Ontology-Based Operators for e-Business Model De- and Re-construction

Jaap Gordijn (gordijn@cs.vu.nl)
Vrije Universiteit - Vuture.net (Centre for e-business research), De Boelelaan 1081a,
NL-1081 HV Amsterdam, The Netherlands,
Cisco Systems - Internet Business Solutions Group, Hoogoorddreef 9,
1101 BA Amsterdam, The Netherlands

Hans Akkermans
Vrije Universiteit - Vuture.net, and
AKMC Knowledge Management, Klareweid 19,
NL-1831 BV Koedijk, The Netherlands

ABSTRACT

We define e-business models as conceptual models that show how a network of actors (a value constellation) creates, exchanges and consumes objects of value by performing value adding activities. In this paper we present a semi-formal ontology-based representation of e-business models that is useful in carrying out a preliminary business and requirements analysis. In particular, we show that a small set of generic 'model deconstruction' operators is able to generate design variations on a given e-business model, so that upfront analysis of the characteristics and consequences of a range of alternative e-business models becomes possible. We illustrate our ontology-based e^3*-value* approach by a commercial project on Internet news services.

Submission: K-CAP 2001, Canada

1. INTRODUCTION

Successful e-business information systems are often characterized by *innovative* ways of doing business. This is usually called the *e-business model*. We define an e-business model as a conceptual model that shows how a network of actors creates, exchanges and consumes objects of *value* by performing value adding activities.

Finding such an e-business model is a creative task. We can, however, support this task by (1) an insightful way of representing e-business models, and (2) a way of finding and analyzing 'design' variations on such models.

To find variations on an initial e-business model, and consequently to assist in the elicitation of such a model, e^3*-value* defines e-business model *de-construction* operators (inspired by [7, 3, 8]). These operators are part of an e-business model de-construction and re-construction process, during which we *de-assign* activities from their performing actors, try to find alternative, and/or more activities by de-constructing existing ones, and re-assign newly found activities to executing actors. Because we assume that activities are profitable for at least one actor, re-assignment is possible. Essentially, to clarify discussions between stakeholders, we split the re-construction process into two questions: (1) which value adding activities exist, and (2) which actors are to perform these activities?

In previous work we have introduced an ontology-based general representation of e-business models ([4], see recent publications at http://www.cs.vu.nl/~gordijn/research.htm). Based on this e^3*-value* ontology, we discuss in the present paper three generic operators for e-business model de-construction: (1) the *value activity* de-construction operator, which breaks an activity into smaller ones, but leaves the products/services offered or requested by the original activity to its environment unchanged, (2) the *value port* de-construction operator, which breaks a service/product offered or requested by a value activity into smaller ones, and (3) the *value interface* de-construction operator, which breaks combinations of value objects offered *and* counter-compensations requested into smaller pieces.

We illustrate e-business model de- and re-construction by one of the e-business projects where we successfully applied our approach. The project at hand is about the provisioning of a value-added news service. With respect to such a service, a regular newspaper called the *Amsterdam Times* (a fictitious name, but based on an actual commercial e-business project) wants to offer to all its subscribers a service to read articles online using the Internet, but such that it will make hardly any additional costs. Therefore, the idea is to finance the execution of this business idea by the telephone connection revenues, which originate from the reader who has to set up a telephone connection for Internet connectivity.

This paper first introduces in brief the core concepts of our e^3*-value* methodology, which we use to formalize e-business models (Sec. 2). In Sec. 3 we discuss both the general theory and an application of e-business model de- and re-construction, and in Sec. 4 we present our conclusions.

2. E^3-VALUE CORE CONCEPTS

To represent an e-business model, we use a *lightweight* ontology consisting of interrelated core concepts, and we utilize a well known *lightweight* scenario technique, called Use Case Maps [1]. This allows us to communicate e-business models easily to intended users such as business consultants, and CxO's. Moreover, the agility of e-business projects (the need to define, explore, and execute a business idea fast [5]) asks for an lightweight approach. Below, we discuss the ontological concepts and the UCM scenario concepts briefly (see Fig. 1 for an example). More information can be found in [4], [1].

2.1 The e^3-value ontology

Actor. An actor is perceived by its environment as an independent economic (and often also legal) entity. By carrying out *value activities* (see below) an actor makes profit or increases its utility. In a sound and viable e-business model *every* actor should be capable of making a profit.

Value Object. Actors exchange value objects. A value object is a service, product or even a consumer experience. The important point here is that a value object is *of economic value* to one or more actors.

Value Port. An actor uses a value port to show to its environment that it wants to provide or request value objects. The concept of port (a notion adopted from engineering systems theory) is important, because it enables to abstract away from the internal business processes, and to focus only on how external actors and other components of the e-business value model can be 'plugged in'.

Value Interface. Actors have one or more value interfaces. A value interface groups individual value ports. It shows the value object(s) an actor is willing to exchange *in return for* other value object(s) via its ports. Such willingness is expressed by a decision function on the value interfaces, which shows on what conditions an actor wants to exchange a value object for another value object. The exchange of value objects is atomic at the level of the value interface. Either *all* exchanges occur as specified by the value interface or *none* at all.

Value Exchange. A value exchange is used to connect two value ports with each other. It represents one or more *potential* trades of value objects between actors.

Value Offering. A value offering is a set of value exchanges. It shows which value objects are exchanged via value exchanges *in return for* other value objects. A value offering should obey the semantics of the connected value interfaces: that is values are exchanged via a value interface on *all* its ports, or *none* at all.

Market segment. In the marketing literature [6], a market segment is defined as a concept that breaks a market (consisting of actors) into segments that share common properties. Accordingly, our concept *market segment* shows a set of actors that share for value interfaces an equal decision function. We realize that in practice all actors behave differently and consequently cannot have equal decision functions. However, to be able to design understandable e-business models, we assume (as in marketing theory is done) that some groups of actors constitute equivalence classes with respect to their decision functions.

Value Activity. A value activity is *performed by* an actor and increases profit or utility *for* such an actor. The value activity is included in the ontology to discuss the *assignment* of value activities to actors. Value activities can be de-constructed into smaller value activities, but the requirement is that these still should be profitable or increase utility for the performing actor.

2.2 Use Case Maps

Scenario path. A scenario path consists of one or more *segments*, related by *connection elements* and *start-* and *stop stimuli*. It represents via *which* value interfaces objects of value must be exchanged, as a result of a start stimulus, *or* as result of exchanges via *other* value interfaces. Thus a scenario path shows causal relations between value interfaces.

Stimulus. A scenario path starts with a **start stimulus**. A start stimulus represents an event, possibly caused by an actor. If an actor causes an event, the start stimulus is drawn within the box representing the actor. The last segment(s) of a scenario path is connected to a **stop stimulus**. A stop stimulus indicates that the scenario path ends.

Segment. A scenario path has one or more segments. Segments are used to relate value interfaces with each other, possibly via connection elements, to show that an exchange on one value interface causes an exchange on another value interface. Using connection elements, sophisticated causal relations can be represented.

Connection. Connections are used to relate individual segments. An **AND fork** splits a scenario path into two or more sub path, while the **AND join** collapses sub path into one path. An **OR fork** models a continuation of the scenario path into one direction, to be chosen from a number of alternatives. The **OR join** merges two or path into on path. Finally, the **direct** connection interconnects two individual segments.

3. E-BUSINESS MODEL RE-CONSTRUCTION IN E^3-VALUE

The e-business model re-construction process consists of the following steps, which we discuss in the following sections in detail:

1. Identification of an initial e-business model.
2. De-construction of the initial e-business model.
3. Re-construction of alternative e-business models.

3.1 The initial e-business model

The process of e-business model re-construction starts with a representation of an initial e-business model. We assume the existence of an innovative e-business idea. Consequently, the goal of this step is to articulate that idea more precisely, so that stakeholders all have a common understanding about the idea.

The idea for the e-business model in this paper is to use a *termination* fee to finance a *news article online* service for subscribers on a regular newspaper. *Termination* means that if someone tries to set up a telephone connection by dialing a telephone number, another actor must pick up the phone, that is, *terminate* the connection. If someone is willing to *cause* termination of a large quantity of telephone calls, most telecommunication operators are willing to pay such an actor for that (the *termination fee*). Because the newspaper has a large subscriber base, it is capable to generate a large number of terminations for an *article online* service. This idea is formalized by an initial e-business model (see Fig. 1).

3.2 e-Business model de-construction

For de-construction, we de-assign actors from value activities, but leave value exchanges between value activities intact. Then, we repeatedly apply one of the three de-construction operators. As we will show, it is possible to apply operators a number of times on an e-business model. The next sections discuss the three operators, along with their business rationale, and an example.

3.2.1 Value Activity De-construction

Business rationale.

Can we split a value activity, which initially is viewed as being performed as a whole by one actor, into smaller activities, together

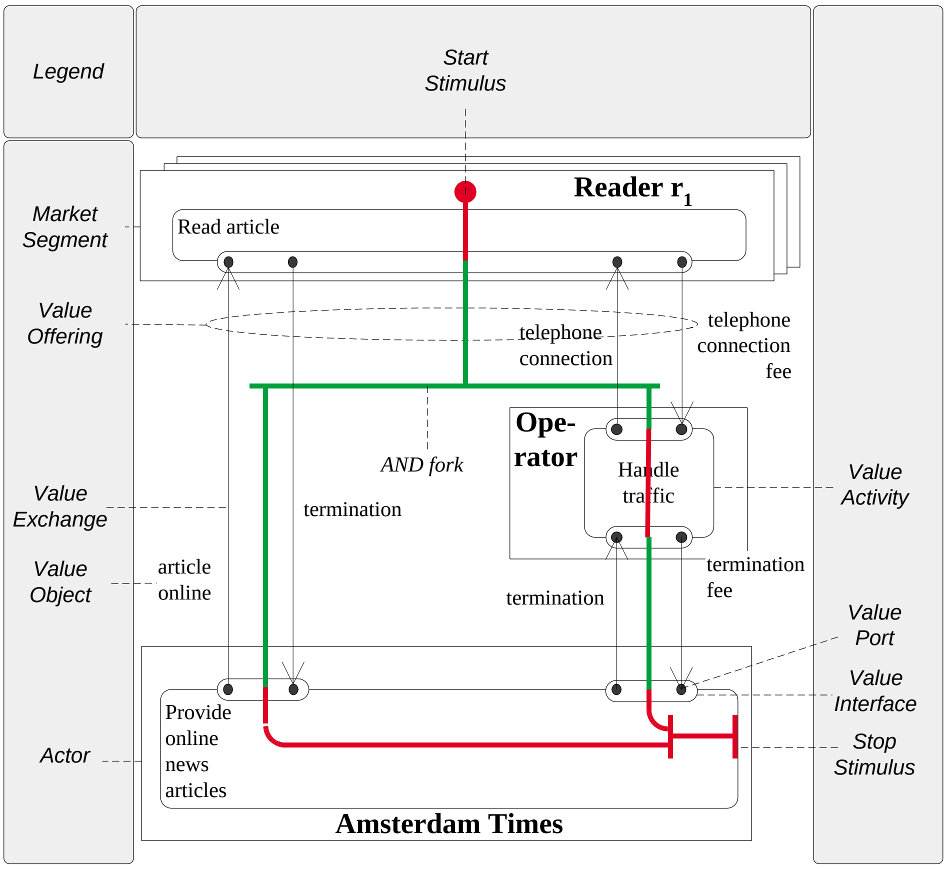

Figure 1: The initial e-business model showing that the Amsterdam Times funds its service by a termination fee offered by a telecommunication operator. The reader offers a termination opportunity and a telephone connection fee and requests in return an article online and a telephone connection. The ports requesting/offering these value objects are grouped into one value interface from a reader's perspective because these objects are only of value in combination to the reader. By following the scenario path, it can be seen that the Amsterdam Times resells the termination to a telecommunication operator. This operator also receives a fee for a telephone connection, as result from reading an article. For each actor, initially one value activity is assumed that describes its value adding process at best.

behaving as the original one, whereby each smaller activity potentially can be performed by individual actors?

Focus.
The value activity de-constructor focusses on the *internal* structure of a value activity while keeping its value interfaces to the environment the same. It breaks down a value activity into smaller ones, for instance to allow specialized actors to perform one of these value activities.

Operator $VAD : a \longrightarrow a_1, \ldots, a_n$**.**

1. De-construct a value activity a with value interfaces $i_1, \ldots i_n$ into value activities $a_1, \ldots, a_n$.

2. Assign each value interfaces $i_1, \ldots i_n$ to one or more of the de-constructed value activities.

3. Add, if necessary, extra value interfaces to the de-constructed value activities, and relate these by value exchanges. Extra value interfaces and exchanges can be necessary to ensure that the de-constructed activities $a_1, \ldots, a_n$ are from an environment perspective equivalent to a.

4. Reconsider scenario segments, which hit the value interfaces of value activity a.

It is possible that for a value activity a multiple alternative de-constructions exist.

Example: De-construct the *Handle traffic* value activity into two other value activities
Fig. 2 de-constructs the *Handle traffic* value activity into two smaller value activities, which each can be potentially performed by a single (different) actor. The two value interfaces of *Handle traffic* can be found at the two smaller value activities, thereby providing the same interfaces to their environment as the original value activity. The value activity *Handle local traffic* offers end-to-end connectivity to a reader and gets paid for this, while it only exploits the local loop: the last miles from a local telephone switch to the reader. Consequently, this activity should 'buy' interconnection from the *Handle long distance traffic* activity, and pays for this in return. The latter activity exploits a telecommunication network between local telephone switches, and a web server for hosting news articles. Buying interconnection is shown by adding value interfaces and value exchanges between *Handle local traffic* and *Handle long distance traffic*. The scenario path is changed but hits the same value interfaces as was the case for the *Handle traffic* value activity.

Example: De-construct the *Provide Online news articles* value activity into two other value activities
The de-construction shown in Fig. 3 essentially separates the content creation (news) from the technical infrastructure needed to deliver content to the reader. It can be seen as out sourcing Internet service provisioning from a news provisioning perspective. Again we need to add value interfaces and value exchanges to represent that the *Provide news articles* value activity must acquire facilities for Internet service provisioning. Note that the scenario path for the de-constructed value activities hits the same value interfaces as the original value activity. However, internally, the scenario path splits to show that as a result of a termination/article online exchange, also a termination/termination fee *and* an Internet service provisioning/fee is necessary.

3.2.2 Value Port De-construction

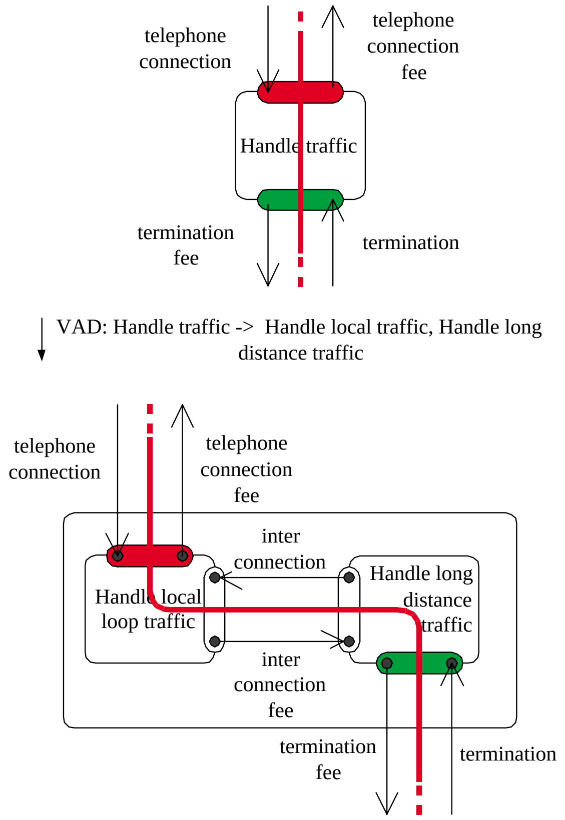

Figure 2: De-construction of the value activity *handle traffic* into two value activities *handle local loop traffic* and *handle long distance traffic*.

Business rationale.
Can we split products, services or combinations into smaller products/services, which each can be delivered and consumed by individual actors?

Focus.
Focus is to untangle offered or requested value objects, which still are of value for actors. These objects can potentially be offered by multiple value activities rather than one, and thus by multiple actors. Because we change the value port, we change the value interface of a value activity to the environment.

Operator $VPD : p \longrightarrow p_1, \ldots, p_n$**.**

1. For each value port p in a value interface:

2. Consider de-construction of value port p with value object o into value ports $p_1, \ldots, p_n$ with value objects $o_1, \ldots o_n$.

3. If de-construction is possible, de-construct also the peer-ports of p. *Peer* ports are the ports p_i, which are connected by value exchanges to value port p. Note that a value port p can be connected to multiple other value ports p_i, representing that a value activity containing port p can exchange objects with multiple other value activities.

   (a) Dis-connect value exchanges connecting value port p and value ports p_i.

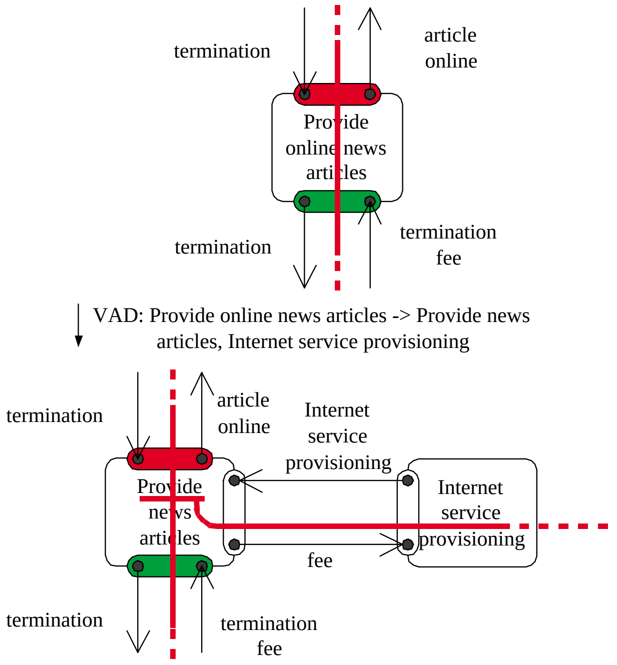
termination
article online
Provide online news articles
termination
termination fee
VAD: Provide online news articles -> Provide news articles, Internet service provisioning
termination
article online
Internet service provisioning
Provide news articles
Internet service provisioning
fee
termination
termination fee

Figure 3: De-construction of value activity *provide online news articles* into two value activities *provide news articles* and *internet service provisioning*.

(b) De-construct value ports p_i into ports $p_{1_i}, \ldots, p_{n_i}$ in the same way as p was de-constructed.

(c) Re-connect ports $p_1, \ldots, p_n$ using value exchanges with ports $p_{1_i}, \ldots, p_{n_i}$.

Example: De-construct the value object *Internet service provisioning* into two other value objects

Fig. 4 de-constructs the value port *Internet service provisioning* into two different ports/value objects: (1) Internet *hosting* provisioning, e.g. hosting a web site, and (2) Internet *access* provisioning, e.g. exploiting a modem pool to offer access to the Internet.

3.2.3 Value Interface De-construction

Business rationale.

A value interface models the notion of *one good turn deserves another*, consisting of value objects offered and value objects requested in return. It is sometimes possible to split up a value interface in more interfaces, for (1) de-bundling, and (2) smaller value activities. Bundling refers to the business notion that an actor believes that if two or more products are offered as a whole, more money can be earned than offering these products seperately (see e.g. [2]). De-bundling refers to the opposite mechanism. We can also apply value interface de-construction to split up the value activity associated with the interface at hand. Essentially, we split up an interface into smaller ones, whereby each value interface can be associated with a new value activity.

Focus.

The focus is to find smaller value interfaces, that is value interfaces with a smaller number of value ports.

Operator $VID : i \rightarrow i_1, \ldots, i_n$**.**

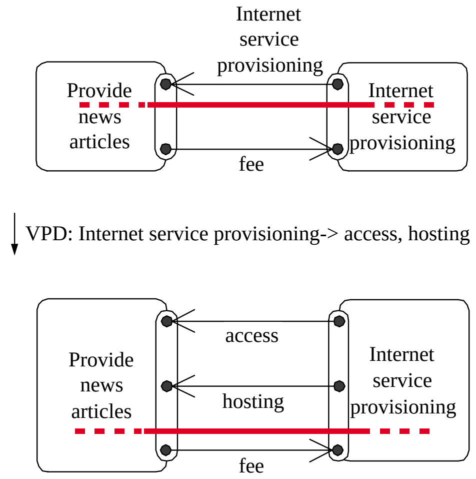
Internet service provisioning
Provide news articles
Internet service provisioning
fee
VPD: Internet service provisioning-> access, hosting
access
Provide news articles
hosting
Internet service provisioning
fee

Figure 4: De-construction of the value object *Internet service provisioning* into two value ports *access* and *hosting*.

1. For each value interface i with value ports $p_1, \ldots, p_n$ of a value activity a:
2. Find (alternative) value interfaces $i_1, \ldots, i_n$ grouping value ports $p_1, \ldots, p_n$.
3. Reconsider scenario segments.

Example: An access and hosting value interface

Fig. 5 introduces two separate value interfaces for the Internet service provisioning activity: one for offering Internet access and one for offering hosting services. Creation of these interfaces takes two steps. First we have to de-construct the *fee* port into two ports: the *access fee* and *hosting fee*. This is necessary due to the definition of value interface. A value interface models objects of value offered to the environment *and* the objects requested in return. We therefore need ports who receive the objects requested in return for offering *access* and *hosting* value objects. Second, we create two value interfaces, representing *hosting* and *access* services.

Note we do *not* split the value interface of the *Provide news articles* value activity. This value interface models that, for offering articles online, we need *both* hosting and access for each scenario occurrence.

Example: Access and hosting via value *activity* de-construction

It also possible to split up the *Internet service provisioning* value activity into *Internet access provisioning* and *Internet hosting provisioning* (see Fig. 6), but there is an important difference compared to the previous example. Fig. 6 still shows a value activity called *Internet service provisioning'* (although smaller than the original one). This activity is profitable by offering a bundle of access and hosting services, but must buy-in access and hosting from another service. In contrast, in Fig. 5, the value activity *Provide news articles* is responsible for acquiring both access and hosting.

3.2.4 Combining de-construction operators.

The three mentioned de-construction operators can be sequentially applied. The following three cases appear regularly:

- A number of sequential value activity de-construction operations. In this case, we try to break up a value activity into (alternative) smaller ones, but do not change anything visible to the outside world.

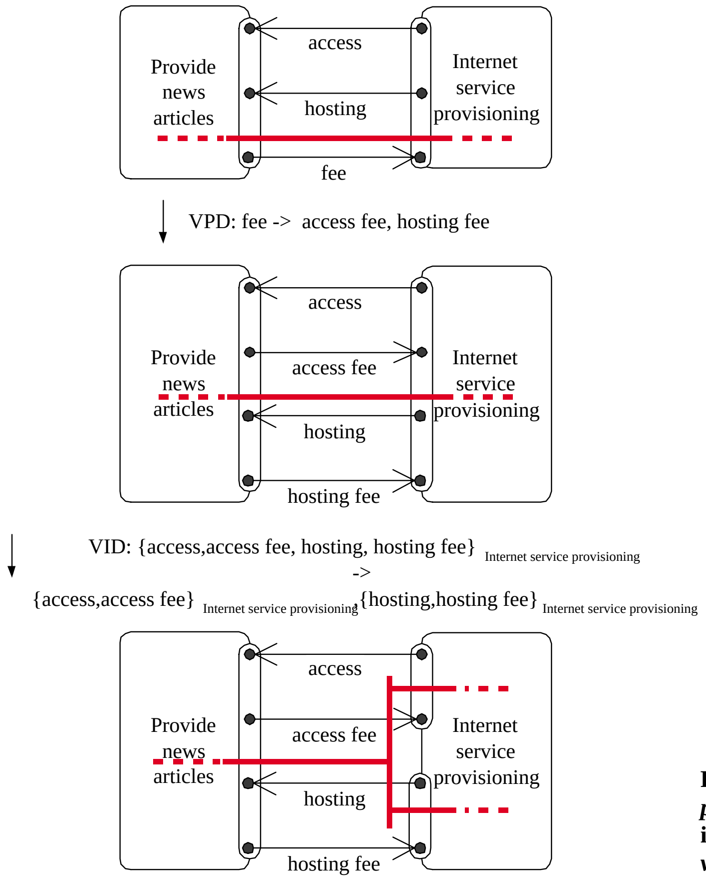
Provide news articles
access
hosting
fee
Internet service provisioning
VPD: fee -> access fee, hosting fee
Provide news articles
access
access fee
hosting
hosting fee
Internet service provisioning
VID: {access,access fee, hosting, hosting fee} Internet service provisioning
->
{access,access fee} Internet service provisioning {hosting,hosting fee} Internet service provisioning
Provide news articles
access
access fee
hosting
hosting fee
Internet service provisioning

Figure 5: De-construction of the value interface with four ports into two value interfaces with each two ports.

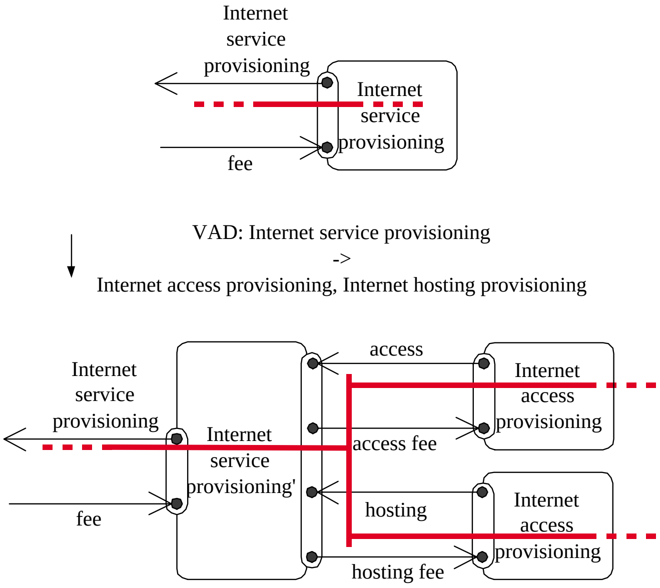
Internet service provisioning
Internet service provisioning
fee
VAD: Internet service provisioning
->
Internet access provisioning, Internet hosting provisioning
Internet service provisioning
fee
Internet service provisioning'
access
access fee
hosting
hosting fee
Internet access provisioning
Internet access provisioning

Figure 6: De-construction of the value activity *Internet service provisioning* into one for access provisioning and one for hosting provisioning. In contrast to Fig. 5, the *Internet service provisioning* ensures that their exist still one bundle of Internet service provisioning, while in Fig. 5 an actor who wants *access* and *hosting* must compose the bundle him/herself.

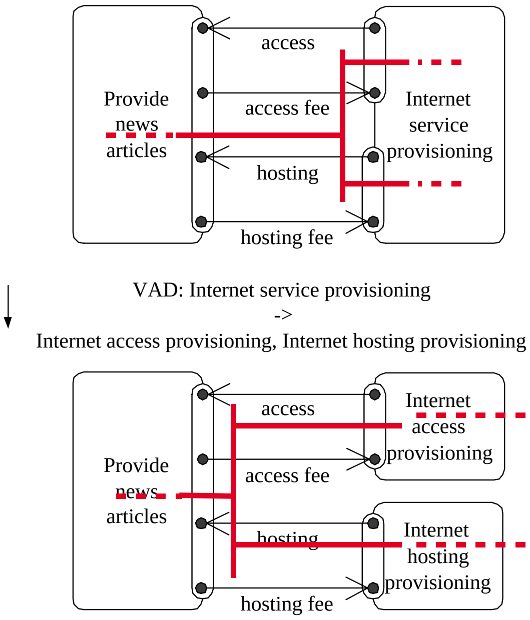
Provide news articles
access
access fee
hosting
hosting fee
Internet service provisioning
VAD: Internet service provisioning
->
Internet access provisioning, Internet hosting provisioning
Provide news articles
access
access fee
hosting
hosting fee
Internet access provisioning
Internet hosting provisioning

Figure 7: De-construction of the value activity *Internet service provisioning* into one for access provisioning and one for hosting provisioning, using the value interfaces de-constructed in Fig. 5

- Value port de-constructions, followed by value interface de-constructions, and finally value activity de-constructions. In this case, we try to find smaller value objects which can be offered by separate value activities, which can be performed by individual actors. Fig. 7 is an example of this. First we de-construct the value interface of *Internet service provisioning* into two smaller ones for access and hosting (see Fig. 5), and then we de-construct the value activity into two smaller ones.

- De-bundling, a number of value port de-constructions, followed by value interface de-constructions. Fig. 5 can be seen as a case of de-bundling: we allow that the services *hosting* and *access* are sold separately rather than as a whole. Note that a value interface means that if a value object is exchanged via one of its ports, value objects on all its other ports must be exchanged too, so after de-bundling, access and hosting can be obtained as separate services rather than as a whole.

3.3 e-Business Model Re-construction

De-construction of an e-business model means de-assigning value activities and actors, and generating new value activities. During e-business model re-construction, we study the re-assignment of value activities to performing actors.

First, we generate value activities *configurations*. These are connected value activities, by means of value exchanges, which represent an e-business model, *without* their performing actors. Because in this case study, we did not consider alternative de-constructions, so we have only one such a configuration (essentially Fig. 8 with omitted actors.).

Second, we re-identify actors, who are potentially interested in executing one or more value activities. Actors are potentially interested, if they expect to make a profit, or to increase utility by performing the value activity. Re-identification means that we consider new actors, which were not identified during development of the initial e-business model. It is reasonable to expect that by finding new, more specialized value activities, other actors than the ones already found are interested to perform these.

Third, we make an *actor-value activity assignment* matrix (see Table 1). This matrix shows actors, which are potentially interested in performing value activities of a specific configuration.

Finally, using the actor-value activity assignment matrix, alternative e-business models can be extracted and represented using our graphical technique. Fig. 8 shows one possible e-business model. Other models are possible by choosing other assignments of value activities to actors.

4. CONCLUSION

Finding innovative e-business models is a creative task. However, finding variations on such an e-business model can be facilitated by e-business model de- and re-construction. The starting consideration for this is to separate the questions (1) which value adding activities exist from (2) which actors are performing these.

To find e-business model variations, we have defined three de-construction operators, which all have a clear business rationale. The value activity de-construction (*VAD*) operator helps in finding smaller value activities, which all can be profitably performed by at least one actor. We keep the value interface invariant using this operator, and only focus on the partitioning of a value activity over a number of actors rather than one actor.

A value interface models that an actor, or value activity, offers something of value to its environment, *and* wants something in return for that. The value interface de-construction (*VID*) operator splits such interfaces into smaller ones. This may be done for two reasons. First, splitting can be done for unbundling reasons: the offering of value objects separately rather than as a bundle. Second, de-constructed value interfaces can be used to de-construct a value activity associated with these interfaces into smaller activities.

Finally, the value port de-construction (*VPD*) operator assists in identifying new value ports/objects, based on an initial one, which each can de delivered or requested by individual actors. Mostly, the *VPD* operator is followed by the *VID* operator to address unbundling, or by the *VAD* operator, to distribute the offering of the original value object over a number of actors.

Also, we have shown how these operators work out in a practical, non-trivial e-business modeling project. The representation proposed in this paper of e-business models appeared valuable in the project to illustrate complicated concepts such as call termination and interconnection to stakeholders, while the presented de- and re-construction process proved important to find new value activities, and to renegotiate assignment of these activities with the performing actors.

Acknowledgement. This work has been partly sponsored by the Stichting voor Technische Wetenschappen (STW), project nr VWI.4949 on a Framework for the Electronic Sale of Information Products. Also, we thank *De PersCombinatie* for their permission to publish some of the e-business project results.

5. REFERENCES

[1] R.J.A. Buhr. Use case maps as architectural entities for complex systems. *IEEE Transactions on Software Engineering*, 24(12):1131–1155, 1998.

[2] Soon-Yong Choi, Dale O Stahl, and Andrew B. Whinston. *The economics of doing business in the electronic marketplace*. Macmillan Technical Publishing, Indianapolis, 1997.

Table 1: Actor - Value activity matrix showing which actors can potentially perform which value activity, *and* while creating profit, or increasing utility by doing an activity.

Value activity	*Actor*				
	Reader	Last Mile	Data Runner	Hoster	Amsterdam Times
Read article	x				
Handle local loop traffic		x	x		
Handle long distance traffic		x	x		
Provide internet access		x	x	x	x
Hosting		x	x	x	x
Provide news articles					x

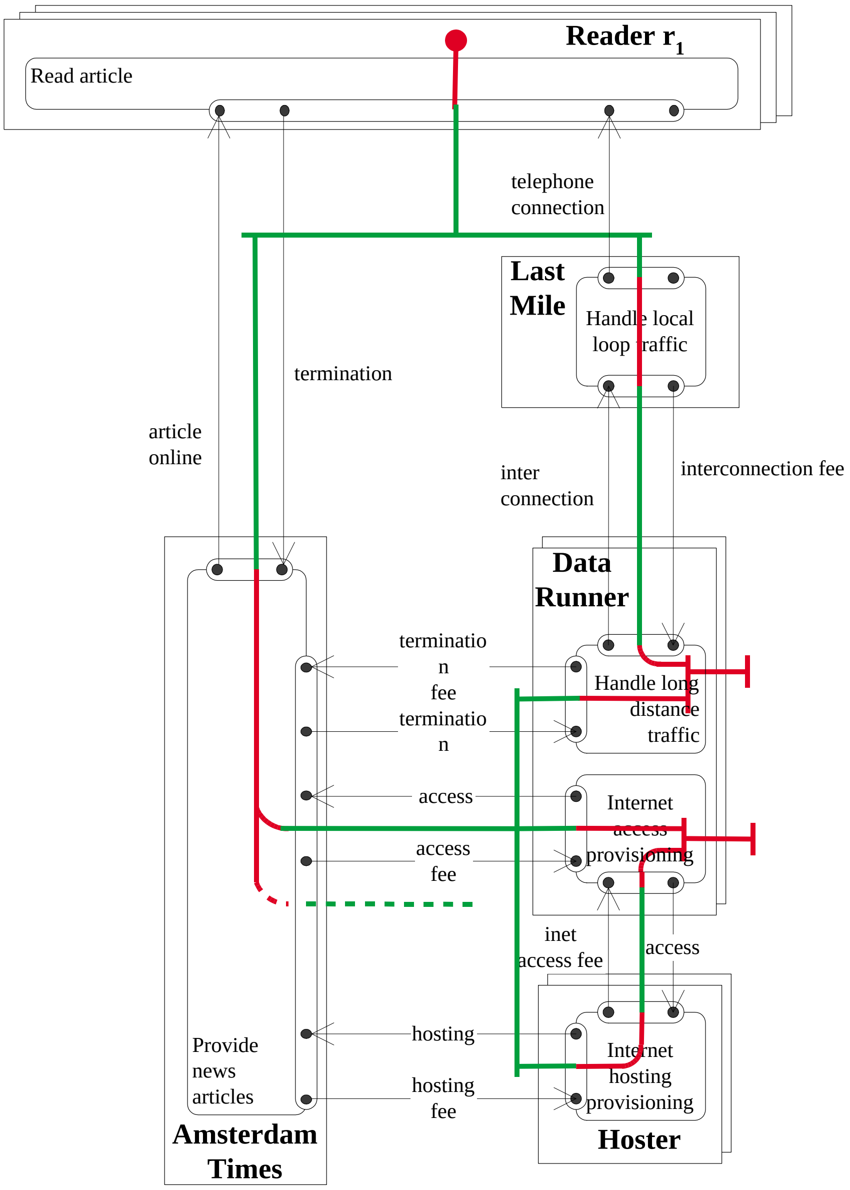

Figure 8: A re-constructed e-business model by assigning newly discovered value activities to actors.

[3] Philip Evans and Thomas S. Wurster. *Blown to Bits - How the New Economics of Information Transforms Strategy*. Harvard Business School Press, Boston, Massachusetts, 2000.

[4] J. Gordijn, J.M. Akkermans, and J.C. van Vliet. What's in an electronic business model. In R. Dieng and O. Corb, editors, *Knowledge Engineering and Knowledge Management - Methods, Models, and Tools, 12th International Conference, EKAW 2000, Juan-les-Prins, France, October 2000*, volume LNAI 1937, pages 257–273. Springer Verlag, 2000. Also available from http://www.cs.vu.nl/˜gordijn.

[5] Amir Hartman, John Sifonis, and John Kador. *Net Ready - Strategies for Success in the E-conomy*. McGraw-Hill, New York, 2000.

[6] P. Kotler. *Marketing management: analysis, planning, implementation and control*. Prentice Hall, Englewood Cliffs, New Jersey, 1988.

[7] Don Tapscott, David Ticoll, and Alex Lowy. *Digital Capital - Harnessing the Power of Business Webs*. Nicholas Brealy Publishing, London, UK, 2000.

[8] Paul Timmers. *Electronic Commerce: Strategies and Models for Business-to-Business Trading*. John Wiley & Sons Ltd., Chichester, England, 1999.