Technical Challenges in Market-Driven Automated Service Provisioning

Anna Chmielowiec
Vrije Universiteit Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
ania@few.vu.nl

Guillaume Pierre
Vrije Universiteit Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
gpierre@cs.vu.nl

Jaap Gordijn
Vrije Universiteit Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
gordijn@cs.vu.nl

Maarten van Steen
Vrije Universiteit Amsterdam
De Boelelaan 1081a
1081 HV Amsterdam
The Netherlands
steen@cs.vu.nl

ABSTRACT

In today's practice, we see readily precomposed commercial service bundles, such as a spam-free email box, consisting of more elementary services like mail storage and a spam-filter. However, these bundles may be suboptimal from the customer's perspective in terms of price and/or the elementary services that constitute the bundle. It would be advantageous to the customer if a service bundle more closely fulfilled the customer's individual requirements, by selecting the most appropriate elementary services included in the bundle. Also, by obtaining the bundle from a consortium of suppliers, rather than just one single supplier, the elementary services of each supplier with the best cost/benefit ratio can be selected. To put this vision into reality, we need middleware facilitating the automated composition of multi-supplier bundles out of basic commercial services available online.

We take the stand that the business nature of commercial services imposes leading requirements on the technical design of the middleware. Most importantly, the middleware should be fair in the sense that no single supplier obtains a preferred position in terms of service selection to satisfy a specific customer need. Also, the middleware should be able to deal with alternative services as offered by many competing suppliers, not to speak about the combinatoric explosion, resulting from combining the available services into candidate services bundles. We present a list of problems to be solved to arrive at middleware for multi-supplier service selection, bundling and provisioning. Also, we review existing work, usable to build a fair and efficient middleware solution for commercial service provisioning.

Categories and Subject Descriptors

H.3.5 [**Online Information Services**]: Commercial services; H.4.0 [**Information Systems Applications**]: General; D.2.11 [**Software Architectures**]: Domain-specific architectures; C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Economics, Design

Keywords

service bundles, service provisioning, commercial services

1. INTRODUCTION

The ubiquity of the Web continues to grow, exemplified by its use as a channel for an increasing number of commercial services. We have now reached a point where new services can be relatively easily facilitated by composition and bundling of basic services into more complex ones. In fact, the coinage of the term Web 2.0 brings to the limelight the idea of users actively contributing to the content of the Web. Users are encouraged to add and modify the information, associate metadata to the content and change the way content is displayed. As a consequence, users have an impact on the shape of the Internet and they can adjust it to their own expectations.

Illustrative in this context are mashups. The term mashup refers to a web application that uses data retrieved from other parties (for example, through the respective web service APIs) to produce a new service. An example of this mashup technology is the combination of cartographic data from Google Maps and real-estate rental/purchase information into a location-based search tool [16].

However, service composition and bundling is still in hands of (commercial) service providers, mainly because the available techniques are simply too complex to be used by end users. For example, creating mashups is not trivial and requires at least basic web programming experience. Moreover, if there is a need to change the source of the data used in the mashup, the modifications caused by the new way of data retrieval (i.e. different API) have to be done by hand.

In today's practice, we therefore see readily precomposed service bundles, which are offered by a single, commercial, service provider. This 'single service provider' approach is however not in line with placing the Web in the hands of end users, and moreover, may not be in their interest. For instance, web hosting services are often sold in combination with e-mail and database services. Although a user is saved the burden of configuration and interoperability management between these services, he is stuck with what is being offered, which may not be the combination of basic services that was actually needed nor at an optimal price.

A contribution of this paper is that we treat *commercial* services and *web* services as different citizens. We understand commercial services as *economic* activities of mostly intangible nature [15] and in our research we restrain ourselves to the commercial services that can be provisioned (almost) entirely online. In contrast, web services are *technical* software components that can be invoked over the Internet. Obviously, commercial services can be implemented by means of web services, but they are not the same.

In our research, we are seeking a solution to facilitate automated composition of bundles out of commercial basic services provided by various parties. The main requirement is that composition should optimize the cost/quality ratio for an end user. We take the stand that much of the technology needed for automated composition is readily available, yet that a market-driven approach will put a specific demand on the technical design and implementation of the middleware that is needed as the "glue" between customers and service providers. As such, middleware designers for Web-based service provisioning may be confronted with hard technical challenges that need to be resolved in order for a next-generation Web to succeed. Our main contribution in this paper is identifying and presenting these challenges, and in doing so, illustrate the impact that a market-driven approach can have on the design of middleware for service provisioning.

2. THE VIRTUAL ISP SCENARIO

To illustrate the problem of compositing commercial service bundles, consider the following scenario. Suppose a customer wants to buy and use an e-mail service. The obvious solution is to obtain the service from a *single* Internet Service Provider (ISP). However, this may have several drawbacks for the customer. Usually, from a commercial perspective, a provider *bundles* a service with other services. Bundling means that services can not be obtained in their own right, but can only be bought in combination with other services. For instance, an e-mail service may be bundled with Internet access. Suppliers do so, because they can generate more profit; they believe that they can sell more service units as bundles, than as separate units [18].

However, if the customer already has Internet access, the provided bundle {e-mail, access} is unnecessary broad for the customer, resulting in paying for an undesired service (here: access). Even in the case when providers offer an e-mail service separately, the customer may obtain a sub-optimal service. There are a few reasons for this. First, an e-mail service can still be decomposed into smaller, elementary services. For instance, an e-mail service of a specific provider may consist of the following sub services: { ingoing SMTP mail, outgoing SMTP e-mail, POP/IMAP mailbox, spam filter, domain name hosting }. Again, the bundle may be too broad, as the customer may prefer not to have a spam filter. Apparently, services can be more complex than they are at first sight. On the other hand, the offered service may be too narrow, as the customer may want to have mailing lists too. Services are also sub-optimal, if some of the underlying elementary services in the bundle produce, as perceived by the customer, insufficient quality. For instance, the customer may be satisfied with the realized quality of service for the ingoing and outgoing e-mail services, but may be dissatisfied with the quality of the offered mailbox (e.g. only a size of 20 MB offered, rather than a desired 10 GB).

What a customer actually would like to do, is to build his own, dedicated service provider for e-mail, such that this virtual provider satisfies the customer's requirements as close as possible. Indeed, some customers are doing so; they *manually* select an ingoing and outgoing mail provider, mailbox provider, and so on. This, however, requires a lot of work, such as searching for providers, comparing prices, and placing the orders. It requires also specific knowledge, since the customer must be able to judge whether the selected services work together seamlessly, and configure the services such that they interoperate. Finally, in case there is a service disruption, the customer himself must trace down the failing elementary service, and solve the problem. All these tasks are not feasible for the average Internet user of today, and most likely also not of the future.

We envision a future scenario, in which a customer can state his requirements for a particular service bundle in a computer-processable way. The customer is then presented a list of alternative service bundles, including the belonging sacrifices (usually money). Then, the customer *selects* a service bundle, which is *provisioned* by the supplier(s) offering the bundle. Once provisioned, the service bundle is continuously *monitored*, to assess whether the requirements of the customer are indeed satisfied all the time. In case the requirements are not satisfied, the service bundle is re-provisioned, e.g. by replacing a supplier. In case such changes require a higher sacrifice (price) from the customer, the customer is contacted. Then the customer is given the opportunity to re-select the service bundle.

From a business perspective, we distinguish two scenario types to accomplish custom-made service bundles, (1) the supplier-hierarchy scenario and the (2) supplier-market scenario. These scenarios stem from the notion that constellations of enterprises can be organized as hierarchies or as markets with respect to decision making [14].

The supplier-hierarchy scenario supposes that all suppliers are organized in *hierarchies.* A supplier A that is hierarchically higher placed than a supplier B, C, D, . . . , is responsible for selecting services of these lower placed actors, and is composing a service bundle. Suppliers that are at the root of a hierarchy obtain the requirements from the customer, and search for relevant available services that can be provided by themselves, or by suppliers lower in the hierarchy. The hierarchy also states monitoring responsibilities; hierarchically higher placed actors monitor service provisioning of the lower placed actors in the hierarchy. The need for this hierarchical service organization can be observed already in practice, for instance in [4], a large telecommunication operator explains that this mechanism is their future service offering scenario.

In a hierarchical service provisioning model, one supplier has strict control over the hierarchically lower placed actors in terms of service selection, provisioning, and monitoring. The advantage is that the top-level supplier can ensure well-integrated service bundles. The drawback for the customer is, however, that the customer is not free to select the elementary services in the bundle. At best, the customer can select the elementary services that are in the hierarchy of a considered top-level supplier, but usually the top-level supplier selects a service bundle himself.

Therefore, we distinguish a second scenario, the supplier-market scenario, which is also the focus of this paper. Suppliers are now organized in a market, rather than in a hierarchy, and are considered as equal when it comes to service selection, provisioning, and monitoring. A customer provides his service requirements to the supplier-*web*, so not to an *individual* supplier anymore. The market proposes alternative service bundles, including the required customer sacrifices, the market provisions the service bundle, and the market monitors the provisioned bundle for requirements satisfaction. For the customer, the advantage of this scenario is that no single supplier determines the service bundle composition anymore, which results in better optimized (in terms of requirement satisfaction) bundles for the customer. However, the scenario obviously requires more capabilities of the network (in terms of service selection, provisioning, and monitoring), whereas the hierarchical scenario supposes these capabilities only at the level of the single supplier.

In the following text, we explore the research problems that occur, if we have to select, provision, and monitor the requested multi-supplier service bundle, in case of the supplier-market scenario.

3. RESEARCH PROBLEMS

3.1 Assumptions

Customer's request. We assume that the customer needs are stated in machine-understandable way. For example, the request is an XML file and contains the list of elementary services and their required properties. How to state the customer need, we have studied in [10], and how to represent the satisfying service bundle, we have explained in [1].

Service catalogs at suppliers. Moreover each supplier in the system we propose has a catalog of the elementary services it offers and determines whether any of the requested services fits the description of some elementary service from that catalog (see also [1]).

Communication among suppliers. We assume that all suppliers in our network use the same protocol for the communication and they understand the semantics of the messages being sent. Moreover, we assume at this point that all nodes (suppliers) adhere to the rules.

3.2 Composite service life-cycle

The focus in this paper is on the market-driven scenario, as outlined in section 2. Choosing the appropriate architecture for the market-driven automated service provisioning system cannot be based only on solving purely technical problems, as the system should also fulfill the requirements imposed by its *business* nature.

To identify business requirements, we distinguish four main phases of the composite service life-cycle: *discovery*, *negotiation*, *provisioning*, *monitoring*.

- *Discovery* takes place right after the customer issues a query for a composite service to the network. In this phase, the suppliers of basic services are found and the providers of different service types are grouped into the sets, each of which can potentially provide the composite service requested by the customer.

- In the *negotiation* phase, suppliers inside each of the sets try to come to an agreement upon the price and terms of the offer. If they reach a consensus they send the offer to the customer.

- When customer selects one of the offers returned by the suppliers web, the instance of the composite service from that offer needs to be created. This happens in the *provisioning* phase.

- Once a composite service is provisioned, it is continuously *monitored* to control that customer's requirements are indeed satisfied. In case these requirements are not fulfilled, the service bundle is modified, for example by exchanging a supplier of one of the elementary services. In effect, the service is re-provisioned.

3.3 Discovery

Fair discovery. An important requirement concerns the fairness of the system. None of the providers should be privileged due to the way the system is implemented. In particular, all suppliers should have equal chances to find the customer's request and to take part in creating the composite service proposal. This entails that the fairness of the system should be ensured from the earliest steps of service provisioning life-cycle and be guaranteed already in the request dissemination and service discovery phase. For instance, if a POP mailbox is required by the customer, possible service providers offering the mailbox as a service should be discovered. The system should guarantee that each supplier offering a POP mailbox has an equal chance to be discovered, and that no bias is given to a specific supplier (as would be possible in case of a hierarchical discovery scenario).

But ensuring fairness of the system is a challenging task. We can observe in the example of ICANN how easily the trust in the fairness can be put in jeopardy when an external party is involved in the control of the system. One of the responsibilities of the Internet Corporation for Assigned Names and Numbers is the management of the generic and country code Top-Level Domain name system. Recently, there have been concerns raised whether ICANN is abusing its dominant (monopolistic) position with regard to domain-name registrations. For example, it came into question [7] if due to the fact that ICANN has granted the exclusive right to the .com and .net name registries to the largest domain name registry company (VeriSign), the corporation has engaged in infringement of European free trade laws.

Thus, the architecture facilitating request distribution and service discovery has to be carefully chosen. If we decided to use for instance a publish/subscribe system, we would make sure that its technical middleware design meets fairness requirements. For example, if we consider publish/subscribe systems like Siena [6], which use a network of servers to distribute messages, we would immediately face the problems similar to the ones of ICANN. If the infrastructure of such a publish/subscribe system is owned by some third party it instantly becomes a weak point in ensuring the fairness of service discovery. Therefore, some means would be needed to decentralize the ownership of the middleware.

Efficient discovery. When the supplier can provide a requested elementary service, the next step is to find the remaining elementary service providers, to jointly create an offer. If the publish/subscribe system is used for customer requests distribution, its infrastructure can be used as well for this task in a way seamless to the customer.

Yet, in the system where thousands of suppliers exist that provide the same type of the elementary service, allowing every A-service supplier to create an offer with every B-service supplier when composite service A-B is requested, would result in a state space explosion of all possible combinations of elementary services A and B. Allowing all the possible bundles to be offered to the customer is neither rational in terms of workload imposed on the network nor indifferent with regard to supplier's resources. But what is more important, it is not advantageous for both suppliers and customers from the business perspective. Service providers have no incentive to create all possible bundle offers. Instead they would prefer to create only the ones that have the highest chances to be chosen by the customer. Likewise, customers are not willing to look through all the possible offers but they would prefer to get only offers that would best fit their needs (just as it happens with our expectations toward search engines).

Thus, there is a need for the elementary service suppliers to narrow the number of the providers they are willing to cooperate with in creation of offers. One of possible solutions is to use heuristics based on (1) history of previous choices, (2) reputation of the suppliers, and (3) technical and business constraints.

The real technical issue is that we need to ensure that local rules as set out by suppliers for selecting other suppliers will assure that (1) enough propositions are formed for the customer to choose from, and (2) the total number of potential propositions is sufficiently limited to warrant acceptable performance. In other words, the choice of local rules should lead to enough, yet not too many propositions.

3.4 Negotiation

After the suppliers have made the preliminary choice for their partners, these potential suppliers negotiate to create a bundled offer. As the discovery step already has reduced the number of relevant bundles, we do not consider the number of concurrent negotiations as a significant problem.

Multi-issue negotiation. However, each bundle may require a number of suppliers. These suppliers have to agree on (1) the price of the entire bundle as well as on (2) the distribution of the profits among themselves. This requires multi-party multi-issue negotiation protocols. Moreover, the negotiating suppliers have to be aware that although they want to maximize their profits they cannot charge the customer too much because he will simply choose a cheaper offer placed by the competition. This gives the suppliers an incentive to offer to the customer a composite service whose quality/cost ratio would be satisfactory. Also, the suppliers have to take into account that many other suppliers are also trying to come with a competitive offering to the customers, as the system is a market, and not a hierarchy of suppliers.

Non-disclosure of business models. While negotiating, none of the suppliers would be happy to reveal their pricing (utility) functions because that would mean disclosing their business models. Therefore, an intermediary party is not an option. Designing this kind of negotiations is technically demanding as they are orders of magnitude more complex than ordinary bilateral single-issue negotiations. Yet multiparty multi-issue negotiations have recently gained attention in the research community, (see, e.g., [9]).

Apart from choosing a negotiation framework we have to make other decisions such as whether a service supplier should be able to take part in more than one negotiation for the same customer need, and how to prevent the customers (and suppliers as well) from collusions, to guarantee a fair process. These are again requirements that come directly from the business environment and that are going to impose challenges on the technical design.

Decision taking. When the suppliers reach a consensus, they create an offer that is sent to the customer. It is not important who sends the offer, each supplier participating in a bundle can do so, as long as the offering is sent to the customer. The detailed specification of the composite service along with the price and other contract details (e.g. lock-in time) is digitally signed by all suppliers. The customer selects then a specific bundle, which will be provisioned.

3.5 Provisioning

We do not foresee specific hard problems in provisioning the service bundle. Research problems do, however, appear once a service bundle needs to be re-provisioned (see section 3.6), e.g. in case the quality of the provisioned service bundle is disappointing. From the business perspective provisioning of the composite service seems to be the least demanding. All business decisions have been already made, the service suppliers selected and offer agreed upon. Of course, it does not imply by any means that it is not a technically complex task. The system needs to assure that the customer gets his composite service delivered. The suppliers that sent the selected offer need to properly configure their services for that customer such that the basic services interoperate correctly. And once the instance of the requested service is created, the customer has to be informed that his new service is ready to use. The possible scenario that realizes this can be the following. First, the customer sends the selected offer digitally signed by himself to all suppliers that provide the elementary services from the offer. Then, after the supplier receives the offer signed by the customer, it creates the instance of the ordered elementary service. After configuration of the elementary service the supplier sends an 'OK' message back to the customer. When the customer collects the 'OK' messages from all suppliers the service is ready to be used. In effect, provisioning can be established by means of a more-or-less standard two-phase commit protocol.

3.6 Monitoring And Re-Provisioning

Fair monitoring. Once the selected service bundle is provisioned, the bundle should be continuously monitored to assess that the customer's requirements which fulfillment the suppliers obliged to are indeed satisfied. As with service discovery (section 3.3), the system should guarantee that no single supplier can bias the monitoring and its results. The monitoring results are used to continuously assess whether the agreed quality of service is met, and if not, a re-provisioning (and if needed, a re-discovery and renegotiation process) is triggered. Note that monitoring the quality of the offered service can technically be efficiently done by the customer. Of course, we need to ensure that

(1) if the QoS drops below an acceptable point, the customer can (possibly automatically) break the contract, but also that (2) customers should be prevented from falsely accusing providers of low quality. There are many potential solutions to these problems, ranging from adopting trusted third parties to handle accusations (as we have explored in GDN [2]), to the more elaborate construction of incentive mechanisms originating from economic game theory (see, e.g., [13]).

Re-provisioning without service-disruption. In case one of the suppliers fails to deliver the basic services as promised, the supplier can be replaced by a different one that offers the same or similar basic service. Sometimes this may result even in a need to replace all suppliers of the basic services from the bundle for new ones. This re-provisioning must be done in a way seamless from the point of view of the customer although the customer may need to be notified to approve the change, especially when the price or the properties of the bundle have altered. Suppose for instance that the customer is not satisfied with the size of its incoming mailbox, and decides, after a re-discovery and re-negotiation process, to switch to another mailbox provider. Also, the customer leaves the DNS service (with the MX-record, needed to locate the mailbox) at the same provider, since the customer is satisfied with that service. Then, at least for a while, the old mailbox provider should forward received e-mails for the customer to the new mailbox provider, since the DNS-MX record is cached in local DNS servers, and therefore e-mail can still arrive at the old mailbox provider. Such additional (temporary and payed) forward services are need to ensure continuous service provisioning, and should be part of the re-provisioning process.

Continuous re-provisioning. As we want to provide automated support for service discovery, negotiation, provisioning, monitoring, and re-provisioning, a logical step of the customer would be to continuously search for better service bundles (e.g. a cheaper price, or a better quality of service). If all customers are doing so, a substantial amount of service discovery, negotiation, and provisioning processes will always be running. A problem is how to prevent such behavior. To do so, we rely on a business-oriented solution, namely to put a price on service re-discovery and re-negotiation, or to include the right to re-discover and re-negotiate as a valuable feature of the service bundle.

4. RELATED WORK

In our work, we focus on market-driven mechanisms to (re)configure multi-supplier *commercial* bundles, which can be provisioned *online*. Therefore, we consider *computer science* oriented research on *semantic* web & IT services relevant to us, but also business oriented research on commercial services (e.g. service marketing, pricing and bundling).

In business oriented research, services are considered as a kind of products, economic activities that often produce intangible outcomes and that are offered to customers by service providers. Bundling of the services, understood in this way, can come, according to [18], in two different flavors: as a *product* bundling or as a *price* bundling. Product bundling stands for "the integration and sale of two or more separate products at any price". We have already discussed computational tool support for this in [1]. Price bundling is used to describe "the sale of two or more separate products as a package at a discount, without any integration of the products". In [5] a formal approach is presented for solving the problem of maximization of the buyer's overall satisfaction in on-line bundle purchasing. To solve this problem, a manifold of satisfactory services combinations needs to be evaluated and a decision has to be taken which bundle should be purchased in order to maximize the utility. However, the work assumes that the alternative bundles have been already composed and the only thing needed is selecting best of them based on price. This leaves us with an open field for designing a middleware that would facilitate automated creation of these bundles.

Yet despite the fact that customized service bundling have been already studied in business literature (see, e.g., [8], [12], [15]), this research is described to high extent in natural language and lacks conceptualization and formalization, while our aim is to create a technical infrastructure that would computationally support automated service bundling and on-line provisioning of these bundles. This kind of infrastructure should enable service suppliers to offer their services, and customers should be able to use this infrastructure to find service bundles that they need. And apart from the facilitation of these functionalities such an infrastructure should above all be able to give its users (both suppliers and customers) guarantees of the compliance to fair-trade rules.

To develop market-driven (re)configuration mechanisms for multi-supplier, commercial, and online service bundles, the web service community is important, although we should be cautious given the different flavor of technical and commercial services. For example, the research on infrastructure for discovery of web services focuses mostly on solving the problem of scalability through decentralization of the registries. METEOR-S [19] and PYRAMID-S [17], which are approaches based on semantic annotations, present scalable peer-to-peer infrastructures for the publication and discovery of services. Each registry is associated with a particular domain and contains descriptions of the Web services that belong to this domain. DIRE [3] is another example of a scalable infrastructure for managing communication between registries but it exploits the publish/subscribe paradigm to provide service publication and discovery. None of this solutions discusses the issue of ownership of the registries and the problem of possible bias in the search results caused by self-interest of some registry owner.

Besides the problem of the fairness of discovery, we have also mentioned the problem of state space explosion of possible service bundles. In [11], the requested and offered services are expressed in the terms of the desired *effects* they should provide and every effect (and transitively every service) may contain a set of parameters by which it can be configured. For example, domain name registration is an effect and can be parameterized with the specific domain name. Naturally a request for a complex service would contain multiple effects and each of these effects may be provided by numerous offered services that can be configured by many parameters. To limit the number of service combinations considered to fulfill the request consisting of multiple effects, the approach of a *value propagation* semantics is used. The main idea of the solution is to order the effects and then fill the parameters according to this ordering optimizing the solution locally. As a result of interdependence among the requested effects, choices for the parameters contained in the preceding effects might restrict the possible choices for the parameters of the succeeding effects. Therefore, the solution may not be globally optimal anymore. The solution proposed in [11] is centralized and requires access to the complete knowledge about available services and their possible configurations. Thus, the proposed algorithm may not be suitable for the system in which suppliers may be unwilling to disclose full information about their services and where the bundling of services occurs in a decentralized manner as we have described it.

The Semantic Web Services field provides also research on Web services re-composition, which we consider as services re-provisioning. For example, [3] presents a framework that facilitates the deployment of the Web services compositions and allows their re-configuration at runtime. Apart form the BPEL specification, it uses a set of rules, constraints and preferences to navigate the dynamic selection and exchange of the services. However, this work does not explicitly consider services as commercial entities and takes no account of more complex market-driven bundling dependencies between the services such as the one we presented in 3.6.

5. CONCLUSIONS

In this paper we outlined the design issues of the middleware for facilitating automated service provisioning. Our main focus was on identification of the technical challenges that stem from the business nature of service bundling. We find that providing a solution for these technical problems is a necessity for the service provisioning system to be successfully adopted by service suppliers and service users communities.

In our future work we plan to deliver solutions to the presented technical problems. This would enable the creation of a middleware facilitating automated service provisioning that fulfills the market-driven requirements.

6. REFERENCES

[1] H. Akkermans, Z. Baida, and J. Gordijn. Value webs: Ontology-based bundling of real-world services. *IEEE Intelligent Systems*, 19(44):23–32, July 2004.

[2] A. Bakker, M. V. Steen, and A. S. Tanenbaum. A law-abiding peer-to-peer network for free-software distribution. In *NCA '01: Proceedings of the IEEE International Symposium on Network Computing and Applications (NCA'01)*, pages 60–67, Washington, DC, USA, 2001. IEEE Computer Society.

[3] L. Baresi, E. D. Nitto, C. Ghezzi, and S. Guinea. A framework for the deployment of adaptable web service compositions. *Service Oriented Computing and Applications*, 1(1):75–91, 2007.

[4] H. Bastiaansen and P. Hermans. Managing agility through service orientation in an open telecommunication value chain. *IEEE Communications Magazine*, 44(10):86–95, October 2006.

[5] S. Buffett and B. Spencer. A decision procedure for bundle purchasing with incomplete information on future prices. *International Journal of Electronic Commerce*, 8(4):131–144, 2004.

[6] A. Carzaniga, D. Rosenblum, and A. Wolf. Design and evaluation of a wide-area event notification service. *ACM Trans. Comput. Syst.*, 19(3):332–383, 2001.

[7] B. N. Dunn. Icann's levy from price increases imposed on europeans (h-0126/07). http://www.europarl.europa.eu/sides/getDoc.do?type=CRE&reference=20070315&secondRef=ANN-01&language=EN&detail=H-2007-0126&query=QUESTION, 15 March 2007. European Parliament, Strasbourg.

[8] C. Grönroos. *Service Management and Marketing: A Customer Relationship Management Approach, 2nd edition.* John Wiley & Sons, Chichester, UK, 2000.

[9] A. Kardan and H. Janzadeh. A multi-issue negotiation mechanism with interdependent negotiation issues. In *Proceedings of Second International Conference on the Digital Society (ICDS 2008)*, pages 55–59, Los Alamitos, CA, USA, 2008. IEEE Computer Society.

[10] S. Kinderen, de and J. Gordijn. Reasoning about substitute choices and preference ordering in e-services. In Z. Bellahsène and M. Léonard, editors, *Proceedings of the 20th International Conference on Advanced Information Systems Engineering*, volume 5074 of *Lecture Notes in Computer Science*, pages 390–404. Springer Berlin (D), 2008.

[11] U. Küster, B. König-Ries, M. Klein, and M. Stern. Diane: A matchmaking-centered framework for automated service discovery, composition, binding, and invocation on the web. *International Journal of Electronic Commerce*, 12(2):42–67, 2007.

[12] C. Lovelock. *Services Marketing, People, Technology, Strategy, 4th edition.* Prentice Hall, Englewood Cliffs, NJ, 2001.

[13] R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Experiences Applying Game Theory to System Design. In *ACM SIGCOMM Workshop on Practice and Theory of Incentives in Networked Systems*, pages 183–190, New York, NY, USA, 2004. ACM.

[14] T. W. Malone, J. Yates, and R. I. Benjamin. Electronic markets and electronic hierarchies. *Communinications of the ACM*, 30(6):484–497, 1987.

[15] R. Normann. *Service Management: Strategy and Leadership in Service Business.* John Wiley & Sons, Chichester, UK, 3rd edition, 2001.

[16] T. O'Reilly. What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software. http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html, September 2005.

[17] T. Pilioura, G.-D. Kapos, and A. Tsalgatidou. Pyramid-s: A scalable infrastructure for semantic web service publication and discovery. In *RIDE '04: Proceedings of the 14th International Workshop on Research Issues on Data Engineering: Web Services for E-Commerce and E-Government Applications (RIDE'04)*, pages 15–22, Washington, DC, USA, 2004. IEEE Computer Society.

[18] S. Stremersch and G. J. Tellis. Strategic bundling of products and prices: A new synthesis for marketing. *Journal of Marketing*, 66(January):55–72, 2002.

[19] K. Verma, K. Sivashanmugam, A. Sheth, A. Patil, S. Oundhakar, and J. Miller. Meteor-s wsdi: A scalable p2p infrastructure of registries for semantic publication and discovery of web services. *Inf. Technol. and Management*, 6(1):17–39, 2005.