no reservation is possible is higher if the difference is larger. We call the time between the reservation and the usage interval the *reservation horizon* or *horizon interval* ($i_h = t_s - t_r$).

The booking of a resource is a business contract. If we cancel this contract we may have to pay some compensation fee. The time of the booking is denoted by t_b. The rules how large the difference $t_b - t_r$ may be, will be dependent of the resource owner's strategy and will differ in different resource management systems. The allowed difference will also depend on the reservation horizon. For a large reservation horizon, the allowed difference will be also longer.

The temporal reasoning is supported by interval algebra [15]. The execution, monitoring and supervision of such processes is supervised by a Web process management system [16].

6. Resource Management System

The backbone of the resource management system is a database system that stores the description of the available resources, the reservations and bookings, and business rules that help to decide how a query may be answered.

Figure 5 shows the data model of a resource management system. The central object in this design model is the resource with a unique identifier (rid), the name of the resource, a capacity and costs as attributes. In the sense of object-oriented systems this object is a virtual object. Real objects must be instances of derived objects. If we reserve or book a resource for a certain time interval, this is an allocation object derived from a time interval. The state attribute distinguishes reservations and bookings.

Sharable resources are resources where allocations may overlap. In contrast, for non-sharable resources allocations may not overlap in time. If a customer books a seminar room, no other customer may reserve or book this room in an overlapping time interval. A resource group gives us the possibility to group resources and to apply rules and algorithms in a consistent way to all members of the group. Restaurant, building, place and room are – in our case – prototypical resources. These may have additional attributes and other resources may be derived from them. The attributes capacity and cost are generic attributes and derived resources may have different interpretations how capacity or costs are measured.

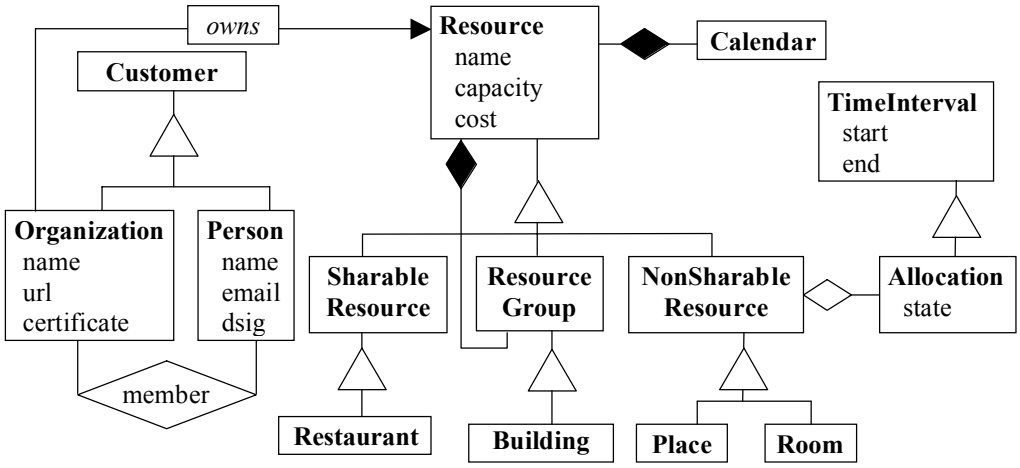

Figure 5: ER-Diagram of the Resource Management System

Resources may have a calendar associated. This can be used to model holidays or other times when the resource is not available. The described model was originally developed for scheduling applications in the DÉJÀVU-project [17] and has be shown to be adequate also for other resource management problems.

Five types of queries can be distinguished. These queries are implemented as Web services and are answered by applying business rules to the data in the database. The Web service is called asynchronously only. Messages are based on the SOAP-protocol [18] that are sent to the resource management system. The content of the message is a standardized XML document. The answer is again an XML document. Although our implementation is realized for booking of conference resources it generic in the sense that also other resources may be managed with this protocol. The temporal attributes of the interface are modeled according to the iCalendar [19] standard. This means a booking of a resource (as well as the reservation) is represented as an event in the sense of iCalendar. This implies that the resource management system also supports recurrent events. The following code specifies the event that ec3's conference room is used by three attendees from 10:00 to 15:00 at April 23.

```
BEGIN:VEVENT
ATTENDEE;CN="Juergen Dorn"
ATTENDEE;CN="Hannes Werthner"
ATTENDEE;CN="Albert Rainer
LOCATION:ec3 meeting room
DTSTAMP:20070329T091802Z
STATUS:TENTATIVE
DTSTART:20070423T130000
SUMMARY:Project Meeting MEMESE
DTEND:20070423T150000
DESCRIPTION:We must decide on
the road map
ORGANIZER;CN="Juergen Dorn":
mailto:juergen.dorn@ec3.at
END:VEVENT
```

This event can be translated to an allocation in our data base. Since iSchedule is not based on XML and Web services use XML, we make a simple transformation from iSchedule attributes to XML elements. The application of iCalendar has the advantage that we can use standard software to visualize events as well as calendar in Web browsers with e.g. phpiCalendar or on smart phones that support the vCalendar standard. In the following we describe the semantic of these Web services.

6.1. Querying for a Resource

The first is a simple query to determine whether a resource is available during a certain time interval (i.e., in the week April 19th to 23rd) for 5 hours. As input parameters we provide the requirements applied to the room and the time when the resource shall be available and for how long. The requirement is that the resource is classified as meeting room and that it must have a capacity for at least 3 persons. The simplified[1] message call is as follows:

```
offerResource(
  <Type>Meeting Room</Type>
  <Capacity>3</Capacity>

<DTSTART>20070419T100000</DTSTART>
  <DTEND>20070423T200000</DTEND>
  <DURATION>PT5H</DURATION>
  <Client id = "Juergen Dorn"/>)
```

In the simplified example, we use the client id as a mechanism for authentication. In the real system, we may use also a digital signature and the resource management system manages besides the resources also a database of clients. For each client or group of client specialized business rules may be associated.

If we assume that no reservations or bookings exists in the resource management system, business rules will be applied to return the most appropriate room in the most appropriate time slots. These business rules will match the queried capacity with the rooms' capacity. Moreover, we may have a usage profile saying that on Fridays the demand for rooms is smaller than on other days. In this case the resource management system will prefer a time slot on Friday. Moreover, the resource may be constrained by cost attributes. If several availabilities exist, the system returns a list of alternative possibilities in an XML message:

```
availableResources(
  <Resource id="ec3-meetingRoom3">
    <VEVENT id="ec3-
meetingRoom3:offer_1">

<DTSTART>20070423T150000</DTSTART>
<DTEND>20070423T200000</DTEND>
    <VEVENT/>
    <VEVENT id="ec3-
meetingRoom3:offer_2">

<DTSTART>20070423T100000</DTSTART>

<DTEND>20070423T150000</DTEND>
    <VEVENT/>
    <Cost>20</Cost>
  </Resource>
  <Resource id="ec3-meetingRoom2">
    <VEVENT id="ec3-
meetingRoom2:offer_1">

<DTSTART>20070423T120000</DTSTART>

<DTEND>20070423T170000</DTEND>
      <Cost>25</Cost>
    <VEVENT/>
  </Resource>)
```

[1] We omit details required for web service administration and show only the arguments required for the functional aspect. The meaning of the temporal arguments is defined in iCalendar.

6.2. Reserving and Booking a Resource

The interface of reservation and booking Web services are very similar. The difference is in the applied business rules. There are two ways to reserve a resource. If we have queried for an available resource we have received an answer with available resources. We may use the "offer id" in the list of events to reserve (or book) this resource. We may also reserve a resource without querying the availability. In the first case the query looks as follows:

```
reserveResource(
  <Resource id="ec3-meetingRoom3">
    <VEVENT id="ec3-
meetingRoom3:offer_1">
      <Client id = "Juergen
Dorn"/> </VEVENT> </Resource>)

bookResource(
  <Resource id="ec3-meetingRoom3">
    <VEVENT id="ec3-
meetingRoom3:offer_1"/>
      <Client id = "Juergen
Dorn"/> </VEVENT> </Resource>)
```

The second possibility to reserve or to book is without the offer.

```
reserveResource(
  <Type>Meeting Room</Type>
  <Capacity>3</Capacity>
  <VEVENT>

<DTSTART>20070419T100000</DTSTART>
    <DTEND>20070423T200000</DTEND>
    <DURATION>PT5H</DURATION>
```

```
  </VEVENT>
  <Client id = "Juergen Dorn"/>)
bookResource(
  <Type>Meeting Room</Type>
  <Capacity>3</Capacity>
  <VEVENT>

<DTSTART>20070419T100000</DTSTART>
    <DTEND>20070423T200000</DTEND>
    <DURATION>PT5H</DURATION>
  </VEVENT>
  <Client id = "Juergen Dorn"/>)
```

For a reservation or a booking we receive a commitment if this reservation or booking is possible. The commitment contains of an event structure again.

```
reserveResource(
  <VEVENT>

<DTSTART>20070419T100000</DTSTART>
    <DTEND>20070419T150000</DTEND>
    <DURATION>PT5H</DURATION>
  </VEVENT>
  <Client id = "Juergen Dorn"/>)
bookResource(
  <Type>Meeting Room</Type>
  <Capacity>3</Capacity>
  <VEVENT>

<DTSTART>20070419T100000</DTSTART>
    <DTEND>20070423T200000</DTEND>
    <DURATION>PT5H</DURATION>
  </VEVENT>
  <Client id = "Juergen Dorn"/>)
```

If we want to cancel or modify a booking, we use the offer id again to identify the booking that shall be changed. The arguments of the message are similar to the other messages. Rules decide whether compensation fees are applied.

7. Distributed Planning

If we have a number of such resource management systems distributed in a certain physical region, we may organize a conference requiring several resources. Starting with a set of required resources, the planning system can query the resource management systems for fitting resources. In the beginning the exact date of the conference may be unknown. There will be several offers from different resource owners.

For each query q_i we may obtain a number of offers o_{i1} to o_{ik}. These offers will vary in different attributes. Price, location, quality and also the temporal attributes may be different. These attributes are in principle service level agreements (SLAs) [20]. For each offer the planning system computes a *utility* value u_{ik}. This value is determined by comparing for each attribute the appropriateness. This appropriateness is determined by the satisfaction of the constraint given implicitly or explicitly by the attribute. If a room with a capacity of 30 persons is queried, the offering of a room with a capacity of slightly larger capacity will lead to a good satisfaction value. The offering of a room with a capacity for only ten persons or a very large room would lead to bad satisfaction, typically. A room for 50 persons would achieve medium satisfaction. There exist default values for measuring the satisfaction, however, it is also possible to define explicit assignments. The satisfaction degree is a value between 0 and 1, where a 1 means full satisfaction.

The utility value for a whole offer is again a normalized value between 0 and 1, where a value of 1 is assigned to the best result. This normalized value is computed by taking the satisfaction value of each attribute and the weighting of these attributes. For example, capacity may be considered as more important than location.

We may now aggregate the best offered resources to a value for a whole solution (all rooms and other resources for a conference). However, there may be constraints between resources. For example, the distance of the location of all rooms should be as small as possible. Therefore, we must compute more utility values that describe the relationship between resources. Thus the best overall solution must not be the selection of the best offer for each query. We may combine resources in different combinations in order to find the optimal solution for the whole problem. This search is governed by an objective function that aggregates different query results.

In principle, the search is a search over the set of all possible combinations making the problem an intractable problem if the task is to find a optimal solution. One possibility to find a good solution in acceptable time is to apply iterative improvement techniques or so called neighborhood search techniques [21]. These techniques apply a certain operator to change an existing solution. With an explicitly specified optimization function the original and the derived solution are evaluated. If the new solution achieves a better evaluation, a new operator is applied to the new solution in order to try to improve further. To escape from local optima, also a new generated solution that is worse than the original may be used for further optimization. Techniques such as genetic algorithms, simulated

annealing or tabu search have different meta-heuristics to allow such a degradation of the optimization function. We have achieved best solutions with tabu search for scheduling problems [21].

8. Conclusions

We have described a service-oriented approach to manage physical resources. In this approach we identify business services and deploy these services on the technical layer as Web service. Two types of Web services are distinguished: simple resource management services and complex inter-organizational services. Simple Web services must be designed so that we can reuse them easily for different business processes. The interface consisting of service name as well as input and output arguments should be understandable and based on standards (e.g. definition of temporal arguments). For automatic planning based on the state-based approach, Web service input parameters should be interpreted as pre-conditions for execution of a Web service and output parameters should be seen as results. Further parameters should be available for trust management. Complex services can be composed from simple services by integration into business processes or by automatic composition. Temporal constraints and their management seems to be a generic task required here.

Which service (and with that which resource in our application) is determined at runtime enabling a flexible reaction on actual conditions in the environment. If the number of participants changes, the system can easily identify alternative resources. Or if a new room becomes available this could be used instead of another resource that was already scheduled. Of course, in the new evaluation a cancellation fee must be considered.

The service-oriented architecture also enables a simple extension for further functionalities. For example, in the application we have not taken into consideration geographical information. But for the case of a conference we should optimize the solution by using rooms not too far away from each other. Such geographical information can be integrated easily: If we have the address of the building, we can use a geo-locating Web service which is provided by different providers (e.g., the city of Vienna offers such a service). From this Web service we receive a coordinate and with simple triangulation we can extend the optimization function of our planning system. Another service required to be included is payment.

In our future work we will extend the system using semantics. With semantic reasoning, for example, classifying resources, one could extend the capability of such a resource management system [22].

9. References

[1] Chesbrough, H. and Spohrer, J. “A research manifesto for service science”, Communications of the ACM Vol. 49, No. 7, 2006.

[2] Margaria, T. and Steffen, B. “Service Engineering: Linking Business and IT”, IEEE Computer, October, 2006.

[3] Dietrich, B., “Resource Planning for Business Services”, Communications of the ACM Vol. 49, No. 7, 2006.

[4] Zeithaml, V. and Bitner, M.J., Services Marketing. McGraw-Hill, New York, 1996.

[5] O’Sullivan, J., Edmond D. and ter Hofstede, A. H. M. , “Service description: A survey of the general nature of services”, report FIT-TR-2003-02. 2002.

[6] Web Service Architectures, http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/, 2004.

[7] Parnas, D.L. On the Criteria to be used in Decomposing Systems into Modules, Communications of the ACM 15 (5), pp. 1053–1058, 1972.

[8] Krueger, C.W. Software Reuse, ACM Computing Surveys 24 (2), pp. 131–183, 1992.

[9] Cheatham, T.E. Jr. Reusability through program transformations, in Software reusability Part 1: Concepts and Models, Ted J. Biggerstaff and Alan J. Perlis (eds), pp. 321–335, 1989.

[10] Dorn, J. Grün, Ch., Werthner, H. and Zapletal, M. “A Survey of B2B Methodologies and Technologies: From Business Models towards Deployment Artifacts”, Proceedings of HICSS07, 2007.

[11] Dorn, J. Hrastnik, P. and Rainer, A. “Web Service Discovery and Composition for Virtual Enterprises”, International Journal of Web Services Research, Vol 4(1) pp. 23-39, 2007.

[12] Dorn, J. Hrastnik, P. and A. Rainer, “Conferences as Virtual Enterprises”, PRO-VE’04 5th IFIP Working Conference on Virtual Enterprises, Toulouse, August, 2004.

[13] Tsang, E., Foundations of Constraint Satisfaction, Academic Press, London and San Diego, 1993.

[14] Zweben and Fox (eds.), Intelligent Scheduling, Morgan Kaufmann, San Francisco, 1994.

[15] Allen, J. F. "Maintaining Knowledge about Temporal Intervals", *Communications of the ACM* **26** (11), 823–843, 1983.

[16] Hrastnik, P., "Execution of Business Processes Based on Web Services", Int. J. Electronic Business Vol. 2, No. 5, September-October 2004.

[17] Dorn, J. "The DÉJÀ VU Scheduling Class Library", in Fayad, Schmidt, and Johnson (eds.) Implementing Application Frameworks, Wiley, pp 521-540, 1996.

[18] SOAP Version 1.2, W3C Recommendation, http://www.w3.org/TR/SOAP/, 2003

[19] The Internet Engineering Task Force, Internet Calendaring and Scheduling Core Object, 1998 Specification (iCalendar) http://www.ietf.org/rfc/rfc2445.txt

[20] Masche, P., Mckee, P. and Mitchell, B., The Increasing Role of Service Level Agreements in B2B Systems, Proceedings of the International Conference on Web Information Systems, Setubal, Portugal, pp. 473-478, 2006.

[21] Dorn, J. , Girsch, M., Skele, G., and Slany, W. Comparison of Iterative Improvement Techniques for Schedule Optimization, European Journal on Operations Research, pp. 349-361, 1996.

[22] Sheth, A., Verma, K. and Gomadam, K. "Semantics to energize the full services spectrum", Communications of the ACM Vol. 49, No. 7, 2006.