Needs-Driven Bundling of Hosted ICT Services

Jaap Gordijn[1], Floris de Haan[2], Sybren de Kinderen[3], and Hans Akkermans[1]

[1] VUA Amsterdam, The Netherlands
{gordijn,elly}@cs.vu.nl
[2] OGD, The Netherlands
frhaan@gmail.com
[3] CRP Henri Tudor, Luxembourg
sybren.dekinderen@tudor.lu

Abstract. Increasingly ICT (Information & Communication Technology) services become *commercial* services. For example, an Internet Service Provider (ISP) offers email, web browsing and content hosting as commercial services. In this paper we present an approach, e^3*service* , to semi-automatically generate such services, satisfying a stated complex customer need. The e^3*service* approach elicits the customer need, the consequences satisfying the need, and services satisfying the need. We show how e^3*service* works in practice using a running, industry strength, case study.

Keywords: service, need, bundling.

1 Introduction

Today's economy increasingly becomes a *service* economy. We consider a service as having an intangible nature [11], a processual nature [5], and producing valueable outcomes [9]. So, we take mainly a *commercial* perspective on 'service'.

The focus in this paper is on commercial *ICT* (Information & Communication Technology)-services. ICT-services are just like normal commercial services; only additionally, ICT-services can be provisioned online. Examples from our case study partner include virtual desktops, accessible via thin-clients, and backup services. Since ICT services are provisioned online, it is therefore important that ordering of these ICT-services can also be done online.

Increasingly, such ICT services are sold as service *bundles*. Service bundles consist of more elementary services. These elementary services may be offered by multiple suppliers. Usually, suppliers bundle services to satisfy more complex customer needs. As an example, consider an Internet Service Provider (ISP). A bundle of an ISP often contains a connectivity service (e.g. for web surfing), an emailbox service, and a webpage hosting service. Note that all these elementary services may be offered by multiple ISPs (suppliers).

To support an online ordering process for ICT services we have developed the e^3*service* ontology. This ontology allows for reasoning about customer needs, services and service bundles. The e^3*service* ontology is capable of representing a service catalogue both from a customer and supplier perspective. The customer perspective catalogue contains concepts such as ‘customer need’ and ‘consequence’ (consequence of satisfying a need). The supplier perspective catalogue entails concepts such as ‘service’ and also ‘consequence’. The notion of consequence is used to connect the supplier perspective with customer perspective. The e^3*service* ontology has also reasoning capabilities as it (1) can semi-automatically derive a set of consequences as a result of satisfying a customer need, (2) can match the found consequences with available service bundles, and (3) can extend service bundles with additional services which also might be of value for the customer.

In sum, the contribution of this paper is that we propose the e^3*service* ontology for reasoning about customer needs, consequences, and service bundles. Additionally, we show how e^3*service* works in a real-life case study.

Finally, it is important to understand that the e^3*service* ontology is different from ontologies in the field of web-services, (e.g. WSMO [8]). The e^3*service* ontology is about *commercial* services, whereas web-services provide a platform to solve interoperability and orchestration between *software components.*

This paper is organized as follows. In Sect. 2 we introduce the running case study. Sect. 3 presents the e^3*service* ontology. In Sect. 4 we show how to reason with the e^3*service* ontology about customer needs and service bundles. Finally, in Sect. 5 we present lessons learned and in Sect. 6 our conclusions.

2 The OGD Case Study

OGD is a Dutch ICT service provider (+/- 750 employees) that has recently started to provide hosted ICT-services. Currently, OGD offers Historium, an online back-up service and Officium a virtual workspace for a client that is accessed remotely through a thin client environment (called ‘hosted desktop’).

Offering hosted services is relatively new to OGD. As a result, OGD currently lacks a coherent idea of the benefits of their service offerings, how individual services are interrelated, and what customer needs their services satisfy. This knowledge is either fragmented throughout OGD, or unknown altogether. For many OGD-employees, this can be problematic. Junior account managers may have trouble in stating why a service is interesting for a customer, while the marketing department may have trouble describing the service in offering texts.

Therefore, OGD wants to create a service catalogue that provides a uniform idea of (1) the benefits of the individual services from OGD, (2) relations between these services and (3) the customer needs these services satisfy.

In addition, OGD is interested in the e^3*service* software ontology to (1) train junior account managers and (2) structure the dialogue for personnel selling services by phone.

3 The e^3*service* Ontology

To represent the service catalogue of an enterprise, we utilize the e^3*service* ontology. This ontology takes two perspectives on services: (1) the supplier perspective, and (2) the customer perpective.

3.1 The Supplier Perspective

The supplier perspective of the e^3*service* ontology is largely based on [2,3]. Fig. 1 shows the e^3*service* ontology expressed as a high-level UML class diagram. Fig. 2 shows a sample of the supplier perspective service catalogue (cf. the e^3*service* ontology). Due to lack of space, we only discuss the most important parts of the ontology in detail.

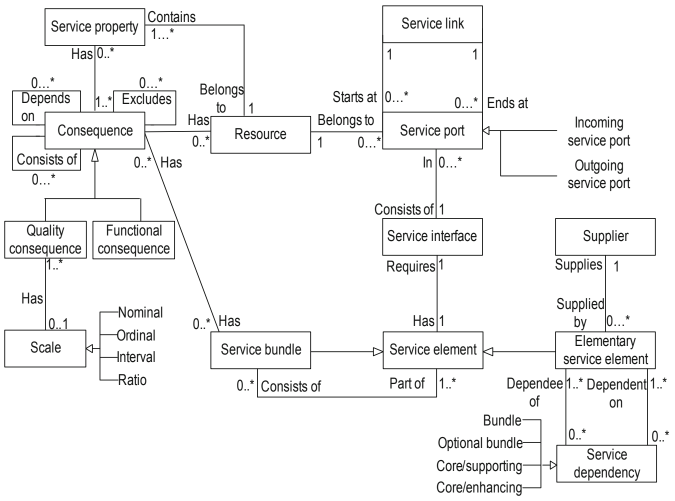

Fig. 1. The e^3*service* supplier perspective ontology

Service element. A service element is a (composed) activity that provides the service. Service elements can be *elementary*; then they can not decomposed further, or service elements can be a service *bundle*; then the bundle is composed of other service elements. Service elements are provisioned by a *supplier.* Service elements have service *interfaces*, which group service *ports.* Service ports provide or request *resources*, which are units of service delivery that can be provisioned in their own right and in a commercially feasible way. Service ports can be connected to each other via service *links.* Resources have service *properties.* As properties can not provisioned independently, they are part of resources.

Case study: For the OGD case (see Fig. 2), service elements include two e-mailing solutions: (1) hosted Exchange basic, a solution offering basic e-mailing capabilities such as sending and receiving e-mail through a standard e-mail client and (2) hosted Exchange complete, a solution that offers the same basic e-mail capabilities, only then supplemented with extra features such as e-mail access via the mobile phone, a mailbox that can be shared with others, etc.

Service dependencies. Next we model dependencies between individual services. Various kind of dependencies between services may exist, cf. [2,3] (where S_1 and S_2 denote service elements):

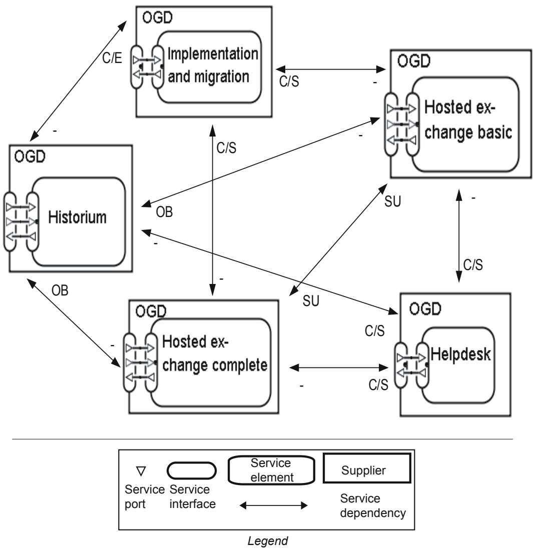

Fig. 2. Sample of the supplier perspective service catalogue

- S_1 is in a *Bundled* dependency with S_2 if S_1 is not provided separately from S_2 for *commercial* reasons. In literature, this is refered to as pure product bundling [12]. Notation: BU.
- S_1 is in a *Core/Supporting* dependency with S_2 if S_1 cannot be provided (for technical or legal reasons) without also providing S_2. The supporting service can be supplied by the same supplier as the supplier of the core service, but another case is that the supporting service is supplied by someone else. Notation: C/S.
- S_1 is in a *Core/Enhancing* relationship with S_2 if (1) S_2 possibly adds value to S_1, (2) acquisition of S_1 is obligatory for acquisition of S_2 and (3) S_1 can be acquired separately from S_2. Notation: C/E.
- S_1 is in a *Optionally Bundled* relationship with S_2 when (1) S_2 possibly adds value to S_1 and (2) S_1 and S_2 can be acquired separately. Note here that, as opposed to the C/E relationship, S_1 does not have to be acquired before S_2 can be acquired. Notation: OB.
- S_1 *excludes* S_2 if the supplier of S_1 prevents the customer to consume S_2, for example because S_2 is offered by a competitor, or because joint consumption is legally prohibited. Notation: EX.

Case study: The following service dependencies are of interest:

- Helpdesk support is a standard supporting service for the basic services.
- Implementation and migration is a standard supporting service for both hosted Exchange service offerings.
- Implementation and migration is an *enhancing* service for the online back up service historium and as such, not included as a standard supporting service.

The reason for this is twofold: (1) installation is relatively straightforward, requiring a single installation of a software tool on a server, and (2) OGD deems historium not to be as 'business critical' as an e-mail service.

– Hosted Exchange complete (the hosted e-mail solution from OGD) is in an OB relationship with Historium (the online backup service from OGD) to indicate that account managers often offer functionality of one of these services in combination with functionality of the other service. However, both services can also be acquired separately.
– Hosted Exchange complete is in an C/E relationship with the service Exchange brick-level. This indicates that account managers often offer functionalities provided by these services in combination, but also that they never sell Exchange brick-level separate from hosted Exchange complete. This is because the Exchange brick level is a component that allows for making backups of individual mailboxes and as such, only makes sense in combination with a basic e-mail service.

Generate possible service bundles. Based on the individual services and dependencies that exist between these services, we generate all possible service bundles. These generated bundles together form the service catalogue of OGD. For example: From the dependency OB(Hosted Exchange complete, Historium) we generate two possible bundles: {Historium (OGD)} and {Historium (OGD), Hosted Exchange complete (OGD)}. A sample of the pregenerated bundles of hosting services can be found in Table 1.

Service consequence. Services may have consequences. A consequence is anything that results from consuming (a combination of) valuable service properties offered by resources of a service (see [10]). There exist several supply-side types of consequences (functional and quality) and relations between these consequences. Since consequences are used in both the customer and supplier perspective of e^3*service* , they form the glue between both ontological perspectives.

Case study: For the OGD case (see Fig. 2), we identify the consequences from the individual services. For a sample of the identified consequences, see Table 1. First, we find functional consequences, such as 'send and receive e-mail' for the hosted Exchange services, and 'generic backup capability' for the Historium service. Second, we find quality consequences, such as 'e-mail access by phone' and 'e-mail access via web browser' for the service Hosted Exchange complete and 'within 4 hours' and 'within 7 hours' for the service Historium.

Then we group quality consequences using the concept of a *scale*. We group 'e-mail access by phone' and 'e-mail access via web browser' under the *nominal* scale 'e-mail access method' because the preference ordering depends on the customer. We group 'within 4 hours' and 'within 7 hours' under the *ordinal* scale 'response time', because this indicates the different response times in recovering a lost dataset. If someone considers response time important, the shorter response time is always preferred over the longer one.

Table 1. Sample of generated service bundles and their respective functional consequences

Service bundle	Functional consequences
Hosted Exchange basic	Access to email, contact support, don't receive unwanted email, don't receive viruses through email, lower total cost of ownership, send and receive email, see progress of open calls, single point of contact
Hosted Exchange complete	Access to email, contact support, don't receive unwanted email, do not receive viruses through email, lower total cost of ownership, make appointments with colleagues, make group email addresses, option to give colleagues access to your email, send and receive email, see progress of open calls, single point of contact
{Historium, hosted Exchange complete, Exchange brick-level }	Access to email, contact support, don't receive unanted email, do not receive viruses through email, lower total cost of ownership, make appointments with colleagues, make group email addresses, option to give colleagues access to your email, send and receive email, see progress of open calls, single point of contact, automatic back-up, contact support, data is secure, easy to use backup, free software updates, know if a backup was successful, receive a report with usage statistics, restore server to original state, see progress of open calls
{Historium, hosted Exchange basic}	Access to email, contact support, don't receive unwanted email, do not receive viruses through email, backup happens without active involvement of local ICT personnel, contact support, data is secure, easy to use backup, free software updates, know if a backup was successful, receive a report with usage statistics, restore server to original state, see progress of open calls, single point of contact

Identify positive consequences for the bundles, additional to the consequences of the individual services. Next, we review if the service *bundles* lead to additional positive consequences.

Case study: For OGD, we find for example that the services in a bundle such as {Historium (OGD), Hosted Exchange complete (OGD)} are supported by the same helpdesk. Thus, for such a bundle of hosted services from OGD, 'single point of contact' can be an additional positive consequence for the bundle.

3.2 The Customer Perspective

Want. A want is a specific, supplier-independent solution that is commercially feasible to be provisioned on its own. As a want indicates a solution available in the market, at least one supplier should be willing to provide the solution.

Case study: Fig. 4 shows various wants: Hosted desktop, online backup, hosted Exchange, mailbox backup. The wants correspond to elementary services offered by OGD.

Consequence. As already explained, the consequences for the customer are the same as the supplier conquence, as the notion of consequence is used to match the supplier perspective with the customer perspective.

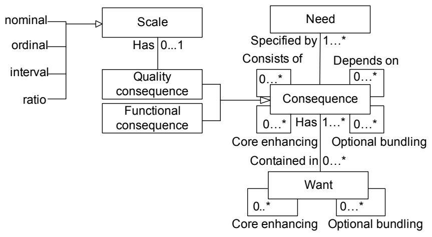

Fig. 3. The e^3*service* customer perspective ontology

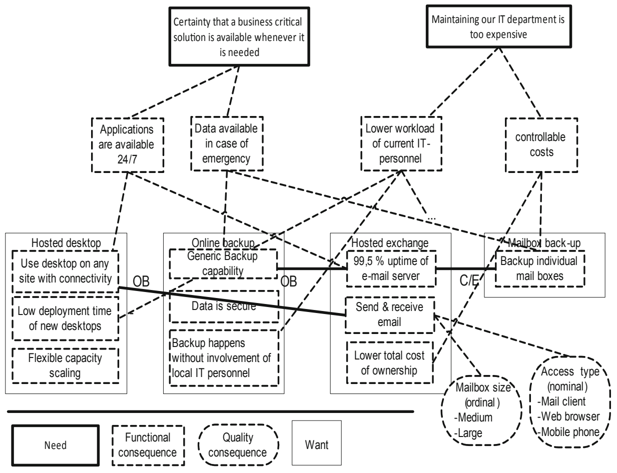
Certainty that a business critical solution is available whenever it is needed
Maintaining our IT department is too expensive
Applications are available 24/7
Data available in case of emergency
Lower workload of current IT-personnel
controllable costs
Hosted desktop
Use desktop on any site with connectivity
Low deployment time of new desktops
Flexible capacity scaling
OB
Online backup
Generic Backup capability
Data is secure
Backup happens without involvement of local IT personnel
OB
Hosted exchange
99,5 % uptime of e-mail server
Send & receive email
Lower total cost of ownership
C/E
Mailbox back-up
Backup individual mail boxes
Mailbox size (ordinal) -Medium -Large
Access type (nominal) -Mail client -Web browser -Mobile phone
Need
Functional consequence
Quality consequence
Want

Fig. 4. Sample of the customer perspective service catalogue

Case study: Fig. 4 presents various functional consequences: send and receive email, generic backup capability; and various quality consequences: mailbox size and access type. We consider 95% uptime as a functional consequence because account managers indicate that cosumers articulate a direct need (applications are available 24/7) for this consequence.

Consequence dependency. Two types of dependencies between consequences may be of interest for the customer: The Core/Enhancing dependency and the Optional Bundling dependency. These dependencies have alreay been discussed for the supplier perspective.

Case study: The Generic backup capability is an Optional Bundling relationship with 99.5% uptime of email server. Both consequences can be separately obtained, but can add value to each other when sold in a bundle.

The 99.5% uptime of email server is in a core/enhancing relationship with Backup of individual mailboxes. The Backup of individual mailboxes consequence can only meaningfully be obtained in combination with a mailbox.

Need. A need represents a problem statement or goal, independently from a solution direction (see[1]).

Case study: Account managers from OGD often mention two customer concerns that lead organizations to acquire hosted services from OGD: (1) an ICT solution should be available when needed, and (2) an IT department is too costly.

4 Reasoning about Hosted ICT Services

4.1 Generic Reasoning Structure of e^3*service*

Fig. 5 shows the high-level reasoning process of e^3*service* . The customer starts the process by selecting a need out of the known needs in the service catalogue. Then the custumer chooses consequences and therefore selects the valuable features s/he wants to obtain from a service. Next, the chosen customer consequences are matched to the supplier consequences, as annotated to service bundles, to find service bundles that can offer these consequences.

In case no service bundles are found, the customer is asked to reconsider the desired consequences. If service bundles are found, chances are high that these bundles come with additional positive and negative consequences (e.g. costs). Therefore, the customer is asked to score these additional consequences, and to consider trade-offs.

In case a bundle is rejected as a result of considering trade-offs, a so-called critique step is done to identify which consequence is responsible for the negatively scored bundle. The customer can then restart the reasoning process, by deselecting the consequence responsible for the negatively scored bundle.

The customer may decide to obtain the bundle, or to consider value-enhancing consequences. The customer then searches for services that provide added value to the basic service bundle already selected. The reasoning process restarts then by considering additional consequences for value enhancing services.

We have implemented this reasoning process in a Java-based software reasoner. The reasoner uses the customer and supplier based service catalogue, and

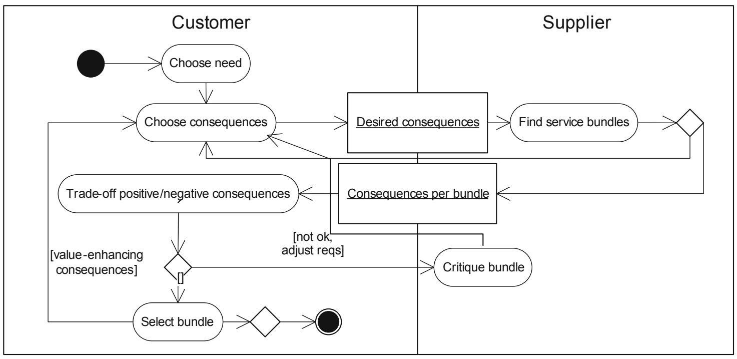

Fig. 5. The generic reasoning structure of e^3*service*

selects by following the aforementioned reasoning process a service bundle. For the OGD case at hand, we illustrate a typical scenario of service bundle selection.

4.2 The Reasoning Process for the OGD Case

We now display the reasoning process for hosted services. In particular, we focus on customer-supplier interaction and reasoning with value-enhancing services. We take the following scenario as a starting point:

'An antique dealer's current e-mail solution is not reliable enough. His server is outdated and sometimes crashes which causes some e-mail to not be received.'

Step 1: Choose need and choose consequences. Of the two most commonly heard needs from OGD, the need 'we need certainty that a business critical technical solution is available whenever needed' is selected for the antique dealer since this comes closest the problem he currently faces: An unreliable e-mail solution.

Next, the antique dealer chooses the consequence '99,5 % uptime of e-mail server' and proceeds to score the attached quality consequences from the nominal scale 'e-mail access method' and from the ordinal scale 'mailbox size'. We assume the antique dealer provides the importance scores presented in Table 2.
Outcome of this step: see Table 2.

Table 2. Mailing preferences entered by antique dealer

For the selected functional consequence, the quality consequences - per scale - are:
We found the following scale of quality consequences: e-mail access method
Please assign a score from 1(not important) to 10(vital) to each of these consequences
e-mail using Microsoft Outlook 6
e-mail access using a web browser 8
e-mail access by mobile phone 2
We found the following scale of quality consequences: mailbox size
1: large mailbox
2: small mailbox
please attach an importance rating (1-10) to this ordinal scale: 8

Step 2: Find and rank service bundles. We find the bundles 'Hosted Exchange Basic' and 'Hosted Exchange complete' for the must-have functional consequence '99,5 % uptime of e-mail server'. For this, the reasoning process:

1. Matches the customer and supplier perspective consequences '99,5 % uptime of e-mail server';
2. The supplier perspective consequence '99,5 % uptime of e-mail server' is traced to the supplier-specific resources 'Hosted Exchange basic' and 'Hosted Exchange complete'
3. The supplier-specific resources are traced to the service ports that correspond to the service bundles 'hosted Exchange basic' and 'hosted Exchange complete'.

Next, bundle scores are calculated for the found bundles. For the bundle 'Hosted Exchange basic' this calculation is presented below.

– 'small mailbox' scores 0.25 because this consequence (1) is defined on an ordinal scale and (2) is defined on an ordinal scale that contains one additional, higher ranked, consequence: 'large mailbox'. We use the Rank-Order Centroid method (ROC) [4], which can transform a qualitative ranking of a consequence into a quantitative ranking whose values are normalized to a value [0...1], to score items on an ordinal scale; the higher ranked consequence scores then 0.75, the lower ranked consequence 0.25.
– Other consequences present in the bundle, such as 'e-mail access method: E-mail access using a web browser' score 1 as they are defined on a nominal scale.

Note here that the scoring of consequences from an ordinal scale is performed differently from consequences defined on an ordinal scale. See Table 3 for a resulting ranked service bundles, as provided by the software reasoner.

Outcome of this step: The service bundles 'Hosted Exchange Basic and 'Hosted Exchange Complete, ranked according to how well they fit with the customer preferences (see Table 3).

Table 3. Tool output for two bundles of mailing services, ranked according to how well they fit with customer preferences

```
...
So the possible bundles, sorted according to preference, are:
1: hosted Exchange Standard Complete with the score: 5.3
2: hosted Exchange Basic with the score: 4.1
Please select a bundle. If none is to your liking, select 0 (zero).
```

Step 3: Trade off positive/negative consequences. The bundle 'Hosted Exchange complete' is more in accordance with the desired consequences than the bundle 'Hosted Exchange basic'. Yet, from the pricing models the antique dealer also observes that the complete bundle costs €10 per month per user, while the basic bundle costs €5 per user per month. For a fair weighing of these costs against the benefits provided by each bundle, the antique dealer therefore decides to also score negative consequences, such as '12 month commitment to using OGD services' and 'hosted Exchange complete fee'.

The provided scores are presented in Table 4. We can see that although the antique dealer scores the fee for Exchange complete higher than the fee for Exchange basic, Exchange complete still outranks Exchange basic, albeit with a smaller margin than for the bundle scores that are based on the positive consequences only (again, see Table 4). Thus, the antique dealer decides to acquire the bundle 'Hosted Exchange complete'.

Outcome of this step: The antique dealer chooses the bundle 'Hosted Exchange complete'.

Table 4. Trade of positive and negative consequences for hosted services

```
Would you also like to score all negative consequences to see how a bundle scores on a trade-off
between benefits and sacrifices?(y/n) y
Please assign a score from 1(does not really matter) to 10(this consequence must be lacking from
the bundle) to each of the following consequences:
no switching of suppliers for 12 months 3
hosted Exchange complete fee 8
hosted Exchange basic fee 4
Taking the negative scores into account, the new ordering of bundles is:
1. hosted Exchange Standard Complete Score: 4.2
2. hosted Exchange Basic: 3.6
Please select a bundle. If none is to your liking, select 0 (zero).
```

Step 4: Find value-enhancing services. Next, the antique dealer chooses consequences that OGD considers to be interdependent in demand with the functional consequences already included in the selected bundle. For the consequence '99,5 % uptime of e-mail server', we find two functional consequences: 'back up individual mail boxes' and 'generic back-up capability' (for reference, see the customer perspective catalogue in figure 4). Because the antique dealer maintains his backups on a server that can itself be considered an antique, we assume that the antique dealer is interested in both functional consequences.

Next, the reasoning process considers the quality consequences relevant for the scored functional consequences. As can be observed from the customer perspective service catalogue however (figure 4) none of the value-enhancing consequences contain quality consequences.

Outcome of this step: The additional, value-enhancing, consequences {back up individual mail boxes, quick recovery of individual mailboxes, automatic backup, data is secure, back-up happens without involvement of local IT personnel}.

Step 5: Find and choose bundle. To find bundles, the reasoning process uses the value-enhancing consequences as input, plus the consequences from the service bundle already selected ('Hosted Exchange complete').

As with the basic consequences, the reasoning process finds service bundles by (1) matching these consequences with all supply-side consequences (2) tracing the consequences on the supplier perspective to service bundles that satisfy these consequences.

Outcome of this step: The bundle 'Hosted Exchange complete Historium Exchange brick-level.

Step 6: Trade-off positive and negative consequences. The antique dealer receives a specification of the positive and negative consequences for the bundle 'Hosted Exchange complete Historium Exchange brick-level', plus a specification of the pricing model for this bundle. After reviewing the pricing model (see Table 5), the antique dealer decides to score the negative consequences for the bundle 'Hosted Exchange complete Historium Exchange brick-level' also.

Table 5. Pricing model for the bundle Hosted Exchange complete Historium Brick-level

For this bundle, we found the following pricing model:
*Hosted Exchange complete Historium Brick-level Exchange pricing model
This pricing model is of the type: single discount pricing model.
The price of this bundle is build up as follows:
* The two-part pricing model Hosted Exchange standard complete pricing model is build up as follows: X euro installation fee + 10 euro monthly/User. This pricing model is attached to the same port as the consequence hosted Exchange complete fee.
* The n-block pricing model Historium pricing model is build up as follows: Condition: Server 1: 60 euro monthly + Condition 2: Server 2: 55 euro/monthly + Condition 3: Server 3 and on: 50 euro/monthly/server. This pricing model is attached to the same port as the consequence online back-up fee.
* The usage-based pricing model Exchange brick-level pricing model is build up as follows: 50 euro/monthly. This pricing model is attached to the same port as the consequence back-up individual mailboxes fee.

The antique dealer has already agreed to the fee for hosted Exchange complete, and so attaches a score '2' (from 1 to 10, where 1 indicates 'does not matter' and 10 'won't have') to the negative consequence 'hosted Exchange complete fee'. The antique dealer attaches the score '6' to the negative consequence 'online back-up fee', considering that while he pays 60 euro monthly (due to the single server he has) he also does not have to worry about having to maintain a backup anymore. Finally, the antique dealer considers the negative consequence 'individual mailboxes fee' as a won't have and thus scores this as a '10'. He considers the investment in a dedicated backup for his e-mail not to be worthwhile €50 monthly, also when considering that he already has a generic backup capability.

Outcome of this step: No possible bundles, proceed with the critique-step.

Step 7: Find culprit. For the negative consequence that is scored as a won't have, the reasoning process now performs a critique-step to find the positive consequences attached to the cost-source. As such, the antique dealer is presented with the dialogue in Table 6.

Outcome of this step: The consequence 'Back up individual mail boxes' is found as the culprit.

Step 8: Choose consequences. Next, the antique dealer is asked to what place in the reasoning process he should return to adjust his requirements.

As can be seen in the dialogue from Table 7, the antique dealer selects (2): Start at value-enhancing requirements. He now returns to the basic bundle

Table 6. Critique: Find the preference that is too expensive to fulfill

The bundle Hosted Exchange complete Historium Exchange brick-level contains the won't have Brick level fee
We shall now investigate what positive consequences from this bundle require the negative consequence Brick level fee
The following group of positive consequences:
1: Back up individual mail boxes
1: Quick recovery of individual mail boxes
requires the negative consequence back-up individual mailboxes fee

Table 7. Adjusting preference scores

1. Start from group of detailed functional consequences you had initially chosen. This was:
*don't receive viruses through e-mail
*don't receive unwanted e-mail
*99,5 % uptime of e-mail server
...
2. Start at the value-enhancing consequences
*Back up individual mail boxes
*quick recovery of individual mailboxes
3. start all over, ie: from the need.
other (> 3): quit.

'Hosted Exchange complete' and answers 'no' when asked if he is interested in the value-enhancing consequence 'Back up individual mail boxes' and 'quick recovery of individual mailboxes'.

Outcome of this step: The consequences 'Back up individual mail boxes' and 'quick recovery of individual mailboxes' are no longer used as input for the reasoning process.

Step 9: Find service bundles. The reasoning process now again finds service bundles for the set of consequences belonging to the bundle 'Hosted Exchange complete Historium Exchange brick-level' *minus* the consequence 'Back up individual mail boxes'. Thus, the bundle 'Hosted Exchange complete Historium' is eventually found. Since this bundle contains an online e-mailing capability and online back-up functionality and the antique dealer finds the costs for both functionalities acceptable, he decides to acquire this service bundle.

Outcome of this step: The bundle 'Hosted Exchange complete Historium'.

5 Lessons Learned and Conclusions

Lesson 1: The reasoning performed by the software tool is similar to the reasoning performed by account managers. In reaction to a demonstration of our software reasoner, the account managers stated that the reasoning process performed by the reasoner is similar to how they sell services. In particular:

- Account managers start from a problem that the customer has, independently of the available solution. This is similar to our separation between needs (problem/goal) and consequences (valuable features that act as solution directions for satisfying the need).
- Constraints do influence decisions on what bundle the customer acquires
- Account managers first seek out a service that satisfies the basic requirements of the customer. Only thereafter, crossselling and upselling is performed. This is similar to our process of first agreeing on a basic service, and only then seeking out any value-enhancing services.

Lesson 2: Graphical representations are useful for together exploring services, but for usage on a daily basis a textual version is more adequate. The service catalogue that OGD uses on a daily basis is a stripped down *textual* version of the conceptual models presented in this paper. This textual representation is preferred by marketing personnel, who perceive a bullet list of services and needs more helpful in writing promotional texts than a conceptual model. Also, the account managers prefer a textual representation of the service catalogue, so that they can write a personalized offer text for customers. OGD-personnel did however declare that the formal conceptual models allowed for a structured exploration of their service offerings, and what needs these offerings satisfy.

6 Conclusion

In this paper we have presented the e^3*service* ontology. This ontology takes two perspectives on commercial services: the customer perspective and the supplier perspective. Both the customer and the supplier perspective contain the notion of 'consequence' of a service, the 'consequence' is therefore used to derive service bundles (supplier perspective) from customer needs (customer perspective). Moreover the e^3*service* ontology is capable of deriving extra, value adding, services to the services already found.

The reasoning process as done by the e^3*service* ontology comes close to the kind of reasoning performed by account managers. Therefore, the e^3*service* ontology can be used to explain the sales process to new account managers.

Two possible lines of further research can be foreseen. First, the service bundles can be derived dynamically, during the matching process of customer consequences and service consequences. Currently, all possible bundles are generated upfront the reasoning process. Second, the reasoning process could include the 'supplier of the supplier'. Now, the reasoning process is restricted to the supplier (OGD) satisfying the customer. But the supplier sometimes becomes a customer because additional services need to be obtained from others to provision a service.

Acknowledgements. The authors wish to thank OGD for the case study. This paper is based on [7, Ch 8] and [6].

References

1. Arndt, J.: How broad should the marketing concept be? Journal of Marketing 42(1), 101–103 (1978)
2. Baida, Z., Gordijn, J., Akkermans, H., Saele, H., Morch, A.Z.: Finding e-service offerings by computer-supported customer need reasoning. International Journal of E-Business Research (IJEBR) 1(3), 91–112 (2005)
3. Baida, Z., Gordijn, J., Akkermans, H., Saele, H., Morch, A.Z.: How e-Services Satisfy Customer Needs: a Software-aided Reasoning, ch. IX. Idea Group, USA (2006)
4. Hutton Barron, F., Barrett, B.E.: Decision quality using ranked attribute weights. Management Science 42(11), 1515–1523 (1996)
5. Bitner, M.J., Gremler, D.D., Zeithaml, V.A.: Services marketing: Integrating customer focus across the firm. McGraw-Hill, Irwin (2008)
6. de Haan, F.: Reasoning about e-services at operator group delft. Master's thesis, Vrije Universiteit Amsterdam (2009)
7. de Kinderen, S.: Needs-driven service bundling in a multi-supplier setting - The computational e^3 service approach. PhD thesis, Vrije Universiteit Amsterdam (2010)
8. Fensel, D., Lausen, H., Polleres, A., de Bruin, J., Sollberg, M., Roman, D., Domingue, J.: The Web Service Modeling Ontology. Springer, Berlin (2006)
9. Grönroos, C.: Service management and marketing: customer management in service competition. Wiley India Pvt. Ltd., Chichester (2007)
10. Gutman, J., Reynolds, T.J.: Laddering theory-analysis and interpretation. Journal of Advertising Research 28(1), 11 (1988)
11. Lovelock, C.: Service Marketing - People, Technology, Strategy, 4th edn. Prentice-Hall, Englewood Cliffs (2001)
12. Stremersch, S., Tellis, G.J.: Strategic bundling of products and prices: A new synthesis for marketing. Journal of Marketing 66(1), 55–72 (2002)