Die approbierte Originalversion dieser Diplom-/Masterarbeit ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this diploma or master thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

MASTERARBEIT

Das Digitale Business Ecosystem als Middleware für Vendor-managed Inventory

zur Erlangung des akademischen Grades

Diplomingenieur (Dipl.-Ing.)

ausgeführt am

Institut für Rechnergestützte Automation
Forschungsgruppe Industrial Software

der Technischen Universität Wien

unter der Anleitung von

Univ.-Prof. Dipl.-Ing. Dr. techn. Thomas Grechenig

Dipl.-Ing. Michael Haselsteiner

durch

Paul Pöltner

Meissauergasse 21/2/2305; 1220 Wien

p@poeltner.co.at

Wien, 01. April 2008

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benützt und die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am ----------

Name

Kurzfassung

In Europa wird ein großer Teil des Bruttoinlandsproduktes von KMUs erwirtschaftet. Diese sehen sich jedoch mit zunehmender Konkurrenz von Seiten der Großunternehmen konfrontiert. Um sich diesen neuen Herausforderungen zu stellen und dabei trotzdem wachsen zu können, werden neue Unternehmensstrukturen benötigt, wie etwa die der virtuellen Organisation. Diese Organisationsform verlangt ein sehr hohes Maß an Kommunikation mittels digitaler Kanäle, um die Prozesse über mehrere Unternehmen hinweg optimieren zu können. Gerade im Bereich des Supply Chain Managements wird diese Herausforderung deutlich. Gesucht werden Konzepte, die im Speziellen den KMUs eine technische Basis für den Aufbau solcher Netzwerke bieten.

Im Rahmen dieser Arbeit werden die Anforderungen an ein modernes Middlewaresystem für das Supply Chain Management erarbeitet und die beiden Systeme SOA (Service Oriented Architecture) und DBE (Digital Business Ecosystem) werden daran gemessen. Als Ergebnis dieser Arbeit wird die Zukunftsfähigkeit dieser beiden Systeme im Rahmen des Supply Chain Managements bestimmt.

In einem weiteren Schritt wird ein Konzept für die Umlegung des Vendor-managed Inventory Modells auf das DBE entwickelt. Der Natur nachahmend wird ein robustes, sich selbst optimierendes System geschaffen, in dem Themen wie intelligente Produkte, Swarm Intelligence oder evolutionäre Algorithmen behandelt werden.

Abstract

Europe is dominated by small and medium-sized enterprises (SME), which generate a big part of Europe's gross domestic product (GDP). These SMEs need a way to face the competition with large enterprises in order to grow and to generate sustainable revenues. New business structures, like a virtual organization, can be a starting point for a solution. But specifically virtual organizations have a lot of technical requirements if they are to be implemented in a real environment. Digital communication is the key to optimizing a business process for more than one company, requiring a reliable business network. Business networks face different challenges. The Supply Chain Management is a god example. Only the appropriate communication concept will work as a solution. On the one hand, this concept has to build up the whole communication infrastructure; on the other hand, it has to be realizable by SMEs.

This thesis investigates the requirements for a modern middleware system that supports the Supply Chain Management. Furthermore new frameworks like SOA (Service-Oriented Architecture) and DBE (Digital Business Ecosystem) will be measured according to these requirements. As a result the sustainability of these frameworks will be evaluated.

The next step is to develop a concept of a vendor-managed inventory system in the DBE. In modeling nature, a new robust and self-optimized system will be created. This system covers new developments in the field of intelligent products, swarm intelligence and evolutionary algorithm.

Inhaltsverzeichnis

Abkürzungen

BPEL	Business Process Execution Language
CBM	Component Business Model
CMM	Capability Maturity Models
CPFR	Collaborative Planning Forecasting and Replenishment
CRP	Continuous Replenishment Programs
DBE	Digital Business Ecosystem
EAI	Enterprise Application Integration
ECR	Efficient Consumer Response
EDLC	Every Day Low Cost
EDLP	Every Day Low Prices System
ERP	Enterprise Resource Planning
ESA	Enterprise Software Architektur
ESB	Enterprise Service Bus
eSCM	elektronisches Supply Chain Management
IKT	Informations- und Kommunikationstechnologie
KMU	Klein- und Mittelbetrieben
OMM	Openness Maturity Models
SCC	Supply-Chain Council
SCM	Supply Chain Management
SCOR	Supply-Chain Operations Reference-Model
SME	Small and medium enterprises
SMR	supplier-manager release

SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
UDDI	Universal Description, Discovery and Integration
VICS	Voluntary Interindustry Commerce Standards
VMI	Vendor-managed Inventory
WS	Web Services
WSDL	Web Service Definition Language

Kapitel 1

Einleitung

Auf dem internationalen Markt stehen Käufer und Verkäufer jeweils einer großen Konkurrenz gegenüber. Um messbare Vorteile gegenüber den Mitbewerbern zu erhalten, werden die internen Prozesse optimiert und die Lieferketten zwischen den einzelnen Handelspartnern synchronisiert. Im Rahmen von Initiativen wie dem CPFR (Collaborative Planning, Forecasting and Replenishment) oder dem ECR (Efficient Customer Response) wird versucht diesen Entwicklungen zu entsprechen und die Kosten auf beiden Seiten signifikant zu reduzieren. Das Ziel ist es Effekten, wie dem „Bullwhip Effect" entgegen zu wirken. Große Handelsketten, wie Wal-Mart, haben solche Systeme bereits erfolgreich umgesetzt.

1.1 Problemstellung

Unter den Gesichtspunkten dieser ständigen Änderungen müssen sich KMUs mit der Bildung von virtuellen Organisationen beschäftigen, um auf dem Markt konkurrenzfähig zu bleiben. Damit eine solche virtuelle Organisation aufgebaut werden kann, verlangt es nach stabilen und für KMUs finanzierbaren technischen Basissystemen. Diese Basissysteme müssen den dynamischen Unternehmensleitbildern von KMUs entsprechen und sich in Folge auch an diese anpassen lassen.

1.2 Motivation

Die aktuellen Entwicklungen im Bereich von verteilten Systemen sprechen vielfach für den Aufbau von serviceorientierten Architekturen. Gleichzeitig hat sich im Rahmen einer Initiative der Europäischen Kommission das Digitale Business Ecosystem entwi-

ckelt, das einen innovativen Weg für den Aufbau von verteilten Systemen darstellt. Im Rahmen dieser Arbeit wird die Frage beantwortet, ob das DBE eine Art SOA ist und ob SOA oder das DBE als Middlewaresystem für das Supply Chain Management und im Speziellen für das Vendor-managed Inventory (VMI) Modell eingesetzt werden können.

1.3 Zielsetzung

Sowohl SOA als auch das DBE basieren im Grundgedanken auf einer serviceorientierten Struktur. Beide erlauben es Prozesse abzubilden und unterschiedlichste proprietäre Systeme miteinander zu verbinden. Es soll daher gezeigt werden, dass zunächst der Zusammenhang zwischen SOA und DBE sehr wohl gesehen werden kann. In der weiteren Folge soll eine Design-Architektur die theoretische Möglichkeit bestätigen, dass VMI unter den Paradigmen des DBE (Evolutionäre Algorithmen, selbstorganisiert, …) im DBE umgesetzt werden kann. Darauf aufbauend soll diese Arbeit eine Grundlage für die Vermittlung des Wissens über das DBE an KMUs bieten.

1.4 Aufbau der Arbeit

Die Arbeit gliedert sich in drei Teilbereiche. Der erste theoretische Teil im Kapitel 2 behandelt die technischen Anforderungen an Middlewaresysteme für das Supply Chain Management (SCM). Dazu fließen Themen wie das „Open ICT Ecosystem“ und die Entwicklungen aus dem SCM ein. Als Ergebnis werden die Anforderungen an ein Middlewaresystem dargestellt. Der zweite Teil der Arbeit stellt die beiden Systeme SOA und DBE vor. SOA wird im Kapitel 3 beschrieben und im letzten Teil des Kapitels mit den Anforderungen aus dem Kapitel 1 verglichen. Dasselbe wird für das DBE im Kapitel 4 durchgeführt. Verstärkt werden hier der evolutionäre Ansatz und der Aufbau des Projektes präsentiert. Der dritte, praktische Teil der Arbeit im Kapitel 5 beschäftigt sich mit der Frage in wieweit ein Vendor-managed Inventory System (VMI) mit Digital Business Ecosystem (DBE) umgesetzt werden kann. Das VMI verlangt vom Käufer, dass dieser dem Verkäufer alle benötigten Informationen über den Point of Sales (POS) mitteilt, welcher daraufhin innerhalb eines Service Level Agreements (SLA) das Lager des Käufers auffüllt. Derzeit wird dies meistens über ein Clearingsystem durchgeführt. Gerade kleinere Unternehmen können auf Grund der Kosten für die Umsetzung an solchen Projekten nicht teilnehmen. Das DBE setzt hierbei einen Kontrapunkt. Gezeigt wird, wie ein solches System ohne Clearingsystem oder sonstige Barrieren für KMUs funktionieren kann.

Teil I

Supply Chain Management

Kapitel 2

Supply Chain Anforderungen an ein Middleware-System

Die Anforderungen an ein modernes Middlewaresystem für die Abbildung der Supply Chain Prozesse sind sehr vielschichtig. Die Umsetzung verlangt einen automatischen Datenaustausch zwischen vielen Teilnehmern und die Rolle der IT-Anwendungen nimmt einen immer höheren Stellenwert ein. Damit neue technische Entwicklungen den Anforderungen entsprechen können, muss ein neues System eine Vielzahl an Eigenschaften aufweisen. In diesem Kapitel werden diese Eigenschaften bzw. Anforderungen an ein mögliches zukünftiges Middlewaresystem beschrieben. In den weiteren Kapiteln wird SOA und DBE diesen Anforderungen gegenüber gestellt.

Die Kriterien für die erfolgreiche Umsetzung ergeben sich auf der einen Seite aus den wirtschaftlichen Bedingungen, die aus den strategischen Zielsetzungen im Unternehmen resultieren. Im ersten Teil wird das Thema Supply Chain Management allgemein besprochen und dessen Einbettung in einem Unternehmen. Im Speziellen wird der „Bullwhip"-Effekt dargestellt. Dieser Effekt wirkt sich entscheidend auf die Performance einer Supply Chain aus und kann durch die richtige Koordination verhindert bzw. minimiert werden.

Auf der anderen Seite werden von verschiedensten Quellen, die in diesem Kapitel erläutert werden, Anforderungen aus technischer Sichtweise vorgegeben. Diese Kriterien für ein modernes und zukunftsweisendes System werden im zweiten Teil des Kapitels dargelegt.

Die Zusammenfassung der wirtschaftlichen und technischen Sichtweise bildet die Basis für den Anforderungskatalog an ein modernes Middlewaresystem. Dieser Katalog wird in den weiteren Kapiteln dem SOA (Kapitel 3.4) und dem DBE (Kapitel 4.6) gegenübergestellt.

2.1 Wirtschaftliche Anforderungen

2.1.1 Warum Supply Chain Management?

Die Koordination der Supply Chain ist keine Aufgabe, die jede Abteilung oder jedes Unternehmen für sich alleine durchführen kann. Es ist eine Managementaufgabe die Unternehmensgrenzen überschreitet. Der Begriff wurde 1982 von Oliver und Webber das erste Mal verwendet.

> *„Supply-chain management covers the flow of goods from supplier through manufacturing and distribution chains to end-user“*[1]

In den Jahren darauf entwickelten sich viele verschiedene Definitionen für den Begriff Supply Chain Management (SCM) und jeder dieser Begriffe bezieht sich auf einen anderen Aspekt des Supply Chain Managements.

Diese Arbeit bezieht sich auf die Definition von Bowersox et al.. Die Autoren beschreiben das Supply Chain Management als ein Beziehungsmanagement zwischen vielen Unternehmen in einem Framework, das durch Beschränkungen in:

- der Kapazität,
- der Information,
- der Kernkompetenzen,
- dem Kapital
- und dem Personal

gekennzeichnet ist. Das integrierte Unternehmen befindet sich in einem Netzwerk von Beziehungen, das auf der einen Seite dem Lieferantennetzwerk (Supplier Network) und auf der anderen Seite dem Absatz-Netzwerk (Distributive Network) gegenüber steht (Abbildung 1). [2]

Die hohe Bedeutung des Supply Chain Managements lässt sich aus den Potentialen erklären, die bei einer richtigen Umsetzung in einem Unternehmen entstehen kann. Wie in den weiteren Kapiteln noch näher erläutert wird, beschreibt Steve New, dass in Zukunft nicht mehr entscheidend sein wird was ein Unternehmen macht, sondern wie es gemacht wird. Unternehmen müssen sich über den Prozess von Konkurrenten unterschei-

[1] (Oliver & Webber, 1982)

[2] vgl. (Bowersox, Closs, & Cooper, 2002) S.6

den.[3] Das Schlagwort, das hier, wie durch viele andere Managementstrategien geht, heißt Prozessmanagement. Im Kapitel 2.1.4 wird dieser Ansatz näher dargestellt.

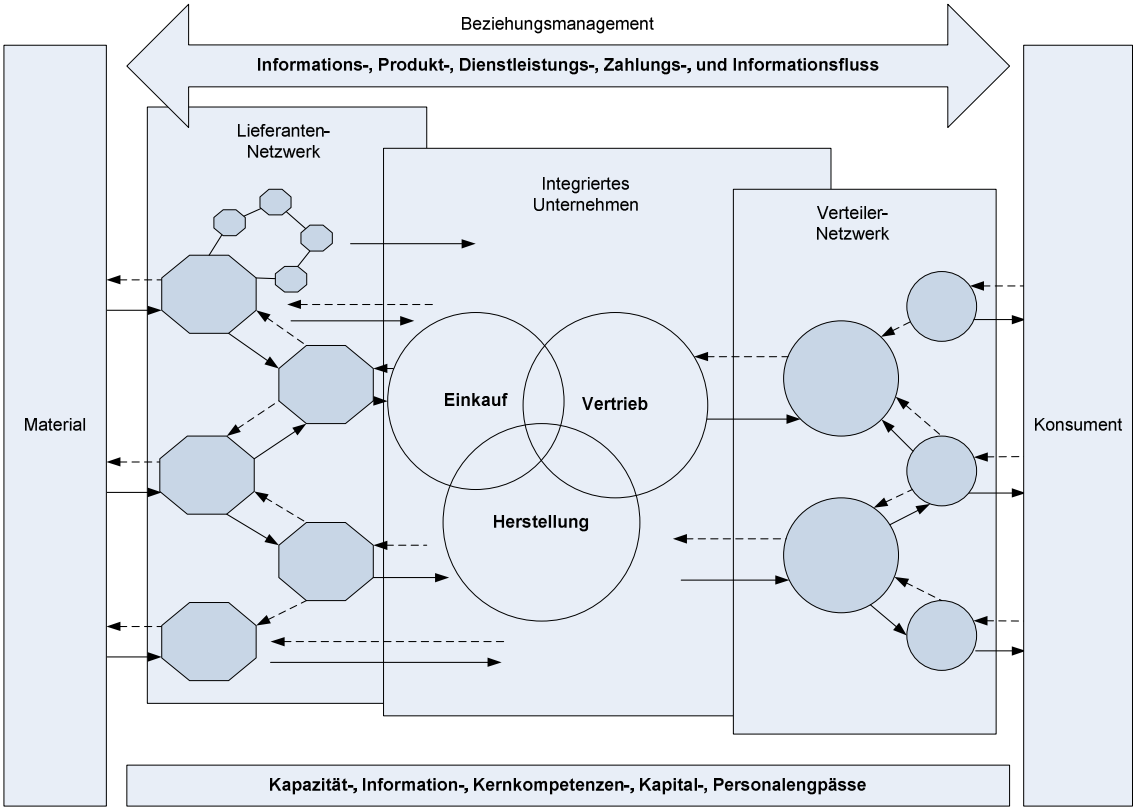

Abbildung 1: Supply Chain Management

Quelle: (Bowersox, Closs, & Cooper, 2002)

Der Nachholbedarf in der Koordination der Supply Chain wurde durch die Darstellung des Bullwhip- sichtbar. Das Phänomen wurde bereits in vielen Artikeln und Untersuchungen aufgearbeitet. Lee et al. haben im Artikel „The Bullwhip Effect in Supply Chains“ versucht die Gründe für den Bullwhip Effekt zu erklären. Spätere Arbeiten vervollständigten den Ansatz von Lee et al. um weitere Aspekte. [4]

Der Bullwhip Effekt ist ein Phänomen, bei dem sich die Bestellschwankungen für ein Produkt von der Ebene des Verkäufers bis zur Ebene des Produzenten merkbar verstärken, ohne dass die Kundennachfrage diesen Schwankungen unterliegt. So kann ein Pro-

[3] (New & Westbrook, 2004)

[4] (Lee, Padmanabhan, & Whang, 1997)

dukt, bei dem der Verbrauch beim Kunden im Durchschnitt über das Jahr gleich ist, beim Produzenten mit starken Schwankungen verbunden sein. Lee et al. beschreiben eine Untersuchung von Hewlett-Packard. Hewlett-Packard erkannte, dass zwar der Verkauf ihrer Drucker gewissen Schwankungen beim Verkäufer unterliegt, jedoch der Bestelleingang bei Hewlett-Packard mit wesentlich höheren Schwankungen verbunden ist. [5] Im Rahmen des Artikels werden vier Gründe für den Effekt genannt.

Der erste Effekt beschreibt die Probleme einer dezentralen Absatzplanung. Jedes Unternehmen erstellt auf Grund der bisherigen Verkäufe eine Bedarfsanalyse. Diese Bedarfsanalyse dient als Basis für die nächste Bestellung. Zusätzlich wird für die Lieferzeit ein Sicherheitsbestand angelegt. Wird von jedem Unternehmen entlang der Lieferkette eine solche Analyse durchgeführt, so erhöht sich der Bedarf sehr rasch, obwohl der Absatz immer gleich bleibt. Der Effekt wird zusätzlich über die Losbildung verstärkt. Um Kosten einzusparen, werden Bestellungen zum Teil nicht kontinuierlich, sondern in bestimmten Intervallen zusammengefasst durchgeführt. Diese Bestellungen werden zum Teil wöchentlich, monatlich oder in noch längeren Abständen weitergeleitet. Gründe für diese Überlegungen sind unter anderem die Kosten, die für einen nicht voll ausgelasteten LKW entstehen. Ein weiterer Faktor sind die Kosten, die bei nicht automatischen Abläufen für den Prozess der Bestellung selbst entstehen. Aus diesen Gründen werden die Bestellungen zusammengefasst und verursachen so Schwankungen.[6]

Preisschwankungen am Markt sind der dritte Faktor, den Lee et al. darlegt. Auf Grund von Werbeaktionen oder Aktivitäten am Markt verändert sich der Preis. Der Händler wird daher dann seine Waren kaufen, wenn der Preis niedrig ist. Gleichzeitig verstärken Engpässe am Markt die Schwankungen, denn der Käufer neigt zu Überbestellung bei Lieferengpässen. Kommt es beim Produzenten zu Engpässen, muss dieser nach einer gewählten Strategie seine Käufer beliefern, damit der Engpass so schnell als möglich behoben werden kann. Dafür wird jeder Käufer zunächst mit einem Teil der bestellten Ware beliefert, damit jeder Käufer etwas bekommt. In einem zweiten Schritt wird die noch ausständige Liefermenge verteilt. Der Käufer wartet jedoch nicht inaktiv bis neue Ware geliefert wird, sondern bestellt bei mehreren Verkäufern eine größere Menge. Erst wenn die Ware geliefert wurde, werden die nicht benötigten Mengen abbestellt. [7]

Das Konzept von Lee et al. wurde von Lödding um vier weitere Gründe für das Entstehen des Bullwhip-Effekts erweitert. Der erste Punkt betrifft den Effekt der konstanten Durchlaufzeiten. Das Problem wird so beschrieben, dass durch starre Durchlaufzeiten

[5] vgl. (Lee, Padmanabhan, & Whang, 1997)

[6] vgl. (Lee, Padmanabhan, & Whang, 1997)

[7] vgl. (Lee, Padmanabhan, & Whang, 1997)

und variable Nachfragen im Lager Schwankungen entstehen. Diese Schwankungen verursachen Kosten. Gleichzeitig basieren Warenbestellungen auf vergangenheitsbasierenden Prognosen. Neue Waren werden dann bestellt, wenn der festgelegte Bestellbestand unterschritten wurde. Die Bestellmenge wird vom Mittelwert und der Standardabweichung bestimmt, die auf vergangenheitsbasierenden Daten beruht. Auf Grund der Änderungen der Mittelwerte und Standardabweichungen entstehen unterschiedliche Sicherheitsbestände, die zusätzlich Bedarfsschwankungen verursachen. Der dritte Punkt betrifft die Informationslaufzeit. Diese ist entscheidend für die Durchführung von adäquaten Prognosen. Werden erhöhte Nachfrageanforderungen zu spät an den Hersteller weitergeleitet, so kann dieser nicht rechtzeitig reagieren. [8]

Der letzte Punkt von Lödding hebt die Kapazitätsrestriktionen hervor. Restriktionen können zu Lieferengpässen führen und somit den Bullwhip-Effekt verstärken. [9]

Die beschriebenen Resultate dieses Effekts sind hohe Warenbestände, schlechte Vorhersagen des Herstellers, Produktionsengpässe, Kundenunzufriedenheit, unsichere Produktionsplanung und damit entscheidend höhere Kosten auf beiden Seiten.

Um diesen Phänomenen entgegen wirken zu können, muss nicht nur die Infrastruktur verändert werden, sondern auch das Verhalten bei der Durchführung der Bestellungen. Lee et al. beschreibt vier Lösungsansätze für die Verringerung des Effekts.

Zunächst sollten auf weniger Ebenen Bedarfsanalysen durchgeführt werden. POS-Daten könnten die Lieferkette aufwärts bis zum Hersteller weitergereicht werden. Dieser kann daraufhin für die gesamte Lieferkette eine Analyse durchführen, bzw. seine Produktion zumindest auf den tatsächlichen Bedarf abstimmen. Ein Modell, das hier eingesetzt werden kann, ist das Vendor-managed Inventory Modell (VMI) als eine Form des Continuous Replenishment Programs (CRP). Darüber hinaus sollten die Bestellungen in kürzeren Abständen erfolgen. In diesem Artikel wird sehr stark der Einsatz von EDI forciert, damit die Bestellungen schnell verarbeitet werden können. Weiters sollten Bestellungen von mehreren Produkten für einen LKW zusammengefasst werden. Der dritte Schritt ist die Stabilisierung der Preise. Über ein Every Day Low Prices System (EDLP), das auf einer fundierten Preispolitik für Großhändler basiert, kann die Zusammenfassung von Bestellungen verhindert werden. Auf der Seite der Distributionszentren kann ein Every Day Low Cost (EDLC) System dasselbe bewirken. Das letzte Problem,

[8] vgl. (Lödding, 2005) S.109ff

[9] vgl. (Lödding, 2005) S.177f

der Überbestellung bei Lieferengpässen, kann über ein klares Informationssystem der Hersteller an den Händler verringert werden. [10]

Die soeben beschriebenen Aktionen haben das Potential den Bullwhip-Effekt entscheidend zu verringern, verlangen allerdings eine starke Kommunikation zwischen den einzelnen Interaktionspartnern. Diese Anforderungen müssen nicht nur über eine funktionierende Managementstrategie umgesetzt werden, sondern auch über eine technische Plattform, die eine schnelle Anpassung an Veränderungen erlaubt. Schnittstellen wie EDI sind diesen modernen Anforderungen in Zukunft nicht mehr gewachsen, wodurch Raum für neue Technologien geschaffen wird.

Ein Ziel ist es daher, durch neue Technologien den Bullwhip-Effekt zu verringern und Kosteneinsparungen dadurch zu erreichen.

2.1.2 Die Komplexität des Supply Chain Managements

Damit neue Technologien gezielt eingesetzt werden können, muss das gesamte komplexe Konstrukt des Supply Chain Managements skizziert werden. Die Komplexität ergibt sich aus den vielfältigen Anforderungen und aus der überaus starken Nachfrage nach Kommunikation. Die Entscheidung des Kunden für ein Produkt wird nicht mehr allein auf Grund des Produktes selbst getroffen, sondern auf Grund der Einbettung in ein erfolgreiches System von Unternehmen, die den Anforderungen des Kunden entlang des Lebenszyklus des Produktes entsprechen.[11]

Den Ausgangspunkt bildet die Strategiedefinition in den jeweiligen Unternehmen. Entscheidend für eine erfolgreiche Umsetzung ist, dass die Strategien in Abhängigkeit mit der Business Strategie entwickelt wurden. Diese gibt den gesamten Fahrplan des Unternehmens vor. Abgeleitet davon beschreibt Cohen et al. fünf Strategien, die für ein erfolgreiches Supply Chain Management entwickelt werden müssen.

Die erste Strategie ist die „Operation Strategy“. Diese Strategie betrifft die Entscheidung über die Art der Warenproduktion (z.B.: make-to-stock, make-to-order, …) selbst. Die zweite Strategie ist die „Channel Strategy“. Diese definiert, wie die Produkte zum Käufer gelangen. Aufbauend auf den vorangegangenen Strategien folgt die „Outsourcing Strategy“. Outsourcing wird auf der Basis der definierten Kernfunktionen des Unternehmens entschieden. Im Supply Chain Management können mit Outsourcing eigene

[10] (Lee, Padmanabhan, & Whang, 1997)

[11] vgl. (New & Westbrook, 2004)

Schwächen ausgeglichen werden. Die „Customer Service Strategy“ erarbeitet die Bedürfnisse der Kunden gereiht nach Größe und Profitabilität der Kundengruppe. Anschließend werden die zu implementierenden Eigenschaften gewählt. Gemeinsam mit den anderen Strategien kann das Asset Network abgeleitet werden. Es definiert welche Bereiche das Unternehmen selbst betreiben soll.[12]

Supply Chain Management sollte jedoch nicht im Unternehmen als eine Strategie für sich alleine stehen, sondern vielmehr die Marketing-Strategie unterstützen. New und Westbrook definieren den Begriff der market-driven Logistik Strategie:

> *„Logistics impact marketing effectiveness through the customer value that is created by superior service“*[13]

Der Kundenwert wird definiert als der Prozentsatz der Division zwischen dem wahrgenommenen Kundenmehrwert und dem Total Costs of Ownership (TCO).

$$\text{Kundenwert} = \frac{\text{Kundenmehrwert}}{\text{Total Costs of Ownership}} * 100$$

Abbildung 2: Kundenwert

Die TCO kann über die Verhinderung des Bullwhip-Effekts stark beeinflusst werden und so den Kundenwert erhöhen. Die Mehrwerte für den Kunden ergeben sich aus den wahrgenommenen Leistungen, die er über den gezahlten Preis, inklusive der Nebenkosten, hinaus bekommt. Ausschlaggebend dafür können Eigenschaften wie pünktliche Lieferung, kurze Lieferzeiten oder flexible Bestellzeitpunkte sein. Das Ziel der market-driven Logistik Strategie ist es die Anforderungen für die Erhöhung des Kundenwertes auf die Supply-Chain Strategie umzulegen. Das Ergebnis bildet die Wahl der richtigen Supply Chain Strategie für ein Produkt. [14] Darauf aufbauend stellt sich die Frage, welche Anforderungen an die Interaktionspartner und Infrastruktur für die Umsetzung gestellt werden.

---

[12] vgl. (Cohen & Roussel, 2005) S.10ff

[13] (New & Westbrook, 2004)

[14] vgl. (New & Westbrook, 2004)

Auf oberster Ebene lassen sich zwei Strategien unterscheiden. Die Lean Design Strategie beschreibt eine Supply Chain für Standardprodukte, die keinen großen Schwankungen unterliegen. Klassische SCM-Ansätze entsprechen diesen Anforderungen. Die Anforderungen können über Vorhersagen auf der Grundlage von bestehenden Bestellungen durchgeführt werden. Das Ziel ist die benötigten Produkte zur gewünschten Qualität am Billigsten zu liefern.[15]

Die zweite Strategie ist vom Markt stark abhängig und wird als Agile Supply Chain bezeichnet. Die Nachfrage kann nur schwer vorhergesagt werden und unterliegt starken Schwankungen. Die Supply Chain sollte von der Nachfrage bestimmt werden und nicht von der Vorhersage der Nachfrage. Da die Umsetzung nicht sehr einfach ist, muss trotzdem noch auf die Vorhersage von Nachfragen zurückgegriffen werden. Die Einführung setzt einen hohen Informationsaustausch zwischen den Unternehmen in einer Lieferkette voraus, der über einen starken Einsatz von Informationstechnologie umgesetzt werden muss. Prozesse müssen unternehmensübergreifend definiert und eingesetzt werden.[16]

Auf der Basis der gewählten Strategie muss eine Eingliederung des Unternehmens in die Supply Chain durchgeführt werden. Für die Umsetzung der Strategien sind, wie in dem von Bowersox et al. definierten integrierten Unternehmen, viele verschiedene Ausprägungen von Beziehungen notwendig.[17] Cohen et al. definiert vier Ebenen der Zusammenarbeit, wobei die Abgrenzung zwischen den einzelnen Ebenen fließend verläuft. Die erste Ebene wird als „Transactional Collaboration“ bezeichnet. Es erfolgt in diesem Fall nur eine geringe Abstimmung zwischen den Interaktionspartnern. Konkrete Ausprägungen sind zum Beispiel Preisfestsetzungen für eine bestimmte Dauer zwischen Käufer und Verkäufer. Die nächste Ebene ist die „Cooperative Collaboration“. Dabei werden bereits Informationen zwischen den strategischen Partnern ausgetauscht. Ein verwendetes Format ist z.B. EDI. Eine tatsächlich abgestimmte Zusammenarbeit wird in der Ebene „Coordinated Collaboration“ eingeführt. Dabei sind die strategischen Partner auf die Fähigkeiten des anderen Partners angewiesen. Eine Ausprägung dieser Strategie ist das Vendor-managed Inventory Modell. Die höchste Steigerungsstufe bildet das „Synchronized Collaboration“ Modell. Auf dieser Stufe werden die Unternehmensprozesse über das Maß der normalen Zusammenarbeit abgestimmt. Es entsteht eine hohe Abhängigkeit zwischen den einzelnen Partnern.[18]

---

[15] vgl. (New & Westbrook, 2004)

[16] vgl. (New & Westbrook, 2004)

[17] vgl. (Bowersox, Closs, & Cooper, 2002) S.5

[18] vgl. (Cohen & Roussel, 2005) S.143ff

New und Westbrook definieren die Anforderungen an zukünftige Lieferketten durch eine Konzentration vom Massenmarkt zu den eins-zu-eins Marktplätzen. Wo in der Vergangenheit die Unternehmen mehr für sich agierten, stehen virtuelle Netzwerke im Vordergrund. Statt reinen „low-cost" Produkten, steht der oben definierte „Kundenwert" im Vordergrund, der unter einer einheitlichen Betrachtung des gesamten Prozesses entsteht. [19]

2.1.3 Frameworks für das Supply Chain Management

Zur Unterstützung der Findung, Planung und Koordination von Kooperationsmodellen wurden mehrere Managementtools entwickelt. Diese helfen die Umsetzung zwischen den einzelnen Partnern besser zu koordinieren und in weiterer Folge überwachen zu können.

Eines dieser Konzepte ist das Supply-Chain Operations Reference-Model (SCOR). SCOR wird in dem gleichnamigen Bericht beschrieben und wurde vom Supply-Chain Council (SCC) entwickelt. Das Modell umfasst einen Handlungsleitfaden für den Aufbau von Partnerprogrammen zwischen Unternehmen. Dieser Leitfaden beschreibt unter anderem Kunden-Interaktionen, Produkt-Transaktionen und Markt-Interaktionen, die auf die jeweilige reale Situation angepasst werden können. Im Detail werden fünf sogenannte SCOR Toplevel-Prozesse beschrieben. [20]

Der erste Prozess wird als „PLAN" bezeichnet. Dieser Prozess behandelt die Nachfrage- und Angebotsplanung für die gewählten Produkte des Partnerprogrammes. In weiterer Folge wird dieser Prozess auch die Verwaltung der Nachfrage- und Angebotsplanung übernehmen.

Die Beschaffung der Waren auf Grund der Lagerstrategien der Unternehmen wird vom Prozess „SOURCE" beschrieben und geht in den Prozess „MAKE" über. Aufgabe des Prozesses „MAKE" ist die Erstellung einer Strategie für die selbsterstellten Produkte. Diese Produkte werden über einen eigenen Prozess verschickt. „DELIVER" kümmert sich um den Warenfluss, dabei werden die Aufgaben Auftragsverwaltung, Lagerverwaltung und Transportmanagement durchgeführt.

---

[19] vgl. (New & Westbrook, 2004)

[20] vgl. (Supply-Chain Council, 2006)

Der letzte Prozess übernimmt die Koordination des After-Sales Services. Dieser Prozess wird als „RETURN“ bezeichnet und nimmt in den modernen Supply Chains einen immer höheren Stellenwert ein.

Im Rahmen dieser fünf Toplevel-Prozesse wird ein Referenzmodell zur Verfügung gestellt, das in drei Detailebenen in den einzelnen Unternehmen implementiert werden kann. Zu jedem Prozess werden zusätzlich Benchmark-Kriterien angeboten, um die Prozesse über die Unternehmensgrenzen hinaus überwachen und vergleichen zu können.

Besonders interessant sind gerade auch aus der Sicht auf ein potentielles Middleware-System die Anforderungen die SCOR an das Supply Chain definiert. Diese Anforderungen werden unabhängig von den einzelnen Prozessen gemessen und optimiert. [21]

Die erste Anforderung des Systems besteht darin zuverlässig zu sein („Reliability“). Zuverlässigkeit in diesem Zusammenhang bedeutet, dass das System die Aufträge in der gewünschten Art und Weise durchführt. Die nächste Anforderung betrifft die Reaktionszeit des Systems („Responsiveness“). Wie im Kapitel zum Thema Bullwhip Effekt bereits beschrieben, spielt die Zeit in der auf Bestellungen reagiert werden kann, eine wichtige Rolle. Nicht nur im Supply Chain Management müssen Unternehmen flexibel auf neue Veränderungen reagieren können. Diese Voraussetzung wird in Zukunft immer wichtiger werden, wenn sich die Technologien immer schneller verändern.[22] Auch SCOR hat diese Forderungen in den Katalog aufgenommen ,(„Flexibility“).[23]

Der vierte Punkt misst die Kosten („Cost“). Dieses Kriterium misst die Performance der Supply Chain Verwaltung und gleichzeitig liefert dieses Kriterium ein Benchmark-Kriterium für die entstehenden Kosten der verkauften Waren. Der letzte Punkt ist ein Maßstab für das Vermögen des Unternehmens („Assets“). Unter anderem wird die Cash-to-Cash Cycle Time, Return on Fixed Assets und dem Return on Working Capital gemessen. [24]

Neben SCOR wurde ein weiteres Framework mit speziellem Augenmerk auf Zusammenarbeit entwickelt. Dieses Modell wird als Collaborative Planning Forecasting and Replenishment (CPFR) bezeichnet.

---

[21] vgl. (Supply-Chain Council, 2006)

[22] vgl. (Universität Duisburg Essen, 2007) S.2

[23] vgl. (Supply-Chain Council, 2006)

[24] vgl. (Supply-Chain Council, 2006)

CPFR wird in dem gleichnamigen Bericht von VICS beschrieben. Das Modell wurde 1998 zum ersten Mal publiziert und seit dem weiterentwickelt. In diesem Referenzmodell wird die Zusammenarbeit von verschiedenen Partnern beschrieben, um die Kundennachfrage zu erfüllen. Wie der Titel des Modells schon erahnen lässt, beschreibt es den Prozess der Planung, Vorhersage und Lagerauffüllung. Innerhalb der vier definierten Aktivitäten „Strategie und Planung", „Nachfrage- und Liefermanagement", „Ausführung" und „Analyse" werden die einzelnen Schritte für eine funktionierende Zusammenarbeit beschrieben. Das CPFR Modell ist ein weiterer Schritt in eine synchronisierte Zusammenarbeit. Vorgehensweisen wie VMI können einen Bestandteil dieses Modelles bilden. Als Beispiel kann hier das CPFR Modell „DC Replenishment Collaboration" erwähnt werden. Dieses beschreibt die Weiterentwicklung bzw. die Umsetzung eines Vendor-managed Inventory Modells. [25]

Sowohl SCOR als CPFR bieten einen guten Ansatzpunkt für die Abbildung der Aufgaben eines Middlewaresystems. Sie bieten darüber hinaus auch die Basis für die Überprüfung einer konkreten Umsetzung in einem Unternehmen und geben an, welche Prozesse während des gesamten Ablaufs einer Neueinführung beachtet werden müssen.

2.1.4 Prozessmanagement

Dr. Wißkirchen schreibt in einem Artikel, dass Unternehmen ihre Kosten durch die Bündelung von Prozessen minimieren können. [26] So sollen gleiche Prozesse aus mehreren Abteilungen zusammengefasst und nur einmal durchgeführt werden. Diese Aussage basiert auf einer Reihe von Entwicklungen, die sich im Rahmen der Prozessanalyse entwickelt haben. Begriffe wie Business Process Reengineering oder Business Process Management sind Strategien, die Unternehmen mit weitreichenden Veränderungen konfrontiert haben. Die Prozesssichtweise erlaubt eine Konzentration des Unternehmens auf die tatsächlichen Aufgaben und deren Umsetzung. Aus diesem Ansatz heraus kann das Component Business Modell (CBM) von IBM vorgestellt werden. [27]

Um in der Zukunft den Erfolg des Unternehmens gewährleisten zu können, verweist Sandy Carter auf die Bedeutung, wie Geschäftsführer ihr Unternehmen sehen. [28] IBM stellt in diesem Modell einen Ansatz vor, wie ein Unternehmen in Prozessen zusam-

---

[25] vgl. (Voluntary Interindustry Commerce Standards (VICS), 2004)

[26] vgl. (Wißkirchen, 2002)

[27] vgl. (IBM Global Business Services, 2007)

[28] vgl. (Carter, 2007) S.24

mengesetzt werden kann. Das Unternehmen ist ein Netzwerk von Bausteinen, die das gesamte Leistungsspektrum abbilden. Nicht mehr die einzelnen Funktionsbereiche stehen im Vordergrund, sondern die kleinsten Teilaufgaben, die zu einem Gesamtkonzept zusammengestellt werden.[29] Diese Sichtweise bildet einen starken Bezug zum SOA-Konzept (Kapitel 3).

Um diese Sichtweise auf das Unternehmen zu erhalten, muss das Unternehmen laut Carter in seine einzelnen Bestandteile zerlegt werden. Nur so kann die Bedeutung jedes einzelnen Bausteins bewertet werden. Das Ziel ist es ein Component Business Model zu erstellen (Abbildung 3).[30]

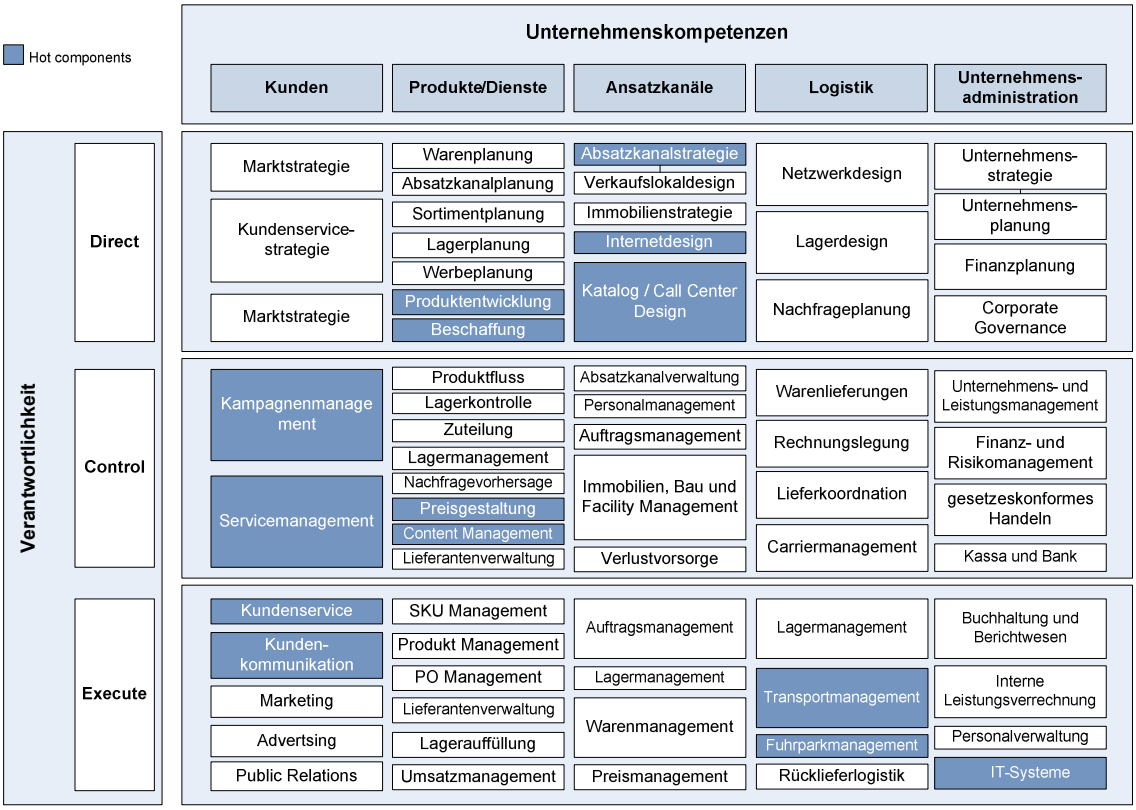

Abbildung 3: Component Business Model

Quelle: (Carter, 2007)S 28

[29] vgl. (IBM Global Business Services, 2007)

[30] vgl. (Carter, 2007) S.28

Die einzelnen Kompetenzen werden auf drei Levels aufgeteilt. Die Ebene „direct“ übernimmt die high-level oder strategischen Aufgaben. Die Ebene „control“ soll das Gleichgewicht zwischen „direct“ und „execute“ herstellen. Die unterste Ebene „execute“ stellt die konkreten Business-Aktionen dar. Durch eine Markierung können die kritischen Komponenten hervorgehoben werden. In einem zweiten Schritt werden die gefundenen Komponenten mit den am Markt befindlichen Unternehmen verglichen. Es soll herausgefunden werden, welche Tätigkeiten das Unternehmen von anderen Unternehmen unterscheidet. Im letzten Schritt muss entschieden werden, in welche Komponenten investiert wird und welche ausgelagert werden. Das Ergebnis soll ein Leitfaden für die nächsten Unternehmensaktivitäten sein. Er wird ebenfalls verwendet um die Aufgaben für die nächsten IT Aktivitäten und die zu verwendende Hard- und Software zu beschreiben. Carter hebt hervor, dass das CBM die Basis für die Verbindung zwischen Business und IT bildet.[31]

Für ein Middlewaresystem, das diesen Anforderungen entsprechen soll, ergibt sich eine stark serviceorientierte Sichtweise. Die Ergebnisse aus dem Prozessmanagement definieren damit die einzelnen Dienste („Services“), die auf die IT-Infrastruktur übertragen werden.

2.1.5 eSupply Chain Management (eSCM)

Im Zeitalter der Vernetzung wurden auch für das Supply Chain Management neue Begriffe entwickelt, die diese Veränderungen beschreiben. Im Bereich des eBusiness wird Supply Chain Management als eSCM definiert. In dem gleichnamigen Buch von Wannenwetsch et al. wird für diesen Begriff folgende Definition dargestellt:

> *„eSCM ist eine Bezeichnung für die Planung, Steuerung und Integration sämtlicher Waren-, Informations- sowie Finanzflüsse entlang der Supply Chain, innerhalb eines Unternehmens sowie unternehmensübergreifend mit Hilfe moderner eBusiness-, Informations- und Kommunikationstechnologien.“* [32]

Das eSCM umfasst eine Vielzahl an Aufgaben, die über das Internet bzw. über moderne Kommunikationstechnologien abgehandelt werden sollen. Für die Neueinführung eines Middlewaresystems in einem Unternehmen müssen diese Aufgaben umgesetzt werden

---

[31] vgl. (Carter, 2007) S.28

[32] vgl. (Wannenwetsch & Nicolai, 2004)

können. Wannenwetsch et al. beschreibt sechs Teilaufgaben des eSupply-Chain Managements.

Das eProcurment beschäftigt sich mit der Beschaffung von Waren für die Produktion oder den Weiterverkauf. Teilaufgaben umfassen das Katalogmanagement, die Lieferantensuche, die Bestellabwicklung sowie die Durchführung von Online-Auktionen. [33]

Aber nicht nur die Beschaffung von Waren kann elektronisch abgehandelt werden. eProduction beschreibt die Produktion von Waren auf der Basis von durchgeführten Bestellungen. Als Leitwort kann hier das Pull-Prinzip genannt werden. Die Produktion beginnt mit dem Eingang des Kundenauftrags und der gesamte Fertigungsprozess wird über ein elektronisches Kommunikationsmedium abgehandelt.[34]

Neben der Erstellung der Waren, spielt im Bereich des Absatzes der elektronische Weg eine gewinnbringende Rolle. Über eMarketing kann ein großer Absatzmarkt für die eigenen Produkte erschlossen werden, der auf Zwischenhändler verzichtet.

Als weitere Aufgabe wird das eService genannt. Durch Zusatzleistungen möchten sich Unternehmen von der Konkurrenz unterscheiden. eService ermöglicht ungeahnte Wege dem Kunden direkt Leistungen anzubieten.[35] Im Zuge dieser Dienste können über eSales dem Kunden neue Produkte oder Verträge verkauft werden. eSales wird als Teilbereich des eCommerce gesehen und beschreibt den Verkauf von Gütern und Dienstleistungen über elektronische Kanäle.[36]

Die letzte Strategie von Wannenwetsch et al. ist eDistribution. Über diese Aufgabe soll die Kundenzufriedenheit durch die Umsetzung einer “Efficient Consumer Response” (ECR) erreicht werden. Moderne Kommunikationsmittel sind dabei nötig um den tatsächlichen Bedarf schnellstmöglich befriedigen zu können.[37]

Das Ziel des eSCM ist die Verbindung der gesamten Wertkette entlang des Supply Chain Managements über moderne Kommunikationsmittel. Neue Systeme müssen daher das Potenzial haben, sowohl interne Systeme miteinander zu verbinden (ERP, CRM) als auch externe Systeme mit Kunden und Lieferanten (SCM).[38]

---

[33] vgl. (Wannenwetsch & Nicolai, 2004) S.7ff

[34] vgl. (Wannenwetsch & Nicolai, 2004) S.119

[35] vgl. (Wannenwetsch & Nicolai, 2004) S.197

[36] vgl. (Wannenwetsch & Nicolai, 2004) S.169

[37] vgl. (Wannenwetsch & Nicolai, 2004) S.213

[38] vgl. (Wannenwetsch & Nicolai, 2004) S.8

Es wird in dem Buch noch der Begriff C-Commerce vorgestellt:

> *„C-Commerce bezeichnet die vernetzte Zusammenarbeit (eCollaboration) und die Integration von Wertschöpfungspartnern sowie unternehmensübergreifender Geschäftsprozesse auf Basis eines echtzeitgetreuen, internetbasierten und plattformunabhängigen Informationsaustausches.“* [39]

Ziel ist der flexible und dynamische Aufbau von „Many-to-Many“ Beziehungen zwischen den Interaktionspartnern und der Aufbau von sogenannten „Trading Communities“.

[39] vgl. (Wannenwetsch & Nicolai, 2004) S.9f

2.2 Technische Anforderungen

Die Grundlage für die Entwicklung eines Middlewaresystems bilden die strategischen Überlegungen aus den betriebswirtschaftlichen Anforderungen, wie im vorherigen Kapitel dargelegt. Technische Umsetzungen sollten diesen Anforderungen bestmöglich entsprechen, um den Mehrwert für die Unternehmen zu gewährleisten. Darüber hinaus darf jedoch nicht außer Acht gelassen werden, dass auf der technischen Seite auch nicht betriebswirtschaftliche Aspekte in den Anforderungskatalog übernommen werden müssen. Diese Ansätze helfen das System auch bei der Entwicklung, in der Eingliederung und bei der Wartung der IKT Systeme optimal zu unterstützen.

In diesem Abschnitt werden diese technischen Anforderungen anhand von verschiedenen Frameworks und Richtlinien herausgearbeitet.

2.2.1 Enterprise Software Architektur (ESA)

Im Buch „Enterprise SOA: Service Oriented Architecture Best Practices“ werden die Anforderungen an ein System beschrieben, die umgesetzt werden müssen, damit eine moderne Enterprise Software Architektur (ESA) realisiert werden kann. ESA ist eine Möglichkeit um die Ergebnisse aus dem Prozessmanagement (Kapitel 2.1.4) flexibel und schnell technisch umsetzen zu können. Dabei werden an ein solches modernes System vier Anforderungen gestellt. [40]

Die erste Anforderung ist „Simplicity“. Die Enterprise Architektur muss einfach sein, damit bei der Spezifikation solcher Systeme, bei der sich viele Mitarbeiter unterschiedlicher Abteilungen und Wissensebenen miteinander austauschen, keine Kommunikationsfehler entstehen. [41]

Die zweite Anforderung versucht den neuen Entwicklungen auf den internationalen Märkten gerecht zu werden, bei denen gerade das Internet einen großen Beitrag leistet. Die Art wie Unternehmen miteinander zusammenarbeiten ändert sich immer schneller und Unternehmensnetzwerke müssen diesen Herausforderungen durch „Flexibility and Maintainability“ entsprechen. Organisationen müssen sich schnell an neue Gegebenheiten anpassen können, denn Verzögerungen in der Umsetzung können mit hohen Kosten

[40] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 1.4

[41] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 1.4

verbunden sein. [42] Carter wertet „Flexibilität“ als eine der wichtigsten Eigenschaften, die moderne Unternehmen aufweisen müssen:

> *„Business flexibility is a key element in a company’s growth strategy. The terrain for today’s businesses is fraught with competitors, complexity, regulations, consolidation, demanding customers, and business models that must change quickly and precisely.“* [43]

Darauf aufbauend soll Unternehmen die Eigenschaft „Reusability“ helfen. Wiederverwendbarkeit kann auf zwei Ebenen die Kosten und die Geschwindigkeit bei der Umsetzung reduzieren. Zum Einen kann der Code selbst, durch den Aufbau von einzelnen Modulen oder Diensten wiederverwendet werden. Zum Anderen ist die Wiederverwendung von Daten bzw. die Reduzierung von Mehrfachspeicherung zu verhindern. [44]

Der letzte Punkt besagt, dass Organisationen unabhängig von der Technologie agieren sollen und müssen („Decoupling of functionality and technology“). Die Informationssysteme müssen nicht nur unterschiedliche Dienste miteinander vereinen können, sondern auch Änderungen zulassen. [45]

2.2.2 Open ICT Ecosystem

Die „Roadmap for OpenICT Ecosystem“ bietet einen sehr guten Anhaltspunkt für die Ableitung von technischen Anforderungen. Dieses Werk wurde in Zusammenarbeit zwischen dem Berkman Center for Internet & Society von der Harvard Universität und IBM sowie Oracle entwickelt. Es bildet einen Leitfaden für ein zukünftiges „open ICT ecosystem“.

In dem Werk werden fünf Prinzipien definiert, die ein Open ICT Ecosystem umfassen soll. [46]

Das erste Prinzip besagt, dass ein Netzwerk vollständig kompatibel sein muss („Interoperable„). Über offene Standards sollen Informationen über mehrere Architekturen aus-

---

[42] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 1.4

[43] (Carter, 2007) S. 13

[44] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 1.4

[45] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 1.4

[46] vgl. (Open ePolicy Group, 2007)

getauscht und verwendet werden können. Gerade durch die jüngsten Zusammenschlüsse von großen Unternehmen ist dieses Prinzip sehr wichtig geworden. In naher Zukunft spielt dieses Thema in der Vernetzung von unterschiedlichsten Unternehmen eine noch wichtigere Rolle. Daher sollen auch Netzwerke gemeinsam entwickelt werden („Collaborative“) damit sie zukunftsfähig sind („Sustainable“). Allen Teilnehmern im Ecosystem soll die Möglichkeit geboten werden, Communities zu erstellen bzw. zu betreiben, um deren Probleme gemeinsam zu lösen. Das gesamte System muss in einer Art und Weise agieren, welches den Anforderungen von Unternehmen, technischen Grenzen, finanziellen Möglichkeiten und Gesetzen entspricht und trotzdem über das Potential zur Weiterentwicklung verfügt. Aus den Erfahrungen mit Software, die nur ihrer Technik wegen entwickelt wurden, wird das Prinzip der Kundenzentriertheit gefordert („User-Centric“). Die Erfüllung von Kundenanforderungen soll an oberster Stelle stehen. Das letzte Prinzip entspricht den Forderungen von ESA. Ein System muss flexibel sein („Flexible„). Das System soll sich schnell an neue Technologien, Informationen und Protokolle anpassen können.

Damit ein OPEN ICT Ecosystem umgesetzt werden kann, wird für Regierungen eine Roadmap erstellt, die aus drei Aufgaben besteht[47].

Die erste Aufgabe wird „Scoping“ genannt. In diesem Schritt wird eine Basis für ein offenes Informations- und Kommunikationssystem geschaffen. Über ein eigens definiertes Capability Maturity Models (CMM)[48] dem Openness Maturity Models (OMM) und einer sogenannten Base Line werden Kriterien für ein offenes IKT Ecosystem geschaffen. Im zweiten Schritt dem „Policymaking“ werden die definierten Kriterien durch Richtlinien und Frameworks umgesetzt. Es wird für die Wirtschaft ein Handlungsrahmen geschaffen, der eine einfachere Umsetzung ermöglicht. Der dritte Schritt „Management“ behandelt die Vorführung der Aktivitäten und die Schaffung eines dynamischen Netzwerkes.

Ein entscheidender Bestandteil des Systems ist das Wort „open“. Die Open ePolicy Group definiert den Begriff so, dass ein System, das sich als “open” bezeichnet, nicht von einer Stelle kontrolliert werden kann und dass es zudem plattformunabhängig, öffentlich publiziert und kostenfrei zugänglich ist. Für die Umsetzung wurde das Openness Maturity Model entwickelt. Dieses beinhaltet sechs Levels und zehn Eigenschaften die auf der Basis dieser sechs Levels umgesetzt werden. Die definierten Eigenschaften sind[49]:

---

[47] vgl. (Open ePolicy Group, 2007)

[48] vgl. (Software Engineering Institute, 2007)

[49] vgl. (Open ePolicy Group, 2007)

- *"Interoperability enabled*
- *Use of open technologies*
- *Architecture framework*
- *Architecture development models*
- *Communication & compliance*
- *Business process led or linked*
- *Linkages among operating units*
- *Active management*
- *Acquisition strategy / ICT investment*
- *Collaborative communities"* [50]

Das „Open ICT Ecosystem" ist ein Beispiel dafür, dass moderne Netzwerke nicht nur zwischen Unternehmen aufgebaut werden sollen, sondern auch die Regierungen ihren Beitrag leisten müssen. Darüber hinaus setzt sich ein modernes Middlewaresystem nur dann durch, wenn auch diese Anforderungen erfüllt werden können.

Im DBE wird genau auf diese Problematik eingegangen und es werden spezielle Strategien vorgestellt (Kapitel 4.2.3).

2.2.3 Smart Business Networks

IKT Systeme unterstehen einem ständigen Wandel und einer ebenso ständigen Weiterentwicklung. Firmen, die sich heute ein neues System anschaffen, müssen dieses bereits nach wenigen Jahren wieder erneuern, damit sie bei aktuellen Entwicklungen mithalten können. Eigenentwicklungen sehen sich mit dem Problem konfrontiert, mit den schnellen Entwicklungen am Markt standhalten zu müssen.

Im Artikel von Heck et al. wird eine neue Art der Unternehmensnetzwerke beschrieben. Unternehmensnetzwerke müssen von einem starren und langsam bewegenden System zu einer „open digital platform" transformiert werden. In diesem Netzwerk müssen sich schnell, zu jeder Zeit und von jedem neue Verbindungen aufbauen lassen. Die genannten Stichworte sind „pick", „plug" und „play". Die Teilnehmer im Netzwerk müssen bereit sein, in Aktion zu treten (pick) und sich einzuklinken (plug), um in der Folge den Situationsanforderungen entsprechend zu handeln (play).[51]

[50] vgl. (Open ePolicy Group, 2007)

[51] vgl. (Heck & Vervest, 2007)

Dabei wird auf das Modell der „Swarm Intelligence“ eingegangen. Das Ziel ist es wie bei Ameisen jeden einzelnen Partner für sich agieren zu lassen. Auf Grund von einfachen Regeln können diese einzelnen Partner wie ein Schwarm von Insekten agieren und so ihr Ziel erreichen. Das so geschaffene Ökosystem entspricht den von Bonabeau et al. definierten Charakteristiken Flexibilität, Robustheit und selbst-organisierend.[52]

Smart Business Networks verfolgen genau den Ansatz von „Swarm Intelligence“. Jeder einzelne Partner ist selbst-organsiert. Der Unterschied zu bestehenden Systemen besteht jedoch darin, dass im Rahmen einer Supply Chain nicht der eigene Prozess beobachtet wird, sondern auf der Basis eines übergreifenden definierten Businessprozesses jeder einzelne Partner seine Rolle in dem System findet.

Konsynski et al. definiert sechs Kernelemente für Unternehmensnetzwerke, mit welchen ein Smart Business Netzwerk entstehen kann. Diese sechs Kernelemente werden in zwei Gruppen geteilt, der Architektur und der Steuerung (Governance). Die Architektur bildet die Rahmenbedingungen in denen die Steuerung eines “Smart Business Networks” erfolgen kann.[53] Sie bestimmt das Framework in dem ein solches Netzwerk agiert:

> *„Effective market practice includes attention to loose coupling and relevant modularization, separation of knowledge sharing from process roles, increased visibility of operations across partnering organization, heterogeneity retention, and a self-organizing swarm architecture“*[54]

Die Steuerung (Governance) gibt den Handlungsspielraum vor, in welchem die Entscheidungen getroffen werden. Für die Umsetzung eines Smart Networks ist dieser zweite Teil die Voraussetzung für eine erfolgreiche Eingliederung in ein Unternehmen.

---

[52] vgl (Bonabeau & Meyer, 2001).

[53] vgl. (Konsynski & Tiwani, 2004)

[54] (Konsynski & Tiwani, 2004)

2.3 Zusammenfassung der Anforderungen

Das Kapitel hat bis jetzt viele Anforderungen, dem ein modernes System entsprechen soll, aufgelistet. In diesem Abschnitt werden diese Anforderungen zu einem Gesamtkonzept zusammen gefasst. In der Tabelle 1 werden die Ausgangspunkte für die einzelnen Anforderungen zusammengefasst. Ein modernes Middlewaresystem sollte, in weiterer Folge diesen Kriterien entsprechen.

In den weiteren Kapiteln wird dieses Gesamtkonzept verwendet um sowohl Serviceorientierte Architektur (SOA) als auch das Digitale Business Ecosystem (DBE) auf ihre Potentiale für ein modernes Middlewaresystem für das Supply Chain Management zu überprüfen.

Anforderung	Wirtschaftlich	Technisch	Kapitel
Zusammenarbeit (Collaborative)	X	X	2.1.1, 2.1.5, 2.2.2, 2.2.3
Serviceorientierung	X	X	2.1.4, 2.2.1
Echtzeitgetreu	X		2.1.1, 2.1.3, 2.1.5
Anpassungsfähigkeit	X	X	2.1.1, 2.1.3, 2.1.5, 2.2.1, 2.2.2
Kundenzentriert	X	X	2.1.2, 2.2.2
Zuverlässig	X	X	2.1.2, 2.1.3, 2.2.2
Kostengünstig	X	X	2.1.2
Kompatibilität	X	X	2.1.2, 2.1.3, 2.1.4, 2.1.5, 2.2.2
Einfach	X	X	2.1.1, 2.1.2, 2.2.1
Wiederverwendbar		X	2.2.1
Zukunftsfähig		X	2.2.2
Selbstorganisiert	X	X	2.1.1, 2.2.3

Tabelle 1: Anforderungen an ein Middlewaresystem

2.3.1 Zusammenarbeit (Collaborative)

Nicht nur aus technischer Sicht ist die Zusammenarbeit von entscheidender Bedeutung. Im Supply Chain Management spielt dieser Faktor eine der Hauptrollen, die alles ermöglichen soll. SCM ist ein Netzwerk von Beziehungen. Neue Prozesse können nur über Zusammenarbeit realisiert werden. Um die Entwicklung von neuen Unternehmensformen und Unternehmenspartnerschaften ermöglichen zu können, wird im Open ICT Ecosystem[55] diese Eigenschaft hervorgehoben.

2.3.2 Serviceorientierung

Unternehmen werden heutzutage zu einem großen Teil über Prozesse definiert. Diese Prozesse können nicht nur mehrere Abteilungen umfassen, sondern viele Unternehmen. Um den Anforderungen auf Grund von Prozessen entsprechen zu können, muss ein Ansatz verwendet werden, der die einzelnen Teilaufgaben eines Prozesses auf technischer Ebene abbildet. Diese Abbildung wird über den serviceorientierten Ansatz umgesetzt, bei dem die einzelnen Aufgaben eines Prozesses als Service dargestellt werden.

2.3.3 Echtzeitgetreu

Schnelligkeit spart auf lange Sicht Kosten. Muss die Planung für das SCM auf der Basis von vergangenen Daten arbeiten, so kann auf Veränderungen nicht schnell reagiert werden. Diese Anforderung ergibt sich aus den Problemen mit dem Bullwhip-Effekt. Durch die Verkürzung der Informationslaufzeit und durch die Lieferung von aktuellen Daten kann dieser Effekt bereits reduziert werden. Weiters wird im Bereich des eSCM diese Anforderung in die Definition von C-Commerce eingebaut. Im SCOR wird die Eigenschaft „Responsiveness“ als Merkmal für ein funktionierendes System herangezogen.

2.3.4 Anpassungsfähig

Sowohl SCOR als auch das Open ICT Ecosystem setzen eine schnelle Anpassungsfähigkeit des Systems an neue Gegebenheiten voraus. Neue Entwicklungen müssen

[55] vgl. (Open ePolicy Group, 2007)

schnell in ein System eingebaut werden können. Für die Optimierung der SCM kann über die Anpassungsfähigkeit zum Teil die Möglichkeit herausgearbeitet werden, die starren Durchlaufzeiten an die tatsächlichen Nachfragen anzupassen.

2.3.5 Kundenzentriert

Im Vordergrund der Entwicklungen dürfen nicht die Software oder Hardware selbst stehen. Das Open ICT Ecosystem verlangt, dass die Erfüllung der Dienste im Vordergrund steht. New und Westbrook beschreiben den oben definierten Kundenwert (Kapitel 2.1.2). Das Ziel muss sein, den Wert für den Kunden auf Grund des Prozesses zu erhöhen.

2.3.6 Zuverlässig

Für den Einsatz im Lieferkettenmanagement müssen sich die einzelnen Partner auf die Funktionsfähigkeit der Systeme verlassen können. SCOR definiert diesen Faktor über die „perfect order fulfilment“ Metrik. Diese Anforderung wird vorausgesetzt egal für welches Produkt eine Supply Chain über ein modernes Kommunikationsmedium aufgebaut wird. Aus technischer Sicht muss das System auch bei hohem Informationsaustausch, wie z.B. bei agilen Supply Chains, zuverlässig handeln.

2.3.7 Kostengünstig

Damit sich ein neues System durchsetzen kann, müssen sich die Kosten innerhalb kürzester Zeit wieder einspielen. Gerade KMUs haben nur begrenzte finanzielle Möglichkeiten. SCOR gliedert die Kosten in den Verwaltungsaufwand und in die Produktkosten. Diese Kosten haben wiederum Einfluss auf den Kundenwert. Je nach Produktstrategie sind die Kosten das einzige Kaufkriterium.

2.3.8 Kompatibilität (Interoperable)

In Abstimmung mit der Unternehmensstrategie werden verschiedene Arten von Zusammenarbeit gewünscht. Für Unternehmen bedeutet das, dass diese sehr oft neue Verbindungen zu Unternehmen, sowohl intern als auch extern, aufbauen müssen.

Gerade im Bereich des SCM kann ein System nur dann funktionieren, wenn das Unternehmen sowohl auf der Händler- als auch auf der Käuferseite, ihre bestehenden ERP-Lösungen miteinander verbinden können, aber auch das Unternehmen selbst seine verteilten Applikationen zu einem gesamten System zusammenstellen kann.

Durch den Einsatz von Open Standards kann die Kompatibilität zwischen den einzelnen Plattformen erreicht werden. Eine Middlewaresoftware muss im Sinne des Open ICT Ecosystems „offen" sein. Sie darf nicht von einer Stelle kontrolliert oder beschränkt werden. Konsynski et al. hebt hervor, dass sogenannte geschlossene Systeme das Problem mit sich bringen, dass die Größe der potenziellen Teilnehmer entscheidend beschränkt wird.[56] Darüber hinaus entstehen höhere Kosten für die Anpassung der jeweiligen Softwaresysteme. Ein System ist laut Open ICT Ecosystems dann offen, wenn:

- *„es nicht von einer Person oder Institution kontrolliert werden kann.*
- *es in einem transparenten und offenen Prozess entwickelt und verwaltet wird.*
- *es plattform- und herstellerunabhängig ist.*
- *Spezifikationen und Anleitungen öffentlich publiziert werden.*
- *es kostenfrei oder mit minimalen Kosten verfügbar ist.*
- *es mit einem breiten Konsens der Teilnehmer entwickelt wurde."* [57]

2.3.9 Einfach

Die Einführung eines neuen Middlewaresystems muss die Eigenschaft haben, einfach zu sein. Nicht auf Grund des Systems selbst, sondern in der Art der Anwendung die auf diesem System laufen. Es muss eine Möglichkeit entwickelt werden, um die komplexen wirtschaftlichen Anforderungen auf die komplexen technischen Anforderungen so zu transferieren, dass ein System entsteht, das sowohl verwaltet, als auch umgesetzt werden kann. Darüber hinaus muss eine Kommunikation zwischen den Personen, die die technische Seite vertreten und denen die die wirtschaftlichen Anforderungen vorgeben, über dieses System möglich sein. Ein Ansatz dahingehend ist bereits die auf dem Prozessmanagement basierende Anforderung der Serviceorientierung. Damit kann die Komplexität durch die Teilung in kleinere Stücke verringert werden.

[56] vgl. (Konsynski & Tiwani, 2004)

[57] vgl. (Open ePolicy Group, 2007)

2.3.10 Wiederverwendbar

Wie im Kapitel 2.2.1 beschrieben, können durch die Wiederverwendbarkeit die Kosten entscheidend verringert werden. Moderne Systeme müssen diesen Anforderungen daher entsprechen.

2.3.11 Selbstorganisation

Nachira et al. definiert als Ziel eines verteilten Systems einen „single point of failure“ zu vermeiden.[58] Daher muss eine zentrale Koordination des Systems vermieden werden. In Anlehnung an die Theorie der „Swarm Intelligence“[59] wird von einem modernen Netzwerk die Selbstorganisation der einzelnen Knoten verlangt.

Konsynski et al. definiert die Selbstorganisation in zwei Ebenen. Zum einem wird ein gemeinsames Vokabular verlangt. Dies kann über Standards wie XML erreicht werden. Zum anderen wird die „Swarm Intelligence“ eingesetzt. Über einfache Regeln kann das System intelligent handeln, ohne dass es zentral gesteuert wird.[60]

2.3.12 Zukunftsfähig

Das Open ICT Ecosystem verlangt von einem System Nachhaltigkeit. Es muss ein Framework gebildet werden, in dem sich das Ecosystem ausbreiten und weiterentwickeln kann. Bestehende Komponenten müssen wiederverwendet werden können und mit neuen Entwicklungen verbunden werden. Nur so kann das System auch in der Zukunft weiterverwendet werden.

---

[58] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007)

[59] vgl. (Bonabeau & Meyer, 2001)

[60] vgl. (Konsynski & Tiwani, 2004)

Teil II

Middlewaresysteme

Kapitel 3

Service Oriented Architecture (SOA)

3.1 Einleitung

Der Begriff Service Oriented Architecture (SOA) wird in vielen Werbeprospekten und Vorträgen verwendet, um Aufmerksamkeit auf das jeweilige Thema zu lenken und den Bezug zu den aktuellen Trends hervorzuheben. Mit der Zeit haben sich viele Definitionen für SOA entwickelt:

> *"Service Oriented Architecture (SOA) is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains."*[61]

> *"Service orientation is a means for integrating across diverse systems. Each IT resource, whether an application, system, or trading partner, can be accessed as a service. These capabilities are available through interfaces; complexity arises when service providers differ in their operating system or communication protocols, resulting in inoperability."*[62]

> *"When coupled with "architecture," service-orientation takes on a technical connotation. "Service-oriented architecture" is a term that represents a model in which automation logic is decomposed into smaller, distinct units of logic. Col-*

[61] (MacKenzie, Laskey, McCabe, Brown, & Metz, 2006) S.8

[62] (Microsoft, 2006)

> *lectively, these units comprise a larger piece of business automation logic. Individually, these units can be distributed."*[63]

Josuttis beschreibt *„SOA ist eine neue Art auf Problemstellungen zu schauen"*, und hebt den Aspekt hervor, dass SOA nicht gekauft werden kann. Es gibt nicht das eine Tool mit dem das gesamte System realisiert werden kann.[64]

Mit dem Konzept von SOA wird versucht eine Verbindung der „Business"-Welt und der technischen Welt zu finden, die leicht verständlich ist. Carter beschreibt SOA als eine neue Art wie Unternehmen ihre Business-Prozesse auf die IT Infrastruktur übertragen können.

> *„Your company must view your business processes as a set of linked components that address the strategic, technical, operational, organizational, and financial issues that businesses deal with today"*[65]

Wie im vorherigen Kapitel 2.1.4 beschrieben, bildet der Prozess die Basis für diese Sichtweise. Im Unterschied zu anderen Entwicklungen, soll SOA nicht als ein neues System in einem Unternehmen einfach alles ersetzen, sondern die bestehenden Systeme und Strukturen können über das Konzept von SOA neu und einfacher miteinander verbunden werden. SOA ist der Ansatz um große Systemlandschaften zu einem agierenden Organismus zu verbinden.[66]

Fünf Anknüpfungspunkte, die im Rahmen einer Studie von IBM ausgearbeitet wurden, sollen den Eintritt in die Welt von SOA erleichtern. Zwei Punkte betreffen die IT Sichtweise (Konnektivität, Wiederverwendung) und drei Punkte den wirtschaftlichen Ansatz (Mitarbeiter, Prozesse, Informationen). Das Ziel ist es Mitarbeiter, Prozesse und Informationen miteinander zu verbinden; dafür wird Konnektivität benötigt, Wiederverwendbarkeit erhöht dabei die Effizienz bei der Umsetzung. In der Abbildung 4 werden diese Anknüpfungspunkte für SOA skizziert.[67]

---

[63] (Erl, Thomas, 2005) Kapitel 3.1

[64] vgl. (Josuttis, 2007) S.2, S.12

[65] (Carter, 2007) S.23

[66] vgl. (Josuttis, 2007) S.3

[67] vgl. (Carter, 2007) S.53, vgl. (IBM, 2007)

Abbildung 4: Eintrittspunkte in SOA

Quelle: (Carter, 2007)

Wie aus dieser Abbildung ersichtlich, steht nicht die Technik selbst im Vordergrund, sondern die wirtschaftlichen Aufgaben, die die Technik umsetzen muss, um die gemeinsamen Ziele und Visionen eines Unternehmens erreichen zu können. Die Punkte Mitarbeiter, Prozesse, Informationen und deren Anforderungen wurden im Kapitel 2 zum Supply Chain Management näher erläutert.

Die weiteren Teile dieses Kapitels beschäftigen sich mit der technischen Umsetzung der beiden letzten Punkte aus der Abbildung 4.

Im letzten Teil des Kapitels wird versucht, die Eigenschaften, die ein Middlewaresystem haben muss, mit SOA in Verbindung zu bringen.

3.2 SOA Architektur

> *"SOA is not a concrete architecture: it is something that leads to a concrete architecture."*[68]

SOA ist kein Tool oder Framework, sondern eine Sammlung von Konzepten und Sichtweisen. Dennoch wird vermehrt versucht Strukturierungen in das Konzept von SOA zu bringen. So beschreibt Carter eine SOA Reference Architektur, die eine Top-Level Ansicht auf die serviceorientierte Architektur darstellt. Im Zentrum des Modells befindet sich die Infrastruktur für die Kommunikation (das Nervensystem). Dieses Nervensystem ist von Diensten umgeben, die jeweils Teilaufgaben in einer Servicearchitektur übernehmen. In der äußersten Schicht befinden sich Dienste, die reale Probleme beschreiben und von den inneren Schichten umgesetzt werden. Bei einem konkreten SOA-Projekt kann und soll nie die gesamte Struktur auf einmal in der Form von SOA abgebildet werden. Stattdessen können mit dieser Reference Architektur gezielt die einzelnen Teilprojekte abgegrenzt werden. [69]

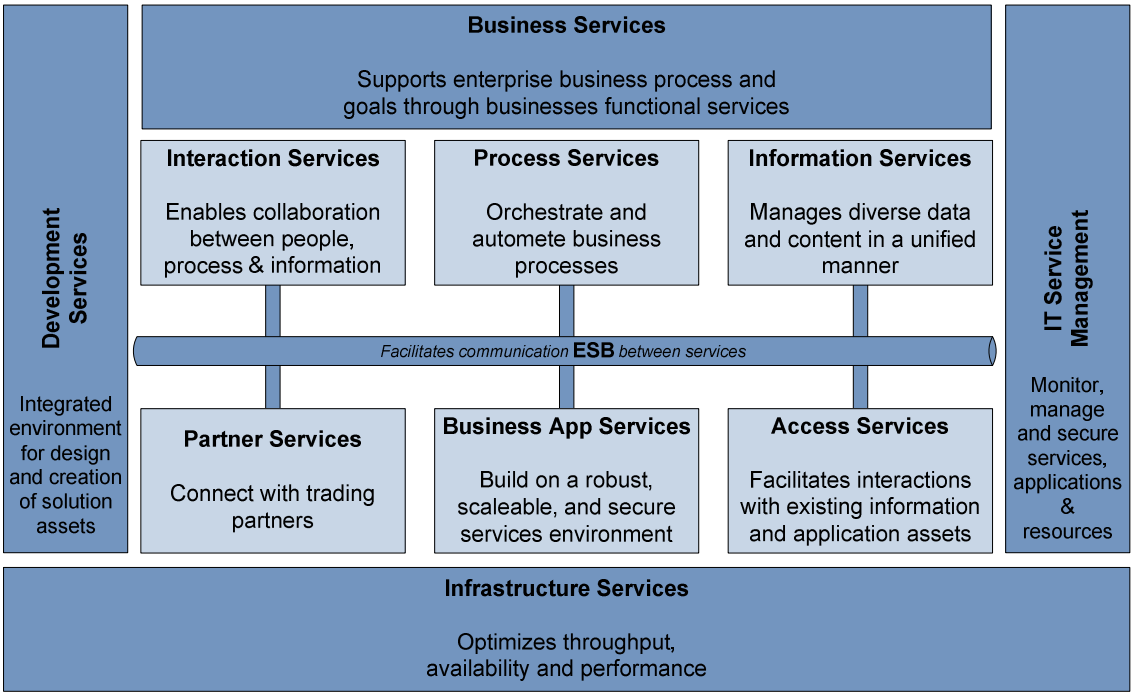

Abbildung 5: Top-Level Architektur von SOA

Quelle: (Carter, 2007) S.65

[68] (Josuttis, 2007) S.12

[69] vgl. (Carter, 2007) S.65

Josuttis nennt drei Hauptbestandteile, welche die Architektur von SOA ausmachen: Services, Infrastruktur, Regeln und Prozesse.[70]

Einzelne Unternehmensaufgaben werden als Services zur Verfügung gestellt. Das Ziel dabei muss sein, die Services so allgemein wie möglich zu definieren, damit diese in weiterer Folge für andere Aufgaben weiterverwendet werden können. Die Services werden in einer sogenannten „Service Registry" gespeichert und für einen späteren Abruf verwaltet. Die Infrastruktur ist das Bindeglied zwischen den einzelnen Services. Bei der Infrastruktur in SOA wird zwischen einer direkten „point-to-point" und einer indirekten „hub-and-spoken" Verbindung unterschieden. Bei einer „hub-and-spoken" Verbindung werden die Applikationen nicht direkt miteinander verbunden, sondern es wird eine Zwischensoftware eingesetzt. Diese Software wird bei SOA Enterprise Service Bus (ESB) genannt und erlaubt die Kombination von Diensten auf eine einfache und flexible Weise.[71]

Innerhalb und außerhalb des Unternehmens stehen viele Personen, die in Interaktion zueinander Aufgaben erledigen. Gerade mit dem Einsatz eines ESB müssen Regeln und Prozesse definiert werden, die das Zusammenspiel der einzelnen Dienste auf eine sichere und zuverlässige Art ermöglichen. Aus diesem Grund müssen in SOA Konzepte wie etwa für Sicherheit (WS-Security) und Zuverlässigkeit (WS-Reliability) miteinbezogen werden.[72]

Anschließend werden diese drei Hauptbestandteile im Detail erläutert, damit in weiterer Folge die Anforderungen aus dem Kapitel 2 mit SOA verglichen werden können.

3.2.1 Services

Der Kernaspekt von SOA sind Services. Die Aufgabe ist es, Unternehmensprozesse in ihre Teilaufgaben aufzubrechen (Kapitel 2.1.4) und die so gewonnenen einzelnen Services in der IT-Infrastruktur abzubilden. Dabei kann ein Service mehrere Funktionen, die zu einem Dienst gehören, umfassen.

---

[70] vgl. (Josuttis, 2007) S.2

[71] vgl. (Bianco, Kotermanski, & Merson, 2007) S.22ff

[72] vgl. (Josuttis, 2007) S.19

3.2.1.1 Paradigmen von SOA

Da SOA keinem konkreten Framework zugewiesen werden kann, sind die Paradigmen, unter denen SOA Systeme aufgebaut werden, die wirklichen Bestandteile von SOA. Die weiteren Darstellungen bieten dann jeweils Sichtweisen auf diese Paradigmen. Obwohl nicht immer alle Teilaspekte in einem realen Projekt umgesetzt werden können, sind diese dennoch die ausschlaggebenden Argumente, die für die Umsetzung von SOA Lösungen sprechen.[73]

3.2.1.1.1 Loose Coupling

Das Problem von großen Unternehmensnetzwerken oder Netzwerken zwischen mehreren Unternehmen ist, dass diese auf unterschiedlichsten Plattformen und Softwareprodukten basieren (Heterogenität). Diese Softwareprodukte haben zum Teil nur geschützte oder selbst programmierte Schnittstellen zur Verfügung, die nur schwer miteinander verbunden werden können. Daher werden speziell für diese Systeme Schnittstellen programmiert, die nur zwischen zwei Systemen funktionieren: „tight coupling“. Was passiert jedoch, wenn ein neues System hinzukommt, das in alle anderen integriert werden muss? In diesem Fall müssen eine Vielzahl von Schnittstellen entwickelt werden, damit die Systeme zusammenarbeiten können. Um dieses Problem zu umgehen, wird bei SOA das Konzept des „Loose Coupling“ verfolgt. Durch offene Standards werden die Schnittstellen so definiert, dass sie zu jedem Zeitpunkt miteinander verbunden werden können. Die Kosten für die Entwicklung sinken entscheidend und damit auch die Kosten und Risiken für die Umsetzung neuer Projekte.[74]

3.2.1.1.2 Service Contract

Die Kommunikation der Services basiert auf der Grundlage von Verträgen. Diese Service Contracts stellen die technische Schnittstelle zum Service dar und umfassen mehrere Dokumente wie zum Beispiel bei Webservices WSDL Definition, XML Schema Definition und eine WS-Policy Beschreibung.[75]

3.2.1.1.3 Autonomy (Service Autonomy)

Jedes Service hat die Kontrolle über die eigene Logik, die es umschließt. Bei der Abfrage eines Dienstes sind nur das Ergebnis und die Schnittstelle bekannt, wie die Aufgabe

---

[73] vgl. (Erl, Thomas, 2005) S.3.1, S.8.3

[74] vgl. (Erl, Thomas, 2008) S.71f, (Josuttis, 2007) S.17f, (Pulier & Taylor, 2006) S.8f

[75] vgl. (Erl, Thomas, 2008) S.127

ausgeführt wird obliegt dem Service selbst. Dadurch wird ein höheres Maß an Zuverlässigkeit erreicht, weil die Programme unabhängig voneinander agieren.[76]

3.2.1.1.4 Abstraction (service abstraction)

Das Ziel ist Informationen, die nicht für die Verwendung des Services benötigt werden, zu verstecken. Im Rahmen von SOA können laut Erl vier Arten von Metadaten unterschieden werden.

Die technischen Informationen geben darüber Auskunft wie das System konkret umgesetzt wurde. Für den Konsumenten ist jedoch ausreichend zu wissen, wie ein Service gefunden und wie es benutzt werden kann. Alle weiteren Details können versteckt bleiben und so vom Provider jederzeit geändert werden. Sogar die gesamte Funktionalität des Services muss dem Konsumenten nicht bekannt sein. Die funktionellen Informationen müssen nur insofern bekannt sein, als sie für die Eingliederung in ein System benötigt werden. Einige Informationen dienen rein der internen Verarbeitung. Die dritte Art der Metadaten ist die Information der Programmlogik. Die Programmlogik wird für die Verwendung des Services zumeist nicht benötigt. Nur in speziellen Anwendungen muss der Konsument wissen, wie das Programm im Hintergrund funktioniert. Die Entscheidung ob diese Informationen freigegeben werden, hängt von der verfolgten Strategie des Betreibers des Services ab. Im Gegensatz dazu sind die Quality of Service Informationen ein entscheidender Bestandteil. Qualitätsinformationen werden zumeist innerhalb des WSDL über die SLA oder die WS-Policy bekannt geben. Die Umsetzung in einem Projekt hängt wiederum von der Aufgabenstellung ab. [77]

3.2.1.1.5 Reusability (service reusability)

Die Verwendung von strukturierten, wiederverwendbaren Programmstücken bietet einen hohen Kostenvorteil gegenüber der Neuprogrammierung. Jedoch kann nicht jeder Code wiederverwendet werden. Bereits bei der Erstellung von Services muss der Dienst abstrakter gestaltet werden, damit mehr als eine Applikation den Dienst verwenden kann. Ein Ziel kann sein, viele kleine generische Services zu generieren, die flexibel miteinander verbunden werden können, wenn ein Business Prozess es verlangt. Darüber hinaus können Services verwendet werden, die bereits in einer laufenden Umgebung funktionieren.[78] Mit der Zeit entstehen so sehr viele Dienste, die strukturiert gesammelt werden sollten. Ein solches Register kann unterschiedliche Namen haben. Dies wird zum Beispiel Service-Pool oder Service Inventory Blueprint genannt. Durch die Ver-

---

[76] vgl. (Erl, Thomas, 2008) S.294, (Papazoglou & van den Heuvel, 2007) S.390

[77] vgl. (Erl, Thomas, 2008) S.212ff

[78] vgl. (Carter, 2007) S.61f

wendung von offenen Standards sind die Services übertragbar auf andere Applikationen.[79]

3.2.1.1.6 Composability (service composability)

Jedes Service hat zwei Rollen (Service role). Zum Einen stellen sie Informationen bereit (Service Provider), zum Anderen konsumieren sie Informationen (Service Consumer). In einem Service können beide Eigenschaften vereint werden, d.h. es können sowohl Informationen von mehreren Services abgefragt werden, als auch an mehrere weitergegeben werden. Diese Rolle nennt sich Service Composition. [80]

3.2.1.1.7 Statelessness (service statelessness)

Je nach Service-Art, werden verschiedene Ebenen für die Speicherung von Stadien in einem Service verlangt. Je weniger Statusinformationen gespeichert werden, um so skalierbarer sind die Services, weil sie weniger Ressourcen benötigen. Erl unterscheidet vier Typen. [81] Der „Primary State“ nimmt den Wert „Aktiv“ oder „Passiv“ an, je nachdem, ob das Service benutzt wird oder nicht. Die Erweiterung ist der „Primary State Conditions“ der „Stateful“ sein kann, wenn der Status eines Services gespeichert werden soll, oder „Stateless“, wenn nicht. Der „Type of State Information“ unterscheidet zwischen „Session Data“, welche Informationen über die aktuelle Sitzung, „Context Data“, welche Informationen über den laufenden Workflow und „Business Data“, welche die Zusatzinformationen für das Service speichert. Der letzte Typ unterteilt sich in „Context Rules“ und „Context Data“ und beschreibt die verschiedenen Aspekte eines Workflows. Diese Informationen werden „Types of Context Data“ genannt.

3.2.1.1.8 Discoverability (service discoverability)

Damit Services gezielt gefunden werden können, ist nicht nur ihr logischer Name von Bedeutung. Bei großen Unternehmensanwendungen mit Hunderten von Services müssen trotzdem exakt jene Services gefunden werden, die für die konkrete Umsetzung benötigt werden. Dies setzt voraus, dass die Services genau beschrieben worden sind (Eigenschaften, Aufgaben und Grenzen) und dass sie über zusätzliche Metadaten verfügen, die diese Beschreibungen beinhalten. Metadaten müssen in einer zentralen Datenbank gespeichert werden, damit dort nach den benötigten Informationen gesucht werden

[79] vgl. (Erl, Thomas, 2008) S.269

[80] vgl. (Erl, Thomas, 2008) S.43, S.333

[81] vgl. (Erl, Thomas, 2008) S.333ff

kann. Als zentrale Datenbank wird zum Beispiel das Service Registry und Repository vorgestellt (Kapitel 3.2.2), die diese Aufgabe in einem SOA Netzwerk übernimmt.[82]

3.2.1.2 Der Aufbau von Services

Nach der Beschreibung der Konzepte, die in einem Service vereint werden, wird im folgenden Kapitel auf die einzelnen Teile eines Services im Detail eingegangen; im Speziellen auf die Interpretation der Paradigmen von SOA auf konkrete Konzepte.

Ein Service besteht aus fünf Bereichen: Vertrag, Interface, Implementierung, Business Logic und Daten (Abbildung 6).[83] Das so erstellte Service kann daraufhin in einem ESB publiziert werden und steht für die Verwendung in einem konkreten Prozess zur Verfügung.

3.2.1.2.1 Vertrag (Contract)

Der Vertrag beschreibt die Aufgabe, die Funktion und die Bedingungen unter denen das Service verwendet werden kann.

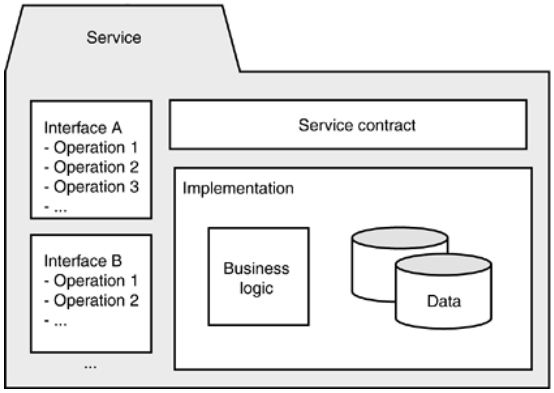

Abbildung 6: Aufbau von Services

Quelle: (Krafzig, Banke, & Slama, 2005) Kapitel 4.3

Bei Webservices wird dieser Teil über das WSDL Protokoll abgebildet. Die WSDL-Datei beschreibt die Anknüpfungspunkte zum Service auf der Basis von URL und beinhaltet die Links zum XML-Schema und zur WS-Policy. Das XML-Schema stellt eine Erweiterung der WSDL-Datei dar, indem es die Struktur der WSDL-Datei beschreibt. Dadurch können komplexe Datentypen übertragen werden. Neben der Beschreibung der Datenstruktur können auch noch Sicherheitsmerkmale an die WSDL-Datei angehängt werden. Dies wird über die WS-Policy realisiert. [84]

3.2.1.2.2 Interface

Das Interface bzw. die Anknüpfung an das Service wird im Vertrag definiert, jedoch muss der Client das Interface zuerst bei sich implementieren, um auf das Service zugreifen zu können. Dabei wird zwischen dynamischen und statischen Interceptoren unterschieden. Je nachdem muss die Referenz zum Service direkt in das Consumer Service

[82] vgl. (Erl, Thomas, 2008) S.362ff

[83] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 4.3

[84] vgl. (Erl, Thomas, 2008) S.127ff

programmiert werden; oder im Fall des dynamischen Interceptors wird nur der Proxy übergeben. Die Suche des Services erfolgt zur Laufzeit des Programms.

3.2.1.2.3 Implementierung

Die Implementierung der Services bildet die physische Umsetzung der im Vertrag definierten Eigenschaften in einen ausführbaren Code. In diesem Zusammenhang besonders interessant ist der MDA-Ansatz[85]; unabhängig von einer konkreten Implementierung kann hier ein Service über UML beschrieben und anschließend über MOF in einen Code umgewandelt werden.

3.2.1.2.4 Business Logic und Daten

Bestimmte Services bilden einen Prozess. Dieser wird über die Implementierung und über den Zugriff auf andere Services realisiert (Business Logic). Neben der Business Logic können in den Services aber auch Daten gespeichert werden (Business Entity).

Für einen effizienten Einsatz bei einem Projekt müssen die Services nach ihren Aufgaben kategorisiert werden. Dies vereinfacht die Arbeit und bildet die Basis für die Systemarchitektur in einer realen Implementierung. Thomas Erl baut sein Service Modell auf drei Kategorien auf. [86]

Entity Services stellen konkrete Wirtschaftseinheiten (Business-Entity) dar, z.B. Rechnung, Mitarbeiter. Jedes Service verkörpert die Funktionen einer dieser Einheiten. [87] Die Umsetzung von Unternehmensaufgabe (Business Task) wird mit Task Services abgebildet. Die Abgrenzung zum Entity Service kann über den Kontext definiert werden. Ein Task Service behandelt mehrere Entity Services, wobei sich ein Entity Service immer nur um das jeweilige Objekt kümmert. Die letzte Kategorie stellen die Infrastrukturdienste dar. Diese übernehmen rein technische Aufgaben wie Import- und Exportaufgaben oder Logging und werden als Utility Services bezeichnet.

Krafzig et al.[88] und Josuttis [89] erweitern das Konzept von Thomas Erl. Sie unterteilen die Aufgaben der Services in vier Layer. Der Basic Layer beinhaltet die von Thomas Erl definierten Entity und Utility Services und werden als „Business logic“ und „Business data“ Services bezeichnet. Der Intermediary Layer oder Orchestration Layer fasst mehrere Services aus dem Basic oder Intermediary Layer zusammen. Der Zugriff auf den

---

[85] vgl. (Miller & Mukerji, 2003)

[86] vgl. (Erl, Thomas, 2008) S.43

[87] CRUD (create, read, update, delete)

[88] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 5.2

[89] vgl. (Josuttis, 2007)

Basic Layer kann über den ESB ermöglicht werden. Die dritte Ebene bildet den Process Layer, welcher die Unternehmensprozesse abbildet. Die letzte Ebene ist der Enterprice Layer, der den Zugriff durch den User darstellt.

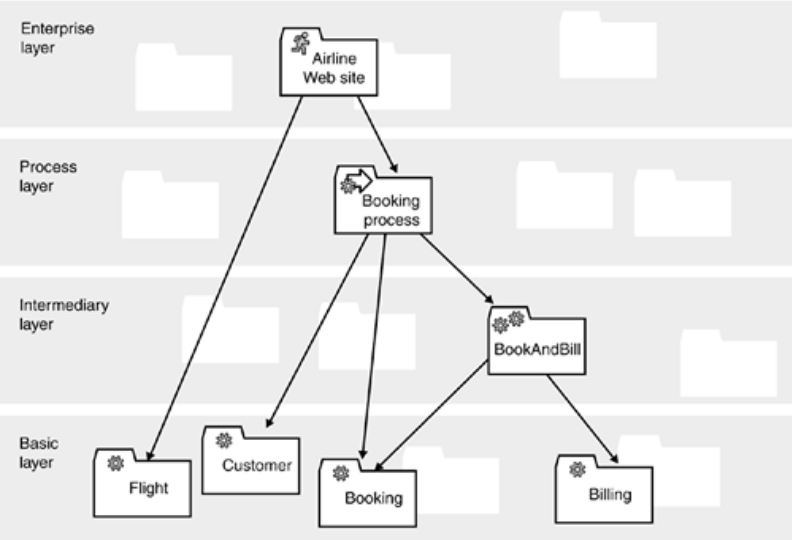

Abbildung 7: Arten von SOA Services

Quelle: (Krafzig, Banke, & Slama, 2005) Kapitel 6.3

3.2.2 Service Registry und Repository

Das Service Registry und Repository stellt einen Mehrwert zum einzelnen Service Contract dar. Es dient dazu, Informationen über das Service, sowie auch zu den Bedingungen unter welchen es verwendet wird, zu speichern. Zusätzlich ist es die Anlaufstelle, um ein Service für seinen Prozess zu finden. Bei den Anforderungen sollte zwischen einem öffentlich zugänglichem und einem unternehmensinternen Netzwerk unterschieden werden, da hierbei unterschiedliche Anforderungen nachgefragt werden[90]. Im unternehmensinternen Netzwerk bildet das Service Registry und Repository alle Services ab. Es erstellt ein sogenanntes „Service inventory“.[91]

Carter definiert das Service Registry und Repository als BSRR (Business Service Registry und Repository). Eine BSRR soll nach Carter wesentlich mehr Aufgaben umfas-

[90] (Krafzig, Banke, & Slama, 2005) Kapitel 4.3.3

[91] (Erl, Thomas, 2008) S.366

sen, als die Umsetzung der ersten Version von Webservices, welche nur über ein Verzeichnis verfügen, das alle Services darstellt. [92]

Zunächst muss die BSRR Services in sich aufnehmen und über Suchalgorithmen anderen Teilnehmern zur Verfügung stellen. („Publish and find Services“). Diese Aufgabe wird von der Service Registry übernommen. Die Suche sollte sich dabei nicht nur auf das Service selbst, sondern auch auf die Beschreibung des Services in der Form von Metadaten beziehen. Die Informationen des Services werden im Service Repository gespeichert. Dieses gibt Auskunft, wie der Dienst verwendet werden kann, wie die Dienste miteinander interagieren, warum der Dienst verwendet werden soll und warum er verwendet wird. Gleichzeitig muss eine große Menge an Services verwaltet werden („Govern“). Die BSRR soll nicht nur die Informationen speichern, sondern eine Umgebung für die Verwaltung der Services von der Erstellung bis hin zur Zerstörung bieten. Fragen wie „Wer darf wann auf welchen Dienst zugreifen?“, können hier definiert werden. Die BSRR ist eine Erweiterung der ESB und hilft bei der dynamischen Auffindung von passenden Services („Enrich“). Im letzten Schritt muss das System sich selbst anpassen können („Manage“). Unter dieser Eigenschaft soll der BSRR die Dienste unter der Einbindung von Metriken in ihrer Verwendung optimieren.

3.2.3 Enterprise Service Bus

Der ESB dient als Schnittstelle zwischen den einzelnen Diensten. Bieberstein et al. beschreiben den ESB als eine

> *„intelligente, verteilt, Transaktions- und Nachrichten-Ebene für die Verbindung von Applikationen, diversen Daten und anderen Diensten die über eine verteilte „Enterprise computing infrastructure“ miteinander kommunizieren“* [93].

Der entscheidende Punkt bei der Verwendung einer ESB ist, dass diese Komponenten unabhängig von den APIs und von Protokollen miteinander kommunizieren können und zwar über den Ansatz als Service. Erreicht wird dieser Zustand über die Verbindung von Service Contracts. Zum Beispiel mittels WSDL werden die Anknüpfungspunkte auf einer abstrakten Ebene beschrieben.

Die Komplexität von einem System kann durch eine ESB verringert werden, weil weniger Verbindungen benötigt werden. Bei der Verbindung von „n“ Systemen würden

---

[92] (Erl, Thomas, 2008) S.369

[93] (Bieberstein, Bose, Fiammante, Jones, & Shah, 2005)

„n*(n-1)/2“ Verbindungen benötigt werden, mit dem ESB lediglich eine Verbindung und ein Interface pro System.[94] Der ESB nimmt daher eine sehr wichtige Rolle in zukünftigen SOA-Umsetzungen ein. Aus diesem Grund muss ein ESB eine Reihe von Aufgaben erfüllen:

Eine der grundlegenden Anforderungen ist der Aufbau von Verbindungen zwischen Systemen („Providing Connectivity“). Wenn sich ein Unternehmen für die Eingliederung eines ESB entscheidet, sollte dies so geschehen, dass möglichst alle Schnittstellen (und nicht nur Webservices) unterstützt werden.[95] Weiters gibt es bei den ESB gravierende Unterschiede bei der Umsetzung: so kann innerhalb eines ESB eine reine point-to-point Verbindung über Adressen unterstützt werden oder über einen „Mediator“, bei dem der konkrete Endpunkt nicht bekannt sein muss. Darüber hinaus unterstützt nicht jeder ESB indirekte Servicecalls über Proxies.[96] Der Erfolg hängt von der Abstimmung der Anforderungen auf die konkrete Umsetzung ab. Je nach Applikation werden für die Kommunikation zwischen den einzelnen Diensten unterschiedliche Modelle benötigt („Communication Models“). Ein ESB sollte daher sowohl synchrone, asynchrone als auch dateibasierende Kommunikation ermöglichen, bzw. auf die jeweilige Applikation abgestimmt sein.[97] Business Continuity vereint die Eigenschaften von „Scalability“ und „High Availability“. Der ESB sollte so gewählt werden, dass das Unternehmen im Wachstum durch den Einsatz des ESB unterstützt und nicht eingeschränkt wird.[98]

Neben der Connectivity und den Kommunikationsmodellen ist die Art der Verteilung der Nachrichten über das Netzwerk für die Geschwindigkeit und Zuverlässigkeit des Systems ausschlaggebend („(Intelligent) Routing“). Die ausschließliche Auslieferung der Daten über eine direkte Verbindung ist nicht immer die beste Wahl, weshalb Informationen über die Daten selbst (Metadaten) die intelligenteste Route für die Nachricht bestimmen.[99] Große SOA-Netzwerke können über intelligente Routen optimaler arbeiten, es wird jedoch mehr Sicherheit für den Zugriff auf diese Services benötigt („Security“). Gerade bei der Integration anderer Unternehmen ist die Frage nach Authentifikation und Autorisation ganz entscheidend für den Einsatz eines ESB.[100] Mit der Anzahl der Dienste wächst auch das Verlangen nach einem gezielten Management für Services

---

[94] vgl. (Josuttis, 2007) S. 5

[95] vgl. (Carter, 2007) S.86

[96] vgl. (Josuttis, 2007) S.50ff

[97] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 9.1

[98] vgl. (Carter, 2007) S.90f

[99] vgl. (Josuttis, 2007) S.74

[100] vgl. (Josuttis, 2007) S.74

in einer Registry. Der ESB muss Funktionen bereithalten, um diese Strukturen gezielt verwalten zu können („Service Management“). [101] Monitoring and Logging unterstützt das Service Management. Der ESB ist das Nervensystem der gesamten Struktur. Probleme während der Entwicklung und im Betrieb sollten sofort erkannt und behoben werden können. [102]

Zuverlässigkeit von Systemen ist für deren Verwendbarkeit unerlässlich, jedoch kann sich der Grad der Zuverlässigkeit von Service zu Service unterscheiden. Quality of Service (QoS) muss daher die Möglichkeit bieten, die Services entsprechend ihrer Service Level Agreements diese Eigenschaften zu gestalten.[103]

Ein weiterer sehr interessanter Aspekt ist die Einführung des MDA-Ansatzes[104] in den ESB. Dabei wird die Service Registry um eine XMI/MOF kompatible Repository erweitert. Services können nach dem MDA-Ansatz in einem abstrakten Modell erstellt werden (UML) und über den MOF-Ansatz in einen plattformabhängigen Code umgewandelt werden. Die plattformunabhängigen Modelle werden in der Service Registry gespeichert.[105]

In den Publikationen wird der ESB hauptsächlich für den Einsatz innerhalb eines Unternehmens oder einer fix definierten Partnerschaft zwischen Unternehmen dargestellt. Eine Möglichkeit, wie an einen ESB weitere Unternehmen auf eine sichere Art und Weise an das Unternehmens-ESB angeschlossen werden können, ist ein ESB-Gateway. Der ESB-Gateway ist ein eigenes Tool, das definierte Dienste für Partner außerhalb des eigenen, sicheren Netzwerkes anbietet. Die externen Unternehmen können dann wiederum über einen ESB-Gateway oder direkt über den eigenen Dienst auf die angebotenen Funktionen zugreifen.[106]

---

[101] vgl. (Josuttis, 2007) S.74

[102] vgl. (Josuttis, 2007) S.74, (Carter, 2007) S.86

[103] vgl. (Josuttis, 2007) S.74, (Carter, 2007) S.109,

[104] vgl. (Miller & Mukerji, 2003)

[105] vgl. (Krafzig, Banke, & Slama, 2005) Kapitel 9.1

[106] vgl. (Keen, et al., 2004) S.306f

3.2.4 Regeln und Prozesse

Bei der Eingliederung von SOA in bestehende soziale Umwelten, in denen viele Personen miteinander interagieren, ist es wichtig, die Umsetzung eines SOA-Projektes in kleinen Schritten zu realisieren und an die Aufgaben des Unternehmens anzupassen.

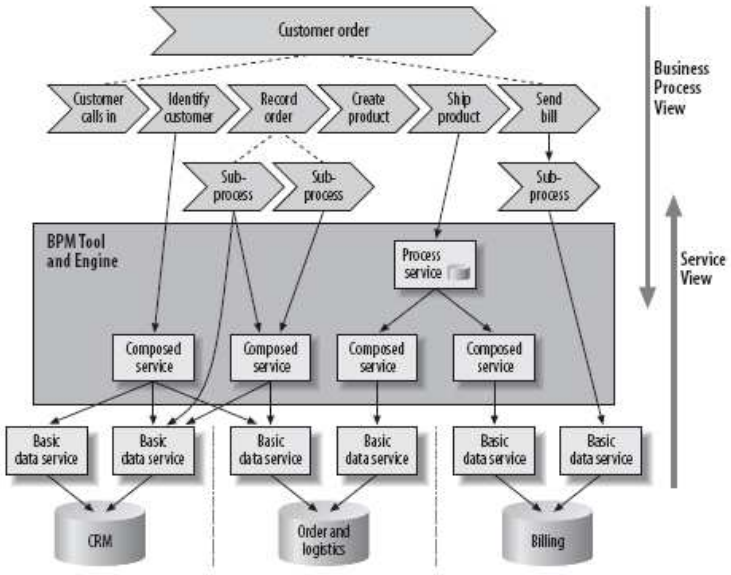

Abbildung 8: Wirtschaftliche und technische Service Landschaft

Quelle: (Josuttis, 2007)

Über die Regeln oder die Governance kann die Umsetzung von SOA direkt auf die IT Governance abgestimmt umgesetzt werden. Dieser Leitfaden gibt die IT-Compliance Richtlinien und die unternehmensinternen Sichtweisen an das Umsetzungsteam von SOA weiter und beugt so unerwünschten Ergebnissen vor. Der Leitfaden sollte von Anfang an bekannt sein, damit alle weiteren Konzepte auf diese Entwicklungen hin abgestimmt werden können.

Die Prozesse sind das Kernthema von SOA, Services stellen einzelne Aufgaben in einem Prozesse dar. Bevor jedoch die Services entwickelt werden können, müssen die Prozess festgelegt worden sein (Kapitel 2.1.4). Die so entdeckten und definierten Aufgaben können dann auf konkrete Services umgelegt werden. Ein Business Modeling Tool kann bei der Umlegung der Prozesse auf die Services helfen und ist ab einer gewissen Größe unterlässlich.[107]

[107] vgl. (Josuttis, 2007) S.88

3.3 Web Services

Web Services sind eine der bekanntesten Möglichkeit SOA zu realisieren. In der Vergangenheit waren Web Services der Ausgangspunkt für den Aufbau von serviceorientierten Projekten. Bei der Entwicklung von Webservices kann zwischen zwei Generationen unterschieden werden, die im Folgenden dargestellt werden.

3.3.1 Die Erste Generation von SOA

Die erste Generation von SOA basiert auf drei Konzepten: WSDL, SOAP und UDDI. Dabei liegt das Hauptaugenmerk auf der Umsetzung mittels Web Services.[108]

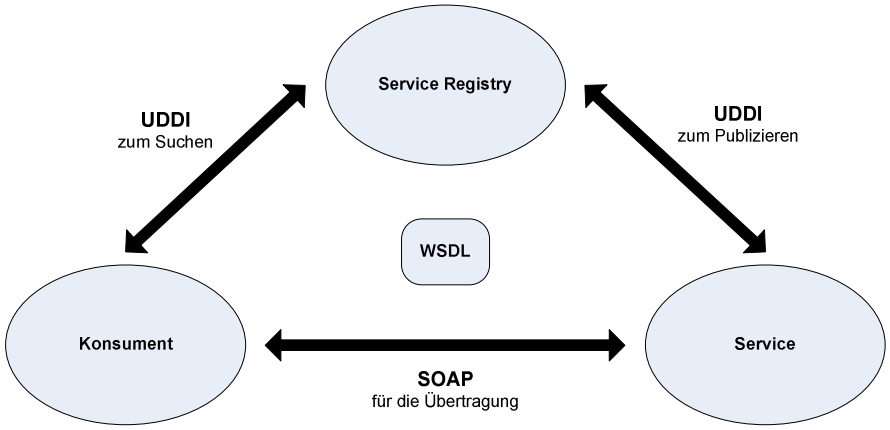

Abbildung 9: Web Services

Quelle: (Margolis & Sharpe, 2007)

Web Services Description Language (WSDL) wird in XML dargestellt und beschreibt, wie ein Dienst erreicht werden kann. Es stellt das Service-Interface für die Kommunikation zwischen den einzelnen Diensten bereit. Simple Object Access Protocol (SOAP) übermittelt Daten in einem XML-Format zwischen den einzelnen Diensten. Innerhalb des WDSL-Interfaces gibt es zwei Attribute, die die SOAP-Kommunikation definieren: „style“ und „use“. Das Attribut „style“ kann die Werte „RPC“ und „document“ annehmen. Dabei entspricht „RPC“ einem XML-basierten Remote Procedure Call. Der Wert „document“ gibt an, dass die SOAP-Nachricht beliebige XML Informationen enthalten kann. Das zweite Attribute „use“ kann die Werte „encoded“ und „literal“ annehmen.

[108] vgl. (Margolis & Sharpe, 2007) Kapitel 5, (Erl, Thomas, 2005) Kapitel 4.1

Diese geben an, ob es eine standardmäßige Codierung gibt, oder ob das Dokument über ein XML Schema definiert wird. Die beiden etablierten Varianten sind RPC-Encoded SOAP und Document-Literal SOAP.[109] Universal Description, Discovery, and Integration (UDDI) ist das zentrale Service Register für die dezentral verteilten Services im Netzwerk. Das UDDI kann durchsucht werden, um Webservices für die Einbindung in den eigenen Code zu finden. Genau genommen werden die Services über einen logischen Namen im Register gespeichert und einem URL für das WSDL zugewiesen. Der logische Name wird in den Programmcode hardcodiert hineingearbeitet. [110] Für die Umsetzung von SOA wird die UDDI zum Teil als Engpass angesehen, da es die Daten zentral speichert und somit einen „single-point of failure“ darstellt.

3.3.2 Die zweite Generation von SOA

Die zweite Generation von SOA wird mit den Begriffen „WS-*“ definiert. Es werden damit Erweiterungen zur ersten Generation beschrieben.[111] In der Abbildung 10 werden einige zusätzliche Aspekte der zweiten Generation abgebildet. Das System ist dabei in drei Ebenen unterteilt. Die unterste Ebene des Modells entspricht etwa der ersten Generation von SOA und wird als „SOA system landscape and document exchange“ bezeichnet. Die Datennachrichten werden mittels SOAP auf der Basis von WSDL definierten Schnittstellen übertragen. In der Registry können die Services gesucht werden. Die mittlere Ebene dient dem Austausch von Daten zwischen den einzelnen Services („Commonly Comprehensible Business Data“).[112] Hier hat sich in den letzten Jahren der ESB langsam entwickelt. Er übernimmt die Kommunikation zwischen den Diensten. Die oberste Ebene beschäftigt sich mit der Eingliederung von mehreren Prozessen zu einem Gesamtsystem („Cross-organizational Business Process“). Als Standard muss hier WS-BPEL genannt werden. [113]

---

[109] vgl. (Bianco, Kotermanski, & Merson, 2007) S.13f

[110] vgl. (Pappas, Kazasis, Anestis, Gioldasis, & Christodoulakis, 2005) S.29

[111] vgl. (Erl, Thomas, 2005)

[112] vgl. (Schroth & Janner, 2007) S.38f

[113] vgl. (Schroth & Janner, 2007) S.38f

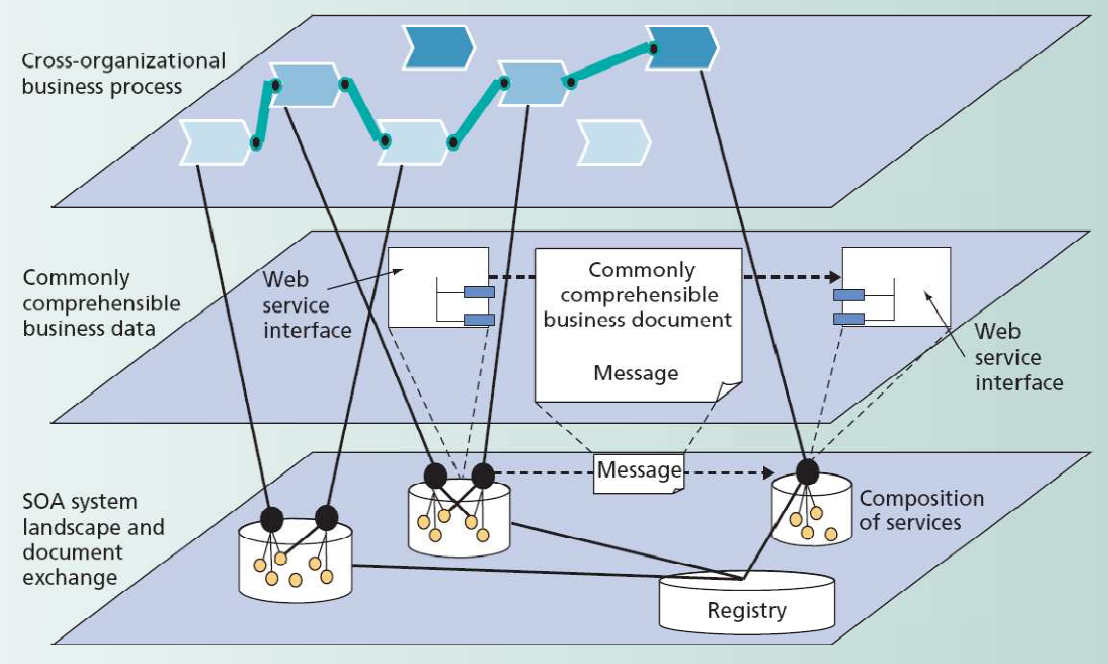

Abbildung 10: Basic service-oriented architecture

Quelle: (Schroth & Janner, 2007)

BPEL steht für Business Process Execution Language und dient der Beschreibung von Business-Prozessen. BPEL wird meistens in XML dargestellt.[114] Der Grundstein zu dieser Definition wurde von IBM, Microsoft und BEA gelegt. Dabei wurden die Standards WSFL und XLANG zu BPEL vereint. Der Begriff WS-BPEL basiert auf der Spezifikation von OASIS. Neben WS-BPEL haben sich auch noch andere Standards entwickelt die hier aber nicht näher beschrieben werden. In der Abbildung 11 wird die Entwicklung des Standards dargestellt[115]:

Josuttis beschreibt die Standards EPC und BPMN als die wichtigsten neben BPEL bestehenden Standards. Wer das Rennen in Zukunft gewinnen wird, ist noch nicht entschieden.

In der Gestaltung von Prozessen wird grundlegend zwischen zwei Varianten unterschieden: Orchestration und Choreography. Orchestration bedeutet, dass es eine zentrale Stelle gibt, die alle Dienste miteinander verbindet. Choreography folgt dem entgegengesetzten Ansatz: niemand kontrolliert den gesamten Prozess, sondern jede Teilaufgabe nur ihren eigenen Bereich.[116]

---

[114] vgl. (Josuttis, 2007) S.89, (Erl, Thomas, 2005) Kapitel 4.1

[115] vgl. (Josuttis, 2007) S.91

[116] vgl. (Josuttis, 2007) S.97

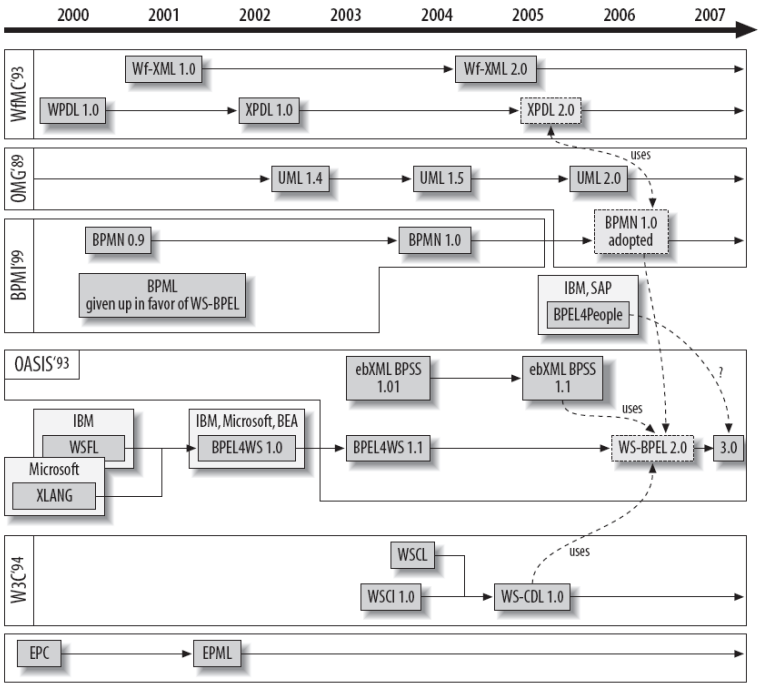

Abbildung 11: Overview of business process modeling standards

Quelle: (Josuttis, 2007)

BPEL verfolgt dabei den Ansatz von Orchestration. Der Prozess wird durch die Abfolge von einzelnen Schritten durchgeführt. Diese Schritte werden „activities“ genannt. Es wird dabei jeder Dienst („partner service“ genannt) nacheinander abgerufen. Mehrere BPEL-Prozesse können miteinander verbunden werden, wobei jeder einzelne Prozess seinen eigenen sogenannten „Scope“ hat. [117]

Neben diesen beschriebenen Standards haben sich noch viele weitere, die bei der Entwicklung von SOA Systemen helfen sollen, gebildet. Gerade im Bereich der Sicherheit haben sich einige Standards entwickelt. WS-Security beschreibt etwa wie Sicherheit bei SOAP realisiert werden kann. WS-Trust beschreibt hingegen sichere Token.[118]

Die Weiterentwicklung der Standards wird von Organisationen wie W3C[119], OASIS[120] und WS-I[121] vorangetrieben.

---

[117] vgl. (Margolis & Sharpe, 2007) Kapitel 7

[118] vgl. (Josuttis, 2007) S.184f

[119] vgl. (The World Wide Web Consortium, 2008)

[120] vgl. (OASIS, 2008)

[121] vgl. (WS-I , 2008)

3.4 SOA und SCM

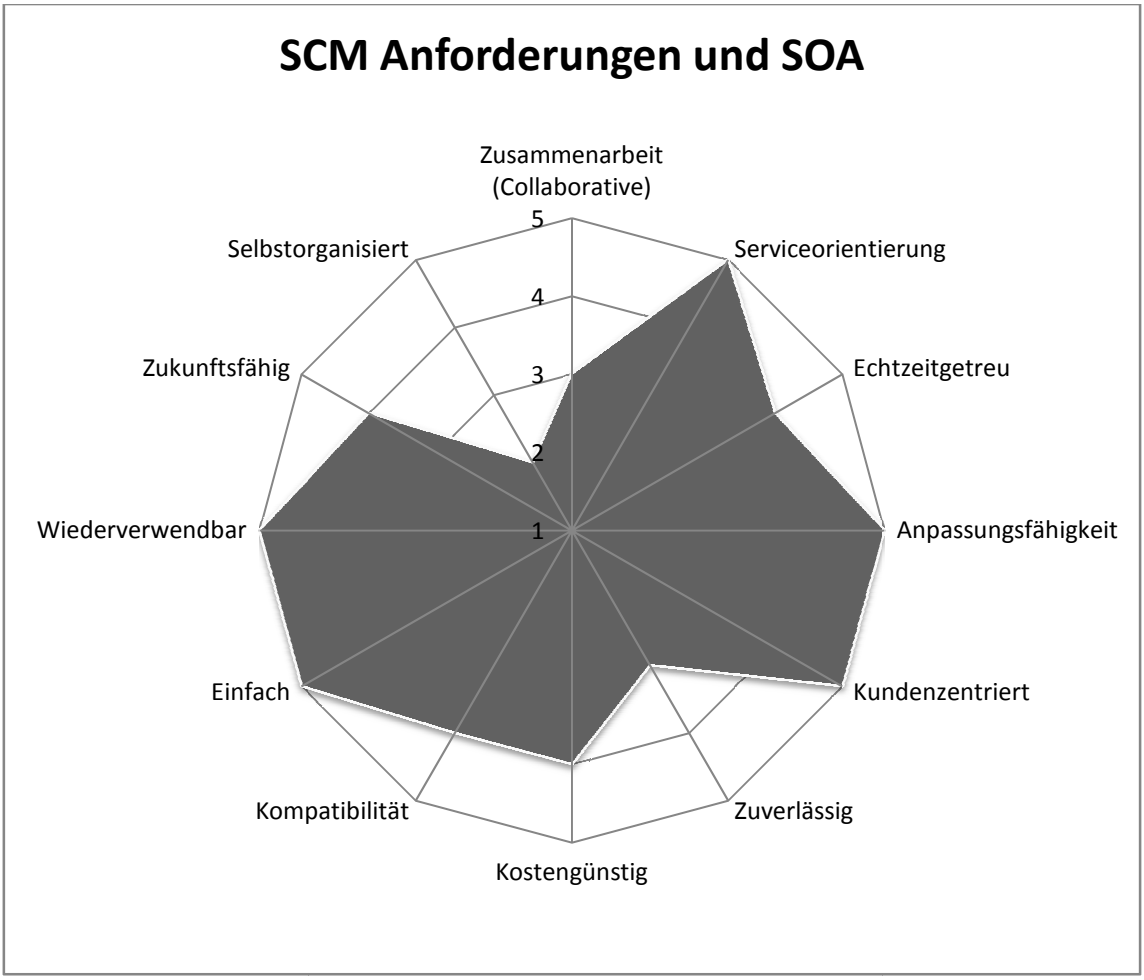

Abbildung 12: SCM Anforderungen und SOA

Im folgenden Teilkapitel werden die Anforderungen aus dem Kapitel 2.3 auf das SOA – Konzept umgelegt und deren Umsetzung erläutert.

In Abbildung 12 wird der Grad der Umsetzung der einzelnen Anforderungen anhand einer Skala von 1 bis 5 bewertet. Die Wertung eins bedeutet, dass diese Anforderung von SOA nicht umgesetzt wird. Die Bewertung zwei definiert, dass diese Anforderung angedacht, aber noch nicht in weiten Teilen des Konzeptes eingearbeitet wurde. Darauf aufbauend stellt die Bewertung drei eine Umsetzung der Anforderung dar, obwohl noch große Potentiale für eine Erweiterung bestehen. Die Bewertung vier sagt aus, dass die Anforderung weitestgehend umgesetzt wurde und die höchste Bewertung gibt an, dass die Anforderung zu 100 % umgesetzt wurde.

Die Tabelle bietet eine Zusammenfassung der Erkenntnisse aus den einzelnen Kapiteln.

Anforderung	Kapitel
Zusammenarbeit (Collaborative)	3.2.1, 3.3
Serviceorientierung	3.2
Echtzeitgetreu	3.2.3, 3.2.1
Anpassungsfähigkeit	3.2.1
Kundenzentriert	3.1, 3.2
Zuverlässig	3.3
Kostengünstig	3.2.1.2
Kompatibilität	3.2.3, 3.2.2
Einfach	3.1
Wiederverwendbar	3.2.1
Zukunftsfähig	3.1
Selbstorganisiert	3.2.1

Tabelle 2: Vergleich SCM und SOA

3.4.1 Zusammenarbeit (Collaborative)

SOA ist kein fixes Framework oder ein Tool, sondern eine Reihe von Paradigmen, die in offene Standards zusammengefasst worden sind. Diese werden von Konsortien wie W3C, OASIS, WS-I weiterentwickelt und in Referenzdokumenten zusammengefasst. Diese Dokumente können von Programmierern eingesehen und selbständig umgesetzt werden. An der Entwicklung selbst können Personen über verschiedenste Foren oder aus den Konsortien teilnehmen und ihre Vorschläge für die Umsetzung bekannt geben.

Zusätzlich zu den Konsortien wird der SOA Ansatz sehr stark bei internationalen Softwareunternehmen wie Microsoft, Oracle, SAP und IBM eingesetzt. Die konkrete Entwicklung in den Unternehmen kann zum Teil nur über Foren gesteuert werden, obwohl auch hier versucht wird die Community in die Entwicklung mit einzubeziehen. Gerade auch beim ESB sind die Konsumenten von den Entwicklungen der Softwareunternehmen abhängig.

Die Entwicklung von SOA wird sehr stark von großen Unternehmen dominiert. Daher wird dieser Bereich mit drei bewertet.

3.4.2 Serviceorientierung

Wie der Name schon sagt, ist das gesamte Konzept auf Serviceorientierung ausgelegt. Das Ziel ist es Unternehmensprozesse auf die IT-Landschaft in der Form von einzelnen Teilaufgaben aus dem Prozess direkt zu übertragen. Es wird keine technische Sprache benötigt, um die Serviceaufgaben zu beschreiben. Die Herausforderung besteht darin, die einzelnen Dienste zu identifizieren und richtig einzuordnen.

3.4.3 Echtzeitgetreu

SOA unterstützt die Anforderungen „Echtzeitgetreu“ durch das gewählte Kommunikationsmodell und die Art der Verbindung der einzelnen Services. Beim Kommunikationsmodell kann eine synchrone Kommunikation gewählt werden, bei welcher die Antwort in Echtzeit auf die gesendeten Nachrichten zurückgeschickt wird.

Bei der Verbindung zwischen den einzelnen Services kann entweder eine point-to-point Verbindung aufgebaut werden, die einen schnellen Datenaustausch auf Grund der direkten Verbindung gewährleistet. Es kann aber auch ein ESB für den Aufbau einer echtzeigetreuen Verbindung verwendet werden. Beim Einsatz eines konkreten ESB ist es wichtig, den richtigen ESB für die Anforderungen aus dem Pool von möglichen Anbietern auszuwählen.

Da diese Anforderung sehr stark von der Umsetzung abhängt, wird diese Anforderung mit vier bewertet.

3.4.4 Anpassungsfähig

„Loose Coupling“ ist das Konzept, das diese Fähigkeit schafft. Services werden so definiert, dass sie schnell miteinander verbunden werden können. Darüber hinaus wird versucht Services wiederzuverwenden. Dabei müssen die Services allgemeiner gehalten werden und sind so eher fähig schnell miteinander verbunden zu werden. Durch die Definition von kleinen Services können neue Funktionen schnellstmöglich umgesetzt und an neue Herausforderungen angepasst werden.

3.4.5 Kundenzentriert

Träger der Entwicklung ist bei SOA nicht nur die Wirtschaft alleine, sondern zusätzlich auch viele Communities und Konsortien. Gerade die Wirtschaft hat das Ziel ihre Versionen von SOA schnellstmöglich an den Kunden zu bringen. Daher sind viele der gelieferten Standards auf die Bedürfnisse der Kunden, bzw. der Unternehmen abgestimmt und versuchen den zukünftigen Entwicklungen in der Unternehmensorganisation zu entsprechen.

Der Kundenwert wird über Konzepte wie Wiederverwendbarkeit und Loose Coupling erreicht. Dem Kunden entstehen dadurch geringere Kosten, weil nicht für jeden Dienst neue Schnittstellen programmiert werden müssen. Auf der anderen Seite können durch die Anpassungsfähigkeit neue wahrnehmbare Werte erstellt werden, die den Kundenwert (Kapitel 2.1.2) erhöhen.

3.4.6 Zuverlässig

Die Zuverlässigkeit bei SOA hängt von der konkreten Umsetzung der Konzepte ab. SOA beinhaltet Referenzmodelle damit genau diese Anforderungen in einem realen Projekte gewährleistet werden können. Entscheidend für diesen Punkt ist die Wahl des richtigen ESB und der richtigen Umgebung, welche die Zuverlässigkeit und Überprüfbarkeit gewährleisten.

Dieser Punkt wird mit drei bewertet, da die Umsetzung von SOA zum Zeitpunkt dieser Arbeit noch in der Entwicklung begriffen ist und für die Verwendung als Middleware dieser Punkt sehr stark von der gewählten Umgebung abhängt.

3.4.7 Kostengünstig

SOA kostet grundsätzlich nichts, weil es ein Konzept ist, das über Referenzmodelle kostenlos erschlossen werden kann. SOA selbst ist aber kein Framework, über das die Software programmiert werden kann, sondern stellt wie gesagt nur eine Sichtweise eines Softwareproblems dar.

Für die Umsetzung eines konkreten Projektes können die meisten Aufgaben jedoch trotzdem in implementierten offenen Tools (wie Java) umgesetzt werden. Auf der anderen Seite gibt es jedoch von Softwareunternehmen kostenpflichtige Umgebungen bzw. werden Zusatzfunktionen kostenpflichtig angeboten, in denen Softwareprojekte zwar

gekauft werden müssen, dafür aber schneller umgesetzt werden. Der trade-off muss vom Kunden selbst bewertet werden.

Auf jeden Fall bietet der Ansatz Kostenvorteile auf Grund der Ersparnisse bei der Programmierung von Schnittstellen zwischen Softwareprodukten. Anstatt 1:1 Verbindungen können hier Schnittstellen für mehrere Systeme verwendet werden.

3.4.8 Kompatibilität (Interoperable)

Bei der Kompatibilität hängt SOA sehr stark von der Umsetzung ab. Die Paradigmen von SOA erlauben alle gewünschten Arten von Zusammenarbeit. Eine Grenze gibt es jedoch bei der Skalierbarkeit der Service Registry. Die Frage ist, wie dies in einem konkreten Projekt umgesetzt wird. Versuche über UDDI haben im großen Stil keine adäquate Lösung gebracht.

Im Bezug auf das Modell der Open ePolicy Group hat SOA je nach Implementierung seine Schranken. SOA ist sehr stark von der Service Registry oder von der direkten Zuweisung zu konkreten Services abhängig. In realen Projekten werden diese Anknüpfungspunkte daher von einem großen Unternehmen vorgegeben, was dem Konzept aber widerspricht.

Der Prozess der Entwicklung von SOA ist grundsätzlich offen und kann von jedem über die einzelnen Konsortien eingesehen werden. Die Entwicklung der Frameworks für die Implementierung von SOA Umgebungen wird ebenfalls sehr offen über Communities umgesetzt, wird aber zum Teil von großen Unternehmen gesteuert. Die Plattformunabhängigkeit ist für SOA von großer Wichtigkeit, damit verschiedenste Softwaresysteme miteinander verbunden werden können. Bei der Wahl eines ESB wird jedoch die Umgebung eines bestimmten Softwareunternehmens zum Teil favorisiert. Spezifikationen zu den Standards können ebenfalls heruntergeladen werden.

Für die Anbindung an bestehende ERP-Systeme im Unternehmen werden von den großen Softwareunternehmen bereits Schnittstellen angeboten, die miteinander verbunden werden können.

SOA wird in diesem Bereich mit vier bewertet, da EAI im Unternehmen sehr gut umgesetzt werden kann, aber im großen Stil, zum Teil noch keine adäquate Lösung vorhanden ist.

3.4.9 Einfach

SOA selbst sollte nie sofort die gesamte Unternehmensstruktur ersetzen. Ein solches Unterfangen würde nicht zum gewünschten Erfolg führen. Die Komplexität von SOA ist im Grunde nicht die technische Umsetzung, sondern die Etablierung des Ansatzes in einem Unternehmen. Zunächst müssen alle Prozesse identifiziert werden, die durch SOA umgesetzt werden müssen, erst dann kann die Implantierung stattfinden. Für die Unterstützung werden Ansätze wie der MDA entwickelt, der eine nichttechnische Beschreibung der Services zulässt und diese über MOF auf die konkreten ausführbaren Services umlegt.

SOA selbst ist daher aus wirtschaftlicher Sichtweise sehr einfach gehalten, gerade auch deswegen, damit Systeme an den SOA-Ansatz adaptiert werden können. Die technische Umsetzung kann mitunter eine Herausforderung für IT-Leute darstellen.

3.4.10 Wiederverwendbar

Diese Forderung ist ein Hauptanliegen an eine SOA-Umgebung. Aufgrund der Architektur von SOA sind die Services dafür ausgelegt, von mehreren Prozessen verwendet werden zu können.

3.4.11 Zukunftsfähig

SOA hat sich von einem Hype zu einem etablierten Softwareentwicklungskonzept entwickelt. Alle großen Unternehmen unterstützen das SOA Konzept und adaptieren ihre Softwareprodukte dahingehend. Rund um SOA haben sich Communities entwickelt die wiederum Tools für SOA programmieren und auf den Universtäten wird SOA als ein zukünftiges Konzept unterrichtet.

Die Idee der Dienstleistungsorientierung hat den Sprung von der realen Welt auf die digitale Welt anscheinend geschafft und wird die IT-Welt noch die nächsten Jahre beschäftigen.

3.4.12 Selbstorganisation

Services sollten vom Grundgedanken her so konzipiert werden, dass sie „autonom“ handeln. Im Bereich von SOA bedeutet das, dass sie ihre Aufgabe eigenständig ausführen.

Jedoch interagieren diese Dienste in einer Umgebung, in der sie andere Dienste beeinflussen oder auf diese zugreifen. Das System selbst ist jedoch weitestgehend nicht selbstorganisiert. Zum einen werden von großen Unternehmen klare Richtlinien über einen Gateway auferlegt. Zum anderen wird SOA oft als reine Schnittstelle für die Kommunikation gesehen. Das System optimiert sich dabei nicht selbst. Neue ESB Umsetzungen werden zeigen, in welche Richtung sich SOA entwickelt.

Kapitel 4

Das Digitale Business Ecosystem

4.1 Einleitung

Der Begriff digitales Business Ecosystem setzt sich aus drei Worten zusammen, die jeweils untereinander bereits oft kombiniert worden sind und Teilbereiche des DBE beschreiben. Dabei werden aus dem Jahr 2002 erhobenen Vergleiche der Wirtschaft mit einem biologischen Ökosystem aufgegriffen und vertieft. Es wird die Metapher des Digitalen Business Ecosystems geschaffen in dem sich „digitale Arten" entwickeln.[122]

Business (ecosystem) bildet im SME networks (Small and medium enterprises networks) die oberste Ebene. Als Grundkonzept wurde der Begriff „business ecosystem" von Moore aus dem Jahre 1996 aufgegriffen. Dabei werden Unternehmen als Organismen in einer Wirtschaftswelt beschrieben, die Produkte und Dienstleistungen für Kunden anbieten, die wiederum Teil dieser Wirtschaftswelt sind. [123]

Digital (ecosystem) wird das Execution Environment gesehen. Es bildet die technische Basis für die Umsetzung des DBE. [124]

Ecosystem hebt den starken Zusammenhang zwischen den einzelnen „Arten" im ökologischen System hervor. Das Evolutionary Environment soll die Umgebung für die Umsetzung eines solchen ökologischen Systems bilden, das Eigenschaften wie selbstorganisiert und selbst-optimiert ermöglicht. [125]

---

[122] vgl. (Nachira, Francesco, 2002) S.11f

[123] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.5

[124] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.5

[125] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.5

Beim Aufbau des DBE werden immer wieder die Vergleiche mit den USA und den dortigen Entwicklungen herangezogen. Dabei verfolgt das Projekt in Europa das Ziel die KMUs zu unterstützen, da diese einen Großteil der Wirtschaft ausmachen. Es soll ein System geschaffen werden, das es KMUs ermöglicht untereinander schneller Netzwerke aufzubauen. Im Vergleich dazu haben sich in Amerika große Unternehmen etabliert, die von kleineren Unternehmen unterstützt werden. Dabei wird ein zentral gesteuertes System angestrebt, das von den großen Unternehmen dominiert wird.[126] In Europa wurden an das Projekt vom ersten Diskussionspapier an Grundanforderungen gesetzt, die eine solche Entwicklung unterbinden sollen:

> *„Equal opportunities of access to the infrastructure, affordability for small organizations, self sustainability, independence from a specific provider, technology, license, ... ".* [127]

Das DBE wird durch die Integration der Hauptbereiche IT, Naturwissenschaften und Wirtschaft entwickelt, wobei viele andere Disziplinen ihren Beitrag leisten. Diese unterschiedlichen Bereiche werden im Abschnitt Forschungsbereiche näher definiert.

4.1.1 Die Entwicklung des DBE

Der DBE Cluster entwickelte sich aus der GO Digital Initiative im Jahre 2001:

> *„GoDigital's overall purpose is to put together and adapt where appropriate support activities to help SMEs to use information and communication technologies (ICT) with best possible efficiency.*"[128]

Im Rahmen dieser Initiative wurde die Förderung von Klein- und Mittelbetrieben entschieden. Der erste Entwurf für die Förderung wurde von Nachira vorgelegt. Er verfasste im Jahre 2002 das Diskussionspapier "Toward a network of digital business ecosystem fostering the local development"[129]. In dem Diskussionspapier werden Lösungen für die im „Lisbon summit 2000" definierten nachhaltige Wege zur Unterstützung von KMUs dargestellt. Im „Lisbon summit 2000" wurden sowohl die Unterschiede zwischen Klein- und Mittelbetrieben (KMU) und Großunternehmen als auch die Unter-

[126] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.7

[127] vgl. (Nachira, Francesco, 2002) S.20

[128] vgl. (Commsision of the European Communities, 2001) S.4

[129] vgl. (Nachira, Francesco, 2002)

schiede zwischen den einzelnen Regionen aufgedeckt. Dabei wurde das klare Ziel festgelegt, die KMUs bei der Entwicklung eines höheren Wachstums und besserer Arbeitsplätze zu unterstützen. Nachira liefert Lösungen in der Entwicklung eines digitalen Ökosystems. Dabei geht es weitgehend nicht um die Unterstützung einzelner kleiner Projekte, sondern um die Umsetzung einer langfristigen Vision.[130]

Die Umsetzung des ersten DBE Projektabschnittes erfolgte im Rahmen des „6th Framework Programme for RTD, and the competitiveness and innovation Programme“ der Europäischen Union. Das Ergebnis wird in weiterer Folge in diesem Kapitel beschrieben. Dabei handelt es sich um ein Projekt, bei dem eine Vielzahl an Universitäten und Partnern aus der Wirtschaft ihren Beitrag geleistet haben. Die Ergebnisse können bereits jetzt in regionalen Testbetrieben besichtigt werden.

Das Projekt DBE ist noch lange nicht abgeschlossen und die Entwicklung wird im Rahmen des „7th Framework Programme for RTD, and the competitiveness and innovation Programme“ fortgeführt. Als ein Ziel wurde die Übernahme der Rolle des weltweiten, technischen Vorreiters festgelegt.

Innerhalb des DBE Clusters haben sich eine Reihe von Teilprojekten entwickelt. Diese Projekte übernehmen die Entwicklung von Diensten im DBE und sollen die Adaptierung bei KMUs unterstützen. In dieser Arbeit liegt die Konzentration auf der Infrastruktur des Systems. Die Dienste befinden sich zum Teil noch in einer sehr frühen Phase der Entwicklung und es gibt nur wenige Informationen darüber.

[130] vgl. (Nachira, Francesco, 2002) S.23

4.2 Forschungsbereiche des DBE

Die Umsetzung des DBE erfordert die Zusammenarbeit von Personen aus den unterschiedlichsten Wissenschaften. Dini und Nicolai haben die einzelnen Forschungsbereiche in vier Ebenen unterteilt.[131]

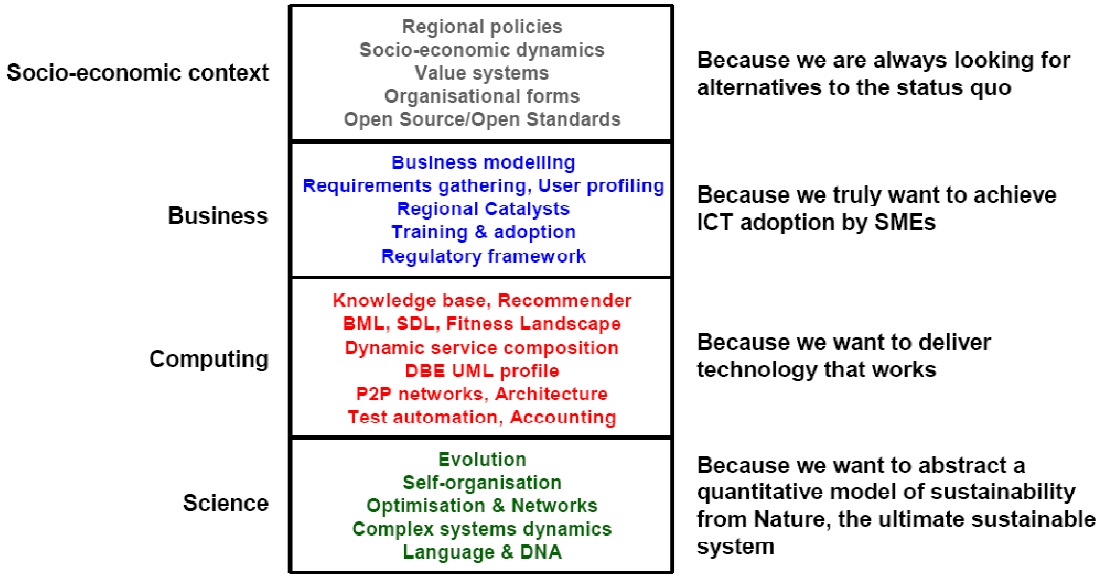

Abbildung 13: Forschungsbereiche des DBE

Quelle: (Dini & Nicolai, The Digital Business Ecosystem, 2003)

Die Schwierigkeit besteht darin die Kommunikation zwischen den einzelnen Bereichen zu ermöglichen. Jede Wissenschaft hat ihre eigenen Definitionen für gleiche Begriffe. Alleine der Begriff Digital Ecosystem hat in den unterschiedlichsten Disziplinen unterschiedlichste Bedeutungen (Abbildung 14).

Um diese Hürde zu bewältigen, müssen die Kontaktpunkte zwischen den Wissenschaften gesucht oder neu gebildet werden. Die Personen dürfen dabei ihre Begriffe und Handlungen nicht als die ausschließliche Wahrheit betrachten, sondern müssen darüber nachdenken, in welchem Zusammenhang sie ihre Aussagen treffen. Die Computerwissenschaft kann dabei als Mediator zwischen den Naturwissenschaften und den Sozialwissenschaften gesehen werden. Aus der Sicht der Sozialwissenschaften übernimmt die Computerwissenschaft die Aufgabe der Kommunikation zwischen den einzelnen Teil-

[131] vgl. (Dini & Nicolai, The Digital Business Ecosystem, 2003) S.8

nehmern. Aus der Sicht der Naturwissenschaft wird eine abstrakte Maschine aufgebaut, die die Aufgaben der Selbstorganisation und Selbstheilung übernimmt.[132]

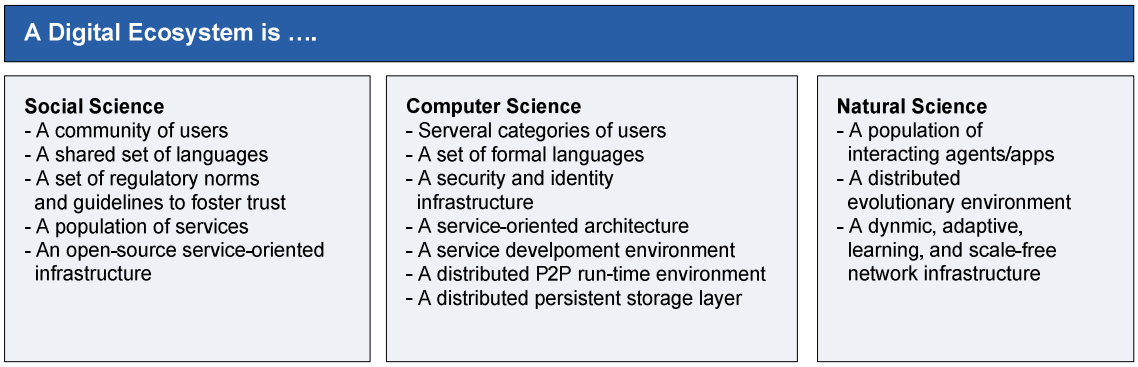

Abbildung 14: Definitionen für Digital Ecosystem

Quelle: (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007)

4.2.1 Science - Naturwissenschaften

Als Ausgangspunkt für die naturwissenschaftliche Betrachtung gilt ein reales Ökosystem. In diesem Ökosystem leben und agieren einzelne selbstorganisierte Organismen, die miteinander in Interaktion treten.

> *„Nothing in biology exists by itself; everything interacts with everything else.“*[133]

Das Ziel ist es, ein solches System digital abzubilden. Dabei gibt es jedoch vom Ansatz her einen wesentlichen Unterschied zwischen dem biologischen Ökosystem und dem digitalen Ökosystem:

> *„Biological ecosystems are ubiquitous natural phenomena, whose maintenance is crucial to our survival.” … “In contrast, digital ecosystems as defined here are technology engineered to serve specific human purposes.”*[134]

---

[132] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.25f

[133] (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.10

[134] (Briscoe, Sadedin, & Paperin, 2007) S.458

Die Entwicklungen in den letzten Jahren haben einen Punkt erreicht, an dem die Komplexität der neuen Applikationen nur noch schwer bewältigt werden kann. Es gilt einen Weg zu finden, der die automatische Suche nach neuen Algorithmen ermöglicht.[135]

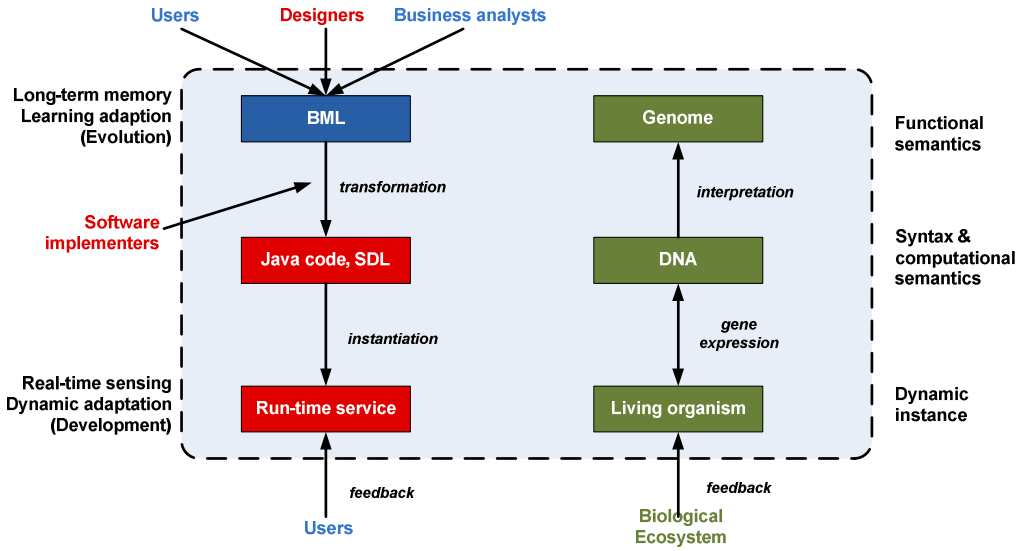

Abbildung 15: Software engineering and gene expression workflows

Quelle: (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007)

Als Grundbaustein wird der Aufbau einer einzelnen Zelle genommen. Alle Zellen sind nach dem gleichen Schema aufgebaut, trotzdem können sie eine sehr hohe Anzahl an Variationen hinsichtlich Aussehen und Funktion aufweisen. Die Organisation des Organismus in Zellen unterschiedlicher Funktionen macht einen Organismus nicht nur lebensfähig, sondern ermöglicht auch die gezielte Reproduktion einzelner Zellen (Wachstum und Entwicklung) oder des gesamten Organismus. Das Aussehen des ganzen Organismus ist in der DNA gespeichert, die zusätzlich zur Reproduktion auch die gesamte Zellfunktion steuert. In der DNA sind die einzelnen Merkmale des Organismus codiert, die Gene genannt werden. Die Summe der Gene bildet das Genom. Der Genotyp ist die Gesamtheit der in der Zelle gespeicherten Informationen, bzw. möglichen Ausprägungen der Merkmale (z.B.: können in der DNA eines Menschen die Anlagen für braune und blonde Haare von den Eltern weitergegeben worden sein). Der Phänotyp beschreibt die konkrete Ausprägung eines Individuums, d.h. welche der möglichen Ausprägungen das Individuum tatsächlich annimmt (z.B. braune Haare). Mutationen sind zufällige und richtungslose Veränderungen der Merkmale des Organismus und passieren bei der Reproduktion der DNA. Diese „neuen“ Organismen müssen sich im bestehenden Ökosys-

[135] vgl. (Tschudin) S.1

tem behaupten. Sie verändern aber auch gleichzeitig das Ökosystem in dem sie wirken. Über die natürliche Selektion überleben nur Organismen, die sich am Besten an die Gegebenheiten zu einem bestimmten Zeitpunkt im Ökosystem angepasst haben.[136]

Im DBE wird die Sichtweise der Natur auf die digitale Welt übertragen, indem die „high-level" Spezifikation der „digitalen Art" (das Genome) zur „design-time" über Business Modeling Language (BML) erfolgt. Die Darstellung der Gene in Form von DNA wird über „automatic code generation" bewirkt und zum Beispiel in Java Code umgewandelt. Es wird aber nicht nur die Syntax (d.h. der Code) beschrieben, sondern auch die Semantik (d.h. welche Aufgaben bewältigt werden sollen).[137]

Der eigentliche Dienst wird aber erst zur Laufzeit umgesetzt. Bis zu diesem Zeitpunkt ist der Algorithmus nicht komplett und könnte nicht vollständig ausgeführt werden. Ein solcher Algorithmus wird „distributed algorithm" genannt.[138]

Die Weiterentwicklung von Organismen auf Grund von natürlicher Selektion wird Evolution genannt. Die Qualität wird in der Natur auf Grund von Stabilität, Komplexität und Diversifikation gemessen. In der Computerwissenschaft hat sich auf der Basis der natürlichen Selektion der Bereich „Genetic Algorithms" entwickelt. Dieser bestimmt die Qualität eines Algorithmus auf der Basis seiner „fitness" ein bestimmtes Problem zu lösen. Diese Lösung muss innerhalb eines bestimmten Raumes oder einer Umgebung basieren, die durch Beschränkungen („Constraints") definiert wird. Alle Möglichkeiten innerhalb eines solchen Raumes werden in einer „Fitness Landscape" dargestellt. Was in der Natur auf Grund der großen Vielfalt nicht möglich ist, kann in der digitalen Welt visualisiert werden. Jeder Punkt stellt in der Fitness Landscape eine Lösung dar. Im DBE werden die optimalen Ergebnisse dadurch definiert, dass sie die Benutzeranforderungen am Effizientesten umsetzen. Durch die automatische Entwicklung von Arten entsteht das Potential zur Selbstorganisation. Die Aufgabe besteht darin, die Beschränkungen so zu wählen, dass unterschiedliche Lösungen entstehen können, aber auch das Problem effizient gelöst werden kann.[139]

Gesucht wird in dem System ein Gleichgewicht, bei dem alle Dienste ihre Aufgaben erfüllen können. Bei komplexen Problemen die selbstorganisiert sind, ist dieses Gleichgewicht aber nicht vorhersagbar. Der perfekte Algorithmus kann nicht im Vorfeld definiert werden. Daher muss eine Möglichkeit entwickelt werden, wie ein Ansatz von Ver-

---

[136] vgl. (Heistracher, Kurz, Marcon, & Masuch, 2005) S.3ff, (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.34f

[137] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.35

[138] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.32

[139] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.49ff, (Heistracher, Kurz, Marcon, & Masuch, 2005) S.15ff

such und Irrtum umgesetzt werden kann. Die Schwierigkeit liegt darin einen Trade-off zwischen Vielfalt an Möglichkeiten und teuren, schlecht adaptieren Systemen zu finden.[140]

4.2.2 Computing

Innerhalb des DBE wird die Domäne „Computing“ in zwei Teilgebiete aufgeteilt. Zum einen wird der Bereich Computerwissenschaften („Computer Science“) damit angesprochen und zum anderen der Bereich Softwareentwicklung („Software Engineering“). Die Computerwissenschaften haben die Aufgabe, die nichtfunktionalen Anforderungen einer Softwarelösung zu definieren. Innerhalb dieses Bereiches werden die Verbindungen zu den anderen Wissenschaften aufgebaut. Dieser Bereich behandelt die theoretische, wissenschaftliche Umsetzung des DBE. Softwareentwicklung übernimmt hingegen die Umsetzung der funktionalen Aufgaben. Dabei wird die konkrete Implementierung von Diensten im DBE behandelt und wie dies mit aktuellen Standards erleichtert werden kann.[141]

Ein sehr wichtiges Thema sind die naturwissenschaftlichen Modelle die, wie im vorherigen Kapitel beschrieben, im DBE realisiert werden. Im Rahmen der Biologie wird eine weitere Unterscheidung getroffen:

Auf der einen Seite gibt es die „Bio-Informatik“, die sich um die Unterstützung der Biologie mit Softwareprodukten kümmert. Auf der anderen Seite wird das Gebiet „Computational biology“ behandelt, das versucht, biologische Modelle auf die Computerwissenschaften zu übertragen. Diese beiden Teilbereiche können aber miteinander verbunden werden und dadurch verstärken sie sich gegenseitig.

Der Zusammenhang der beiden Systeme wird über die Metapher des Ökosystems hergestellt. Darin leben, wie im vorherigen Kapitel definiert, digitale Arten. Die Bioinformatik beschreibt in diesem System die Natur mit der Zuhilfenahme von Softwareprodukten. Die Erkenntnisse aus diesen Entwicklungen vertiefen das theoretische Verständnis des Menschen vom ökologischen Gleichgewicht der Natur. Auf der anderen Seite werden die Modelle der Natur auf das DBE übertragen und bauen so das digitale Ökosystem auf. Die Anknüpfungspunkte sind sowohl die Entwicklung und der Aufbau des Ökosystems und deren Interaktion und Strukturen zueinander, bei dem die digitale Welt und die reale Welt miteinander kommunizieren.[142]

---

[140] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.52

[141] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.30

[142] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.36f

4.2.3 Wirtschaft

Das Ziel des DBE ist, wie in der Einleitung definiert, die KMUs beim nachhaltigen Aufbau des Wirtschaftswachstums zu unterstützen. Im ersten Diskussionspapier des DBE wurde als wesentlicher Faktor für die Erreichung eines nachhaltigen Wachstums die Informations- und Kommunikationssysteme definiert. IKT erlaubt es, virtuelle Unternehmensnetzwerke aufzubauen, damit neue Synergieeffekte entstehen können. [143]

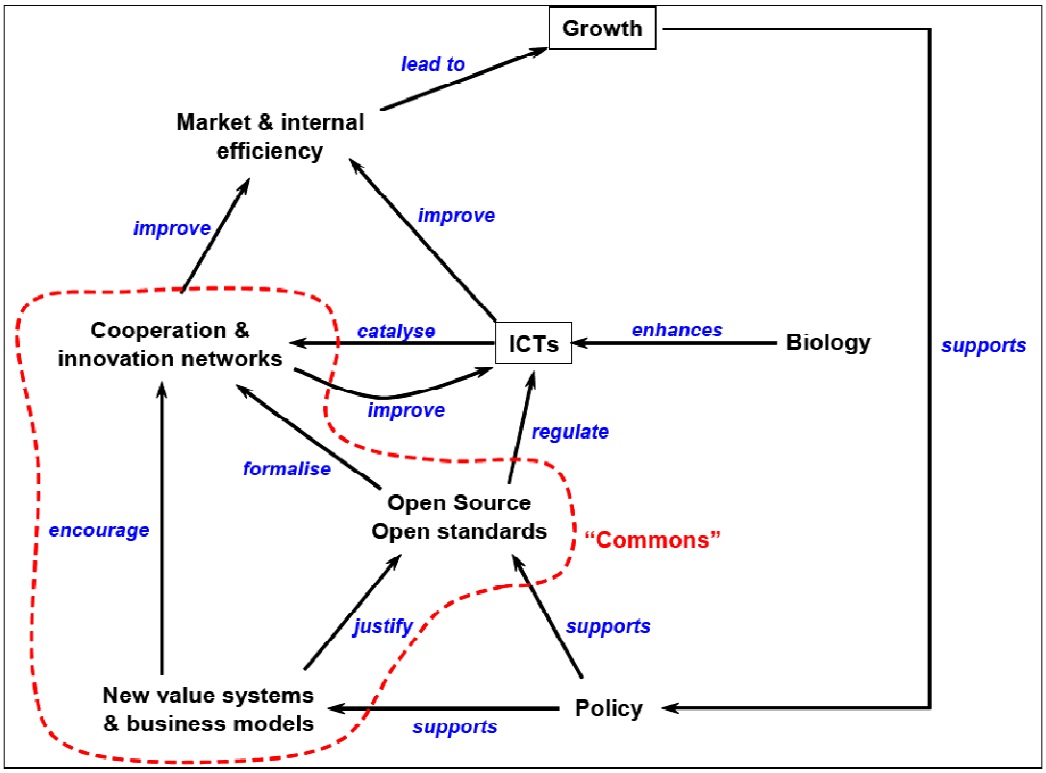

Abbildung 16: Flow diagram for growth catalysed by ICTs

Quelle: (Dini & Nicolai, The Digital Business Ecosystem, 2003)

Der Kreislauf für die Unterstützung von KMUs wird in Abbildung 16 dargestellt. Nachhaltiges Wachstum („Growth“) kann über eine höhere Effizienz am Markt erreicht werden („Market & internal efficiency“). Wie im nächsten Kapitel beschrieben, dient die IKT als Medium für die Bildung von neuen Kooperationen und Innovationsnetzwerken („Cooperation & innovation networks“). Diese Netzwerke werden durch Open Source - Software für KMUs finanzierbar. Offene Standards wiederum ermöglichen die Umset-

[143] (Dini, et al., 2005) S.5

zung von Systemen, die schnell miteinander verbunden werden können. Für Implementierung vor Ort beim Kunden helfen „regional catalysts“. Diese „regional catalysts“ spielen bei der Einbindung des DBE in die reale Wirtschaftswelt eine ganz zentrale Rolle.[144]

Die KMUs selber haben jedoch nur schwer die Möglichkeit diese Aufgaben ohne Unterstützung zu bewältigen. Nachari hat die Probleme im ersten Diskussionspapier zum DBE bereits herausgearbeitet und gleichzeitig Lösungsvorschläge angegeben (Abbildung 17). [145]

Obstacle	Actions
Shortage in skills and knowledge	Creation of local competence centres on e-business Building virtual learning communities Sharing e-learning and e-training modules Sharing knowledge bases of e-business models and DBE instances
Lack of technological interoperability	Use and promotion of open standards Sharing common solutions Implementations of DBEs
Costs	Software sharing, open-source software Open and distributed common infrastructure Reliance on DBEs
Regulatory complexity	Knowledge base of norms and laws Alternative methods of conflict resolution e-learning and e-training modules
Shortage of capital	Support for venture capital, investment fora

Abbildung 17: DBE Aktionen für KMUs

Quelle: (Dini & Nicolai, The Digital Business Ecosystem, 2003)

Die Umsetzung der Aktionen wurde in drei Bereiche unterteilt. Auf der untersten Ebene wird ein Netzwerk entwickelt, das die Grundfunktionen des DBE umfasst. Dabei handelt es sich um die Infrastruktur und die allgemeinen DBE Dienste. Die nächste Ebene beschreibt die sektorspezifischen Aufgaben. In dieser Ebene werden Ontologien, Systeme und Dienste entwickelt, die sektorübergreifend eingesetzt werden können. Die oberste Ebene stellt die lokale Implementierung über die „regional catalysts“ dar.[146]

Im Deliverable 18.1 werden die Anwendungsfälle zum DBE zusammengefasst (Abbildung 18). Im Zentrum entsteht das Business-Ökosystem, das aus einer Reihe von

[144] vgl. (Dini & Nicolai, The Digital Business Ecosystem, 2003) S.4f

[145] vgl. (Nachira, Francesco, 2002) S.19

[146] vgl. (Dini & Nicolai, The Digital Business Ecosystem, 2003) S.4

Unternehmen besteht. Über drei Zyklen soll nachhaltiges Wachstum erreicht werden. Im ersten Schritt über das Social-Economic Environment (SEE). Dabei werden Informationen bei den lokalen Unternehmen gesammelt, zusammengefasst und in regionale, wirtschaftliche Regeln umformuliert. Diese Regeln dienen als Input, um über den MDA-Ansatz, neue Dienste in der Form von Service Factories (Kapitel 4.3.1) zu beschreiben. Die DBE Services werden im ExE (Kapitel 4.3.2) publiziert und über das inkludierte peer-to-peer Netzwerk FADA verteilt. Im letzten Schritt kann sich das System über „Feedback“ und dem genetischen Ansatz innerhalb des EvE (Kapitel 4.3.3) selbst optimieren und neue DBE Services ausbilden.[147] Dadurch kann nachhaltiges Wachstum über die Schaffung von virtuellen Unternehmen, basierend auf DBE Services erreicht werden, weil diese den Markt effizienter mit Angeboten beeinflussen.

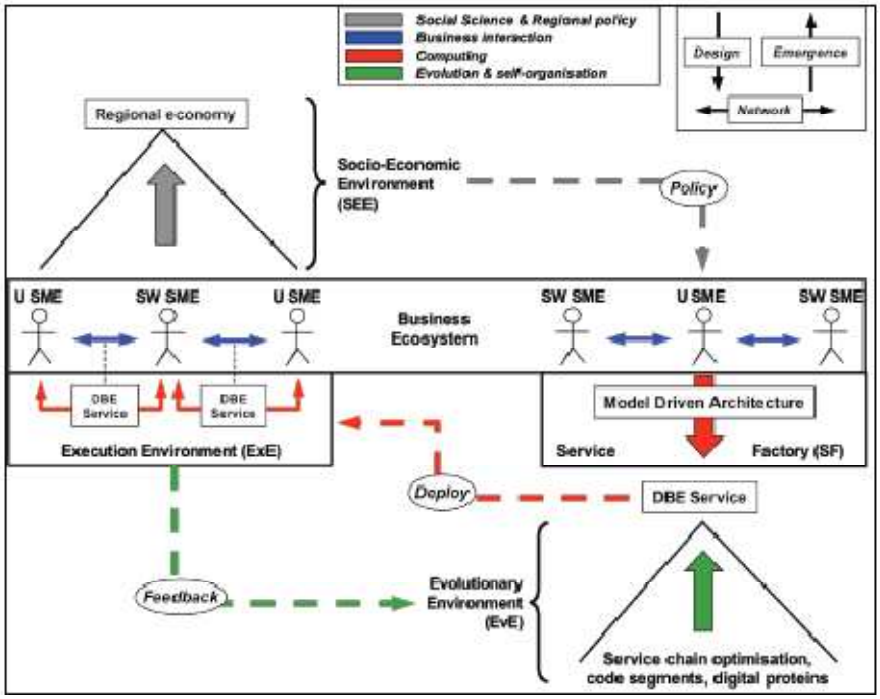

Abbildung 18: The constructive cycles of the DBE

Quelle: (Dini & Berdou, Del.18.1 Report on DBE-specific Use Cases, 2004)

[147] vgl. (Dini & Berdou, Del.18.1 Report on DBE-specific Use Cases, 2004) S.7ff

4.2.4 Sozialwissenschaften

Wie in der Einleitung zum DBE beschrieben, ist die Herausforderung dieses Projektes, die Zusammenarbeit von Personen aus den unterschiedlichsten Bereichen zu ermöglichen. Ein Bereich der sozialwissenschaftlichen Zugangsweise betrifft die Schaffung einer Umgebung, in der die Zusammenarbeit effizient möglich ist.

Ein weiterer Bereich betrifft die Umsetzung der Projekte selbst und die Schaffung einer nachhaltigen Umgebung für die Weiterentwicklung des DBE. Wie im ersten Diskussionspapier definiert wurde, soll das DBE einen starken regionalen Bezug haben und Klein- und Mittelunternehmen unterstützen.[148] Dini definiert die Herausforderung wie folgt:

> *„... develop a consciously and explicitly reflexive methodology of research that can stimulate and inspire a continuous process of self-renewal and innovation in the SMEs and support their participation in the broader scientific community. ... The working assumption so far has been that combining*
>
> 1. *a greater openness to innovation,*
> 2. *a community building process that maximises network effects, and*
> 3. *an enabling open source digital ecosystem infrastructure*
>
> *should lead to an environment where knowledge is constantly created and shared, flowing freely and dynamically where it is needed.“*[149]

Der erste Punkt bezieht sich auf die Art und Weise wie neues Wissen geschaffen werden soll. Bestehende Forschungs- und Entwicklungsabteilungen in Unternehmen sind nicht mehr ausreichend, weil sie zu einem großen Teil über die Abteilungsgrenzen nicht hinaus kommen. Schlagworte, die neue Entwicklungen hervorheben, sind „Open Innovation“ und „Crowdsourcing“.[150] Durch aktuelle Entwicklungen, die zum Teil unter dem Begriff Web2.0 zusammengefasst werden können, aber auch mit dem Open Source Phänomen Linux verbunden sind, entstehen im Internet immer mehr Gemeinschaften, die nicht mehr primär das Ziel der Gewinnerwirtschaftung vor Augen haben, sondern an

---

[148] vgl. (Nachira, Francesco, 2002)

[149] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.26

[150] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.7

einer größeren Sache teilnehmen wollen. Unternehmen können und müssen diese Entwicklungen in ihre eigenen Forschungsarbeiten einarbeiten um mit den aktuellen Entwicklungen mithalten zu können. Geschieht dies nicht, sind Kosten und Dauer für eine Eigenentwicklung unter Umständen zu hoch. Xerox wurde in diesem Zusammenhang von Chesbrough als Beispiel genannt.[151]

Innerhalb des DBE muss daher eine Umgebung für „open flow of ideas“[152] geschaffen werden. Der freie Austausch von Wissen zwischen Universitäten, Wirtschaft und lokalen Regierungen muss ermöglicht werden. [153]

Der zweite Punkt kann als ein Anknüpfungspunkt zwischen den Naturwissenschaften und den Sozialwissenschaften gesehen werden. Im Bereich der Naturwissenschaften wurde die „autopoietic“ beschrieben, d.h. die Eigenschaft der Selbstreproduktion und Selbstverwaltung. Als Anknüpfungspunkt zwischen diesen beiden Theorien kann das soziale Ökosystem gesehen werden. In diesem Bereich agieren selbstorganisierte Agenten und bauen untereinander Assoziationen und Interaktionen auf, die in der Form von Communities dargestellt werden. Um effizient in einem sozialen Ökosystem handeln zu können, müssen sie reflexiv agieren können. Die Frage, die sich stellt ist, in wieweit diese Prozesse in einem gesamten System verselbstständigt werden können.

Als zusätzliches Bindeglied dient eine darunterliegende Software, die das digitale Ökosystem ermöglicht.[154]

Im Rahmen des DBE wurde ein Framework für den Austausch von Free/Open Source (F/OS) Artefakten geschaffen. Als Zielgruppen werden F/OS Gemeinschaften, Unternehmen und öffentliche Institutionen definiert. Eine Hauptaufgabe besteht darin eine Umgebung zu schaffen, in der diese unterschiedlichen Zielgruppen effizient zusammenarbeiten können. Ein entscheidender Faktor ist der Einsatz von F/OS Software. Der Erfolg des DBE-Projektes wird zu einem großen Teil von freiwilligen Helfern abhängig sein, daher wird eine gemeinschaftliche Entwicklung des Systems als unbedingt notwendig erachtet. In diesem Zusammenhang werden die Communities of Practise (CoP) genannt, die Personen aus unterschiedlichen Bereichen vereinen und so den Austausch von Wissen und Innovationen während der Arbeit an dem Projekt erreichen.[155]

Um diese sozialen Netzwerke aufbauen zu können, ist eine Art von „governance“ notwendig, die dem Projekt die richtige Richtung vorgibt. Die Grundprinzipien wurden

---

[151] vgl. (Chesbrough, 2003)

[152] vgl. (Lessig, 2001)

[153] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.7

[154] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.28f

[155] vgl. (Berdou & Dini, 2005) S.5ff

bereits im Diskussionspapier definiert. [156] Die weiteren Charakteristika verlangen eine klare Vorgabe der Werte um die Vision zu erreichen.

Characteristic of digital ecosystem	Dimension of digital ecosystem governance
Shared values, common vision, participation and membership expressed in constitutional documents such as manifesto, bill of rights, codes of practice	Constitution and balance of interests
Transparency, inclustion, due process, policy, procedure and accountability	Culture of communication
Alliance forming and regional coordination, allowing for diverse governance models and diverse membership	Credibility, attunement and trust
Distributed template, lightweight organisation and synchronisation for aligning codebase infrastructure develpoment: association and alliance forming	Organisation and synchronisation
Knowldge and technology licensing, regulatory framework for digital ecosystems e-business interactions and legal definitions relevant to DBE entity	Licensing and regulation
Choice of software develpoment methodologies, technological directions and infrastructural standrads; association and alliance forming	Technological dimension

Abbildung 19: Characteristic of digital ecosystems

Quelle: (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007)

[156] vgl. (Nachira, Francesco, 2002)

4.3 Die Architektur des DBE

Im Bericht von Soluta.net wird die funktionelle Architektur des DBE auf zwei Ebenen beschrieben die für das Businessnetzwerk sichtbar sind. Dies ist das Service Execution Environment, welches ab jetzt mit ExE abgekürzt wird und das Evolutionary Environment EvE. Innerhalb des ExE existiert das Service Factory Environment SF. Jeder Dienst, der im DBE publiziert werden soll, wird im Service Factory Environment mittels eines Service Manifests beschrieben. Das Service Manifest wird im ExE publiziert. Das EvE dient dazu DBE Dienste innerhalb des DBE zu optimieren (Abbildung 20). [157]

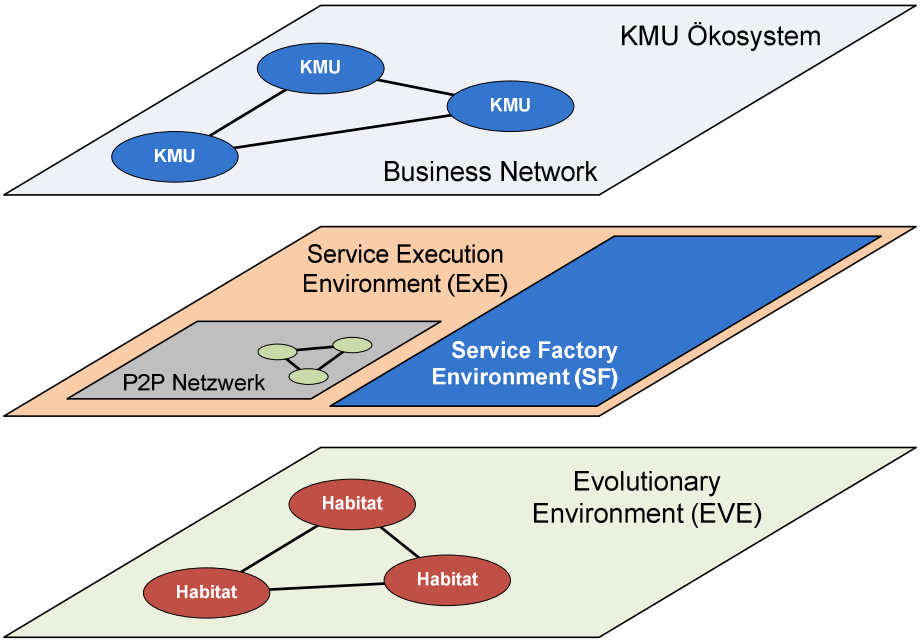

Abbildung 20: DBE Architektur

Quelle: (Soluta.net, 2004)

Für die Umsetzung eines Dienstes im DBE wird der MDA-Ansatz verwendet. Ziel ist, über Modelle die Funktionalität eines Systems abzubilden. MDA kennt drei Sichten (=Viewpoints) die das System von unterschiedlichen Seiten beschreiben. Das Computer Independent Viewpoint zeigt die Anforderungen des Benutzers ohne auf die Struktur bzw. auf die technische Umsetzung zu achten. Es wird auch als Domainmodell bezeichnet und soll die Lücke zwischen den Anwendern und den Programmierern bilden. Darauf aufbauend stellt die Platform Independent Viewpoint die Funktionen dar, ohne je-

[157] vgl. (Soluta.net, 2004)

doch auf eine bestimmte Plattform Rücksicht nehmen zu müssen. In einer Umsetzung kann diese Eigenschaft über virtuelle Maschinen realisiert werden. Die unterste Ebene ist die Platform Specific Viewpoint. Diese Erweiterung stellt das plattform-unabhängige Modell für eine konkrete Umgebung dar. [158]

Über eine Transformation werden die Modelle aus den verschiedenen Sichten ineinander übergeleitet. Der MDA Ansatz basiert auf der Meta Object Facility MOF Spezifikation.[159] Das Ziel von MOF ist es, ein Framework für ein Metadata-Modell zu bieten, das jederzeit angepasst werden kann. Das Framework basiert auf vier Layer. Die oberste Ebene des Meta-Metamodells beschreibt die Struktur und Semantik des Meta-Metamodells. Diese Ebene wird auch als M3 bezeichnet. Der Metamodell Layer beschreibt die Metadaten selbst M2. Die Ebene darunter bezeichnet ein konkretes Modell M1 und die unterste Ebene wird als M0 bezeichnet und stellt die Informationen dar. Sämtliche Strukturen im DBE basieren auf diesem Modell.

4.3.1 Service Factory Environment

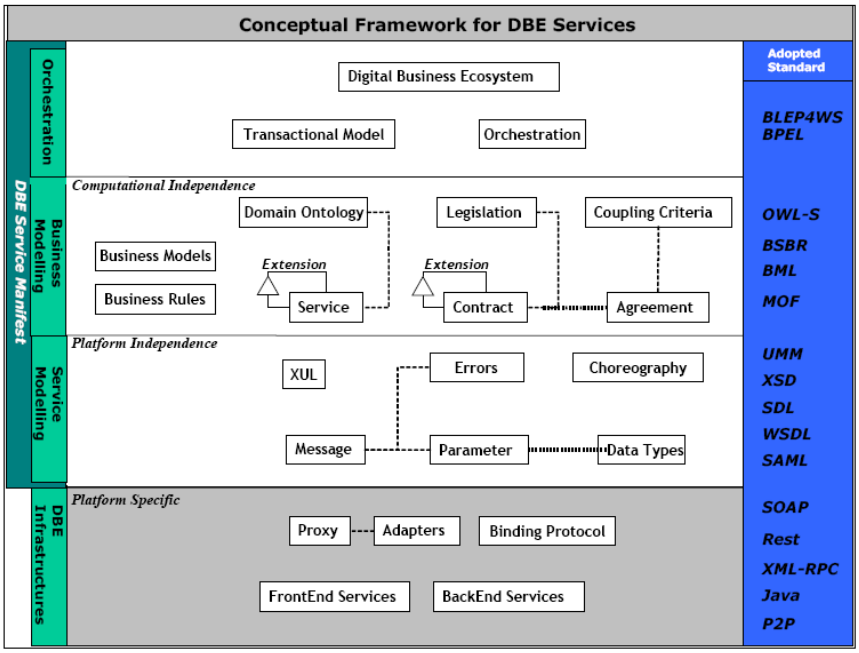

Abbildung 21: Framework for DBE Services

Quelle: (Soluta.net, 2005)

[158] vgl. (Miller & Mukerji, 2003)

[159] vgl. (OMG, 2005)

Paul Malone beschreibt das Service Factory Environment als eine Sammlung von Tools um einen Dienst für das DBE zu beschreiben. Es wird dies als die „design-time of the DBE“ bezeichnet.[160] Das Ergebnis ist das „Services Manifest“ in dem alle Informationen über einen Dienst zusammengefasst werden. Daher wird das Service Manifest auch als „Service DNA“ bezeichnet. Das Framework für die Erstellung der DBE Dienste wird in der Abbildung 21 dargestellt.[161] In diesem Teilkapitel werden die einzelnen Teilaufgaben genauer erläutert. Klar ersichtlich ist der Zusammenhang mit den verwendeten Modellen und Protokollen aus SOA (Kapitel 3).

Die konkrete Erstellung eines DBE Services erfolgt über das DBE-Studio.[162] Das DBE-Studio ist eine Sammlung von Plugins, die in die Umgebung von Eclipse implementiert wurden.[163]

DBE - Plugin	Beschreibung
BML Editor	Der BML Editor dient der Erstellung und der Speicherung der BML Modelle im DBE Knowledge Base (KB)
BML Data Editor	Damit können in die BML Modelle Daten eingefügt werden.
Ontology Analysis Tool	Dieses Tool erlaubt den Import von OWL – Dateien in das DBE Studio.
Service Exporter	Mit dem Exporter können die DBE Dienste zum Servent (bzw. ExE Umgebung) exportiert werden.
Metering Wizard	Ist eine Erweiterung des Service Exporter.
BPEL Editor	Dient der Erstellung von Web Workflows.
SDL Editor	Definiert die technische Seite der Dienste.
SDL2Java Compiler	Damit können aus den SDL Modellen Java Interfaces erstellt werden.
Service Manifest Composer	Das Service Manifest wird in der Semantic Registry deployed.

Tabelle 3: DBE Plugins

[160] vgl. (Soluta.net, 2004), (Malone, 2006)

[161] vgl. (Soluta.net, 2005)

[162] vgl. (DBE Studio, 2007)

[163] vgl. (Edmonds, Dahlem, & McKitterick, 2005) S.56, (Malone, 2006)

Bei der Erstellung eines DBE Services wird strikt nach dem MDA – Modell vorgegangen. Im Gegensatz zum SOA-Ansatz wird im DBE nicht nur die technische Sichtweise beschrieben, sondern auch die wirtschaftliche des neuen DBE-Dienstes. Dadurch entsteht die Möglichkeit intelligente Suchalgorithmen für die Zusammensetzung der Dienste einzusetzen. Wie in der Liste der DBE Studio Plugins ersichtlich, gibt es innerhalb des DBE nicht nur visuelle Tools für die Beschreibung der einzelnen Dienste. Es wird der gesamte Prozess von der ersten Erstellung mittels BML bis zum deployment im ExE über DBE Studio abgebildet.[164]

Dabei kann, wie in der Einleitung beschrieben, die Erstellung der Dienste in drei Schritte aufgeteilt werden.

Im ersten Schritt gilt es die CIM-Ebene („Computer Independent Model“) zu beschreiben. Dabei wird zwischen dem Business Modell und dem Service Modell unterschieden. Das Erstgenannte beschreibt das Unternehmen, die Prozesse und Artefakte die mit dem DBE Dienst verbunden sind. Das Zweitgenannte charakterisiert die konkreten Dienste die für eine Implementierung in einer anderen Software zur Verfügung stehen sollen.

Mittels BML werden das Business Modell und die Business Regeln für den DBE Prozess beschrieben. Das Business Modell, oder der wie im MDA genannte Computational Independence Viewpoint, wird über die zusätzlich zur Business Modeling Language (BML) mittels der Semantic Service Language (SSL) und des Ontology Definition Metamodels (ODM) abgebildet. Über diese Ansicht werden die kundenspezifischen Anforderungen an die Umgebung erstellt. Es erfolgen keine technischen Angaben, sehr wohl aber kann eine Definition der Service Level Agreements in dieser Ebene stattfinden.

Die Business Modeling Language BML Sprache basiert auf der von der OMGs BSBR RFP Anfrage an die WC3 definierten Semantics of Business Vocabulary & Business Rules (SBVR). Der Vorteil von SBVR ist die Definition in einem computerunabhängigen Modell. SBVR kann vom Customer Independent Model CIM in ein Platform Independent Model PIM transformiert werden. Darüber hinaus entspricht SBVR auch den Anforderungen von MOF. SBVR besteht aus Vokabeln, Fakten und Regeln.[165]

Über die Semantic Service Language SSL wird das Service selbst beschrieben. Es gibt darüber Auskunft welche Aufgabe der Dienst hat, wie er arbeitet und welche Voraussetzungen er benötigt. Die Umsetzung basiert in Anlehnung an OWL-S. OWL-S ist ein Modell für den Aufbau von Ontologien für Webservices.[166]

---

[164] vgl. (Soluta.net, 2004) S.18f

[165] vgl. (Corallo, Tommasi, & Cisternino, 2005), (Marcon, Okada, Heistracher, & Kurz, 2006) S.11f

[166] vgl. (TUC, 2005)

Die Informationen aus dem SSL werden für die Suche von bestehenden DBE Diensten im Ecosystem verwendet. Die Voraussetzungen geben darüber Auskunft, wie die benötigten Ressourcen in einem neuen DBE Dienst auszusehen haben.

Das Ontology Definition Metamodel ODM wurde speziell für die Darstellung von domänenspezifischen Ontologien entwickelt. Informationen, die der Ontology Web Language OWL entsprechen, können über XMI in das DBE übernommen werden.[167]

Der zweite Schritt beschreibt das Service Modell und ist die Spezifikation der PIM-Ebene. In dieser Ebene wird die SDL automatisch aus der BML generiert. Zusätzlich können in dieser Ebene die abstrakt definierten Metamodelle des Unternehmens mit konkreten Daten verbunden werden. Dies wird im DBE Studio über den BML Data Editor umgesetzt.

Die Service Modeling Ebene erlaubt es die Modelle in eine plattform unabhängige Struktur zu bringen. Das BML wird in die SDL Service Description Language transformiert. Die SDL beschreibt die technischen Schnittstellen der DBE Dienste[168]. Es wurde in Anlehnung an die Web Service Definition Language WSDL für das DBE angepasst. DBE Dienste können ebenfalls mit einer WSDL Datei versehen werden.[169]

Die letzte Ebene ist die Umwandlung der SDL in ein Interface für die Java-Programmierung. Diese Aufgabe übernimmt der SDL2Java Compiler. Für die Publikation der Dienste im Exe muss noch das Service Manifest erstellt werden. Dafür wurde ein eigener Editor programmiert. In das Service Manifest werden die im Vorfeld erstellten Artefakte angehängt und in weiterer Folge in der Service Registry gespeichert.[170]

Die erstellten Artefakte (BML-Dateien, SDL-Dateien und BML-Daten Dateien sowie zusätzliche Definitionen über Service Level Agreements oder BPEL-Prozesse) werden in das sogenannte Service Manifest zusammengepackt. [171] Wird ein neuer Dienst in der ExE deployed, so wird in einem ersten Schritt überprüft, ob das Service Manifest vorhanden ist. Dieses Service Manifest wird in der Semantic Registry (Kapitel 4.3.2) gespeichert und eine SM-ID wird zurückgeliefert. Existiert der Dienst bereits, so wird der bestehende Dienst gestoppt und der Neue gestartet. Das Service Manifest repräsentiert den publizierten Dienst in der Service Registry. Nur die Beschreibung des Dienstes ohne Daten wird in der Knowledge Base abgespeichert und kann so weiterverwendet werden. In Semantic Registry wird keine Referenz erstellt, sondern eine Kopie. Damit kann

---

[167] vgl. (TUC, 2005)

[168] vgl. (Soluta.net, 2006)

[169] vgl. (Christensen, Curbera, Meredith, & Weerawarana, 2001)

[170] vgl. (Soluta.net, 2004) S.18f

[171] vgl. (Soluta.net, 2004) S.25

das Knowledge Base jederzeit geändert werden, unabhängig von den publizierten Diensten.[172]

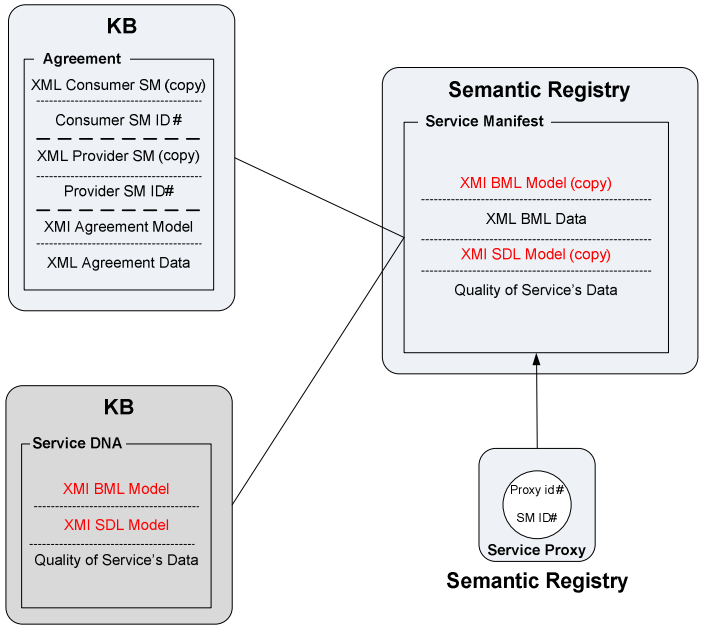

Abbildung 22: Service Manifest, Agreement and Proxy

Quelle: (Soluta.net, 2004) S.25

Neben diesen Standardprozessen für die Implementierung eines DBE Services über das DBE Studio werden in dem laufenden Projekt immer neuere Funktionen hinzugefügt. Diese Funktionen umfassen zum Beispiel die Möglichkeit der Bildung von Workflows über Orchestration. Für die Erstellung eines Workflows wird die Business Process Execution Language BPEL verwendet. BPEL ist ein akzeptierter Standard für die Ausführung von Businessprozessen (Kapitel 3.3.2)[173] Der Standard ist eine formale Spezifikation für die Interaktion und Ausführung von Prozessen. Im DBE wird eine abgewandelte Form der ActiveBPEL in der Execution Environment implementiert. Durch diese Abwandlung können Webservices, DBE Dienste und DBE Workflows in einem Prozess zusammengefasst werden.

Eine weitere Funktion ist die Erstellung von Service Level Agreements. Diese werden an den DBE Dienst im Service Manifest angehängt und können beim Aufruf des Dienstes abgefragt werden. Dadurch können die definierten Bedingungen bereits zu Beginn der Zusammenarbeit über das DBE gelesen werden.

[172] (Edmonds, Dahlem, & McKitterick, 2005) S.61f

[173] (Trinity College Dublin, 2006)

4.3.2 Execution Environment – ExE

Jeder einzelne Knoten in einem DBE Netzwerk beinhaltet eine eigene ExE. Die ExE besteht aus mehreren ineinander greifendenden Tools.[174] Sie bilden die Umgebung, in der die DBE Dienste deployed, gesucht und konsumiert werden können.[175] Die Implementierung des ExE wird als Servent bezeichnet und kann über Sourceforge heruntergeladen werden:

> *„The SERVENT (SERVer + cliENT) "DBE virtual machine" deployed on an SME's computer that enables the consumption of services by acting either as local client or as remote server"* [176]

Der Servent ist das Basissystem für die Bereitstellung der DBE Dienste im Ecosystem.

4.3.2.1 Federated Advanced Directory Architecture FADA

Federated Advanced Directory Architecture (FADA)[177] stellt ein verteiltes Register für die Speicherung von service-proxies zur Verfügung. Das Netzwerk ist in der Form eines P2P-Netzwerkes aufgebaut und basiert auf einem früheren EU-Projekt, das für das DBE adaptiert wurde. Es stellt das sogenannte Nervensystem des DBE zur Verfügung.

> "*The main role of the nervous system is to sore and distribute service proxies. "*[178]

4.3.2.2 Distributed Hashtables DHT

Das DHT ist ein über das P2P-Netzwerk verteilter Hashtable. Er bietet eine Überleitung von einem Identifier auf einem bestimmten Host. Dieser Hashtable wird über das ge-

---

[174] (Kennedy, John, 2007)

[175] (Edmonds, Dahlem, & McKitterick, 2005) S.59f

[176] (Dini, et al., 2005) S.26

[177] (FADA, 2008)

[178] (Wang & De Wilde, 2006) S.14

samte Netzwerk verteilt und erlaubt das Suchen, Einfügen und Löschen von neuen Einträgen.[179]

4.3.2.3 Identity Service IS

Der IS Dienst übernimmt die Authentifikation von DBE-Usern und DBE Diensten im Netzwerk. Sowie die gesamte Struktur des DBE, verfolgt auch dieser Dienst einen dezentralen Ansatz, der entity-centric identity management genannt wird.

Die Identität im DBE wird über ein IDBE (Identity in DBE) erstellt. Dieser basiert auf einem X509 Zertifikat, wobei der IDBE dem Hashcode des öffentlichen Schlüssels entspricht. Der private Schlüssel wird über ein Passwort im PKCS#12-Format gespeichert. Die Zertifikate im DBE können sowohl selbst zertifiziert als auch von einer „Certificate Authority" erstellt worden sein. Zusätzlich hat jeder DBE-User die Möglichkeit mehrere IDBE (Pseudonyme) zu erstellen. Die IDBE und credentials werden im Credential Server (CRES) gespeichert.[180]

4.3.2.4 Semantic Registry

Die Semantic Registry, die von der TUC entwickelt wurde, speichert sämtliche Service Manifeste ab. Diese Datenbank wird bei jeder Suche von einem Benutzer aufgerufen. Die Semantic Registry ist Teil der DBE Knowledge Base KB. Für das Finden eines geeigneten Service Manifests wurde von der TUC der Recommander entwickelt, der die Daten auf Grund einer QML Anfrage sucht.[181]

Für die Verarbeitung von Service Manifesten, die als XML-Dokument gespeichert werden, gibt es den sogenannten SM-Toolkit innerhalb des KB-Toolkits (Abbildung 23: The KB infrastructure). Dieser teilt die SM-XML-Datei in ihre Bestandteile auf (BML, SSL, …).[182]

4.3.2.5 Knowledge Base

---

[179] vgl. (Biskupski, et al., 2005) S.34

[180] vgl. (Biskupski, et al., 2005) S.40

[181] vgl. (Pappas, Kazasis, Anestis, Gioldasis, & Christodoulakis, 2005) S.25, (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007)

[182] vgl. (Pappas, Kazasis, Anestis, Gioldasis, & Christodoulakis, 2005) S.25

Das Knowledge Base ist innerhalb des Servent angesiedelt und bietet alle Funktionen für die Speicherungen der Meta-Daten in einem P2P Netzwerk. Das Knowledge Base ist auf einer nativen XML Datenbank aufgebaut und bietet die Infrastruktur für den Semantic Registry Dienst, den Knowledge-Base Dienst und den Recommander Dienst. Die Kernfunktionen der Knowledge Base Infrastruktur werden in der Deliverable 14.3 zusammengefasst: OMG MOF Unterstützung (M1 Information, M0 Daten); Metamodelle für Ontologien, Business and Services; Unterschiedliche Zugriffsmöglichkeiten. [183]

Die Architektur des KB, in der folgenden Abbildung 23 dargestellt, wird für alle drei Modell Dienste (SR, KB, RC) verwendet. Sie ist ebenfalls in der Deliverable 14.3 detailiert beschrieben.

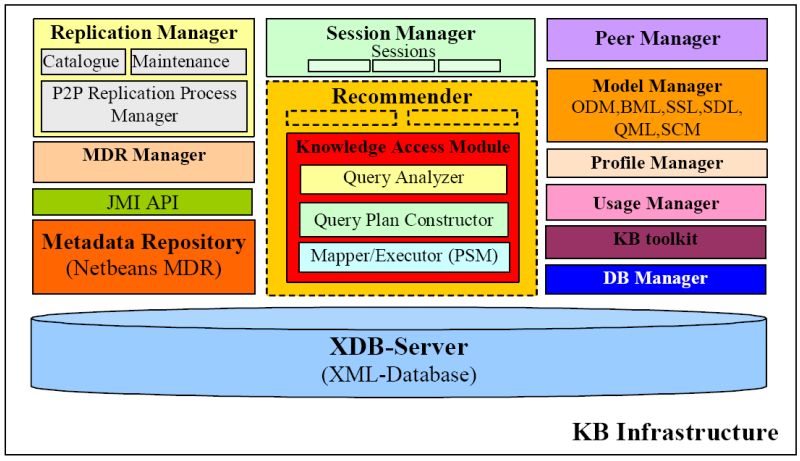

Abbildung 23: The KB infrastructure

Quelle: (Pappas, Kazasis, Anestis, Gioldasis, & Christodoulakis, 2005) S.23

Das KB selbst verwaltet die einzelnen Nodes im Netzwerk über den Peer Manager. Im DBE arbeitet eine große Anzahl an Nodes über ein P2P-System zusammen, auf dem das KB aufbaut. Dabei wird zwischen KB-Peers unterschieden, die Teile des DBE-Wissens speichern, und Simple-Peers die nur auf die Daten zugreifen. Für die Suche in einem solchen Netzwerk wird ein „flooding mechanism“ verwendet. Dieser schickt die Anfragen jeweils an den Nachtbarhost weiter. Die Verteilung der Replikate übernimmt der „Replication Manager“. Der Modellmanager übernimmt die Verarbeitung der MOF kompatiblen Metamodelle (ODM, BML, SSL, SDL, QML, SCM) und speichert sie im

[183] vgl. (Pappas, Kazasis, Anestis, Gioldasis, & Christodoulakis, 2005) S.21f

Data-Management System ab. Die gesamten Daten werden in einer Berkeley XML-Datenbank, dem XDB-Server, abgespeichert. [184]

4.3.2.6 Recommander

Suchanfragen werden durch den Recommander verarbeitet. Diese Suchanfragen werden als QML-Anfragen (Query Metamodel Language) gestellt, die auf OCL 2.0 basieren und für MOF 1.4 adaptiert worden sind. [185]

Das Query Analysis Modul analysiert die QML Anfragen in jedem Peer auf Grund von semantischen Informationen aus dem MDR-Manager (Metadata Repository). Das Query Tree Construction Modul bereitet die Anfrage für die Ausführung im Executor vor. Im Semantic Query Expander Modul wird die Suchanfrage auf Grund von semantisch äquivalenten Informationen erweitert und an das Code Generator Modul weitergegeben, das die Anfrage für die XML-Datenbank in einen XQuery Code umwandelt. Der Executor führt die Anfrage aus. [186]

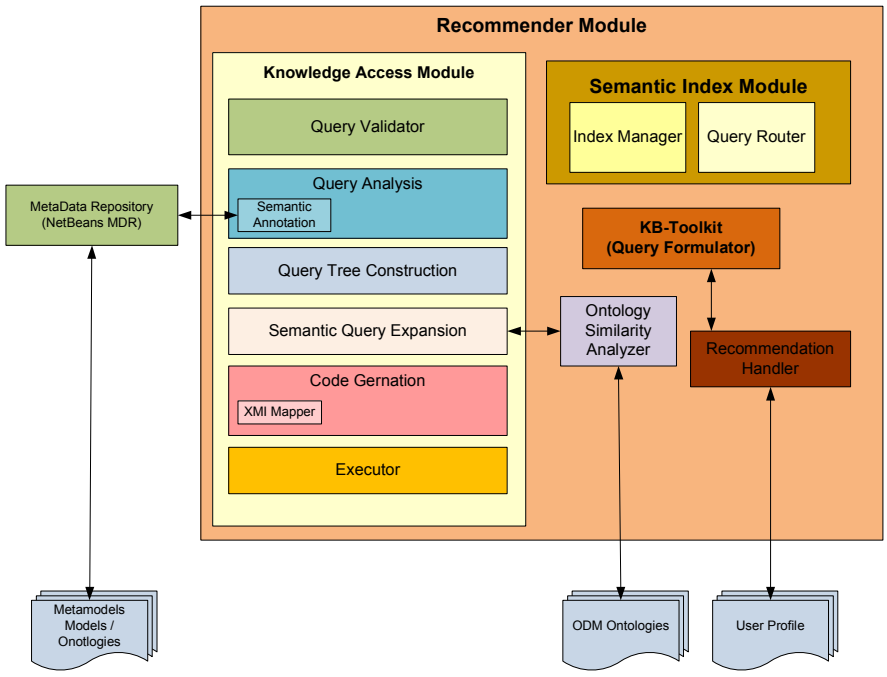

Abbildung 24: The Recommender Module

Quelle: (Kotopoulos, Kotopoulos, & Kazasis, 2006)

[184] vgl. (Pappas, Kazasis, Anestis, Gioldasis, & Christodoulakis, 2005) S.13f, S.22f

[185] vgl. (Pappas, Kazasis, Anestis, Gioldasis, & Christodoulakis, 2005) S.23ff

[186] vgl. (Kotopoulos, Kotopoulos, & Kazasis, 2006) S.15ff

4.3.2.7 DBE Portal

Das DBE Portal wurde ebenfalls von der Firma Intel entwickelt und bietet eine Benutzeroberfläche für das DBE. Es dient dazu Dienste im DBE zu suchen und auszuführen, sowie weiters einen schnellen und unkomplizierten Zugang zum DBE herzustellen. Die Aufgabe bestand darin den DBE Desktop abzulösen und einen kleinen Client zu entwickeln, der unabhängig von der Browserwahl ausgeführt werden kann.

4.3.2.8 Distributed Storage System DSS

Das DSS dient dazu Byte-Arrays redundant im DBE zu speichern. Diese Byte-Arrays können über einen eindeutig identifizierten Hashcode von jedem Node im Netzwerk abgerufen werden. Suchanfragen im DSS betreffen ausschließlich den Hashcode und sind daher sehr einfach. Damit das DBE nicht überlastet wird, haben sich einige Mechanismen etabliert, die die Verteilung der Daten steuern.[187]

Das DSS selbst teilt sich in das DSS und das DFS (DBE File System). Ist es nicht notwendig Daten in einem File System zu speichern, so kann der Overhead gespart werden. Der Dienst baut auf den Servent und den damit verbunden Diensten (FADA, DHT und dem IS) auf. Damit das Netzwerk nicht überfüllt wird, hat jede Information einen time-to-live Parameter, nach welchem die Daten gelöscht werden. Arrays bekommen einen eindeutigen DSS Index, welcher im DHT gespeichert wird. Mittels dieser Referenzen wird die Verbindung zu den gespeicherten Daten hergestellt.[188]

4.3.3 Evolutionary Environment - EvE

EvE hat die Aufgabe für das DBE ein verteiltes Optimierungssystem zu bieten, das auf „evolutionary algorithms“ basiert.[189] Das Ziel ist es, auf der Basis von evolutionären Prinzipien in einem großen Suchraum die beste Lösung für ein konkretes Problem über „try & error“ zu finden.[190]

Das EvE ist in das ExE eingebettet und stellt fünf konkrete Dienste zur Verfügung um die geforderten Aufgaben zu erfüllen.

---

[187] vgl. (Biskupski, et al., 2005) S.15

[188] vgl. (Kennedy & Lee, 2007) S.24f

[189] vgl. (Heistracher, Kurz, Marcon, & Masuch, 2005) S.18

[190] vgl. (De Wilde & Briscoe, 2006) S.2

Jeder Dienst in der ExE wird in der EvE als EveService mit einem Pointer zur Service Factory gespeichert und im EveService-Pool abgelegt. Innerhalb der Service Factory befindet sich das Service Manifest mit der Beschreibung des Dienstes in BML und den BML Daten. Das EveService-Pool repräsentiert alle gespeicherten Dienste eines einzelnen Nodes im Netzwerk. Der Begriff EveSerivce-Pool ist auch als Agent-Pool bekannt.[191][192]

Die EvE Optimierung basiert auf zwei Ebenen. Die erste Ebene ist das verteilte peer-to-peer Netzwerk FADA wie in Kapitel 4.3.2 beschrieben. Auf der zweiten Ebene erfolgt die Optimierung innerhalb eines Peers, dem sogenannten Habitat. Ein Habitat stellt einen Business User im Ökosystem dar und repräsentiert eine Umgebung, in der eine Spezies (ein konkreter DBE Dienst) lebt.[193]

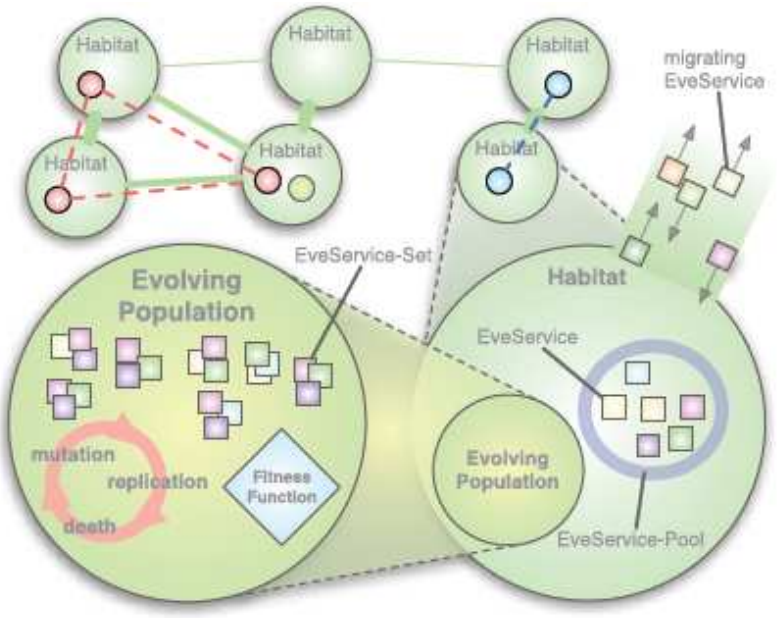

Abbildung 25: Das Habitat

Quelle: (Briscoe, Gerard; De Wilde, Philippe, 2005)

Das Habitat ist die Kernumgebung des EvE (Abbildung 25). In ihm ist der EveService-Pool angesiedelt, einzelne EveServices (Agents) werden zwischen den Habitaten ausgetauscht. Der Begriff dafür nennt sich „migrating EveService“. Ziel ist es die Dienste auf andere Habitate zu übertragen, die das EveService unter Umständen verwenden kön-

[191] vgl. (Briscoe, Gerard; De Wilde, Philippe, 2005) S.53f

[192] vgl. (Briscoe, Gerard; De Wilde, Philippe, 2005) S.5

[193] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.53f

nen.[194][195] Die Theorie hinter dieser Funktion nennt sich „Mobile Agent System“. Dabei soll eine bestimme Aufgabe auf einem anderen Host ausgeführt werden. Auf jedem Host existiert eine Agent-Station, die die Ausführung eines Agents übernimmt. Sie stellt lokale Dienste für die Durchführung der Anfrage zur Verfügung. Schlussendlich sollen die Dienste in Form von Agents unabhängig von herstellerabhängigen Spezifikationen auf allen Hosts ausgeführt werden können und die vom Eigentümer aufgetragenen Aufgaben ausführen.[196]

Damit eine konkrete Applikation entstehen kann, müssen einzelne EveServices und EveService-Sets miteinander verbunden werden. Eine solche Verbindung wird in den unterschiedlichen Publikationen service-set, agent-set, eveservice-set oder Applikation genannt. In diesem Dokument wird in weiterer Folge EveService-Set verwendet.

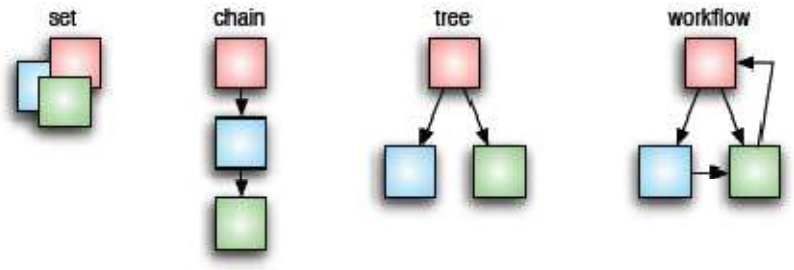

Abbildung 26: EveService-Set

Quelle: (Briscoe, Gerard; De Wilde, Philippe, 2005)

Ein solches EveService-Set kann, wie in Abbildung 26 dargestellt, unterschiedliche Zusammensetzungen haben und repräsentiert jeweils eine eigene Applikation. Diese Verbindung wird danach im EveService-Pool gespeichert und steht für weitere Anfragen zur Verfügung.[197][198] Ein „Workflow“ wird im DBE über BPEL beschrieben (Kapitel 4.3.1).[199]

Im Kapitel 4.2.1 wurde der Begriff Evolution erklärt. Dieser Ansatz wird im EvE durch die unterschiedliche Zusammensetzung von EveServices realisiert und in Form eines

[194] vgl. (Briscoe, Gerard; De Wilde, Philippe, 2005) S.53f

[195] vgl. (Briscoe, Gerard; De Wilde, Philippe, 2005) S.6

[196] vgl. (Briscoe, Gerard; De Wilde, Philippe, 2005) S.13

[197] vgl. (Briscoe, Gerard; De Wilde, Philippe, 2005) S.56

[198] vgl. (Briscoe, Gerard; De Wilde, Philippe, 2005) S.22

[199] vgl. (Briscoe, Gerard; De Wilde, Philippe, 2005) S.15

EveService-Sets im Habitat abgebildet. Welches EveService-Set für eine konkrete Benutzeranfrage die beste Lösung bietet, wird mittels der Fitness-Funktion ermittelt. Ein genetischer Algorithmus stellt verschiedenste Möglichkeiten von EveServices und EveServices-Sets zusammen und prüft diese entlang der Fitness-Funktion. So entsteht ein Optimum, das im EveService-Pool gespeichert und über „migration" an andere Habitate weitergegebenen wird. Wichtig ist, dass nicht alle EveService-Sets in allen Habitaten gespeichert werden. In einem solchen Fall wäre die Suche nach möglichen Lösungen zu zeitintensiv. Tatsächlich werden nur solche EveServices-Sets gespeichert, die das Potential haben von dem Unternehmen genutzt zu werden.[200]

Neben der evolutionären Optimierung wurde in das EvE noch ein distributed Intelligence System (DIS) implementiert. Dessen Aufgabe ist die direkte Migration von Diensten zwischen den einzelnen Habitaten auf der Grundlage von neuralen Netzwerken unter Zuhilfenahme der gespeicherten „Events" im EveService. Das bedeutet im EveService werden „usage events" gespeichert, welche als Input für die Auswahl der Zielhabitate dienen.[201] Die Aufgabe der DIS ist es die Zeit für die evolutionäre Optimierung auf lokaler Ebene zu verringern. Durch die DIS werden gefundene Lösungen gezielt auf andere Habitate übertragen. Dadurch kann die evolutionäre Optimierung schneller eine optimale Lösung für ein Problem finden.[202]

Zwischen den einzelnen Habitaten erfolgt ein ständiger Austausch an Diensten. Umso mehr positive Zusammenhänge es zwischen den einzelnen Habitaten gibt, umso stärker erfolgt die Bindung zwischen ihnen. Die Abhängigkeit wird über das sogenannte „migration feedback" verstärkt. Sie werden verteilt, wenn es eine erfolgreiche Migration gegeben hat. Über die „service-history" kann das Heimathabitat bestimmt werden, bei Durchführung einer positiven Benutzeranfrage (Auslösung eines Migration Feedbacks) wird der Wert „migration probability" mit diesem Habitat erhöht. Der Wert „migration probability" basiert auf den positiven und negativen Migrationen und gibt an, ob ein EveService mit diesem Habitat ausgetauscht werden soll oder nicht.[203]

Die Folge dieser Feedback-Loops ist die Ausprägung von „Habitat Clustern". Diese Cluster bilden reale Wirtschaftscluster ab und symbolisieren Gemeinschaften.

---

[200] (Briscoe, Gerard; De Wilde, Philippe, 2005) S.56f, (Briscoe, Gerard; De Wilde, Philippe, 2005) S.22

[201] (McKitterick, 2006) S.9f

[202] (Marcon, Okada, Heistracher, & Kurz, 2006) S.19

[203] (De Wilde & Briscoe, 2006) S.4

Ein DBE Service im EvE hat folgende Informationen gespeichert:

- **SM-ID**: Ist der Pointer zum Service Manifest, gespeichert in der Semantic Registry in der ExE. Darin ist die Beschreibung des Dienstes als BML und im Speziellen als SBVR Modell gespeichert. Ein Service Manifest kann mehrere Darstellungen im EvE haben.
- **Home Habitat-ID**: Bezeichnet die Reference für das Heimathabitat für die Speicherung von Informationen in der History.
- **MUH (event-history)**: Das MUS speichert die Informationen eines „migration events“ und eines „usage events“ ab.
- **Failure counter**: Gibt die Anzahl der Fehlverwendungen bei Useranfragen an.
- **Intelligence**: Diese Information wird für das DIS benötigt.
- **DIS migrations counter**: Zählt die Anzahl der positiven Migrationen.

Der Zusammenhang zwischen dem EveService und den restlichen Diensten im DBE wird in der folgenden Abbildung nochmals verdeutlicht.

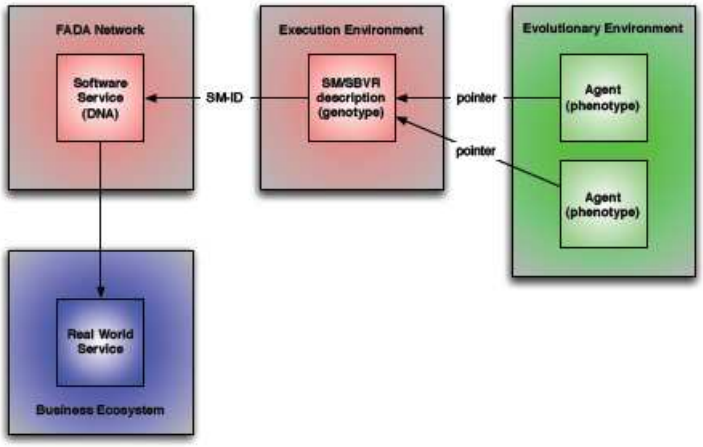

Abbildung 27: EveService relation to DBE service

Quelle: (Briscoe, Gerard; De Wilde, Philippe, 2005)

Das EveService-Set enthält statt dem SM-ID eine Aggregation, der im Set verwendeten Service Manifeste. Der gespeicherte Dienst, der durch das Service Manifest in der Semantic Registry gespeichert wurde, wird über den SM-ID im P2P-Netzwerk FADA publiziert und so für die Verwendung zur Verfügung gestellt.[204]

[204] (Briscoe, Gerard; De Wilde, Philippe, 2005) S.16, S.30ff

4.4 DBE Service Kreislauf

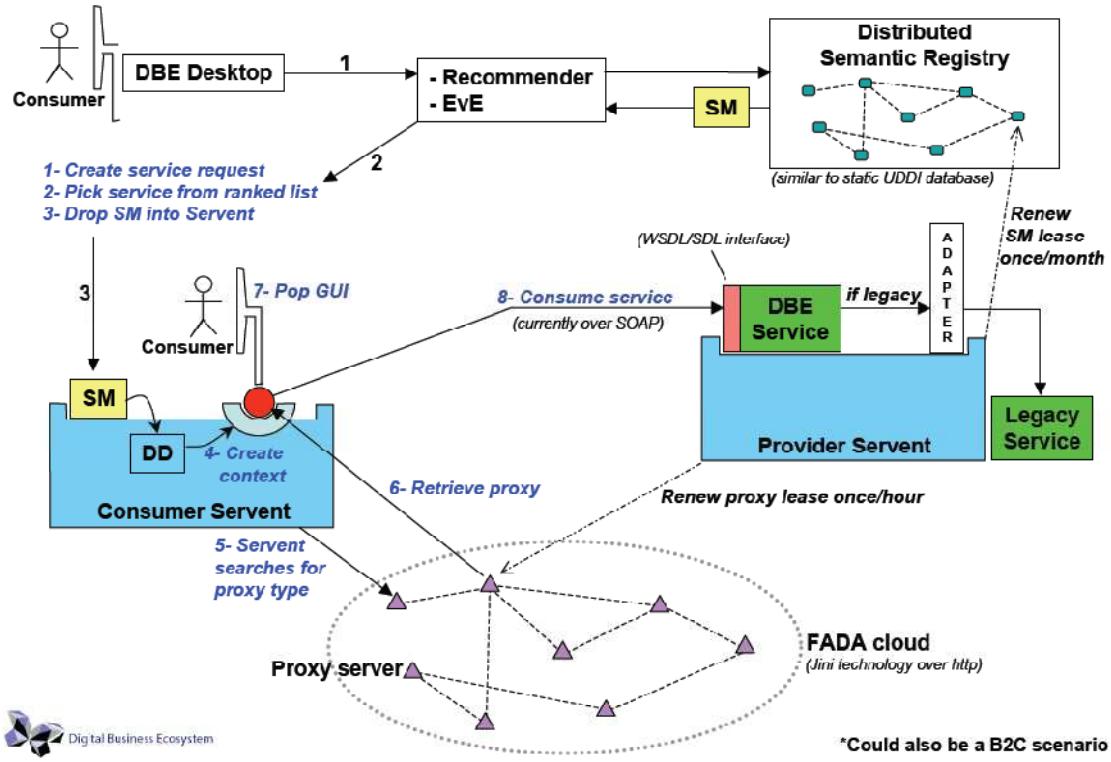

Abbildung 28: Consume Service: Behind the Scenes

Quelle: (Dini, Paolo, 2007)

Der folgende Kreislauf beschreibt die Lösungsfindung eines konkreten Problems eines Konsumenten im DBE. Es soll dabei der genaue Zusammenhang der einzelnen Dienste erläutert werden.[205]

Der erste Schritt (1) wird durch den Konsumenten gesetzt. Dieser gibt eine Anfrage in Form eines SBVR über den DBE Desktop oder das DBE Portal an einen Servent weiter.

Das EvE (2) erhält die Anfrage und bildet eine „envolving population“. Dabei werden der Anfrage entsprechende EveService aus dem EveService-Pool (Genotype) ausgewählt und miteinander verbunden. Es entstehen mehrere „service-chains“ (Genetic Algorithmus - GA Core). Die Dauer dieser Funktion ist mit einer Grenze definiert. Es werden mittels des Fitness Algorithmus mehrere service-chains ausprobiert und dann über die Fitness Landscape die beste digitale Spezies bestimmt. Diese neue optimale Lösung wird an die Nachbarhabitate weitergegeben. Werden die EvE-Dienste nicht

[205] (Wang & De Wilde, 2006) S.40 f, (Kurz, Marcon, Okada, Heistracher, & Passani, 2006) S.10f

mehr verwendet, dann werden sie auch nicht mehr weiter geben und sterben aus. Dadurch wird der Lebenszyklus für einen DBE Dienst bestimmt.

Eine zusätzliche Optimierung erfolgt über das Distributed Intelligence System, das die EveServices auf anderen Habiaten migriert. (In den Habitaten selbst werden nur Pointer zu den Service Manifesten gespeichert.) Der Zusammenhang wird in der Abbildung 29 dargestellt.

Das so gefundene Eve-Service gibt das Service Manifest an den Servent des Konsumenten weiter (3).

Im vierten Schritt (4) wird die Umgebung für die Ausführung der digitalen Spezies (dem DBE Service) geschaffen. Dabei werden über das FADA-Netzwerk (5) die Service-Proxy für die konkreten Dienste gesucht. Dafür wird der im Kapitel 4.3.2 beschriebene Flooting Algorithmus verwendet. Die gefundenen Service Proxy werden an den Consumer Servent weiter gegeben (6).

Im DBE Desktop wird die GUI für den Dienst geöffnet und anschließend auf eine Benutzeranfrage gewartet (7). Nach Eingabe der Daten wird der Dienst am remote Servent ausgeführt (8).

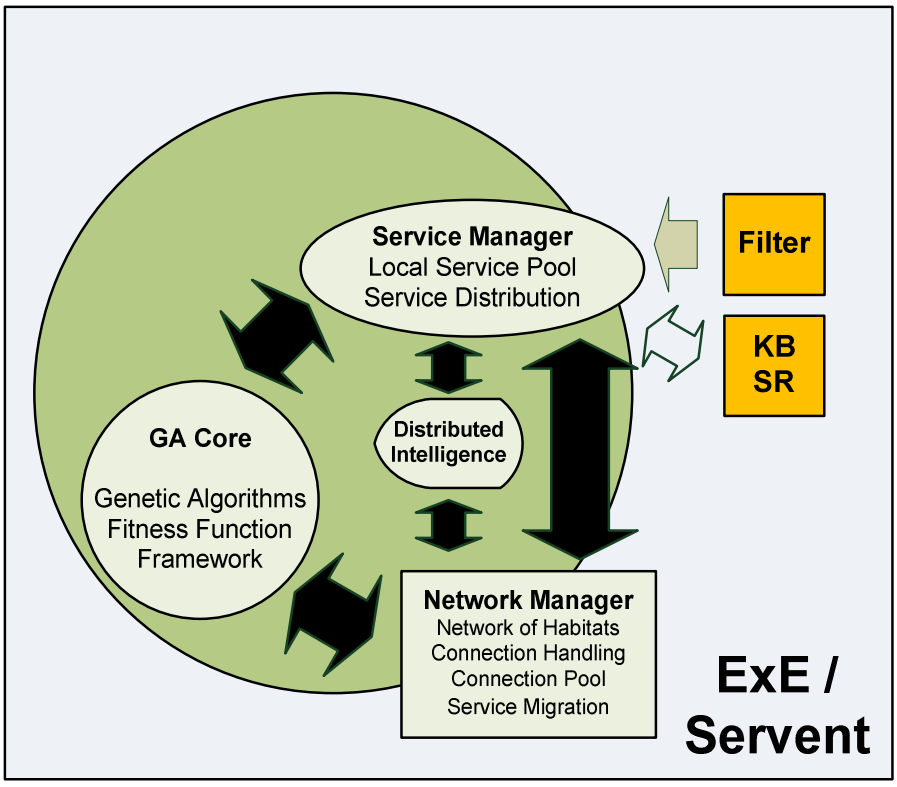

Abbildung 29: Schematic visualization of the EvE

Quelle: (Kurz, Marcon, Okada, Heistracher, & Passani, 2006)

4.5 DBE und SOA

Der Zusammenhang zwischen SOA und DBE wird in mehreren Artikeln zum DBE beschrieben. Je nach Artikel wird das DBE als eine Erweiterung des SOA-Konzepts gesehen oder als eine völlig neue Entwicklung. Auf jeden Fall werden sehr viele Ideen und Standards aus dem SOA-Konzept im DBE übernommen.

Innerhalb der Habitate eines jeden Unternehmens werden die Dienste auf der Basis von SOA entwickelt. Der EvE hat die Aufgabe diese Dienste miteinander zu verbinden.[206] Marcon et al. schreiben:

> *"... the Digital Business Ecosystem is fundamentally an advanced version of a Service Oriented Architecture where several mechanisms help the offering and consumption of services."*[207]

Im Gegensatz dazu schreibt Ferronato:

> *„Business Ecosystem is based on a dynamic interaction of organizations which evolve over time in terms of capabilities and roles ... the Service Oriented Architecture (SOA) is not adequate to face such challenges which are unique in the context of a Digital Ecosystem (DE)"*[208]

Das Kernkonzept von SOA basiert auf Services, der Infrastruktur sowie den Regeln und Prozessen (Kapitel 3.2). Entstanden ist SOA aus der Notwendigkeit, proprietäre Softwareprodukte innerhalb eines Netzwerkes miteinander zu verbinden (Stichwort Enterprise Application Integration). In weiterer Folge wurde das Konzept von SOA vermehrt verwendet, um über das Internet verteilte Aufgaben ausführen zu lassen und dies über eine direkte Verbindung auf der Basis von Webservices.

Ferronato argumentiert, dass SOA gerade in der Verbindung mit UDDI dafür ausgelegt wurde, um technische Informationen (also die WSDL-Dateien) in einer zentralen Datenbank zu speichern. Diese Datenbank bildet die hauptsächlich technische Sichtweise eines Dienstes ab und liefert keine Metadaten. Daher werden bei SOA keine Informationen über die damit verbundenen Unternehmen sowie Business Regeln geliefert. Zusätzlich stellt die zentrale UDDI einen „single point of failure" dar.[209]

---

[206] vgl. (Kurz, Marcon, Okada, Heistracher, & Passani, 2006) S.8

[207] (Marcon, Okada, Heistracher, & Kurz, 2006) S.8 vgl., s.46

[208] (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.97

[209] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.99

Im Hinblick auf die erste Version von SOA (Kapitel 3.3.1) ist Ferronato auf jeden Fall im Recht. Die Ausführung von SOA basiert auf einer zentralen Service Registry UDDI, in der alle Services gespeichert werden. Dort können die Services über eine Referenz auf das Service zugreifen, welches sich verteilt im Netzwerk befindet.

Jedoch befindet sich SOA genauso wie das DBE in einer ständigen Entwicklung. Die Frage ist im ersten Schritt, ob diese Entwicklungen überhaupt miteinander verglichen werden müssen. In der Einleitung zu SOA wurde erklärt (Kapitel 3.1), dass SOA keinem konkreten Framework zugrunde liegt noch mit bestimmten Tools umgesetzt werden muss. SOA ist also eine eigene Art auf Probleme zu schauen.

Auf der Grundlage dieser Sichtweise gibt es natürlich sehr viele Verbindungen zwischen SOA und dem DBE.

Sowohl SOA als auch das DBE basieren auf Diensten, die über das Netzwerk verteilt aufgerufen werden können. Über Standards wie BPEL können bei beiden Prozesse über ein Netzwerk ausgeführt werden.

Ein großer Kritikpunkt bei SOA ist die Service Registry. Diese kann im Fall von UDDI auf keinen Fall dem DBE entsprechen, doch ist die UDDI nur eine Ausprägungsform. Carter definiert eine viel weitreichendere Service Registry als die UDDI, die sehr viele Funktionen des DBE übernehmen kann (Kapitel 3.2.2):

Die Grundfunktion „Publish and find services“ wird von beiden Service Registries gelöst. Im Vergleich zum DBE (Kapitel 4.3.2) wird bei SOA dies aber noch nicht über Metadaten beschrieben, die die Umgebung, in der die Dienste eingebettet sind, beschreiben. Bei der Entwicklung eines ESB wird aber genauso versucht auf Basis von MDA und MOF eine Umsetzung mittels Metadaten zu realisieren (Kapitel 3.2.3). Daher gleichen sich beide Systeme an.

Das Dienste im DBE einen Lebenszyklus haben, wird über den evolutionären Ansatz verstärkt dargestellt (Kapitel 4.3.3). Im SOA wird diese Funktion ebenfalls über ein Managementtool angedacht.

Die dynamische Findung von passenden Lösungen im SOA hängt sehr stark von der Implementierung ab. Im DBE wird dies auf jeden Fall über das EvE und SBVR umgesetzt.

Auch im SOA wird versucht die Dienste zu optimieren. Das funktioniert in diesem System über Metriken, die innerhalb der Service Registry definiert werden. Das DBE verwendet den EvE und die DIS.

Nach der ersten Version von SOA gehen sehr viele Gedanken in eine ähnliche Richtung wie bei der Umsetzung des DBE. Genauso sieht es beim ESB in SOA aus. Auch dieses System soll, wie beim DBE das FADA Netzwerk, das Nervensystem darstellen. Die Frage, die es zu klären gilt ist, inwieweit dieses Netzwerk skaliert werden kann. In den

Darstellungen wird meistens von einer bestimmbaren Anzahl an Partnern gesprochen, die unter Umständen über einen ESB-Gateway auf eine unbestimmbare Anzahl erweitert werden können.

Aus dieser Schlussfolgerung kann das DBE als eine Art von SOA interpretiert werden, weil SOA als ein Konzept und das DBE auf Diensten basiert, die über eine gemeinsame Infrastruktur miteinander verbunden sind und die auf der Basis von Regeln und Prozessen miteinander interagieren.

Was DBE jedoch noch von anderen „SOA-Umsetzungen“ unterscheidet ist die strikte Orientierung am Aufbau eines ökologischen Systems sowie die Einbindung von genetischen Algorithmen, die dieses Systems selbstständig innerhalb des Habitats weiter entwickeln. Darüber hinaus ist das DBE nicht nur eine technische Umsetzung einer Middleware-Lösung für die Zusammenfassung von Softwareapplikationen. Das DBE ist auf Grund seiner Wurzeln eine Bewegung die KMUs unterstützen möchte. Daher sind sehr viele Projekte, die zwar den DBE Cluster, aber nicht die DBE Kerninfrastruktur betreffen, in nicht technischen Disziplinen zusätzlich angesiedelt.

Aufgrund der Gesichtspunkte, dass SOA und DBE beide auf den Paradigmen von SOA aufbauen und der Betrachtungsweise der Kernfunktionen des DBE, ist meine Meinung, dass das DBE eine Variante des SOA umsetzt: gerade auch deswegen, weil viele Standards die bei SOA verwendet werden, auch im DBE zum Einsatz kommen. Es muss aber berücksichtigt werden, dass für den Fall, dass das DBE eine Art von SOA ist, es noch keine vergleichbare Umsetzung in einem anderen Framework gibt und rein das Konzept von SOA auf das DBE übertragen werden kann. SOA in der Ausprägung als Webservices über einen Enterprise Service Bus vermag es nicht, ein einziges Netzwerk aufzubauen, über das alle Dienste für alle zur Verfügung stehen.

4.6 DBE und SCM

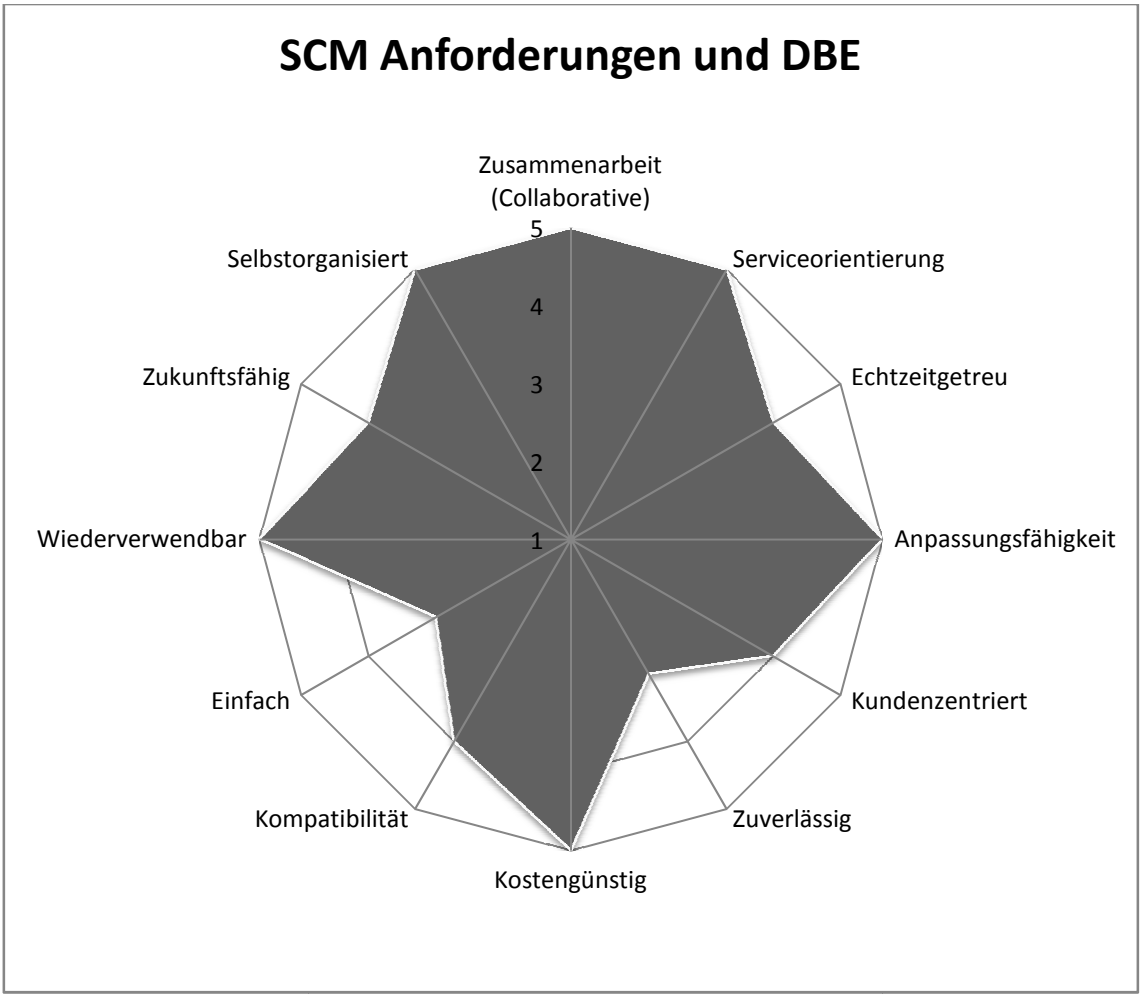

Abbildung 30: SCM Anforderungen und DBE

Im folgenden Teilkapitel werden die Anforderungen aus dem Kapitel 2.3 auf das DBE umgelegt.

Nach den selben Kriterien wie im Kapitel 3.4 werden in der Abbildung 30 die Umsetzung der einzelnen Anforderungen im DBE bewertet und in weiterer Folge näher erläutert. Die Tabelle bietet eine Zusammenfassung der Erkenntnisse mit einem Bezug auf die relevanten Teile in diesem Kapitel.

Anforderung	Kapitel
Zusammenarbeit (Collaborative)	4.2.4
Serviceorientierung	4.3.1, 4.4
Echtzeitgetreu	4.3.2, 4.3.3
Anpassungsfähigkeit	4.3.1
Kundenzentriert	4.2.3
Zuverlässig	4.3.3
Kostengünstig	4.2.3, 4.2.4
Kompatibilität	4.3.1, 4.3.2
Einfach	4.2.1, 4.2.2
Wiederverwendbar	4.3.1
Zukunftsfähig	4.1
Selbstorganisiert	4.2.1

Tabelle 4: Vergleich SCM und DBE

4.6.1 Zusammenarbeit (Collaborative)

Einer der Kernpunkte für das DBE ist das Thema Zusammenarbeit. Im Bereich der Sozialwissenschaften werden Themen wie „Community of Practice/Network of Practice“ aufgearbeitet die Personen aus unterschiedlichsten Berufsfeldern zusammenbringen.

Das Projekt selbst hat das Ziel, die Netzwerke unter den Klein- und Mittelbetrieben zu verstärken bzw. neu aufzubauen. Neben einem starken regionalen Bezug, bei dem auf die kulturellen Unterschiede eingegangen wird, soll ein Netzwerk auf internationaler Ebene entstehen.

4.6.2 Serviceorientierung

Das Thema SOA wird in sehr vielen Artikeln zum Thema DBE angeschnitten und es werden Vergleiche mit der vorherrschenden Meinung zu SOA gezogen (Kapitel4.5).[210] Auf jeden Fall entsprechen die gesamte Struktur und auch der Aufbau einer serviceorientierten Sichtweise. Dies geht sogar so weit, dass die Dienste immer kleiner gestaltet werden, damit sie in weiterer Folge auf Basis von evolutionären Algorithmen selbständig neue Lösungen finden.

Darüber hinaus wird der Einsatz von Workflows auf Basis von BPEL eingesetzt, damit auch den Anforderungen des Prozessmanagements Genüge getan wird (Kapitel Teil I2.1.4).

4.6.3 Echtzeitgetreu

Das DBE erlaubt es, Dienste von unterschiedlichsten Unternehmen schnell miteinander zu verbinden. Die Geschwindigkeit des DBE wird in sehr vielen Algorithmen zu optimieren versucht (zum Beispiel DIS über Service Migrationen). Im Bullwhip-Effekt geht es darum Daten von vielen Kunden beim Verkäufer zusammen zu tragen, damit eine zentrale Vorhersage über die benötigten Produkte erstellt werden kann. Genau das bietet das DBE.

Da jedoch das DBE noch in den Kinderschuhen steckt, kann die Auswirkung der Optimierungsalgorithmen in einem großen System mit mehreren tausend Teilnehmern nur sehr schwer vorhergesagt werden, daher wird diese Anforderung mit vier bewertet.

4.6.4 Anpassungsfähig

Im DBE wird von der Möglichkeit gesprochen, spontane Netzwerke aufzubauen. Es ist wichtig, dass die Infrastruktur die KMUs unterstützt und zudem die Wendigkeit der KMUs nicht behindert werden. Durch die Verwendung von bestehenden Standards und aktuellen Technologien können nicht nur die Systeme untereinander schnell angepasst werden, sondern auch mit bestehenden Systemen im Unternehmen verbunden werden.

Darüber hinaus werden Dienste im DBE in einem ersten Schritt nicht technisch beschrieben. Es wurde eine eigene Sprache für das DBE angepasst, damit die nichtfunkti-

[210] vgl. (Nachira, Nicolai, Dini, Le Louarn, & Leon, Digital Business Ecosystems, 2007) S.97ff

onalen Anforderungen schnell eingearbeitet werden können. Dadurch ist ein Weg geschaffen worden, durch den auch nicht technisch ausgebildete Personen ihre Bedürfnisse aus dem SCM schnell auf das Netzwerk übertragen können. Durch die Struktur im DBE werden die Änderungen sofort wirksam.

4.6.5 Kundenzentriert

Bereits im ersten Diskussionspapier zum DBE wurde auf die Aufgabe der Unterstützung der KMUs eingegangen.[211] Damit ist das gesamte Projekt mit all seinen Teilgebieten auf dieses Ziel ausgerichtet und orientiert sich dahingehend.

Wie im Kapitel 2 gefordert, muss der Kundenwert (Kapitel Teil I2.1.2) erhöht werden. Da das Projekt zu einem großen Teil durch die EU gesponsert wird, entstehen keine Kosten für die KMUs. In der Umsetzung eines konkreten Projektes beim Kunden sind die Kosten daher entscheidend billiger als vergleichbare Technologien (Kapitel 4.2.3).

4.6.6 Zuverlässig

Für den Einsatz im Lieferkettenmanagement müssen sich die einzelnen Partner auf die Funktionsfähigkeit des Systems verlassen können. Innerhalb des DBE wird versucht dieser Anforderung über die Struktur des Servent zu entsprechen. Alle Dienste sollen über das peer-to-peer Netzwerk immer verfügbar sein. Informationen in der Knowledge Base und in der Service Registry werden als Kopie gespeichert, damit Änderungen nicht das gesamte System betreffen. Zusätzlich werden noch weitere Dienste für das DBE entwickelt, die die Zuverlässigkeit erhöhen. Derzeit kann dieser Punkt noch nicht mit 100% Sicherheit garantiert werden.

4.6.7 Kostengünstig

Wie in der Anforderung zur Kundenzentriertheit, soll auch hier gesagt werden, dass sich durch geringe Kosten für die KMUs der Kundenwert erhöht. Durch die durchgängige Verwendung von Free/Open Source Anwendungen entstehen für die KMUs keine Kosten für Lizenzen oder für die Verwendung des Netzwerkes auf Grund eines zentralen

[211] (Nachira, Francesco, 2002)

Service Broker. KMUs sollen durch dieses Projekt unterstützt und damit der Engpass an möglichen Mitteln für Investitionen verringert werden.

4.6.8 Kompatibilität (Interoperable)

Das DBE entspricht seit seiner Entstehung den Anforderungen laut dem OpenICT Ecosystem da:

- *„es nicht von einer Person oder Institution kontrolliert werden kann.*
- *es in einem transparenten und offenen Prozess entwickelt und verwaltet wird.*
- *es plattform und herstellerunabhängig ist.*
- *Spezifikationen und Anleitungen öffentlich publiziert werden.*
- *es kostenfrei oder mit minimalen Kosten verfügbar ist.*
- *es mit einem breiten Konsens der Teilnehmer entwickelt wurde.“* [212]

Es werden alle Entwicklungen als F/OS publiziert und auf mehreren Portalen zur Verfügung gestellt.[213] Im Projekt werden Open Standards verwendet, die es jedem Unternehmen ermöglichen, sich schnell in dieses System einzugliedern und seine eigenen Dienste anzubieten. Die Spezifikationen zum Projekt können von jedem eingesehen werden.[214] Der Prozess der Erstellung des Projektes ist transparent und kann auf der Internetseite des DBE mit verfolgt werden. Darüber hinaus wurden die aktuellen Entwicklungen im Bereich von Open Innovation aufgegriffen und so Communities aufgebaut, die beim weiteren Aufbau des Projektes mithelfen sollen.

Die letzte Forderung wird durch die Einbindung von den drei identifizierten Zielgruppen erreicht: Universitäten, Wirtschaft (KMUs) und lokale Regierungen.

Weiters werden bestehende Webservice-Schnittstellen unterstützt. Dadurch können die Vorzüge aus dem SOA Konzept in die Entwicklung des DBE mit einfließen. Über diesen Weg ist eine Eingliederung an bestehende Systeme jederzeit möglich. Unternehmensintern ist die Frage, in wieweit das DBE hier zum Zug kommt. Ganz klar müssen dafür klare und überschaubare Sicherheitsrichtlinien geschaffen werden, damit unerwünschten Zugriffen Einhalt geboten werden. Für die Integration von Unternehmensinternen Applikation ist DBE womöglich nicht die richtige Wahl.

---

[212] (Open ePolicy Group, 2007)

[213] http://www.sourceforge.net, http://www.digital-ecosystems.org, http://www.digital-ecosystem.org, …

[214] http://www.opaals.org/

4.6.9 Einfach

Das DBE selbst nimmt seit seiner Entstehung ständig an Komplexität zu. Durch die Verwendung von neuartigen Algorithmen aus den Naturwissenschaften und unter Berücksichtigung von sozialen Entwicklungen sowie dem damit verbundenen Einfluss vieler anderer Wissenschaften ist die Einfachheit des Projektes selbst nicht mehr gegeben.

Dies wird aber auch nicht gefordert, denn nur dadurch kann die Verwendung durch die KMUs für diese so einfach wie möglich gestaltet werden. Für KMUs können Dienste in einer nichttechnischen Sprache dargestellt werden, sodass mit der Umsetzung schneller begonnen werden kann. Die Struktur selbst übernimmt die Verteilung der Dienste und optimiert die Ergebnisse für den Kunden.

4.6.10 Wiederverwendbar

Das DBE setzt sehr stark auf Wiederverwendbarkeit in einer neuen Art und Weise. Dienste und deren Aufgabengebiet werden zunächst sehr klein definiert und daraufhin miteinander verbunden. Jeder Dienst wird zu einer digitalen Spezies, die in der digitalen Welt aufwächst und sich selbst reproduzieren kann. Wenn die Spezies nicht mehr gebraucht wird, dann stirbt sie. Das Ziel ist es nicht mehr alles neu zu programmieren, sondern die digitalen Spezies zu verwenden.

4.6.11 Selbstorganisation

Das gesamte Framework von DBE ist dafür entwickelt worden, sich selbst zu organisieren und die optimale Lösung für eine Benutzeranfrage zu liefern. Als Hauptansatzpunkt dient dafür das Evolutionary Environment. Für jeden DBE Dienst wird eine Referenz im EvE angelegt. Das EvE übernimmt die Optimierung der Zusammensetzung der einzelnen Dienste. Auf Basis eines evolutionären Algorithmus werden für eine Kundenanfrage verschiedene Dienste zu einer Applikation zusammengesetzt. Für die Bestimmung einer optimalen Lösung wird eine „Fitness Landscape“ gebildet. Mittels dieses Algorithmus erfolgt die „natürliche Selektion“ in der digitalen Welt. Diese Applikation wird gespeichert und kann für weitere Anfragen verwendet werden.

Darüber hinaus werden über ein DIS Distributed Intelligent System die einzelnen Dienste auf Basis von vergangenen Werten ausgetauscht. Das passiert ebenfalls automatisch.

4.6.12 Zukunftsfähig

Das Open ICT Ecosystem verlangt von einem System dessen „Sustainability“. Es muss ein Framework gebildet werden, in dem sich das Ecosystem ausbreiten und weiterentwickeln kann. Bestehende Komponenten müssen wiederverwendet werden können und mit neuen Entwicklungen verbunden werden.

Das DBE hat alle Voraussetzungen zukunftsfähig zu sein. Nicht nur aus technischer Sicht werden Standards mit hohen Fähigkeiten für zukünftige Programme verwendet, sondern der gesamte Prozess für die Erstellung des Projektes bis hin zu seiner Verwendung und späteren Weiterentwicklung verlangt die Bildung von nachhaltigen Werten für dieses Projekt.

Teil III

VMI im Digital Business Ecosystem

Kapitel 5

Design eines VMI Multi-Agent Systems im DBE

5.1 Einleitung

Im Kapitel 2 wurde bereits beschrieben, dass das Vendor-managed Inventory Modell (VMI) eine Form eines Continuous Replenishment Programms (CRP) ist. Ziel des VMI ist es, die Aufgaben im Supply Chain Netzwerk neu zu verteilen und dadurch die Probleme der traditionellen Supply Chain zu minimieren.

Abbildung 31: Schematic of a traditional supply chain

Quelle: (Disney & Towill, 2003) S.202

Ausgangspunkt ist die sogenannte traditionelle Supply Chain. Die traditionelle Supply Chain liefert Waren von der Fabrik in das Verteilerzentrum zum Kunden (innerhalb der Supply Chain arbeiten eine Vielzahl an Teilnehmern mit: Wiederverkäufer, Großhändler, Zwischenlager, usw.). Informationen werden an die vorherige Ebene ausschließlich über den Verlauf der Bestellungen weitergegeben. Die Fabrik erhält am Ende der Supply Chain ebenfalls nur die Zahl der konkreten Bestellungen, die zum größten Teil nichts mehr mit der tatsächlichen Nachfrage vom Kunden zutun haben und stark variie-

ren. Jeder Teilnehmer in der Lieferkette muss für sich die Menge an Waren, die bestellt bzw. produziert werden soll vorhersagen und hat dafür nur seine eigenen Daten zur Verfügung. Damit das System effizient funktionieren kann, werden zwischen den Teilnehmern in einer Lieferkette Verträge ausgearbeitet, die die Verfügbarkeit der Waren gewährleistet. Die sogenannten Service Level Agreements (SLA) definieren die Zeiten und Qualitäten innerhalb derer die Waren geliefert werden müssen. Damit ein Lieferant diese Verträge einhalten kann sind zum Teil enorme Sicherheitsbestände in den Warenhäusern notwendig.[215]

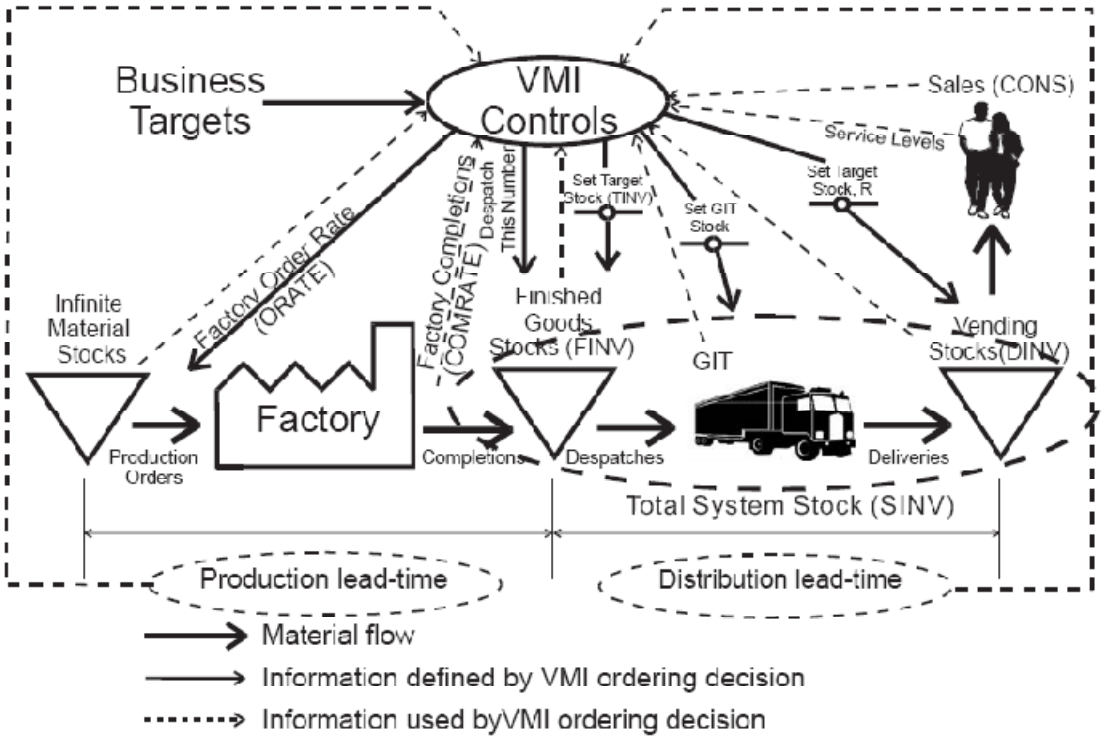

Abbildung 32: Overview of the VMI scenario

Quelle: (Disney & Towill, 2003) S.202

Die Probleme aus der traditionellen Lieferkette wurden in Form des Bullwhip-Effektes dargelegt und greifbar gemacht. Um Teile dieses Problems und die damit verbundenen Kosten zu umgehen, sind neue Wege der Zusammenarbeit gefragt. Als eine Lösung wird das Vendor-managed Inventory Modell gesehen (Abbildung 32). [216]

[215] vgl. (Disney & Towill, 2003) S.200

[216] vgl. (Disney & Towill, 2003) S.201f

Durch die Einführung von VMI sollen auf der einen Seite die Lagerbestände reduziert und auf der anderen Seite die Liefertreue erhöht werden. Dieses Ziel kann nur dadurch erreicht werden, dass die Daten für die Vorhersage der zukünftigen Mengen direkt an alle Teilnehmer weitergegeben werden.

Damit das System entsprechend funktioniert, benötigt es den Austausch von aktuellen Daten. Dieser Austausch ist auf Grund der hohen Mengen und der Regelmäßigkeit zum überwiegenden Teil nur auf elektronischem Wege möglich. Mittels diesen Daten, den sogenannten POS-Daten, kann der Lieferant die benötigte Menge des Kunden bestimmen und die Lager auf der Basis von definierten SLA (service level agreements) auffüllen. [217] POS Daten werden definiert als:

> *"product related information captured at an identified point of sale and generated by the act of purchase."*[218]

Ein einfaches Beispiel für ein Vendor-managed Inventory Modell wird in der Abbildung 33 dargestellt.

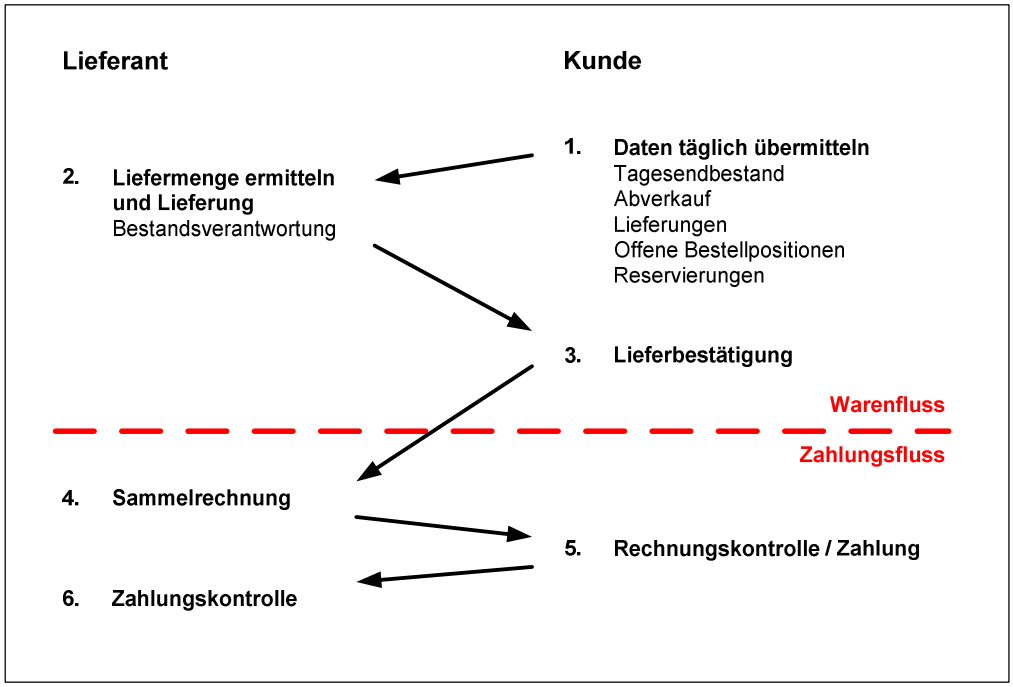

Abbildung 33: ECR/VMI – Logistik

Quelle: (stratEDI Gesellschaft für Kommunikationskonzepte und -lösungen mbH, 2002)

[217] vgl. (Disney & Towill, 2003) S.200f

[218] (Deloitte Consulting, 2003) S.8

Nach Abschluss einer Vereinbarung über die Durchführung von VMI (1) sendet der Kunde die POS Daten an den Lieferanten. (2) Der Lieferant bestimmt über ein DSS (Decision Support System)[219] die optimale Bestellung für den einzelnen Kunden (unter Berücksichtigung seiner anderen Kunden). Die Lieferung wird durch den Lieferanten eingeleitet. (3) Es wird nur die Lieferbestätigung an den Kunden geschickt. (4) Zum Abschluss wird eine Sammelrechnung an den Kunden geschickt, (5) der diese begleicht und (6) welche vom Lieferanten kontrolliert wird.

VMI wird in verschiedensten Varianten dargestellt und beschrieben. Eine der ersten Versionen des VMI wurde im Rahmen des Efficient Consumer Response (ECR) Modells dargestellt. In diesem Modell wird das sogenannte Continuous Replenishment Programm (CRP) vorgestellt. Es beschreibt, wie der Verkäufer das Lager für seinen Kunden auf Basis der Übergabe der Lagerdaten aus dem Verteilerlager des Kunden, auffüllt. CRP umfasst zwei Varianten des Prozesses. Der Unterschied der beiden Modelle liegt darin, wer den konkreten Auftrag für die Bestellung der Waren abgibt. Die Berechnung der benötigten Mengen ist zum größten Teil gleich. In der Form des VMI gibt der Lieferant auch gleichzeitig die Bestellung für den Kunden ab. Dabei ist diese Strategie in ein Gesamtkonzept eingegliedert.[220]

Eine Weiterentwicklung des ECR stellt das Collaborative Planning, Forecasting and Replenishment (CPFR) dar. Es wird als eine der Hauptaktivitäten für die Umsetzung von ECR angesehen und gibt genaue Vorgehensmodelle an.[221] Für die Umsetzung des VMI in dieser Arbeit dient das CPFR Modell als Vorlage.

In dieser Arbeit wird die Herausforderung der Umsetzung dadurch erschwert, dass dieses System über das Digital Business Ecosystem stabil auf einem Massenmarkt etabliert werden soll. In einem solchen Fall kann nur über sehr hohe Kosten mit jedem einzelnen Partner eine eigene Verbindung aufgebaut werden. Gerade für KMUs sind diese Kosten und die damit verbundenen Risiken ein Hinderungsgrund für die Umsetzung eines solchen Projektes. Darüber hinaus sollten mit der Umsetzung eines VMI die technischen Anforderungen an ein SCM-System ebenfalls umgesetzt werden (Kapitel 2.3): Zusammenarbeit, Serviceorientierung, echtzeitgetreu, Anpassungsfähigkeit, kundenzentriert, zuverlässig, einfach, wiederverwendbar, zukunftsfähig und selbstorganisiert.

---

[219] (Achabal, Mcintyre, Smith, & Kalyanam, 2000)

[220] vgl. (Roland Berger & Partner, 1996) S.10ff

[221] vgl. (Accenture, 2001) S.5

5.2 Vorgehensweise

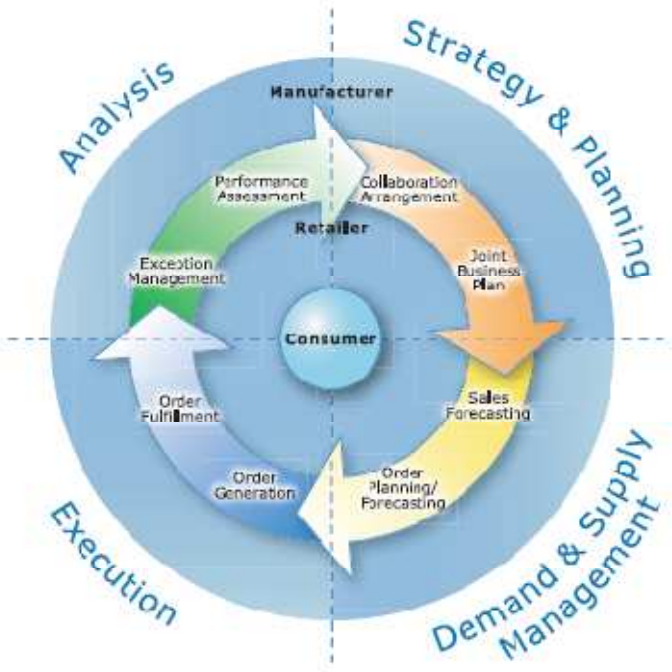

Abbildung 34: DC Replenishment Collaboration Process Overview

Quelle: (Bozarth, 2007)

In diesem Kapitel wird die Architektur des VMI unter den im Punkt 5.1 definierten Gesichtspunkten auf das Digitale Business Ecosystem umgelegt. Die Prozesse für das VMI basieren, wie in der Einleitung definiert, auf dem CPFR DC Replenishment Process. Dabei wird rein auf das SMR (supplier-manager release) Modell und im Speziellen auf die Option D des Modells (Retailer VMI) eingegangen.[222] Dabei werden entlang der vier Bereiche „Strategy & Planning“, „Demand & Supply Management“, „Execution“ and „Analysis“ die einzelnen Aufgaben des VMI abgearbeitet.

Für die Umsetzung eines VMI über das DBE wird ein Multi-Agent System aufgebaut. Jeder Agent wird als ein DBE Service entwickelt, das wiederum aus mehreren DBE Services besteht: eine sogenannte service-chain. Diese DBE Services können mittels BPEL zusammengesetzt werden. Als Grundkonzept wird die „Gaia Methodology“ für die Spezifikation des Multi-Agent Systems verwendet. Dabei fasst Zambonelli et al. zumindest vier Eigenschaften jedes Agenten zusammen: „Autonomy“, „Situatedness“,

[222] vgl. (Voluntary Interindustry Commerce Standards (VICS), 2004) S.17

„Proactivity“, „Agents interact“. [223] Jeder Agent folgt seinen eigenen inneren Zielen, er handelt autonom von anderen Agenten. Weiters ist jeder Agent von seiner Umgebung abhängig. Sein Handeln wird durch diese Umgebung bestimmt. Da Agenten in einer nichtvorhersagbaren Umgebung handeln, müssen sie aktiv an der Erreichung ihres Ziels arbeiten. Damit müssen Agenten für das Lösen ihrer Aufgaben sich untereinander koordinieren, kooperieren und verhandeln. Es wird damit ein Multi-Agent System (MAS) aufgebaut, das im Fall von DBE auf einem offenen System basiert, bei dem mehrere Agenten oder DBE Spezies miteinander agieren und gleichzeitig auch in Konkurrenz zueinander stehen. Jeder Agent nimmt in einem MAS eine bestimmte Rolle ein, die er über Interaktion und in einer definierten Umgebung (Environment) zu erreichen versucht. Über Evolution kann sich das System weiterentwickeln (die Dienste neu zusammensetzen); da es mit einer Vielzahl an Agenten aus anderen Firmen interagiert, kann das genaue Ergebnis nicht vorhergesagt werden. Der technische Design-Prozess für das VMI basiert auf der Gaia Methode. [224]

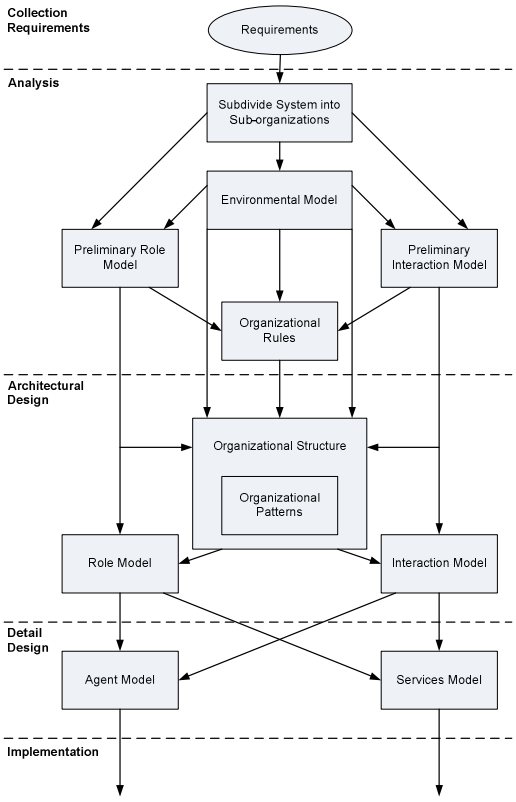

Abbildung 35: Gaia methodology

Quelle: (Zambonelli, Jennings, & Wooldridge, 2003)

[223] vgl. (Zambonelli, Jennings, & Wooldridge, 2003) S.320

[224] vgl. (Zambonelli, Jennings, & Wooldridge, 2003) S.324ff

5.3 Architektur des VMI im DBE

5.3.1 Anforderungen

Die erste Aufgabe im Rahmen der Gaia-Methode besteht darin, die Anforderungen an das System zu definieren. Dieser Schritt wurde bereits sehr ausführlich im Kapitel 2 durchgeführt. Zusammenfassend wurden folgende Anforderungen an ein modernes System gestellt:

Anforderung	Wirtschaftlich	Technische	Kapitel
Zusammenarbeit (Collaborative)	X	X	2.1.1, 2.1.5, 2.2.2, 2.2.3
Serviceorientierung	X	X	2.1.4, 2.2.1
Echtzeitgetreu	X		2.1.1, 2.1.3, 2.1.5
Anpassungsfähigkeit	X	X	2.1.1, 2.1.3, 2.1.5, 2.2.1, 2.2.2
Kundenzentriert	X	X	2.1.2, 2.2.2
Zuverlässig	X	X	2.1.2, 2.1.3, 2.2.2
Kostengünstig	X	X	2.1.2
Kompatibilität	X	X	2.1.2, 2.1.3, 2.1.4, 2.1.5, 2.2.2
Einfach	X	X	2.1.1, 2.1.2, 2.2.1
Wiederverwendbar		X	2.2.1
Zukunftsfähig		X	2.2.2
Selbstorganisiert	X	X	2.1.1, 2.2.3

Tabelle 5: Anforderungen für VMI

5.3.2 Analyse

Auf der Basis der Anforderungen gilt es das System zu analysieren. Im ersten Schritt müssen die Organisationen des Systems bestimmt werden. Eine Organisation ist definiert als:

> *„(i) exhibit a behavior specifically oriented towards the achievement of a given sub goal, (ii) interact loosely with other portions of the system, or (iii) require competences that are not needed in other parts of the system.*[225]

Im zweiten Schritt wird das Umgebungsmodell definiert, das Ressourcen angibt, auf die ein Agent zugreifen kann. Dabei sollen Dienste, die nur Informationen liefern, ebenfalls als Ressourcen dargestellt werden und Dienste, die komplexe Aufgaben lösen können als quasi Agenten dargestellt werden.[226]

Im dritten Schritt werden die Basisrollen definiert. Diese stellen nur die Hauptrollen (inklusive Verantwortungen und Rechte der Agenten) dar, das gesamte Konzept wird erst bei der Architektur zusammengestellt. In dieser Ebene werden noch Hilfsagenten hinzugefügt. Dasselbe gilt für das Interaktionsmodell, das die Protokolle für die Interaktionen definiert, sowie die Organisationsregeln, welche die Verantwortungen für das gesamte System analog zu den Regeln der Agenten definieren.[227]

Die Analyse erfolgt entlang der definierten Prozesse des CPFR und versucht dabei so viele Aufgaben wie möglich auf die digitale Welt des DBE umzulegen. Der Teilprozess „Joint business Plan“ kann im Rahmen dieser Architektur nicht abgebildet werden und findet sich daher direkt im Business Process Plan des CPFR.

Als Ausgangspunkt für die Entwicklung des Agentensystems wurde die Architektur von Negri et al. genommen.

> *„OrderManager: receives orders and uncertain forecasts from the buyers …*
> *DemandPlanner: determines the forecasted output of the factory on the long-term …*
> *DistributionPlanner: assigns finished goods in stock to buyers …*
> *TransportationPlanner: organizes the delivery of products*
> *SupplyPlaner: ensures the necessary raw materials“*[228]

---

[225] vgl. (Zambonelli, Jennings, & Wooldridge, 2003) S.338

[226] vgl. (Zambonelli, Jennings, & Wooldridge, 2003) S.339f

[227] vgl. (Zambonelli, Jennings, & Wooldridge, 2003) S.341ff

[228] (Egri & Váneza, 2005) S.351

Die Architektur stellt alle Agenten für die gesamte Supply-Chain dar (vom Kauf der Rohmaterialien bis hin zum Verkauf).[229] Für die Entwicklung des VMI in dieser Arbeit liegt die Beschränkung rein auf den Prozessen für das VMI.

5.3.2.1 Strategy & Planning

Die erste Phase des CPFR Modells besteht aus dem Strategy & Planning Prozess. Dieser Prozess wird wiederum in zwei Teilprozesse geteilt: Collaboration Arrangement und Joint Business Plan. Für die technische Umsetzung über ein Middlewaresystem werden wir uns in diesem Prozess nur auf das Collaboration Arrangement konzentrieren. Der zweite Teilprozess muss wie bereits erwähnt außerhalb der digitalen Welt vereinbart werden.

In Abbildung 36 werden die Teilaufgaben des Collaboration Arrangements dargestellt.

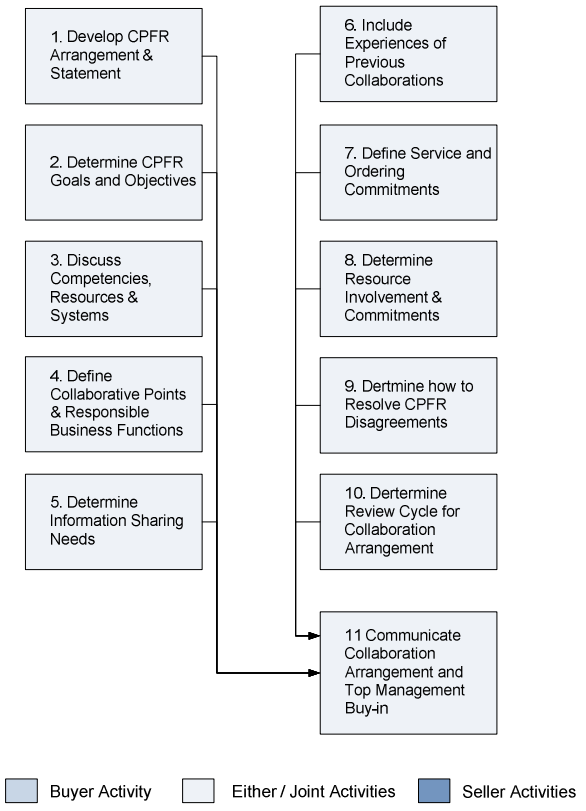

Abbildung 36: Collaboration Arrangement

Quelle: (Bozarth, 2007)

[229] (Egri & Váneza, 2005)

Collaboration Arrangement definiert die generelle Zusammenarbeit. Darunter fällt die Definition der Produkte und Standorte aber auch der Parameter für das Service Level Agreement.

Im ersten Schritt muss ein Partner für die Einführung von VMI gefunden werden. Dies passiert meistens außerhalb der digitalen Welt, jedoch kann diese Aufgabe auch über das DBE erfüllt werden.

Als Voraussetzung muss eine gemeinsame Wissensbasis zwischen den einzelnen Partnern aufgebaut werden. Dieses Wissen dient dazu die Prozesse und Aufgaben der einzelnen Partner zu definieren. Gerade bei KMUs kann dieses Wissen und die Ausprägung in den einzelnen Unternehmen sehr unterschiedlich verlaufen. Um ein gemeinsames Wissen zu schaffen, müssen Ontologien entwickelt werden, die die Supply-chain Prozesse und Teilnehmer beschreiben.

Es sollte daher eine SCM Domain Ontologie entwickelt werden, die die generelle Struktur des SCM vorgibt. In weiterer Folge wird diese Ontologie auf die einzelnen individuellen Unternehmensontologien angepasst („mappen“).[230] Das Problem bei der Erstellung einer SCM Domain Ontologie ist die Akzeptanz der Partner in der Community. Daher sollte diese Ontologie von bestehenden Standards abgeleitet werden.[231]

Über die definierte Ontologie können die Partner ihre Unternehmen mit allen Strukturen beschreiben. Im DBE werden die Unternehmen mittels der Business Modeling Language dargestellt. Damit Ontologien in dieses Modell übernommen werden können, bietet das DBE-Studio eine Importfunktion von Ontologien. Auf einer grafischen Oberfläche können die Strukturen der Unternehmen einfacher und schneller dargestellt werden. Diese Definitionen werden an das DBE-Service für die Umsetzung von VMI als Metadaten angehängt und erlauben es DBE Services zu suchen, die den genauen Anforderungen entsprechen.

Die Rolle, die diese Aufgabe übernimmt, ist der CustomerManger beim Verkäufer. Auf der Seite des Käufers übernimmt der SupplyManager die Suche nach einem adäquaten Partner.

Die Informationen über das Unternehmen sind aber nicht ausreichend. Viel interessanter sind die Produkte, die angeboten oder nachgefragt werden. Dafür kann als eigenes DBE Service ein Katalog angeboten werden, der die benötigten Informationen beinhaltet.

---

[230] vgl. (Blomqvist, Levashova, Öhgren, Sandkuhl, Smirnov, & Tarassov, 2005) S.251ff

[231] vgl. (Fayez, Rabelo, & Mollaghasemi, 2005) S.2364

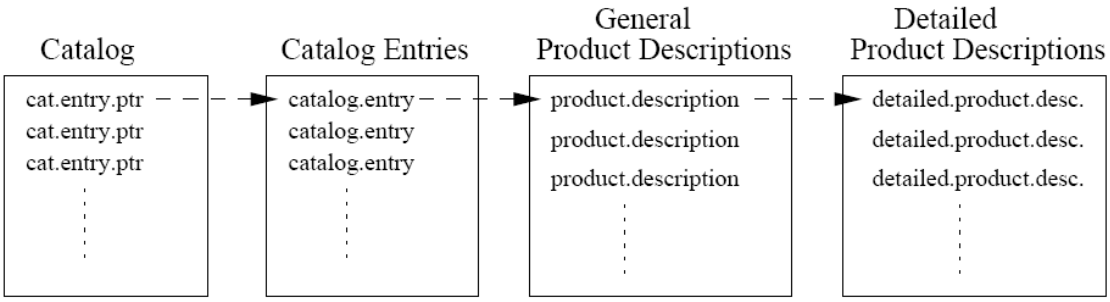

Abbildung 37: The electronic catalog structure in CBL

Quelle: (Cingil & Dogac, 2001) S.62

Das Common Business Language CBL beinhaltet mehrere Konzepte für die Umsetzung von Businessmodelle in der Form von XML Schema. Cingil et al. hebt dieses Konzept als eine sehr gute Wahl für die Umsetzung eines eigenen Agenten hervor, der die folgenden Aufgaben des Kataloges übernimmt: [232]

- *"Facilitating product comparisons and customized catalogs*
- *Locating complementary products*
- *Bi-directional catalog integration*
- *Automation of processes on the supply chain"*[233]

Die CBL kann dabei als Ontologie verwendet werden. Der Agent wird im DBE als eigener DBE Service publiziert und liefert jeweils die aktuellen Informationen über das Produkt eines Unternehmens zurück. Dabei erfolgt die Unterscheidung der Produkte auf der Basis von SKUs (Stock Keeping Units). Die Rolle ist der KatalogManager und wird in der Architektur als KatalogAgent dargestellt.

Um ein Produkt identifizieren zu können, benötigt es einen eindeutigen Code. Auf vielen Produkten befindet sich bereits ein Barcode, der zwar eine Produktgattung identifiziert, aber das Produkt nicht als ein Produkt eines bestimmten Unternehmens. Auch bei zusammengesetzten Produkten kann nur schwer eine eindeutige Identifikation durchgeführt werden. Främling et al. stellt ein System vor, bei dem ein Produkt eindeutig durch einen ID und eine URI beschrieben wird. Zusammengesetzte Produkte sind dann eine Komposition aus der Summe dieser eindeutigen Codes.[234]

---

[232] vgl. (Cingil & Dogac, 2001) S.61f

[233] (Cingil & Dogac, 2001) S.59f

[234] vgl. (Främling, Kärkkäinen, Ala-Risku, & Holmström) S.1ff

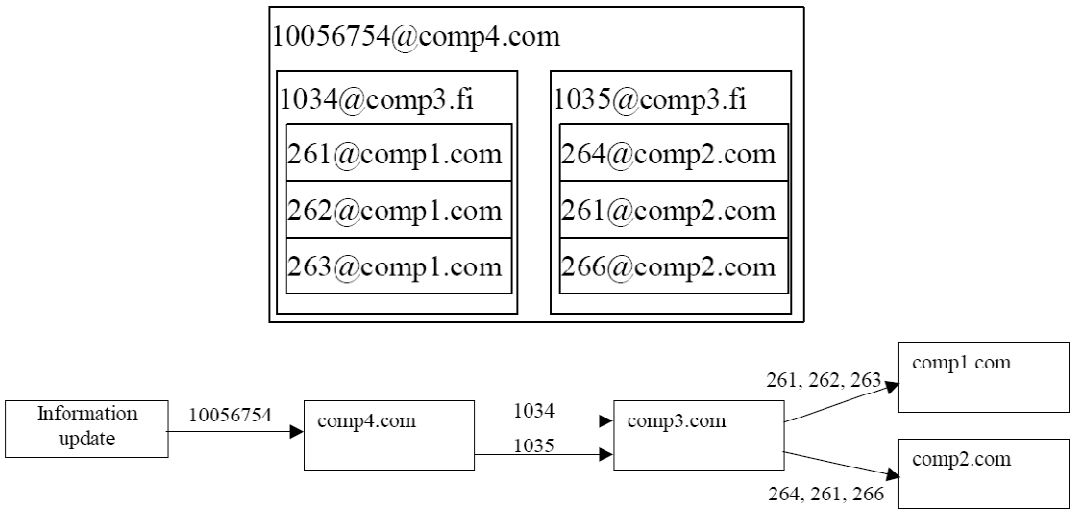

Abbildung 38: Produktidentifizierung im Internet

Quelle: (Främling, Kärkkäinen, Ala-Risku, & Holmström) S.5

Im DBE könnte diese Art der Identifizierung über einen ID und der Nummer des DBE Services erfolgen, der das Produkt identifiziert. Bei der Order Fullfilment wird ein ähnliches System für jede einzelne Ware dargestellt.

Die Angaben zum Service Level Agreement werden im DBE Services des Lieferanten zusätzlich als Metadaten über die BML abgespeichert und können daher vom potenziellen Kunden direkt eingesehen werden.

Ein Großteil der Informationen, die für eine Umsetzung des DBE benötigt werden, könnte direkt über das DBE abgefragt werden und müssen somit nicht angefordert werden. Kunden können bereits im Vorfeld überprüfen, inwieweit eine Zusammenarbeit möglich ist. Die konkrete Vertragsunterzeichnung bzw. die rechtlichen Konsequenzen müssten trotzdem noch in der realen Welt vereinbart werden.

Im Rahmen des ersten Schrittes wurden die Rollen CustomerManager, KatalogManager und SupplyManager identifiziert. Die definierten Protokolle und Ressourcen werden im Anhang dargestellt und genau spezifiziert.

5.3.2.2 Demand & Supply Management

Der Prozess „Demand & Supply Management“ besteht aus zwei Teilprozessen. Der erste Prozess beschäftigt sich mit der Vorhersage der Bestellmengen und der zweite Prozesse mit der Planung der Bestellungen.

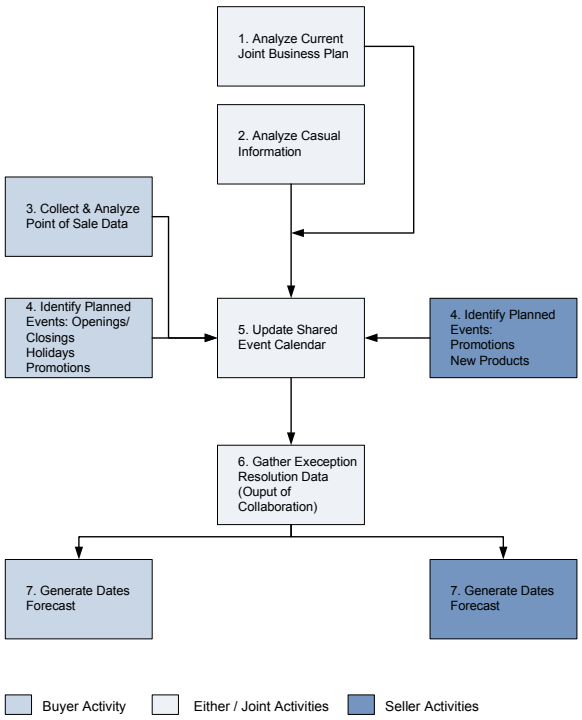

Abbildung 39: Sales Forecasting

Quelle: (Bozarth, 2007)

Die Vorhersage der Verkäufe dient der Planung der Bestellungen. Im Fall des DBE soll diese Möglichkeit einer breiten Masse zur Verfügung gestellt werden. Um dies effizient durchführen zu können, darf die gesamte Aufgabe nicht auf den Verkäufer übertragen werden sondern auch der Käufer sollte seinen Beitrag leisten. Andernfalls würde der Berechnungsaufwand für einen einzigen Teilnehmer in der Lieferkette zu groß werden und es würde kein Optimum entstehen können, weil die Berechnung vor Ende auf Grund eines Timers abbricht.

Wie in Abbildung 39 dargestellt, teilt sich die Vorhersage der Verkäufe auf beide Partner in dem Netzwerk auf. Holmström et al. schlägt vor, dass der Käufer die Vorhersage auf sein Categorymanagement aufbauen soll. Die Hauptaufgabe besteht darin nicht eine

Produktgruppe zu bestimmen, sondern auf SKU Ebene die Vorhersage der Verkäufe zu betreiben.[235]

Dies sollte so durchgeführt werden, dass in jeder Kategorie der Rang der SKU und die Rolle bestimmt werden. Über die Verkäufe der gesamten Kategorie und dem Anteil einer SKU an den gesamten Verkäufen kann eine Vorhersage getroffen werden.[236] Diese Daten werden dann an den Verkäufer übermittelt, der auf Basis aller seiner Einkäufer eine Vorhersage der Verkäufe treffen kann. Dabei schlägt Holmström et al. folgende Vorgehensweise vor:

> *"… promotion and product introduction plans in three basic steps:*
>
> *(1) rank the products in the category based on the retailer's assortment decision and planned activities of the supplier;*
>
> *(2) estimate the total category sales; and*
>
> *(3) estimate the scaling function for relative shares for the items in the category. "*[237]

Damit wird ein hohes Maß an Datenaustausch zwischen den einzelnen Partnern verlangt. Zusätzlich müssen Informationen über Änderungen auf Grund von bevorstehenden Werbeaktionen kommuniziert werden. Diese Daten können wiederum von der Rolle des SupplyManagers erstellt werden und an den CustomerManager übermittel werden.

Auf Grund geschickter Vorhersagen der Verkaufszahlen kann die benötigte Produktmenge berechnet werden.

Die Berechnung der Verkaufszahl einer Vielzahl an Kunden kann unter Umständen sehr komplex ausfallen. Um dieser Eigenschaft zu entsprechen, gibt Nachiappan et al. einen anderen Ansatz vor, bei dem der „Demand" über einen genetischen Algorithmus bestimmt wird. Über diesen Algorithmus wird die optimale Bestellmenge für jeden einzelnen Kunden über die Artikelart, die Verkaufsstärke des Kunden und der Ausprägung des Produktes (Lagerfähigkeit) bestimmt. Auf der Basis dieser Werte wird eine Fitness-Funktion erstellt die die verschiedensten Variationen der Bestellmengen für die Kunden überprüft und so eine optimale Lösung liefert.[238]

---

[235] vgl. (Holmström, Främling, Kaipia, & Saranen, 2002) S.136ff

[236] vgl. (Holmström, Främling, Kaipia, & Saranen, 2002) S.139f

[237] (Holmström, Främling, Kaipia, & Saranen, 2002) S.140

[238] vgl. (Nachiappan & Jawahar, 2007)

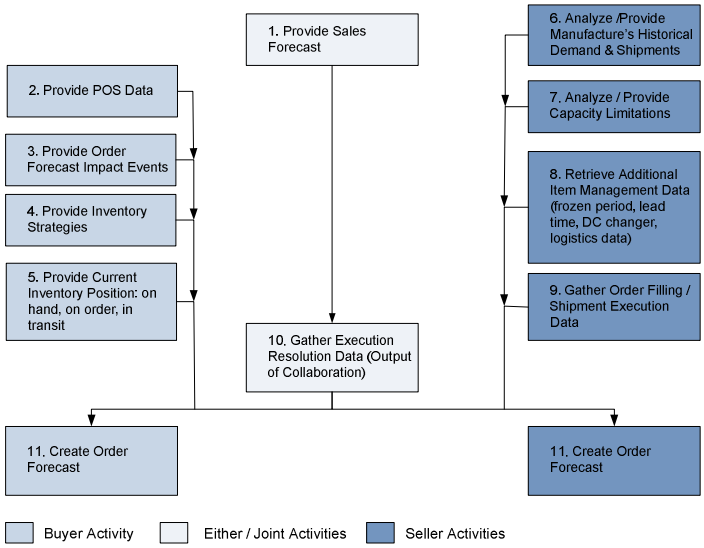

Abbildung 40: Order Planning

Quelle: (Bozarth, 2007)

Diese Daten werden verwendet um zusätzlich die Sicherheitsbestände auf Grund des aktuellen Lagerstandes vom StorageManager und den Lieferzeiten aus dem Transport-Manager zu bestimmen. Diese addierten Werte ergeben die Bestellmenge für jeden einzelnen Kunden.

Der StorageManager übernimmt noch eine zusätzliche Aufgabe für die Weitergabe von Eskalationen an den DemandManager. Sollte der Lagerstand nicht ausreichen, so schickt der StorageManager an den DemandManager eine entsprechende Mitteilung: Eskalation

5.3.2.3 Execution

Nachdem die benötigten Mengen für einen Kunden definiert worden sind, kann die Bestellung aufgegeben werden. Dabei haben sich verschiedene Standards entwickelt, die von den aktuellen ERP-Systemen verstanden werden.

In der Vergangenheit hat sich bei vielen Unternehmen der Standard EDI durchgesetzt, der auch von vielen Softwareprodukten unterstützt wird. Die konkrete Umsetzung von EDI ist jedoch meistens sehr teuer. Was gesucht wird, ist eine Ontologie die auf eine Bestellbestätigung umgesetzt und mittels Mapping für das konkrete Unternehmen angepasst wird. Diese Voraussetzung ist jedoch nur sehr schwer erfüllbar, weil sich auch Standards wie EDI nicht überall durchgesetzt haben.

Als Basis für eine solche Ontologie können bestehende Standards wie xCBL, OAGIS, UBL/ebXML oder RosettaNet herangezogen werden. Zhao hat die verschiedenen Konzepte auf ihre mögliche Umsetzung auf eine Ontologie überprüft. Dabei wurde als po-

tentieller Kandidat das UBL Modell von der OASIS[239] herausgefunden. Dieser Standard basiert auf xCBL und kann mit ebXML verbunden werden.[240]

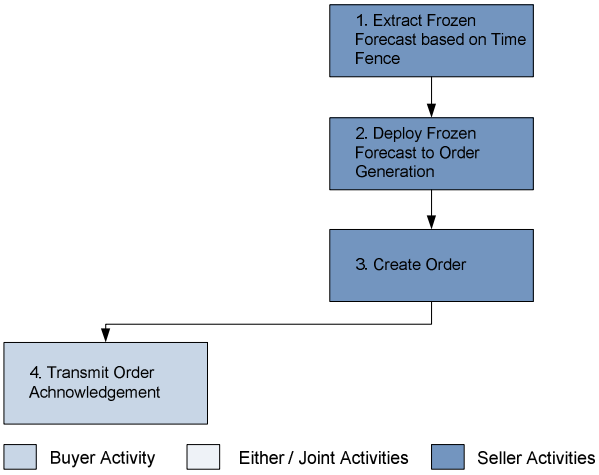

Abbildung 41: Order Generation

Quelle: (Bozarth, 2007)

Die in der definierten Ontologie kodierten Daten werden an den Kunden als Bestellbestätigung geschickt. Der DemandManager im DBE schickt die Daten an den OrderManager, der die Bestellung an das ERP-System übergibt.

Nachdem die konkrete Bestellung für den Kunden fertig ist, müssen die Produkte dem Plan nach produziert werden. Dabei werden die Informationen an den internen ProduktionManager weitergeben und in einen Produktionsplan umgewandelt.

Sobald die Produkte fertig für die Lieferung sind, muss die Auslieferung der Produkte an den Kunden geplant und an ein Logistikunternehmen weitergegeben werden. Dieser Prozess kann unter Umständen sehr komplex verlaufen: gerade wenn sehr viele Produkte die Produktionsstätte verlassen, wird die Planung für die Logistik aufwendig.

Wie im ersten Kapitel definiert, sollten die Systeme selbst-organisiert und einfach sein. Im Fall der Lieferung kann dies über sogenannte „intelligente Produkte“ realisiert werden. Dabei erfolgt die Steuerung der Logistik nicht über ein zentrales System, sondern die Produkte selbst veranlassen ihre Lieferung. Jedes Produkt bekommt seinen eigenen ID und einen privaten Schlüssel mit auf den Weg. Mittels der ID wird gleichzeitig der Dienst am Heimat-Habitat des Produktes definiert. Dieser kann über das peer-to-peer

[239] vgl. (OASIS)

[240] vgl. (Yuxiao, 2003)

Netzwerk des DBE jederzeit wiedergefunden werden. Der DBE Dienst im Habitat erlaubt den Zugriff auf die Informationen des Produktes. Dort werden nach Fertigstellung des Produktes die Angaben gespeichert, wo das Produkt hingehen muss, welche Eigenschaften es hat und wie es behandelt werden soll. In einem zweiten Schritt initialisiert das Produkt selbst seine Lieferung.[241]

Durch einen Scan beim Ausgang aus der Fertigungshalle wird der DBE Service TransportManager aufgerufen, der ein Logistikunternehmen für die Lieferung sucht. Durch den privaten Schlüssel kann sich das Produkt eindeutig identifizieren. Gleichzeitig wird das Produkt bei jedem Checkpoint des gesamten Weges gescannt und gibt somit seine Position bekannt (Track&Trace). Der Empfänger kann auf diese Weise den Verlauf der Produkte ständig überwachen. Technologien die hierbei helfen sind RFID oder Barcodes.[242]

Die interessante Aufgabe ist jedoch die Suche nach einem Logistikunternehmen für die Auslieferung der Produkte. Hierbei kann das DBE alle seine Vorzüge ausspielen. Die Aufgabe besteht darin den optimalsten Weg vom Herstellerlager bis hin zum Kunden zu finden.

Über das EvE des DBE wird eine Anfrage als SBVR-Code gestellt. Auf Grund der Metainformationen, die im DBE gespeichert sind, versucht das EvE die optimalste Lösung für das Problem zu finden und gibt den DBE Dienst bzw. die Abfolge der Dienste zurück, die dieses Problem lösen.

Die Anfrage an das DBE kann über eine Erweiterung der Swarm Intelligenz abgeleitet werden. Kasinger und Bauer definieren das Modell der „Self-Managing Systems Based on Pollination“.[243] Als Vorbild wird dabei die Bestäubung von Blüten genommen: Blüten haben die Eigenschaft Bienen anzuziehen, die ihre Pollen vom Staubbeutel der einen Pflanze auf die Narbe der anderen Pflanze tragen. Dabei handelt es sich um zwei unterschiedliche miteinander agierende Komponenten. Um die Bienen auf sich aufmerksam zu machen, bieten die Blüten Anreize, wenn die Insekten die Pollen wegtragen oder zu einer Pflanze derselben Gattung bringen. Können Blüten keine Bienen mehr anziehen, sterben sie aus.[244]

Dieses System kann auch auf die Anforderungen der Produkte umgelegt werden: sobald die Produkte die Fertigungshalle verlassen, wird der DBE Service TransportManager aktiv und bietet den Logistik-Unternehmen Anreize diese abzuholen, d.h. es werden

---

[241] vgl. (Kärkkäinen, Holmström, Främling, & Artto, 2003) S.6ff

[242] vgl. (Teuteberg & Ickerott, 2007)

[243] vgl. (Kasinger & Bauer, 2006) S.169ff

[244] vgl. (Kasinger & Bauer, 2006) S.171

über das DBE Unternehmen gesucht; als Gegenleistung wird ihnen Geld geboten. Den Unternehmen werden der Standort, das Ziel und die Eigenschaften der Produkte wie ein Lockstoff mitgeteilt und sie werden aufgefordert Angaben über das Lieferdatum und die Kosten mitzuteilen. Auf Grund dieser Angaben kann der optimale Weg für ein Produkt bestimmt werden. Wurde ein DBE Service gefunden, das den Anforderungen entspricht, wird der Weg für das Service des Produktes abgespeichert und das Produkt kann sich auf die Reise machen.

Eine Erweiterung des Systems könnte so ausschauen: steuert ein Logistikunternehmen nur das Ende oder den Anfang der Lieferstrecke an, so kann es automatisch eine eigene Suche nach adäquaten Logistikunternehmen durchführen, die das Produkt weiterführen. Die Kosten werden dem Logistikunternehmen mitgeteilt und zu den eigenen Kosten addiert. Am Schluss werden diese Kosten dem Produkt mitgeteilt und fließen in die Optimierung ein.

Gleichzeitig kann das Logistikunternehmen die eigenen Netzwerke insofern optimieren, indem die Kosten niedriger angesetzt werden, wenn die LKW des Unternehmens in der Nähe eines Kunden sind. Somit wird die Chance auf den Auftrag erhöht, das Unternehmen muss dafür aber einen Teil des Gewinnes für diese Optimierung mit dem Kunden teilen.

Auf diese Art wird für die Steuerung der Lieferungen kein zentrales System mehr benötigt. Vielmehr finden die Produkte ihren eigenen Weg. Springt ein Logistikunternehmen ab, so kann es von alleine eine neue Route suchen und bei Nichteinhaltung eines Liefertermins eine „Eskalations-Nachricht“ an den Leiter des Prozesses schicken.

5.3.2.4 Analysis

Im letzten Schritt des CPFR Prozesses muss der gesamte Prozess überwacht werden, damit Fehler sofort behoben werden können. Die Überwachung wird dabei auf mehrere Rollen verteilt: auf Lagerebene übernimmt der StorageManager die Überwachung der Lagerstände und meldet Fehler dem DemandManager über eine EscalationMessage.

Darüber hinaus existieren sowohl beim Verkäufer als auch beim Käufer eigene Rollen für die Überwachung des gesamten Prozesses auf Basis der in der ersten Phase definierten Paramter: SupplyControlManager und OrderControlManager.

Diese beiden Rollen sammeln die Daten von den anderen Rollen und erarbeiten einen Performanceplan. Dieser Plan wird nicht nur an die realen Personen übermittelt um die Performance des Systems zu messen, sondern auch an den DemandManager, der damit die Vorhersagen optimieren kann. Eskalationen werden ebenfalls entweder direkt an den DemandManager weitergeleitet oder an die reale Person, die für die Steuerung des Prozesses verantwortlich ist.

5.4 Architektur

In dieser Phase müssen die Erkenntnisse aus der ersten Phase der Analyse strukturiert und in ein Gesamtkonzept zusammengefasst werden. Unter Einflussnahme der organisatorischen Regeln und der erkannten Rollen kann nun in dieser Phase die Hierarchie aufgebaut werden, unter der das gesamte System agieren soll. Hinzu kommen noch die Einwirkungen aus der realen Welt, da die Organisation zum Teil das konkrete Unternehmen widerspiegeln muss.[245] Die Struktur des gesamten Systems wurde im Anhang dargestellt.

5.5 Detaildesign

Im letzten Schritt werden die identifizierten Rollen auf konkrete Agenten und Services umgelegt. Im Fall des DBE gibt es dabei keine Unterscheidung zwischen Agenten und Services. Sie sind im DBE gleich aufgebaut. Von ihren Fähigkeiten her entsprechen sie aber den Anforderungen aus der Gaia Methode.

5.6 Zusammenfassung

Wie das Design für VMI im DBE zeigt, spielen sehr viele neue Entwicklungen in der Umsetzung eine wichtige Rolle. Eine konkrete Implementierung dieses Systems verlangt auf jeden Fall eine genaue Simulation der einzelnen Agenten und deren Zusammenspiel. Viele Einflüsse durch unterschiedliche Programmierungen in unterschiedlichen Unternehmen machen das System angreifbar und verwundbar. Können jedoch diese ersten Anfangsschwierigkeiten überwunden werden, ermöglicht dieses System eine selbstgesteuerte Optimierung und kann sich an Einwirkungen aus der realen Welt anpassen. Vielleicht können sogar neue Wege der Interaktion gebildet werden. Als ein Beispiel kann hier die Präsentation von Hod Lipson erwähnt werden. Er hat einen Roboter vorgestellt, der seine eigene Form selbst nicht kannte und auch nicht wusste, wie er sich vorwärts bewegen kann. Über den selben Ansatz, den auch ein Wissenschaftler verfolgt, konnte der Roboter eine eigene Entscheidung darüber treffen, wie er sich fortbewegen kann und das in einer ganz anderen Art und Weise wie wir uns das vielleicht denken.[246] Vielleicht passiert so ein ähnlicher Effekt auch im DBE.

---

[245] vgl. (Zambonelli, Jennings, & Wooldridge, 2003) S.351

[246] vgl. (Lipson, 2007)

Teil IV

Ausblick

Kapitel 6

Zusammenfassung

Im Rahmen des ersten Kapitels wurde zunächst Supply-Chain Management im Allgemeinen beschrieben und auf verschiedene aktuelle Entwicklungen Bezug genommen. Im Bereich des SCM haben dabei Modelle wie das CPFR, ECR oder SCOR eine hohe Bedeutung beim Aufbau von Netzwerken über mehrere Unternehmen. Aus technischer Sichtweise spielen das Prozessmanagement und die schnellen Entwicklungen im Rahmen des E-Commerce eine wichtige Rolle in Bezug auf ein zukünftiges Middlewaresystem für SCM Anwendungen. Sehr interessante Aspekte liefern auch Themen wie „Swarm Intelligence“. Dabei sollen einfache, selbstorganisierte Systeme über einfache Regeln komplexe Aufgaben lösen.

All dies nimmt Einfluss auf zukünftige SCM Systeme und muss daher in die Überlegungen für eine richtige Wahl eines Middlewaresystems miteinbezogen werden. Dieses System soll das „Nervensystem“ späterer Systeme wiederspiegeln. Einer der wichtigsten Faktoren ist, dass es für KMUs in einer Art und Weise umgelegt werden kann, sodass diese es auch einsetzen können und es nicht von Großunternehmen aufoktroyiert bekommen.

Das erste System, das diesen Anforderungen zu entsprechen versucht, ist SOA. Ganz klar wird in dieser Arbeit SOA als ein Konzept gesehen, das Paradigmen für eine konkrete Umsetzung vorgibt. SOA ist daher kein Framework oder bestimmtes Softwareprodukt. Die Umsetzung von SOA kann also sehr viele Ausprägungen haben.

Die gängigste Variante für die Umsetzung von SOA sind sogenannte Webservices. Für den Vergleich mit den Anforderungen an das SCM wurde die Verbindung dieser Webservices über einen modernen Enterprise Service Bus angenommen. Innerhalb dieses ESB befindet sich eine Service Registry, die zusätzlich zu den Services auch deren Metadaten abspeichern kann. Unter Bezugnahme dieser Variante von SOA können moderne SCM Systeme auf jeden Fall abgebildet werden. Die Engpässe bei der Umsetzung

sind der Enterprise Service Bus und die Service Registry. Diese sind vom Grundgedanken her so ausgelegt, dass ein großer Konzern eine eigene ESB aufbaut und andere Subunternehmen über einen ESB-Gateway an diesem System partizipieren können.

Die Entwicklung von SOA basiert auf offenen Standards, die im Rahmen von Konsortien entwickelt werden. KMUs können diese Standards aufgreifen und in eigene Softwareprodukte umwandeln bzw. ihre eigenen Programme an bestehende Systeme anpassen. Für die Umsetzung werden darüber hinaus ebenfalls opensource Produkte angeboten. Damit entspricht SOA allen Voraussetzungen für ein modernes SCM System, so wie in Kapitel 2 dieser Arbeit definiert. Was SOA noch nicht kann, ist der Aufbau eines einzigen Netzwerkes für alle Dienste an dem alle Unternehmen teilnehmen.

Das Digitale Business Ecosystem DBE ist das zweite System, das an Hand der Anforderungen des SCM überprüft worden ist. Das DBE hat sich 2002 aus einem EU Projekt entwickelt, das sich derzeit in der zweiten Förderungsphase befindet. Das DBE hat das Ziel KMUs auf regionaler Ebene zu unterstützen, damit diese nachhaltiges Wachstum erreichen können. Für die Erreichung dieses Ziels ist es erforderlich, effizienter am Markt aufzutreten und in Konkurrenz zu anderen, vor allem Großunternehmen, eine bessere Performance zu zeigen. Virtuelle Organisationen sind die neuen Formen von Unternehmensstrukturen, die diesen Entwicklungen entsprechen sollen. Umgesetzt werden diese über IKT-Systeme. Das DBE hat in seinem ersten Diskussionspapier bereits hervorgehoben, dass dies nicht über einzelne Projekte, sondern über die Schaffung einer nachhaltigen Infrastruktur erreicht werden kann. Dadurch wurde der Weg für das DBE geebnet.

Das DBE versucht über ein Peer-to-Peer Netzwerk alle Firmen über eine Plattform miteinander zu verbinden. Aufgaben aus der realen Welt werden über DBE Dienste in dieser Struktur abgebildet und können von allen Teilnehmern aufgegriffen werden. Damit wird aber nicht nur der Dienst im DBE publiziert, sondern auch die gesamten wirtschaftlichen Informationen die diesen Dienst betreffen (Metadaten). Der gesamte Dienst kann ohne technische Angaben über den MDA-Ansatz beschrieben werden. Diese Menge an Diensten würde die Suche im DBE bald unbewältigbar machen, weshalb eine Umgebung geschaffen wurde, in der sich das System selbst optimieren kann. Über einen genetischen Algorithmus werden DBE Services auf Basis von Benutzeranfragen selbstständig zusammengestellt und das Optimum innerhalb eines Fitness-Landscape ausgewählt. Kommen neue Dienste oder neue Versionen von Diensten in das DBE, kann sich das System von selbst optimieren und alte Dienste sterben aus.

Das DBE selbst erfüllt alle Voraussetzungen, die ein modernes SCM von einem Middlewaresystem verlangt. Es können über ein System ad-hoc Verbindungen mit sämtlichen Benutzern des DBE hergestellt werden bzw. Anpassungen sofort umgesetzt werden. Der gesamte Prozess der Entwicklung, als auch die Tools selbst, sind über offene

Standards und über opensource Software umgesetzt worden. Jeder Teilnehmer kann die Informationen des DBE selbständig abfragen und in einem eigenen System umsetzen.

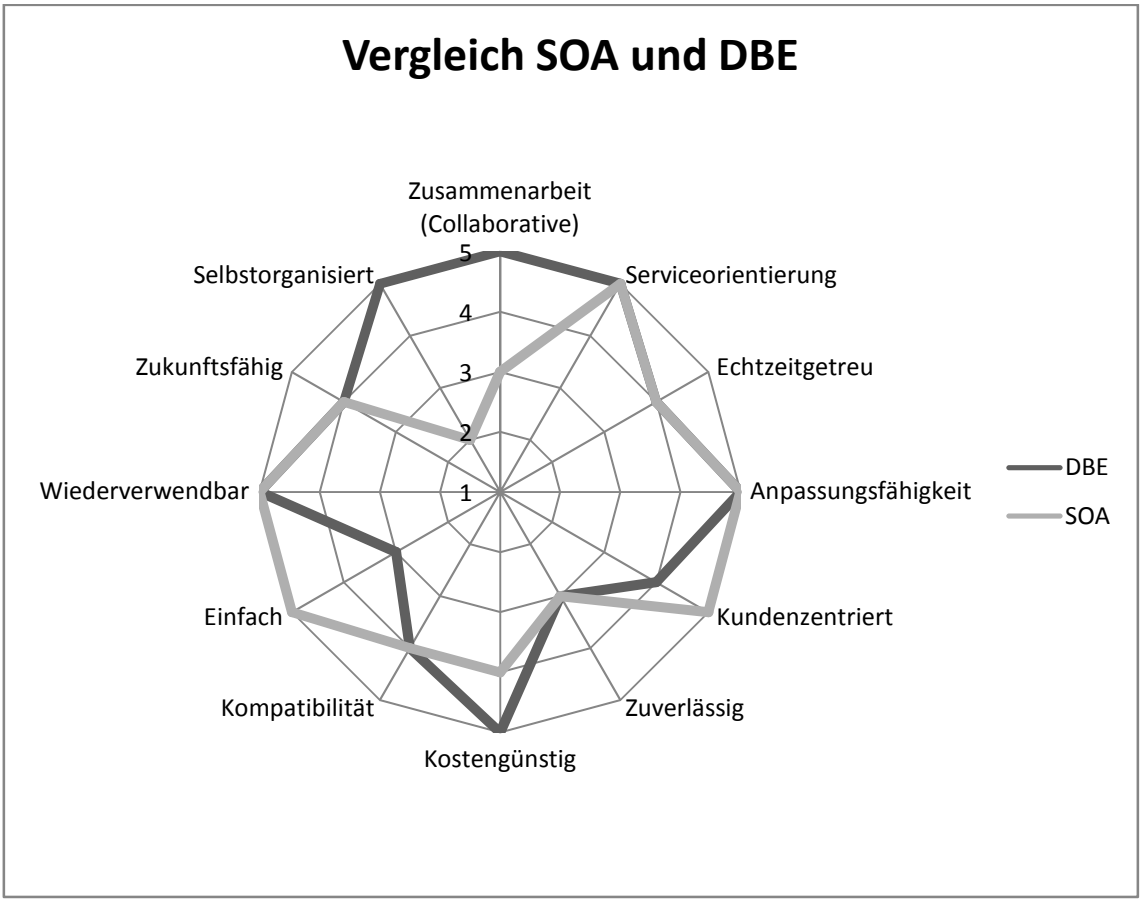

Abbildung 42: SOA und DBE

Das DBE bildet daher genauso wie das SOA alle Aufgaben des SCM ab und kann für eine Umsetzung von SCM-Lösungen eingesetzt werden. In Abbildung 42 werden die beiden Systeme SOA und DBE noch einmal zusammenfassend dargestellt.

Ob das DBE eine Art SOA ist, kann nur sehr schwer beantwortet werden, weil es dabei auf die Sichtweise des Problems ankommt. SOA wurde als technische Schnittstelle für proprietäre Systeme entwickelt und erst jetzt werden zusätzliche Informationen über die Dienste gespeichert. Die ESB sind ebenso in Entwicklung und werden ständig verbessert. Das DBE hingegen bietet eine Lösung für die Umsetzung eines einzigen Netzwerks für alle Dienste an, was SOA in der Form als Webservice über ESB noch nicht anbietet. Als Konzept hingegen bildet das DBE alle Eigenschaften einer SOA Lösung im großen Stil an. Meiner Meinung nach ist daher aus konzeptueller Sichtweise das DBE eine Art von SOA.

Bei der Umsetzung eines realen Projektes (wie VMI im DBE) spielen aber noch viele andere Faktoren eine wichtige Rolle. Wie im SOA Kapitel beschrieben, geht es bei der Umsetzung eines serviceorientierten Systems um die Verbindung von Personen, Informationen und Prozessen. Bei VMI wird dies über eine Vielzahl von Agenten oder digitalen Spezies erreicht. Jeder dieser Agenten übernimmt seine Aufgabe und versucht wie auch im realen Leben die optimalste Lösung zu finden. Dabei agieren diese Systeme in einem digitalen Ökosystem, an dem eine Vielzahl an Unternehmen teilnehmen können und müssen, damit es zur Selbstoptimierung fähig ist. Entdeckungen aus den Naturwissenschaften, die über die Entwicklung des DBE hinausgehen, können bei der Umsetzung der einzelnen Agenten helfen und diese selbst wiederum optimieren. Die Eingangsfrage, ob das VMI über das DBE umgesetzt werden kann, wird eigentlich schon beim allgemeinen Vergleich mit dem SCM mit ja beantwortet. Mit der Darstellung einer Architektur wird diese Theorie noch verstärkt.

Die Frage, die sich jetzt stellt ist „Welches System nun gewählt werden soll?“. Nur die Zukunft vermag die Lösung auf diese Frage zu liefern. Beide Systeme, getrennt betrachtet, haben das Potential ein allgemein akzeptierter Standard zu werden der von Unternehmen für SCM eingesetzt werden kann. Unter der Prämisse, dass das DBE die Konzepte von SOA umsetzt, wird vermutlich eine Mischung aus beiden Systemen die Zukunft von Firmennetzwerken über das Internet bestimmen.

Teil V

Anhang

Anhang A:

Rollen, Protokolle und Ressourcen

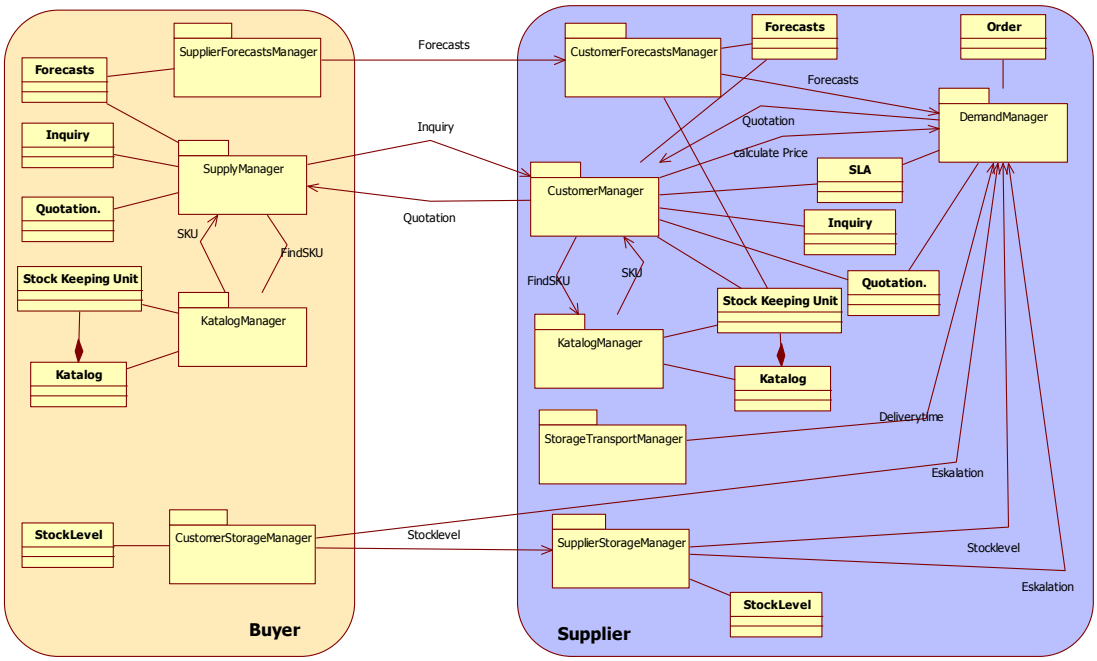

Abbildung 43: Strategie & Planning, Demand & Supply Management

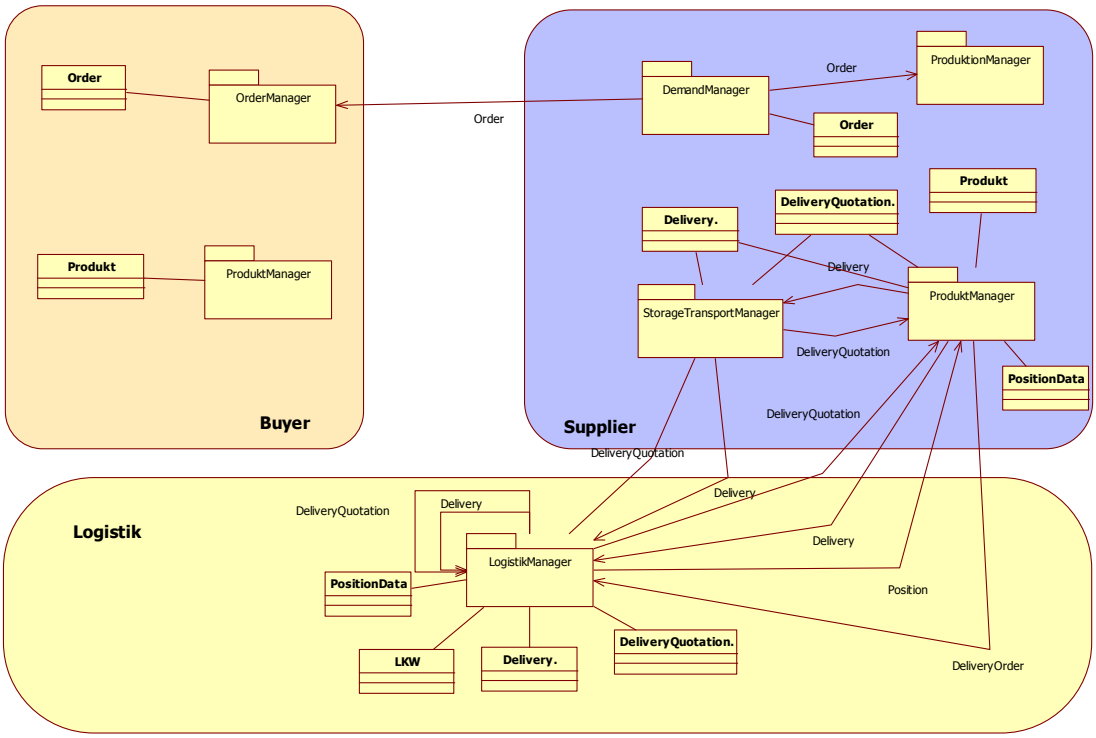

Abbildung 44: Execution

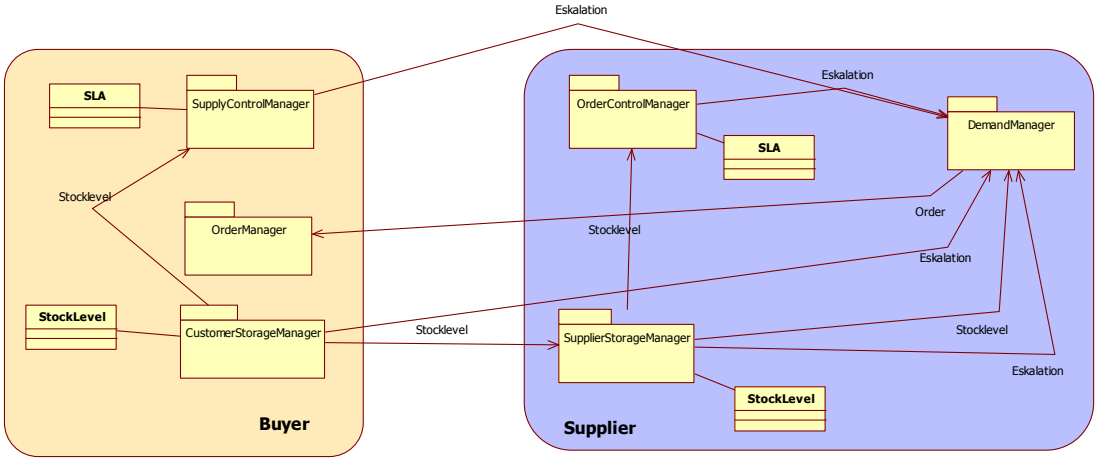

Abbildung 45: Analysis

Rolle: CustomerManager CM	
Beschreibung:	CM übernimmt Angaben über die benötigte Liefermenge auf Grund von Vorhersagen bzw. von konkreten Bestellungen des Kunden und bestimmt den Preis für das Produkt. Dieser Preis wird als Angebot zurückgeliefert.
Protokolle und Aktivitäten:	ReceiveInquiry, FindSKU, CalculatePrice, CreateQuotation, TransmitQuotationToCustomer
Rechte:	reads Forecasts, Inquiry, Quotation, SKU, SLA creates Quotation changes Quotation
Verantwortungen:	
Liveness:	CustomerManager = (ReceiveInquiry. FindSKU . CalculatePrice . CreateQuotation . TransmitQuotationToCustomer)$^{\omega}$
Safety:	• number_of_inquery = number_of_quotation

Rolle: CustomerForecastsManager CFM	
Beschreibung:	Der CFM übernimmt die Vorhersagen des Kunden und bereitet sie für den DemandManager auf.
Protokolle und Aktivitäten:	ReceiveForecasts, FindSKU, SaveForecasts, RequestForForecasts, TransmitForecasts
Rechte:	reads Forecasts, SKU
Verantwortungen:	
Liveness:	CustomerForecastsManager = (ReceiveForecasts . FindSKU . SaveForecasts)$^{\omega}$ || (RequestForForecasts . TransmitForecasts)$^{\omega}$
Safety:	

Rolle: SupplierForecastsManager SFM	
Beschreibung:	Der SFM bestimmt die Produktvorhersagen des Kunden und übermittelt sie an den Lieferanten.
Protokolle und Aktivitäten:	FindSKU, ForecastDemand, TransmitForecasts
Rechte:	reads Forecasts, SKU creates Forecasts
Verantwortungen:	

Liveness:	SupplierForecastsManager = (FindSKU^{for_all_products} . ForecastDemand . TransmitForecasts)$^{\omega}$
Safety:	• duration_of_calculation < time_between_forecasts

Rolle: SupplyManager SM	
Beschreibung:	Der SupplyManager übernimmt die Suche nach geeigneten Partnern für die Durchführung eines VMI.
Protokolle und Aktivitäten:	ReceiveNewInput, FindSKU, SearchForSupplier, CreateInquiry , TransmitInquiry, SelectSupplier, AppendToSystem
Rechte:	reads Forecasts, Quotation, SLA, Inquiry creates Inquiry
Verantwortungen:	
Liveness:	SupplyManager = (ReceiveNewInput . FindSKU . SearchForSupplier . CreateInquiry . TransmitInquiry^{all_supplier} . SelectSupplier . AppendToSystem)$^{\omega}$
Safety:	• Num_ supplier >= 1

Rolle: KatalogManager KM	
Beschreibung:	Der KatalogManager verwaltet die Daten über die einzelnen SKUs des Unternehmens. Es können sowohl Daten abgefragt, als auch geändert werden.
Protokolle und Aktivitäten:	UpdateSKU, UpdateSKUInformation, UpdateKatalog, UpdateKatalogInformation, SubmitSKU, RequestForSKU, CheckKatalog
Rechte:	reads Katalog, SKU, Forecasts, Inquiry changes Katalog, SKU
Verantwortungen:	
Liveness:	KatalogManager=(UpdateSku . UpdateSKUInformation)$^{\omega}$ || (UpdateKatalog . UpdateKatalogInformation)$^{\omega}$ || (RequestForSKU . CheckKatalog . SubmitSKU)$^{\omega}$
Safety:	

Rolle: DemandManager DM	
Beschreibung:	Der DemandManager berechnet die benötigten Warenmengen für den Kunden und bestimmt den Preis.

Protokolle und Aktivitäten:	RequestForecasts, RequestStorageLevel, RequestDeliveryTime, CalculateDemand, CreateOrder, SubmitOrder, ReceiveEskalationMessage, AdjustDemand, RequestForPrice, SubmitPrice, CalculatePrice
Rechte:	reads Forecasts, Stocklevel, Order, Escalation creates Order, Quotation
Verantwortungen:	
Liveness:	DemandManager = ((RequestForecasts . RequestStorageLevel . RequestDeliveryTime)$^{\text{for all customer}}$. CalculateDemand . CreateOrder . SubmitOrder)$^{\text{defined period}}$ || ([ReceiveEskalationMessage].AdjustDemand)$^{\omega}$ || (RequestForPrice . CalculatePrice . SubmitPrice)$^{\omega}$
Safety:	• Liefermengen innerhalb SLA

Rolle: CustomerStorageManager	
Beschreibung:	Im Kundenlager werden die Bestände kontrolliert und an den StorageManager des Lieferanten gemeldet. Gibt es ein Problem, so wird dies direkt dem DemandManager gemeldet.
Protokolle und Aktivitäten:	CheckStorageLevel, CompareToRequiredLevel, SubmitEscalation, SubmitStocklevel
Rechte:	reads Stocklevel creates Escalation
Verantwortungen:	
Liveness:	CustomerStorageManager = (CheckStorageLevel . CompareToRequiredLevel . (SubmitEscalation | SubmitStocklevel)$^{\omega}$
Safety:	• Stocklevel > min_stocklevel • Stocklevel < max_stocklevel

Rolle: SupplierStorageManager	
Beschreibung:	Die Lagerverwaltung des Lieferanten speichert seine eigenen Lagerbestände und die der Kunden. Bei einer Anfrage liefert er die Daten an den DemandManager
Protokolle und Aktivitäten:	CheckStorageLevel, CompareToRequiredLevel, SubmitEscalation, SubmitStocklevel, RequestForStocklevel, ReceiveStocklevel, SaveCustomerStocklevel

Rechte:	reads Stocklevel changes customer_stocklevel creates Escalation
Verantwortungen:	
Liveness:	SupplierStorageManager = (CheckStorageLevel . CompareToRequiredLevel . [SubmitEskalation])$^{\omega}$ || (ReceiveStocklevel . SaveCustomerStocklevel)$^{\omega}$ || (RequestForStocklevel. SubmitStocklevel) $^{\omega}$
Safety:	• Stocklevel > min_stocklevel • Stocklevel < max_stocklevel

Rolle: OrderManager OM	
Beschreibung:	Der OrderManager übernimmt die Bestellungen und speichert sie im ERP System des Kunden.
Protokolle und Aktivitäten:	ReceiveOrder, CheckOrder, SaveOrder
Rechte:	reads Order
Verantwortungen:	
Liveness:	OrderManager = (ReceiveOrder . CheckOrder . SaveOrder)$^{\omega}$
Safety:	

Rolle: ProductManager PM	
Beschreibung:	Der ProductManager verwaltet die Daten für die einzelnen Produkte. Über diese Rolle kann die Position eines bestimmten Produktes nachverfolgt werden.
Protokolle und Aktivitäten:	UpdateProduct, UpdateProductInformation ,ReceivePosition , SavePosition, FindDelivery, RequestDeliveryFromStorageTransportManager, RequestDeliveyFromLogistik, SelectDelivery, OrderDelivery
Rechte:	reads Product, DeliveryQuotation, PositionData changes Product creates Delivery
Verantwortungen:	
Liveness:	ProductManager = (UpdateProduct . UpdateProductInformation)$^{\omega}$ || (ReceivePosition . SavePosition)$^{\omega}$ || (FindDelivery .

	liveryFromStorageTransportManager | RequestDeliveyFromLogistik^{all_known_companies}) . SelectDelivery . OrderDelivery)$^{\omega}$
Safety:	• num_delivery_for_product = 1

Rolle: StorageTransportManager STM	
Beschreibung:	Für ein bestimmtes Lager übernimmt diese Rolle die Aufgabe für die Koordination der Warenlieferungen.
Protokolle und Aktivitäten:	RequestForDelivery, RequestDeliveyFromLogistik, SelectLogistik, SubmitPrice
Rechte:	reads Product, DeliveryQuotation create Delivery
Verantwortungen:	
Liveness:	StorageTransportManager = (RequestForDelivery . RequestDeliveyFromLogistik^{all_known_logistikcompanies} . SelectLogistik . SubmitPrice)$^{\omega}$
Safety:	

Rolle: LogistikManager LM	
Beschreibung:	Der LogistikmManager übernimmt den Transport der Waren.
Protokolle und Aktivitäten:	RequestForDelivery, CalculatePrice, SubmitPrice, OrderforDelivery, StartDelivery, ReceivesProductScan, SubmitProductScan, RequestDeliveyFromLogistik
Rechte:	read Delivery, LKW creates PositionData, DeliveryQuotation
Verantwortungen:	
Liveness:	LogistikManager = (RequestForDelivery . [RequestDeliveyFromLogistik$^{all_known_logistikcompanies]}$. CalculatePrice . SubmitPrice) || (OrderforDelivery . StartDelivery)$^{\omega}$ $^{\|\|}$ (ReceiveProductScan . SubmitProductScan)$^{\omega}$
Safety:	

Rolle: OrderControlManager OCM	
Beschreibung:	Der OCM übernimmt die Überwachung des gesamten Systems. In regelmäßigen Abständen werden die Lagerbestände und Bestellungen abgefragt. Tritt ein Problem auf, so wird dies dem DemandManager und dem Prozesseigentümer gemeldet.
Protokolle und Ak-	RequestStocklevel, CheckSLA, MessaurePerformance, Submi-

tivitäten:	tEskalation, InformController
Rechte:	reads Stocklevel, SLA creates PerformanceReport creates Escalation
Verantwortungen:	
Liveness:	OrderControlManager = (RequestStocklevel$^{\text{for all buyer}}$. CheckSLA . MessaurePerformance . ([SubmitEscalation].[InformController]))$^{\omega}$
Safety:	

Rolle: SupplyControlManager SCM	
Beschreibung:	Der SCM überwacht das VMI auf der Seite des Kunden. Es überprüft die Einhaltung der SLA.
Protokolle und Aktivitäten:	RequestStocklevel, CheckSLA, MessaurePerformance, SubmitEskalation, InformController
Rechte:	reads SLA, Stocklevel creates Escalation creates PerformanceReport
Verantwortungen:	
Liveness:	OrderControlManager = (RequestStocklevel$^{\text{for all buyer}}$. CheckSLA . MessaurePerformance . ([SubmitEskalation].[InformController]))$^{\omega}$
Safety:	

Protokollname: ReceiveInquiry		
Initiator: SupplyManager	Partner: CustomerManager	Input:
Beschreibung:	Der CustomerManager erhält eine Anfrage für die Durchführung einer VMI-Partnerschaft.	Output: Inquiry

Protokollname: FindSKU		
Initiator: CustomerManager SupplyManager	Partner: KatalogManager	Input: Inquiry, Forecasts

CustomerForecastsManager SupplierForecastsManager		
Beschreibung:	Der KatalogManager bestimmt die SKU des Unternehmens und liefert diese zurück.	Output: SKU

Protokollname: CalculatePrice		
Initiator: CustomerManager	Partner: DemandManager	Input: Forecasts, SKU
Beschreibung:	Der CustomerManager liefert die Forecasts und die SKU-Bezeichnung an den DemandManager, der die Bestellung in seine Berechnungen miteinbezieht und einen Preis bestimmt.	Output: Quotation

Protokollname: TransmitQuotationToCustomer		
Initiator: CustomerManager	Partner: SupplyManager	Input: Quotation
Beschreibung:	Der CustomerManager schickt das Angebot an den Kunden zurück.	Output:

Protokollname: ReceiveForecasts		
Initiator: SupplyerForecastsManager	Partner: CustomerForecastsManager	Input:
Beschreibung:	Die Vorhersagen des Kunden werden an den Lieferanten geschickt.	Output: Forecasts

Protokollname: RequestForForecasts		
Initiator: DemandManager	Partner: CustomerForecastsManager	Input: Customer
Beschreibung:	Anfrage für Vorhersagen durch den DemandManager	Output:

Protokollname: TransmitForecasts		
Initiator: CustomerForecastsManager SupplierForecastsManager	Partner: DemandManager CustomerForecastsManager	Input: Forecasts
Beschreibung:	Die Vorhersagen werden weitergeleitet.	Output:

Protokollname: ReceiveNewInput		
Initiator: Employee	Partner: SupplyManager	Input: Anfrage für neues VMI
Beschreibung:	Ein Mitarbeiter möchte ein neues VMI einführen.	Output:

Protokollname: TransmitInquiry		
Initiator: SupplyManager	Partner: CustomerManager	Input: Inquiry
Beschreibung:	Eine Anfrage wird an den Lieferanten verschickt.	Output: Quotation

Protokollname: UpdateSKU		
Initiator: Employee	Partner: KatalogManager	Input: Änderungsanfrage
Beschreibung:	SKU Informationen werden erneuert.	Output:

Protokollname: UpdateKatalog		
Initiator: Employee	Partner: KatalogManager	Input: Änderungsanfrage
Beschreibung:	Katalog Informationen werden erneuert.	Output:

Protokollname: RequestForSKU		
Initiator: SupplierManager CustomerManager SupplierForecastsManager CustomerForecastsManager	Partner: KatalogManager	Input: Inquery, Forecasts
Beschreibung:	Anfrage für eine SKU.	Output:

Protokollname: SumitSKU		
Initiator: KatalogManager	Partner: SupplierManager CustomerManager SupplierForecastsManager CustomerForecastsManager	Input:
Beschreibung:	SKU Informationen werden übermittelt.	Output: SKU

Protokollname: RequestForecasts		
Initiator: DemandManager	Partner: CustomerForecastsManager	Input: Customer
Beschreibung:	Der DemandManager holt sich die Vorhersagen	Output: Forecasts

Protokollname: RequestStocklevel		
Initiator: DemandManager OrderControlManager SupplyControlManager	Partner: CustomerStorageManager SupplierStorageManager	Input: Customer
Beschreibung:	Stocklevel-Daten werden angefragt.	Output: Stocklevel

Protokollname: RequestDeliveryTime		
Initiator: DemandManager	Partner: StorageTransportManager	Input: SKU
Beschreibung:	Lieferzeiten werden nachgefragt.	Output: DeliveryTime

Protokollname: SubmitOrder		
Initiator: DemandManager	Partner: OrderManager	Input: Order
Beschreibung:	Der DemandManager verschickt die Bestellung an den Kunden.	Output:

Protokollname: ReceiveEskalationMessage		
Initiator: CustomerStorageManager SupplyerStorageManager OrderControlManager SupplyControlManager	Partner: DemandManager	Input:
Beschreibung:	Sumits Eskaltion to DemandManager.	Output: Escalation

Protokollname: RequestForPrice		
Initiator: CustomerManager	Partner: DemandManager	Input:
Beschreibung:	Get Request for a price	Output: Forecasts

Protokollname: SumitPrice		
Initiator: DemandManager	Partner: CustomerManager	Input: Quotation
Beschreibung:	Schickt Angebot zum CustomerManager.	Output:

Protokollname: SubmitStocklevel		
Initiator: CustomerStorageManager SupplierStorageManager	Partner: SupplierStorageManager DemandManager	Input:
Beschreibung:	Der aktuelle Lagerstand wird weitergeleitet.	Output: Stocklevel

Protokollname: SubmitEscalation		
Initiator: CustomerStorageManager SupplierStorageManager OrderControlManager SupplyControlManager	Partner: DemandManager	Input:
Beschreibung:	Wenn der Lagerbestand nicht dem Soll entspricht wird das direkt dem Demand-Manager gemeldet.	Output: Escalation

Protokollname: ReceiveStocklevel		
Initiator: CustomerStorageManager	Partner: SupplierStorageManager	Input:
Beschreibung:	Stocklevel-Daten werden kontinuierlich an den Lieferanten geschickt.	Output: Stocklevel

Protokollname: RequestForStoragelevel		
Initiator: DemandManager	Partner: SupplyStorageManager	Input: Customer
Beschreibung:	Anfrage des DemandManagers betreffend der Lagerbestände.	Output:

Protokollname: ReceiveOrder		
Initiator: DemandManager	Partner: OrderManager	Input: Order

Beschreibung:	Liefert die Bestellung an den OrderManager.	Output:

Protokollname: UpdateProduct		
Initiator: Employee	Partner: ProductManager	Input: Update Information
Beschreibung:	Produktinformation erneuern.	Output:

Protokollname: ReceivePosition		
Initiator: LogistikManager	Partner: ProductManager	Input: PositionData
Beschreibung:	Die neue Position des Produktes wird festgehalten.	Output:

Protokollname: FindDelivery		
Initiator: Product	Partner: ProductManager	Input: PositionData
Beschreibung:	Das Produkt passiert eine Kontrollstelle und ist für die Auslieferung bereit.	Output:

Protokollname: RequestDeliveryFromStorageTransportManager		
Initiator: ProductManager	Partner: StorageTransportManager	Input: Delivery
Beschreibung:	Es wird eine Anfrage über die Lieferung an das Lagerverwaltungstool gestellt.	Output: DeliveryQuotation

Protokollname: RequestDeliveyFromLogistik		
Initiator: ProductManager StorageTransportManager	Partner: LogistikManager	Input: Delivery

LogistikManager		
Beschreibung:	Es wird eine Anfrage über die Lieferung direkt an das Logistikunternehmen gestellt.	Output: DeliveryQuotation

Protokollname: OrderDelivery		
Initiator: ProductManager	Partner: LogistikManager StorageTransportManager	Input: OrderDelivery
Beschreibung:	Es wird die Bestellung der Lieferung abgesegnet.	Output:

Protokollname: RequestForDelivery		
Initiator: ProductManager StorageTransportManager	Partner: StorageTransportManager LogistikManager	Input:
Beschreibung:	Es wird eine Anfrage über die Lieferung an das Lagerverwaltungstool gestellt.	Output: Delivery

Protokollname: SumitPrice		
Initiator: LogistikManager StorageTransportManager	Partner: ProductManager	Input:
Beschreibung:	Der Preis wird an den Produktmanager geliefert.	Output: DeliveryQuotation

Protokollname: OrderforDelivery		
Initiator: ProductManager	Partner: LogistikManager	Input: DeliveryOrder
Beschreibung:	Bestellung der Lieferung.	Output:

Protokollname: ReceiveProductScan		

Initiator: Product	Partner: LogistikManager	Input: PositionData
Beschreibung:	Ein Scan des Produktes wird durchgeführt.	Output:

Protokollname: SubmitProductScan		
Initiator: LogistikManager	Partner: Productmanager	Input:
Beschreibung:	Die Position des Produktes wird an den ProduktManager weitergeleitet	Output: PositionData

Protokollname: InformController		
Initiator: SupplyControlManager OrderControlManager	Partner: Controller	Input: PerformanceReport
Beschreibung:	Die Performance des VMI wird berichtet.	Output:

Ressourcen

Name	Beschreibung	Rollen
Forecasts	Informationen über die zukünftigen Verkäufe	CM, CFM, SFM, SM, KM, DM
Inquery	Anfrage für einen Preis auf der Basis von Vorhersagen	CM, SM, KM
Quotation	Angebot für VMI	CM, SM, DM
SKU	Informationen über eine SKU	CM, CFM, KM, SFM
SLA	Informationen über die Zusammenarbeit	CM, SM, OCM, SCM
Katalog	Information über die gesamten Produkte des Unternehmens	KM
Stocklevel	Informationen über die Lagerbestände	CSM, SSM, DM, SCM,

		OCM
Escalation	Nachricht über einen Vorfall	CSM, SSM, OCM, SCM, DM
Order	Produktbestellung	DM, OM
Product	Informationen über ein bestimmtes Produkt	PM
Delivery	Anfrage einer Lieferung	PM, STM, LM
DeliveryQuotation	Angebote für eine Lieferung	PM, STM, LM
DeliveryOrder	Bestellung einer Lieferung	PM, LM
DeliveryTime	Lieferzeit	DM, STM
PostionData	Positionsdaten eines Produktes	PM, LM
LKW	Stellt ein Fahrzeug des Logistikunternehmens dar	LM
PerformanceReport	Bericht über die Leistung des VMI	SCM, OCM

Anhang B:

Architektur

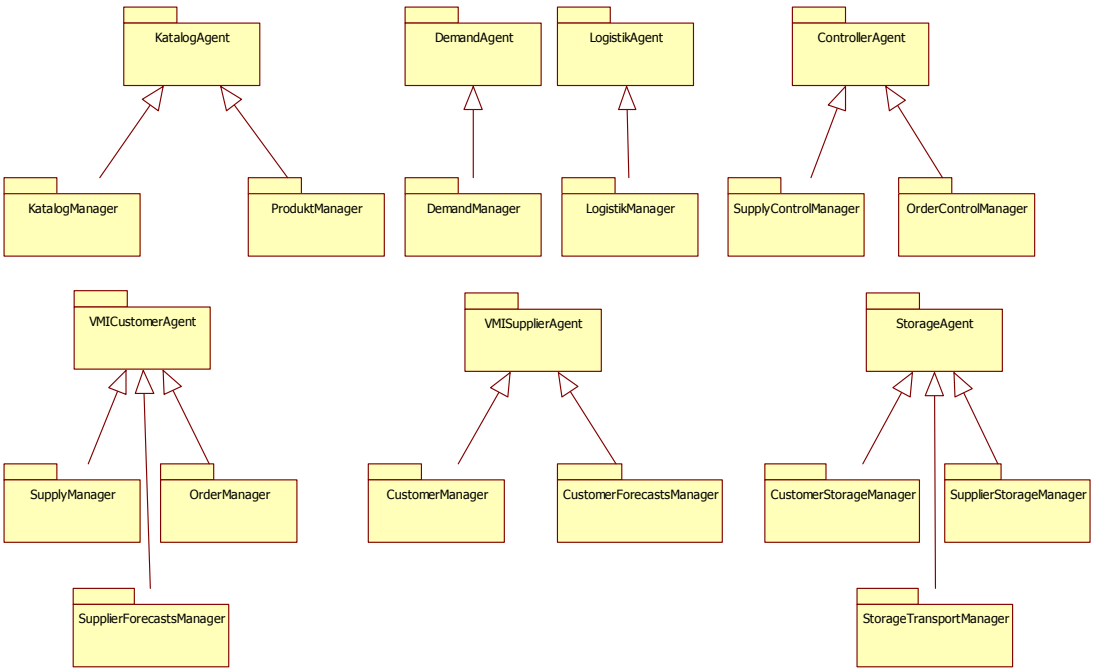

Abbildung 46: Architektur

Anhang C:

Detaildesign

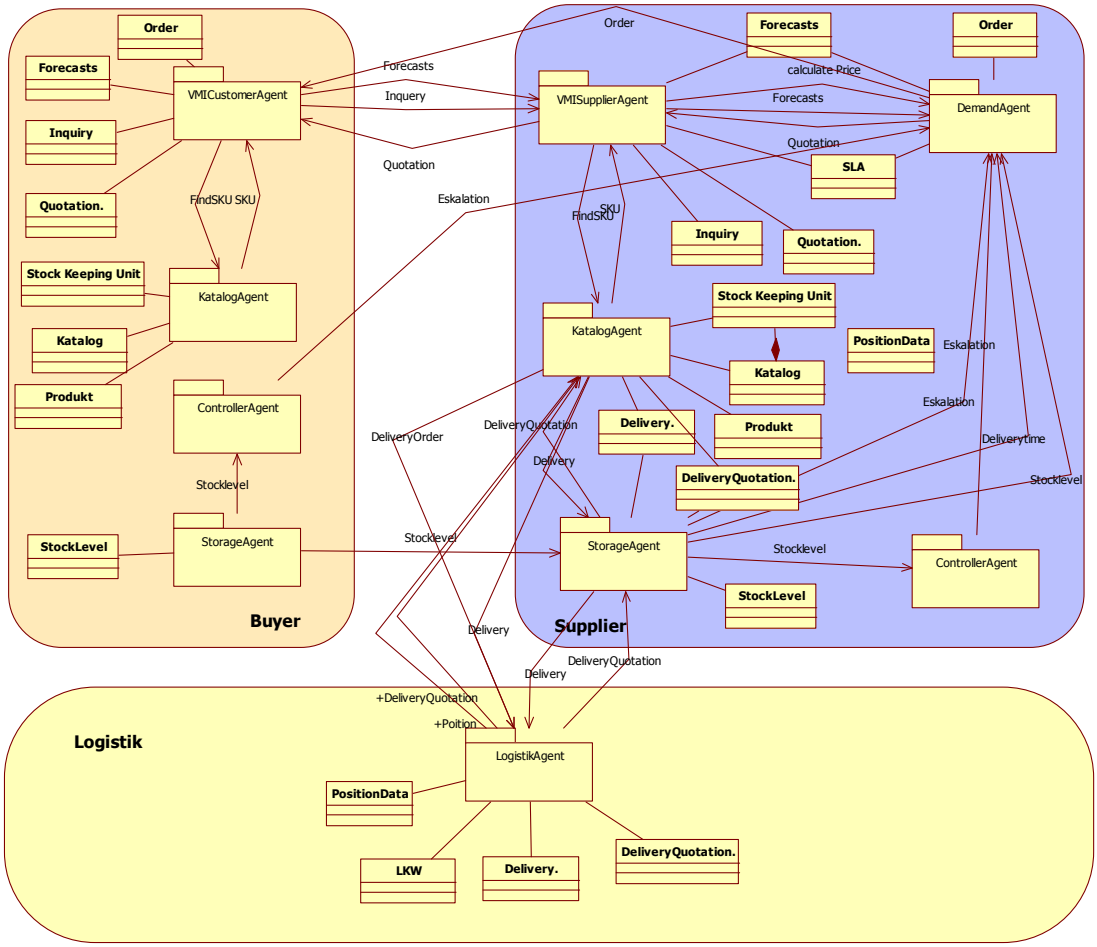

Abbildung 47: Detaildesign

Abbildungen

Tabellen

Literaturverzeichnis

Accenture. (2001). *A Guide to CPFR Implementation*. Retrieved 2 1, 2008, from http://www.ecr.no/data/f/0/70/62/2_2401_0/2001_a_guide_to_cpfr_implementation.pdf

Achabal, D. D., Mcintyre, S. H., Smith, S. A., & Kalyanam, K. (2000). A Decision Support System for Vendor Managed Inventory. *Journal of Retailing Volume 76(4)* , 430-454.

Berdou, E., & Dini, P. (2005). *Del.18.3 Report on the socio-economics of Free/Open Source.* London: DBE Consortium.

Bianco, P., Kotermanski, R., & Merson, P. (2007). *Evaluating a Service-Oriented Architecture.* Hanscom AFB: Software Engineering Institue.

Bieberstein, N., Bose, S., Fiammante, M., Jones, K., & Shah, R. (2005). *Service-Oriented Architecture Compass: Business Value, Planning, and Enterprise Roadmap.* IBM Press.

Biskupski, B., Sacha, J., Dowling, J., Seigneur, J. M., Edmonds, A., Meier, R., et al. (2005). *Del.24.3 DBE Peer-to-Peer Architecture Design.* DBE Consortium.

Blomqvist, E., Levashova, T., Öhgren, A., Sandkuhl, K., Smirnov, A., & Tarassov, V. (2005). Configuration of Dynamic SME Supply Chains Based on Ontologies. *HoloMAS 2005* , 246-256.

Bonabeau, E., & Meyer, C. (2001). Swarm Intelligence: A Wohle New Way to Think About Business. pp. 106-114.

Bowersox, D. J., Closs, D. J., & Cooper, M. B. (2002). *Supply Chain Logistics Management.* New York: The McGraw-Hill Companies, Inc.

Bozarth, C. (2007). *Collaborative Planning, Forecasting and Replenishment (CPFR): A Tutorial*. Retrieved 01 04, 2007, from http://scm.ncsu.edu/public/cpfr/index.html

Briscoe, G., Sadedin, S., & Paperin, G. (2007). Biology of Applied Digital Ecosystems. *In IEEE First International Conference on Digital Ecosystems and Technologies* .

Briscoe, Gerard; De Wilde, Philippe. (2005). *Del.6.2 Self-organisation of Evolving Agent Populations.* London: DBE Consortium.

Briscoe, Gerard; De Wilde, Philippe. (2005). *Del.6.3 How Software Develpment in the DBE Differs From Normal Buinsess Software Development.* London: DBE Consortium.

Briscoe, Gerard; De Wilde, Philippe. (2005). *Del.6.6 High-Level Design Specification of the Distributed Intelligence System.* London: DBE Consortium.

Carter, S. (2007). *The New Language of Business: SOA & Web 2.0.* Upper Saddle River: Pearson plc.

Chesbrough, H. W. (2003). *Open Innovation: The New Imperative for Creating and Profiting from Technology.* Boston: Harvard Business School Publishing Corporation.

Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (2001). *Web Services Description Language (WSDL) 1.1*. (W3C) Retrieved 09 2007, from http://www.w3.org/TR/wsdl

Cingil, I., & Dogac, A. (2001, Juli). An Architecture for Supply Chain Integration and Automation on the Internet. *Distributed and Parallel Databases Volume 10, Number 1* , pp. 59-102.

Cohen, S., & Roussel, J. (2005). *Strategic Supply Chain Management - The five Disciplines for Top Performance.* New York: The McGraw-Hill Companies, Inc.

Commsision of the European Communities. (2001). *Helping SMEs to "Go Digital".* Brussels: Commsision of the European Communities.

Corallo, A., Tommasi, M. D., & Cisternino, V. (2005). *Del 15.3 BML framework 2nd release.* ISUFI.

DBE Studio. (2007). Retrieved 09 2007, from DBE Studio: http://dbestudio.sourceforge.net/

De Wilde, P., & Briscoe, G. (2006). *Digital Ecosystems: Evolving Service-Oriented Architecture.* IEEE.

Deloitte Consulting. (2003). *Collaborative POS Data Management.* ECR Europe.

Dini, P., & Berdou, E. (2004). *Del.18.1 Report on DBE-specific Use Cases.* DBE Consortium.

Dini, P., & Nicolai, A. (2003). *The Digital Business Ecosystem.* Retrieved 12 29, 2007, from http://www.digital-ecosystems.org/cluster/dbe/dbe_summary_cc.pdf

Dini, P., Darking, M., Rathbone, N., Hermandez, P., Briscone, G., Ferronato, P., et al. (2005). *The Digital Ecosystem Research Vision: 2010 and Beyond.* DBE Consortium.

Dini, Paolo. (2007). *The Role of Fundamental Science in Digital Ecosystems Research*. Retrieved 12 28, 2007, from http://files.opaals.org/OPAALS/Presentations/PDini/Fundamental_Science_for_DE.pdf

Disney, S. M., & Towill, D. R. (2003). The effect of vendor managed inventory (VMI) dynamics on the Bullwhip Effect in supply chains. *International Journal of Production Economics, Volume 85, Issue 2* , pp. 199-215.

Edmonds, A., Dahlem, D., & McKitterick, D. (2005). *Del.24.1: DBE First Implementation.* DBE Consortium.

Egri, P., & Váneza, J. (2005). Cooperative Planning in the Supply Network - A Multiagent Organization Model. *CEEMAS 2005, LNAI 3690* , 346-356.

Erl, Thomas. (2005). *Service-Oriented Architecture: Concepts, Technology, and Design.* Upper Saddle River: Pearson Education, Inc.

Erl, Thomas. (2008). *SOA Principles of Service Design.* Bosten: Pearson Education, Inc.

FADA. (2008). *FADA*. Retrieved 01 14, 2008, from http://fada.sourceforge.net/

Fayez, M., Rabelo, L., & Mollaghasemi, M. (2005). Ontologies for supply-chain simulation modeling. *Einter Simulation Converence 2005* , 2364-2370.

Främling, K., Kärkkäinen, M., Ala-Risku, T., & Holmström, J. (n.d.). *Managing Product Information in Supplier Networks by Object Oriented Programming Concepts*. Retrieved 01 04, 2008, from http://www.tuta.hut.fi/library/working_paper/pdf/Framling_et_al_WP.pdf

Grundmann, M. (2005). *Diplomarbeit: Prozessmodellintegration.* Reutlingen: Hochschule Reutlingen.

Heck, E. V., & Vervest, P. (2007). *Smart Business Networks: Hoe the Network wins.* New York: ACM.

Heistracher, T. J., Kurz, T., Marcon, G., & Masuch, C. (2005). *Del.9.1 Report on Fitness Landscape.* Salzburg: DBE Consortium and European Commission.

Holmström, J., Främling, K., Kaipia, R., & Saranen, J. (2002). Collaborative planning forecasting and replenishment: new solutions needed for mass collaboration. *Supply Chain Management: An International Journal Volume 7 . Number 3* , pp. 136-145.

IBM. (2007). *Einstiegspunkte in eine serviceorientierte Architektur (SOA)*. (IBM) Retrieved 11 07, 2007, from http://www-306.ibm.com/software/at/solutions/soa/soaentrypoints.html

IBM Global Business Services. (2007). *A component-based approach to strategic change*. Retrieved 11 12, 2007, from A component-based approach to strategic change: http://www-935.ibm.com/services/us/igs/cbm/html/bizmodel.html

Josuttis, N. M. (2007). *SOA in Practice.* Sebastopol: O'Reilly Media, Inc.

Kärkkäinen, M., Holmström, J., Främling, K., & Artto, K. (2003). Intelligente products - a step towards a more effective project delivery chain. *Computers in Industry Vo. 50 No. 2* .

Kasinger, H., & Bauer, B. (2006). Beyond Swarm Intelligence: Building Self-Managing Systems Based on Pollination. *Lecture Notes in Informatics (LNI), volume P-93* , pp. 169-176.

Keen, M., Acharya, A., Bishop, S., Hopkins, A., Milinski, S., Nott, C., et al. (2004). *Patterns: Implementing an SOA Using an Enterprise Service Bus.* IBM Corp.

Kennedy, J., & Lee, R. (2007). *Del.24.7 Implementation of the Distributed Storage System.* DBE Consortium.

Kennedy, John. (2007). Distributed Infrastructural Services. In F. Nachira, A. Nicolai, P. Dini, M. Le Louarn, & L. R. Leon, *The Digital Business Ecosystems* (pp. 134-140). Bruxelles: European Commission.

Konsynski, B., & Tiwani, A. (2004). The Improvisation-Efficiency Paradox in Inter-firm Electronic Networks: Governance and Architecture Considerations. *Journal of Information Technology, Vol. 19, No. 4* , pp. 234-243.

Kotopoulos, G., Kotopoulos, Y., & Kazasis, F. (2006). *Del.14.6: Final Release of the Recommender.* European Commission.

Krafzig, D., Banke, K., & Slama, D. (2005). *Enterprise SOA: Service-Oriented Architecture Best Practices.* Upper Saddle River: Pearson Education, Inc.

Kurz, T., Marcon, G., Okada, H., Heistracher, T. J., & Passani, A. (2006). *Del.9.2 Report on Evolutionary and Distributed Fintess Environment.* DBE Consortium.

Lee, H. L., Padmanabhan, V., & Whang, S. (1997). The Bullwhip Effect in Supply Chains. *MIT Sloan Vol. 38, No. 3* , pp. 93-102.

Lessig, L. (2001). *the future of ideas.* New York: Random House, Inc.

Lipson, H. (2007, 03). *Robots that are "self-aware"*. Retrieved 01 04, 2008, from http://www.ted.com/talks/view/id/195

Lödding, H. (2005). *Verfahren der Fertigungssteuerung - Grundlagen, Beschreibung, Konfiguration.* New York: Springer Berlin Heidelberg.

MacKenzie, M. C., Laskey, K., McCabe, F., Brown, F. P., & Metz, R. (2006). *Reference Model for Service Oriented Architecture 1.0.* OASIS.

Malone, P. (2006). DE Services in Ecosystem Oriented Architecture. In F. Nachira, P. Dini, A. Nicolai, M. Le Louarn, & L. Riv, *Digital Business Ecosystems* (pp. 119 - 124). European Commission.

Marcon, G., Okada, H., Heistracher, T. J., & Kurz, T. (2006). *Del.16.3 Report on Adaptive Service Generator.* DBE Consortium.

Margolis, B., & Sharpe, J. (2007). *SOA for the Business Developer: Concepts, BPEL, and SCA, First Edition.* Lewisville: MC Press Online, LP .

McKitterick, D. (2006). *Del.6.7 Impelemntation of Distributed Intelligence System.* Ireland: Intel Ireland Ltd.

Microsoft. (2006). *Learn About Service Oriented Architecture (SOA)*. (Microsoft) Retrieved 11 03, 2007, from Learn About Service Oriented Architecture (SOA): http://www.microsoft.com/biztalk/solutions/soa/overview.mspx

Miller, J., & Mukerji, J. (2003). *MDA Guide Version 1.0.1.* Object Management Group, Inc.

Nachiappan, S. P., & Jawahar, N. (2007). A genetic algorithm for optimal operating parameters of VMI system in a two-echelon supply chain. *European Journal of Operational Research Volume 182 Issue 3* , pp. 1433-1452.

Nachira, F., Nicolai, A., Dini, P., Le Louarn, M., & Leon, L. R. (2007). *Digital Business Ecosystems.* European Commission.

Nachira, F., Nicolai, A., Dini, P., Le Louarn, M., & Leon, L. R. (2007). *Digital Business Ecosystems.* Luxembourg: European Commission.

Nachira, Francesco. (2002). *Towards a network of digital Business Ecosystems fostering the local develpment.* Bruxelles: European commission.

New, S., & Westbrook, R. (2004). *Understanding Supply Chains: Concepts, Critiques & Futures.* Oxford: Oxford University Press.

OASIS. (n.d.). *OASIS Universal Business Language (UBL) TC*. Retrieved 01 04, 2008, from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ubl

OASIS. (2008). *Organization for the Advancement of Structured Information Standards*. Retrieved 01 04, 2008, from http://www.oasis-open.org/

Oliver, R. K., & Webber, M. D. (1982). Supply-Chain Management: Logistics Catches up with Strategy. *Logistics: The Strategic Issues* , pp. 63-75.

OMG. (2005). *Meta Object Facility (MOF) Specification 1.4.1.* ISO/IEC .

Open ePolicy Group. (2007, 07). Roadmap for Open ICT Ecosystem. http://cyber.law.harvard.edu/epolicy/, USA: Open ePolicy Group.

Papazoglou, M. P., & van den Heuvel, W.-J. (2007). Service oriented architectures: approaches, technologies and research issues. *The VLDB Journal (2007) 16* , 389-415.

Pappas, N., Kazasis, F. G., Anestis, G., Gioldasis, N., & Christodoulakis, S. (2005). *Del.14.3 1st P2P distributed implementation of the DBE Knowledge Base and Semantic Registry.* DBE Consortium.

Pulier, E., & Taylor, H. (2006). *Understanding Enterprise SOA.* Greenwich: Manning Publications Co.

Roland Berger & Partner. (1996). *Efficient Replenishment Techniques.* ECR Europe.

Schroth, C., & Janner, T. (2007). Web 2.0 and SOA: Converging Concepts Enabling the Internet of Services. *IT Pro May/June 2007* , 36-41.

Software Engineering Institute. (2007). *CMMI® for Development Version 1.2.* Pittsburgh: SEI.

Soluta.net. (2006). *Del 16.2: SDL specification for the DBE Platform.* DBE Consortium.

Soluta.net. (2004). *Del 21.2: Architecture Scope Document.* DBE Consortium.

Soluta.net. (2005, 04 30). *Del 21.3 DBE Architecture High Level Specification.* Retrieved 09 2007, from Digital Business Ecosystem: http://www.digital-ecosystem.org

Soluta.net. (2004). *Del.21.2: Architecture Scope Document.* DBE Consortium.

stratEDI Gesellschaft für Kommunikationskonzepte und -lösungen mbH. (2002, 09 12). *ECR in der Praxis: Das Beispiel Grohe*. Retrieved 01 04, 2008, from http://www.ecin.de/edi/grohe/

Supply-Chain Council. (2006). *Supply-Chain Operations Referenfce-model 8.0.* Supply-Chain Council.

Teuteberg, F., & Ickerott, I. (2007). Mobile Supply Chain Event Management Using Auto-ID and Sensor Technologies - A Simulation Approach. *Trends in Supply Chain Design and Management* , pp. 93-125.

The World Wide Web Consortium. (2008). *The World Wide Web Consortium*. Retrieved 01 04, 2008, from http://www.w3.org/

Trinity College Dublin. (2006). *Del 17.4: Automatic Composer.* Dublin: Trinity College Dublin.

Tschudin, C. F. *Fraglest - a Metabolistic Execution Model for Communication Protocols.* Basel: University of Basel.

TUC. (2005). *Del 14.1: DBE Knowledge Representation Models.* DBE Consortium.

Universität Duisburg Essen. (2007). *ICT-Research Report No.21.* Essen: Universität Duisburg Essen.

Voluntary Interindustry Commerce Standards (VICS). (2004). *Collaborative Planning, Forecasting and Replenishment (CPFR).* VICS.

Voluntary Interindustry Commerce Standards (VICS). (2004). *Collaborative Planning, Forecasting and Replenishment (CPFR®).* VISC Association.

Wang, J., & De Wilde, P. (2006). *Del.19.5 Report on Distributed Mechanism.* DBE Consortium.

Wannenwetsch, H. H., & Nicolai, S. (2004). *E-Supply-Chain-Management: Grundlagen - Strategien - Praxisanwendungen.* Wiesbaden: Betriebswirtschaftlicher Verlag Dr. Th. Gabler/GWV Fachverlage GmbH.

Wißkirchen, D. F. (2002). Shared Services Organisationen (Teil 1). *HR Services* (02), pp. 38-40.

WS-I . (2008). *WS-I* . Retrieved 01 04, 2008, from http://www.ws-i.org

Yuxiao, Z. (2003, 03). Develop the Ontology for Internet Commerce by Reusing Existing Standards. *Proceedings of International Workshop on Semantic Web Foundations and Application Technologies (SWFAT)* , pp. 51-57.

Zambonelli, F., Jennings, N. R., & Wooldridge, M. (2003). Developing Multiagent Systems: The Gaia Methodology. *ACM Transactions on Software Engineering and Methodology (TOSEM) Volume 12 , Issue 3* , pp. 317–370.