SERV PAY

Zahlungsbereitschaften für
Geschäftsmodelle produktbegleitender Dienstleistungen

DOKUMENTATION

Modellierung, Konfiguration und Kalkulation hybrider Leistungsbündel mit H2-ServPay

- Becker, J.
- Beverungen, D.
- Brune, C.
- Bunge, M.
- Knackstedt, R.
- Meschede, B
- Müller, O.
- Philipp, M.
- Stepanow, K.

ERCIS

IAS

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

SERVPAY ARBEITSPAPIER
WORKING PAPER

Nr. 4 (2008)

Becker, J.; Beverungen, D.; Brune, C.; Bunge, M.;
Knackstedt, R.; Meschede, B; Müller, O.; Philipp, M.; Stepanow, K.

Modellierung, Konfiguration und Kalkulation hybrider Leistungsbündel mit H2-ServPay

Bericht über die Toolentwicklung und -anwendung auf Basis eines Szenarios aus der Fertigungsindustrie

Ein Beitrag zum Forschungsprojekt
„ServPay – Zahlungsbereitschaften für Geschäftsmodelle produktbegleitender Dienstleistungen“

Dieses Forschungs- und Entwicklungsprojekt wird mit Mitteln des Bundesministeriums für Bildung und Forschung (BMBF) innerhalb des Rahmenkonzeptes „Forschung für die Produktion von morgen“ (Förderkennzeichen: 02PG1010) gefördert und vom Projektträger Forschungszentrum Karlsruhe, Bereich Produktion und Fertigungstechnologien (PTKA-PFT), betreut.

BETREUT VOM
Projektträger Forschungszentrum Karlsruhe (PTKA)

Prof. Dr. Jörg Becker
Westfälische Wilhelms-Universität Münster
European Research Center for Information Systems (ERCIS)
Institut für Wirtschaftsinformatik
Leonardo-Campus 3
48149 Münster
Tel.: 0251-83-38100
Fax: 0251-83-28062
Email: becker@ercis.uni-muenster.de

Dipl.-Wirt.Inform. Daniel Beverungen
Westfälische Wilhelms-Universität Münster
European Research Center for Information Systems (ERCIS)
Institut für Wirtschaftsinformatik
Leonardo-Campus 3
48149 Münster
Tel.: 0251-83-38092
Fax: 0251-83-28068
Email: daniel.beverungen@ercis.uni-muenster.de

Dr. Ralf Knackstedt
Westfälische Wilhelms-Universität Münster
European Research Center for Information Systems (ERCIS)
Institut für Wirtschaftsinformatik
Leonardo-Campus 3
48149 Münster
Tel.: 0251-83-38094
Fax: 0251-83-28062
Email: ralf.knackstedt@ercis.uni-muenster.de

Dipl.-Wirt.Inform. Oliver Müller
Westfälische Wilhelms-Universität Münster
European Research Center for Information Systems (ERCIS)
Institut für Wirtschaftsinformatik
Leonardo-Campus 3
48149 Münster
Tel.: 0251-83-38064
Fax: 0251-83-28064
Email: oliver.mueller@ercis.uni-muenster.de

Unter dem damaligen Arbeitstitel „HyProDesign“ dieses sog. Projektseminars am ERCIS wurde in 6 Monaten in einem Team von 15 Studierenden und 6 Betreuern die beschriebene Toolunterstützung neben weiteren Funktionalitäten konzipiert, implementiert und getestet. Wir danken an dieser Stelle dem gesamten studentischen Team, das hier durch die Co-Autoren des Arbeitsberichts vertreten wird, für die tatkräftige Unterstützung.

Mitglieder des Teams waren: Cedric Berlin, Christian Brune, Michael Bunge, Marc Deinert, Robin Fischer, Volker Frehe, Alexander Hachmann, Björn Meschede, Steffen Müller, Marc Philipp, Konstantin Stepanow, Carl Stolze, Nico Straßenburg, Holger Sunke und Daniel Schulze-Zumkley.

Als weitere Betreuer fungierten: Dipl. Wirt.-Inform. Stefan Fleischer, Magister Lukasz Lis und Dipl. Wirt.-Inform. Armin Stein

Inhaltsverzeichnis

1 Einführung und Einordnung ... 2
1.1 Einführung in die Bündelung von Sach- und Dienstleistungen ... 2
1.2 Einordnung in das Forschungsprojekt ServPay ... 3
1.3 Aufbau des Arbeitsberichts und Projektszenario ... 4
2 Projektspezifische Ansätze zur Modellierung hybrider Leistungsbündel ... 6
2.1 ServPay Modellierungstool (1) ... 7
2.2 Leistungskonfigurator (2) ... 8
2.3 Zahlungsbereitschaftstool (3a) ... 9
2.4 Kostenkalkulationstool (3b) ... 10
2.5 Analysefunktionalität (4a, 4b, 4c) ... 11
3 Fachkonzept ... 13
3.1 Anbietersicht ... 13
3.1.1 Hybride Leistungsbündel auf Typebene ... 13
3.1.2 Lebenszyklusphasen ... 15
3.1.3 Regelkonstrukte ... 18
3.2 Kundensicht ... 20
3.2.1 Hybrides Leistungsbündel auf Instanzebene ... 20
3.2.2 Lebenszyklusphasen ... 22
3.2.3 Präferenzen und Präferenzwerte ... 24
3.2.4 Assoziationsregeln ... 25
4 Vorgehensmodelle und Anleitungen ... 27
4.1 Modellierung anhand der HLB-Sprache ... 27
4.1.1 (Kundenseitige) Ressourcen, Kostenstellen, Organisationseinheiten ... 29
4.1.2 Lebenszyklusphasen, Leistungseigenschaften, Präferenzen, Präferenzwerte ... 33
4.1.3 Leistungen ... 35
4.1.4 Module und Hybride Leistungsbündel auf Typebene ... 40
4.1.5 Assoziationsregeln ... 44
4.2 Anleitung zur Nutzung des Kostenkalkulationstools ... 45
4.3 Anleitung zur Nutzung des Leistungskonfigurators ... 51
5 Implementierung ... 71
5.1 Kostenkalkulationstool ... 71
5.2 Leistungskonfigurator ... 76
6 Fazit und Ausblick ... 90
Anhang ... 92
A H2 Sprachdefinition ... 92
B Wissensbasierte Vorschlagsgenerierung nach Ardissono und Goy ... 94
C Detaillierte Darstellung der Hilfsklassen ... 96

1 Einführung und Einordnung

1.1 Einführung in die Bündelung von Sach- und Dienstleistungen

In den letzten Jahren hat im Industriegüterbereich aufgrund von steigendem Konkurrenzdruck eine Erweiterung des Leistungsangebots stattgefunden. Bei der Erweiterung des Leistungsangebots werden zusätzlich zu Sachleistungen auch industrielle Dienstleistungen mit in das Angebot aufgenommen. Unternehmen, die eine Kombination aus Sach- und Dienstleistungen anbieten, entwickeln sich dadurch zu einem Problemlöser und steigern somit ihre Attraktivität für Kunden.[1]

Problemlösungen für den Kunden kommen durch Leistungen zustande. Eine Leistung kann generell als das Ergebnis eines (Geschäfts-) Prozesses verstanden werden und kann sich aus verschiedenen Leistungsarten, wie beispielsweise Sach- und Dienstleistungen, zusammensetzen.[2] Sachleistungen zeichnen sich durch ihren materiellen Charakter aus. Der Begriff Sachleistung wird in Rohstoffe, Produktions-/Betriebsmittel und Verbrauchsgüter unterteilt. Sachleistungen sind beispielsweise Material, gefertigte Zwischenprodukte und gefertigte Endprodukte.[3]

Im Gegensatz zu Sachleistungen findet sich zum Dienstleistungsbegriff keine eindeutige Definition in der Fachliteratur. Stattdessen existiert eine Vielzahl möglicher Ansätze zur Definition und Abgrenzung des Dienstleistungsbegriffs. Die maßgeblichen Eigenschaften einer Dienstleistung, die zur Abgrenzung einer Sachleistung genutzt werden können, lassen sich wie folgt zusammenfassen: Immaterialität des Outputs, keine Lagerung und keine Vorratsproduktion, Gleichzeitigkeit von Leistungserstellung und Konsum, sowie die Integration des Kunden als externen Faktor in den Produktionsprozess.[4]

Es lässt sich feststellen, dass sich Dienstleistungen aufgrund dieser Eigenschaften zwar grundsätzlich von Sachleistungen unterscheiden. Der Ausgangspunkt jedes Angebotes ist jedoch die Nachfrage des Kunden, die sein Bedürfnis widerspiegelt, für ihn wichtige Probleme zu lösen. Diesen Bedürfnissen begegnet ein Anbieter mit einer individuell zusammengesetzten Gesamtlösung, bestehend aus materiellen und immateriellen Leistungsbausteinen, die er auf die Kundenbedürfnisse abstimmt. Die Kombination von Sach- und Dienstleistungen wird dabei als *hybrides Leistungsbündel* bezeichnet. Das Ergebnis der Leistung wird kooperativ durch eine Endkombination von Ressourcen des Anbieters und Ressourcen des Kunden (als sog. externer Faktor) erzeugt.[5] Beispielsweise ist zu der Dienstleistung „Schulung von Bedienpersonal einer technischen Anlage“ neben geeigneten Maschinen und dem Trainer (Ressourcen des Anbieters) auch die Anwesenheit des zu schulenden Personals erforderlich (Ressourcen des Kunden). Diese Kooperation muss bei der Erstellung von

1 vgl. Backhaus, Lütgemüller, Weddeling (2007), S. 2
2 vgl. Scheer, Grieble, Klein (2006), S. 21.
3 vgl. Scheer, Grieble, Klein (2006), S. 21 f.
4 vgl. Beier (2006), S. 15 f.
5 vgl. Schmid (2005), S. 68.

Dienstleistungen berücksichtigt werden und findet somit auch Einzug in die Erstellung hybrider Leistungsbündel.

SCHEER, GRIEBLE und KLEIN verdeutlichen die generelle Notwendigkeit der integrierten Betrachtung von Sach- und Dienstleistungen in ihrer Arbeit an einem einfachen Beispiel: Bei einem Wechsel eines defekten Autoreifens wird auf der einen Seite eine Dienstleistung erbracht (das Auswechseln des Reifens). Auf der anderen Seite wird auch eine Sachleistung mit eingebracht (der neue Reifen). Wenn nun versucht wird, die beiden Leistungskomponenten zu teilen, stellt sich heraus, dass es lediglich möglich ist, die Sachleistung separat zu betrachten. Eine alleinige Betrachtung der Dienstleistung (d.h. eine Reifenwechseln ohne Reifen) hingegen wäre nicht sinnvoll.[6]

Angesichts der Schwierigkeit der Abgrenzung von Sachleistungen und Dienstleistungen wird in der Literatur von verschiedenen Autoren ein Ansatz verfolgt, bei dem keine strikte Trennung erfolgt, sondern der Übergang von Sach- zu Dienstleistungen als fließend erachtet wird.[7] Diesem Ansatz folgend, wird innerhalb dieses Projektes ebenfalls nicht strikt zwischen Sach- und Dienstleistungen unterschieden, sondern Sach- und Dienstleistungen werden im Folgenden unter dem Begriff Leistungen zusammenfassend betrachtet.[8]

1.2 Einordnung in das Forschungsprojekt ServPay

Das Management hybrider Leistungsbündel gestaltet sich als sehr komplex. Oftmals sind Anbieter kaum in der Lage zur systematischen Beschreibung der von ihnen angebotenen hybrider Leistungsbündel, da existierende Modellierungssprachen zur Entwicklung und Beschreibung hybrider Leistungsbündel nicht existieren. Dies macht insbesondere auch eine wirtschaftliche Vermarktung hybrider Leistungsbündel schwierig, da weder das eigene Leistungsangebot in ausreichender Tiefe bekannt ist, noch Kostentreiber und Zahlungsbereitschaften für hybride Produkte bekannt sind. Die Folge sind ungenutzte Chancen – bis zu 50 % des Ertragspotentials im Servicegeschäft werden somit leichtfertig verschenkt.[9]

Im Rahmen des vom Bundesministerium für Bildung und Forschung (BMBF) geförderten Forschungsprojektes ServPay wird ein Entscheidungsunterstützungskonzept entwickelt, um Anbietern eine wirtschaftliche Vermarktung ihrer Leistungen zu erleichtern. Der Preis für angebotene Leistungen muss dabei sowohl kostendeckend, als auch am Markt durchsetzbar sein. Daher sind einerseits die Kosten der Leistungserbringung systematisch zu kalkulieren (untere Grenze des Preisspielraums), andererseits die Zahlungsbereitschaft des Kunden zu ermitteln (obere Grenze des Preis-

6 vgl. Scheer, Grieble, Klein (2006), S. 26 f.
7 vgl. Schmid (2005), S. 67 f.; Engelhardt, Kleinaltenkamp, Reckenfelderbäumer (1993), Hilke (1989), Vargo und Lusch (2004)
8 Gleichwohl ist auch das Abbilden „reiner“ Sachleistungen oder Dienstleistungen möglich, falls es zweckmäßig erscheint.
9 vgl. Innovationsreport.de (Zugriff: 10.10.06)

spielraums)[10]. Um die Berechnungen zu ermöglichen, basieren beide Aspekte auf einer aussagekräftigen Modellierung des Leistungsangebotes, die mithilfe einer speziellen Modellierungssprache ermöglicht wird. Gleichermaßen wird auch die Konfiguration von maßgeschneiderten Lösungen für den Kunden aus vordefinierten Lösungsbausteinen – ähnlich wie in einem Produktkonfigurator – ermöglicht. Dadurch kann mit einer begrenzten Anzahl an Lösungsmodulen eine Vielzahl an Lösungen gebildet werden, was die Effizienz der Leistungserstellung begünstigt. Sollten Kunden sehr spezielle Anforderungen an die Lösung haben, kann die aus den Bausteinen erstellte Lösung als Ausgangslösung dienen, die noch „von Hand“ angepasst werden kann.

Diese beschriebene Funktionalität wird im Rahmen der eigens entwickelten Software zur Verfügung gestellt. Ihr Einsatz bildet damit insgesamt die Basis für ein wirtschaftliches Management und eine optimale Vermarktung hybrider Produkte. Technisch gesehen handelt es sich bei der Software um ein H2-Toolset-Plug-In, das speziell für die Modellierung, Konfiguration und Kalkulation von hybriden Leistungsbündeln ausgelegt ist. Das Tool wird der Einfachheit halber verkürzt **H2-ServPay** genannt, um den Leser nicht mit softwaretechnischen Begriffen zu verwirren.

1.3 Aufbau des Arbeitsberichts und Projektszenario

Die hier präsentierten Ergebnisse waren und sind Gegenstand einiger wissenschaftlicher Veröffentlichungen, auf die hier lediglich verwiesen werden soll[11]. Ziel des Berichtes ist stattdessen eine praxisnahe Vorstellung des im Projekt ServPay entwickelten Softwarewerkzeugs **H2-ServPay**. Das Tool wird momentan evaluiert und wird nachfolgend auf der Webseite des ServPay Projektes (www.servpay.de) kostenlos zum Download bereitgestellt.

Zunächst werden in Kapitel 2 Ansätze zur Modellierung von hybriden Leistungsbündeln und die beiden Kernfunktionalitäten von **H2-ServPay**, Kostenkalkulation und Produktkonfiguration, vorgestellt. Im Sinne des üblichen Vorgehens bei Softwareentwicklungsprojekten schließt sich dem die Fachkonzeption des Softwaretools an. Das Fachkonzept in Kapitel 3 enthält die Vorstellung von Sprachkonstrukten zur Modellierung hybrider Leistungsbündel mit **H2-ServPay** aus Anbieter- und Kundensicht – bzw. anders ausgedrückt – aus Typ- oder Instanzsicht. Dieses Kapitel dient der Dokumentation und Erläuterung der Modellierungssprache und ist primär an interessierte Fachexperten gerichtet. In Kapitel 4 wird anschließend umfassend die Vorgehensweise bei der Modellierung von hybriden Leistungsbündeln anhand eines Praxisszenarios erläutert und in einem Vorgehensmodell zusammengefasst. Auch die Benutzung des Leistungskonfigurations- und Kostenkalkulationstools wird detailliert beschrieben. Ziel ist es, Praktikern einen Eindruck von der Funktionalität des Tools zu geben und gleichsam als eine Art „Anleitung“ zu fungieren, um das Tool von Unternehmen selbst-

[10] Die Ermittlung von Zahlungsbereitschaften mithilfe von Conjoint Analysen wird momentan noch ungesetzt und wird daher noch nicht in diesem Arbeitsbericht vorgestellt.

[11] Die Beiträge wurden auf angesehenen Konferenzen vorgestellt und teilweise als bestes Paper ausgezeichnet, Becker et al. (2008a) und (2008b). Sie stehen auf der Projektwebseite zum Download zur Verfügung.

ständig einsetzbar zu machen. Nachdem in Kapitel 5 kurz auf die Implementierung eingegangen wird und die erstellten Klassen vorgestellt werden, schließt dieser Arbeitsbericht mit einer Zusammenfassung der Funktionalitäten von **H2-ServPay** sowie der Einordnung des Tools in diverse Anwendungskontexte. Erweiterungen der Funktionalität (bspw. durch ein Conjoint-Modul zur Messung der Zahlungsbereitschaft von Kunden) werden in späteren Arbeitsberichten und wissenschaftlichen Veröffentlichungen vorgestellt werden.

Projektszenario

Um den Zusammenhang zwischen Sach-, Dienstleistungen sowie hybriden Leistungsbündeln und die daraus entstehenden Anforderungen für das Tool besser zu verdeutlichen, soll im Folgenden eine kurze Einführung in das dem Projekt zugrunde liegende, fiktive Szenario gegeben werden.

Das Szenario findet seinen Ursprung im Bereich der industriellen Fertigungstechnik. Im gesamten Projekt wird dazu eine Fräsmaschinenbaureihe zur Bearbeitung von Werkstoffen betrachtet. Die Fräsmaschinenbaureihe mit dem Namen „T-Serie“ beinhaltet drei verschiedene Fräsmaschinen (T-800, T-1000 und T-2000) die sich untereinander in Größe, Ausstattungsmerkmalen und weiteren Details, die an dieser Stelle nicht näher erläutert werden sollen, unterscheiden. Diese Fräsmaschinen stellen die zentrale Leistung mit hohem Sachleistungsanteil in dem konfigurierbaren hybriden Leistungsbündel dar. Weiterhin verfügt die Baureihe über verschiedene optionale Leistungen, die in einer Reihe von Konfigurationsschritten ergänzt werden können. Zu diesen gehören beispielsweise verschiedene Kühlmittelanlagen, Werkzeugwechsler oder Palettenoptionen mit denen eine Fräsmaschine den genauen Vorstellungen eines Kunden angepasst werden kann. Leistungen mit hohem Dienstleistungsanteil innerhalb dieses hybriden Leistungsbündels sollen in diesem Szenario durch Wartungsservice, Finanzierung, Schulung und Beratung repräsentiert werden. Kunden haben beispielsweise die Möglichkeit, eine Fräsmaschine zusammen mit einem Wartungsservice als Bündel zu erwerben. Zusammen ermöglichen all diese Leistungen die Konfiguration eines hybriden Leistungsbündels.

Das zugrunde gelegte Szenario ist frei erfunden und spiegelt nicht die tatsächliche Komplexität eines „realen“ Szenarios wieder. Vielmehr soll es zur Veranschaulichung der Funktionalität von **H2-ServPay** dienen. Die Software kann jedoch zur Modellierung, Konfiguration und Kalkulation beliebig komplexer hybrider Leistungsbündel eingesetzt werden.

2 Projektspezifische Ansätze zur Modellierung hybrider Leistungsbündel

Im Rahmen des Projektseminars zur Entwicklung von **H2-ServPay** wurde die Modellierung beliebiger Varianten und somit Unterstützung des Komplexitätsmanagements hybrider Leistungsbündel forciert. Zwecks Erfüllung dieser Anforderungen wurde das am Lehrstuhl für Wirtschaftsinformatik und Informationsmanagement der Westfälischen Wilhelms-Universität Münster entwickelte Meta-Modellierungswerkzeug H2-Toolset-Solution eingesetzt. Dies ist ein Werkzeug, in dem beliebige Modellierungssprachen definiert werden können. Neben einigen technischen Weiterentwicklungen wurde vor das H2-Toolset vor allem dazu genutzt, eine (Meta-)Sprache zur Modellierung hybrider Leistungsbündel zu entwerfen. Damit wird – ganz ähnlich zu Modellen, die in der Produktentwicklung eingesetzt werden – u.a. die systematische Beschreibung hybrider Leistungsbündel ermöglicht.

Ziel ist es, eine möglichst domänenübergreifend anwendbare Metasprache zu entwickeln, die aber speziell auf die Modellierung industrieller Maschinen und entsprechender produktbegleitender Dienstleistungen abzielt. Da hybride Leistungsbündel aus Sachleistungen und Dienstleistungen bestehen, erfordert der Entwurf dieser Sprache die integrierte Berücksichtigung von Anforderungen der Produktentwicklung (Product Engineering) und der systematischen Dienstleistungsentwicklung (Service Engineering). Dabei werden insbesondere kundenindividuelle Konfigurationsmöglichkeiten berücksichtigt und auch die Bildung verschiedener Kennzahlen ermöglicht. Durch einen Vergleich der Kennzahlen verschiedener konfigurierter Leistungsbündel kann also ein aus Kunden- und Anbietersicht optimales hybrides Leistungsbündel ausgewählt werden. Somit leistet das Tool einen Beitrag zur Konstruktion und Auswahl einer optimalen Kundenlösung und kann die Abstimmung zwischen Anbietern und Kunden fundieren und ermöglichen.

Das Modellierungswerkzeug H2-Toolset-Solution wurde als Ausganglösung für die Entwicklung von **H2-ServPay** weiterentwickelt. Das H2-Toolset ermöglicht sowohl die Erstellung neuer hierarchischer Modellierungssprachen als auch die Modifikation bereits existierender Sprachen.[12] Das Gesamtkonzept der Funktionalität von **H2-ServPay** findet sich in Abb. 2.1. Die Komponenten werden nachfolgend detaillierter vorgestellt.

[12] vgl. Becker, Janiesch, Knackstedt et al. (2006), 109, S. 9.

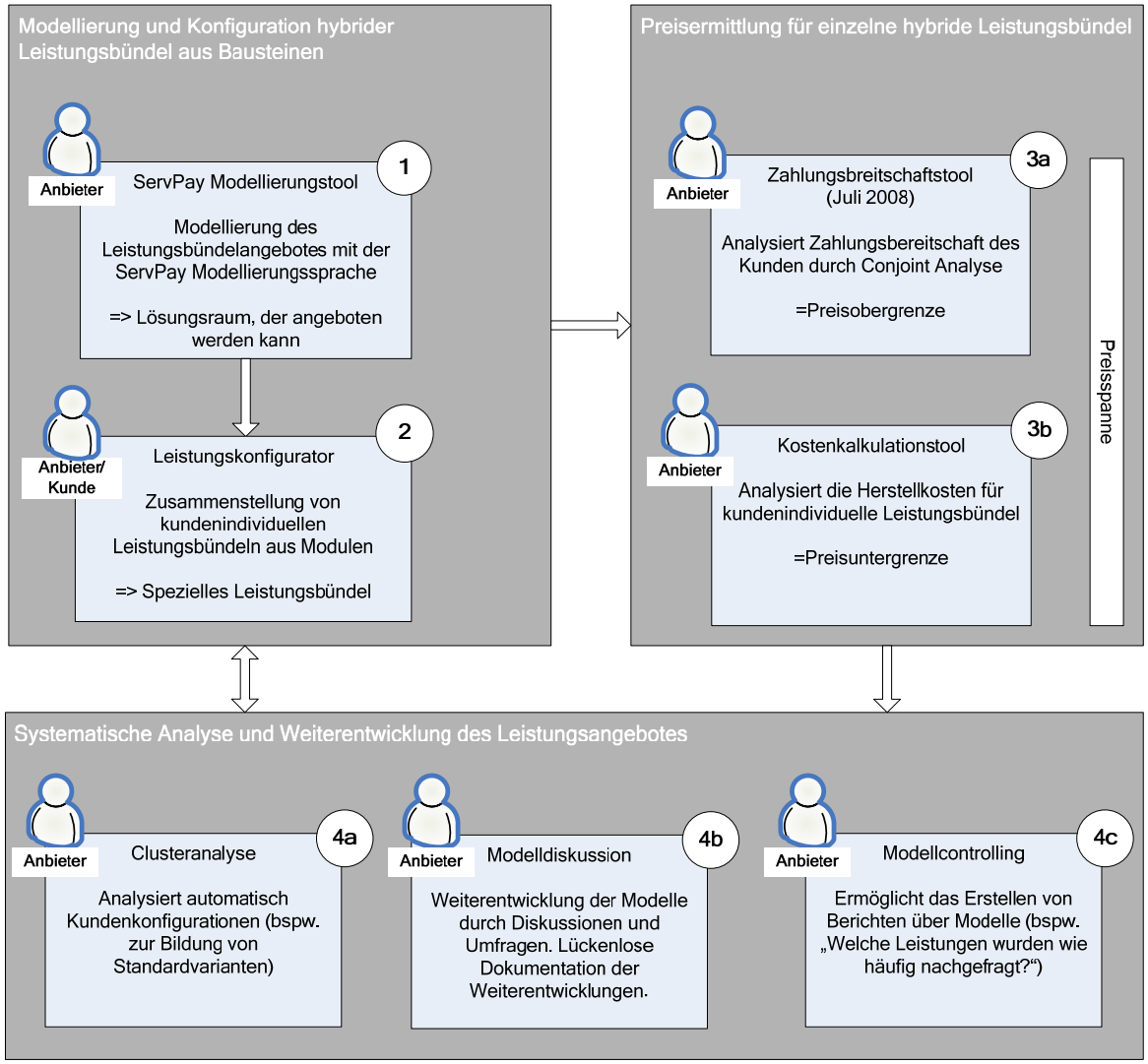


Abb. 2.1: Überblick der Gesamtfunktionalität von H2-ServPay

2.1 ServPay Modellierungstool (1)

Kunden wollen durch den Kauf eines hybriden Leistungsbündels die Lösung eines bestimmten Problems erreichen. Diese Lösung kann sehr kundenindividuell sein und ist in diesem Fall von Anbietern schwer wirtschaftlich zu erbringen. Eine Möglichkeit ist es, Kundenlösungen aus wieder verwendbaren Modulen zu konfigurieren, die als Ausgangspunkt zur Lösung des speziellen Kundenproblems dienen können. Durch die Wiederverwendung von Modulen kann der Anbieter somit viele verschiedene Kundenwünsche effizienter bedienen. Daher ist es eine besondere Herausforderung bei der Entwicklung der Sprache für hybride Leistungsbündel (im folgenden HLB genannt), die Kundenperspektive und die Anbieterperspektive miteinander zu verbinden, obwohl eine unterschiedliche Sicht auf hybride Leistungsbündel besteht.

Anbieter definieren Kombinationsregeln für ihre Lösungsmodule. Damit wird ein Möglichkeitsraum ausgespannt, in dem alle erlaubten Konfigurationen (Kombination aus Modulen) verortet werden können. Die Konfiguration kann aufgrund der Unterstützung verteilter Modellierung und webbasierter Leistungskonfiguration räumliche unabhängig erfolgen. Eine weitere Besonderheit des hier vorge-

stellten Ansatzes ist die Ableitung ökonomischer Konsequenzen aus den zugrundeliegenden Modellen, wie sie z.B. von VOM BROCKE für die Auswahl von Konfigurationen Serviceorientierter Architekturen (SOA) vorgestellt wurde.[13] Um die zuvor genannten Anforderungen zu erfüllen, wurden neben der HLB-Sprache ein webbasierter Leistungskonfigurator sowie ein Softwaretool zur Kalkulation der beim Anbieter anfallenden Kosten für die Erstellung hybrider Leistungsbündel entwickelt.

Anforderungen an die Modellierungssprache ergeben sich aus den verschiedenen Anwendungskontexten, in denen die Modellierungssprache eingesetzt wird. Auch für die Modellierung hybrider Leistungsbündel lassen sich verschiedene Anforderungen zusammenstellen. Eine Gliederung der vielfältigen Anforderungen legt die für die Beschreibung von Dienstleistungen verbreitete Unterscheidung der Sichten Potential, Prozess und Ergebnis nahe. Aus Potentialsicht ist z. B. zu fordern, dass sowohl die für die Dienstleistungs- als auch für die Sachleistungserstellung vorzuhaltenden Ressourcen und ihre Kapazitäten modelliert und aufeinander abgestimmt werden können. Hinsichtlich der Prozesssicht sind die geplanten Abläufe der Dienstleistungs- und Sachleistungserbringung darzustellen und miteinander zu integrieren. Hierbei ist insbesondere den Erfordernissen zur Integration des Kunden in den Dienstleistungsprozess geeignet Rechnung zu tragen. Aus Ergebnissicht ist zu fordern, dass die Ergebnisbeiträge für den Kunden modelliert werden können. Kombinationsregeln für Sach- und Dienstleistungsmodule stellen eine Besonderheit der Modellierung hybrider Leistungsbündel dar.[14]

2.2 Leistungskonfigurator (2)

Die Produktkonfiguration beschreibt das Zusammensetzen eines Produktes aus vorgegebenen Produktkomponenten. Gerade im Internet können Produktkonfiguratoren als entscheidender Wettbewerbsvorteil besonders von produzierenden Unternehmen angesehen werden. Durch die Vorteile eines Produktkonfigurators, wie z.B. fehlerfreie Angebote, kurze Reaktionszeiten, Minimierung der Reklamationsrate, Steigerung der Kundenzufriedenheit, Unterstützung des Vertriebs, sind Produktivitätssteigerungen im gesamten Unternehmen zu erwarten.

Die kundenseitige Konfiguration der Leistungen wird im Rahmen des Projektes **H2-ServPay** mit Hilfe eines webbasierten Leistungskonfigurators durchgeführt. Anhand von vorgefertigten hybriden Leistungsbündeln, welche der Anbieter vordefiniert, können Kunden für sich relevante Leistungen nach sprachlich vorgegebenen Konfigurationsparametern gestalten. Hierbei wird einerseits die Konfiguration für in der Datenbasis bereits bekannte Kunden anhand vorgegebener Präferenzen, als auch andererseits die Konfiguration eines Modells für unbekannte Kunden, die über einen Gastzugang das System nutzen, ermöglicht.

Beim Aufruf des Leistungskonfigurators über einen Webbrowser (z.B. Internet Explorer oder Firefox) wird zunächst eine Anmeldemaske angezeigt. Der Kunde hat hier die Möglichkeit sich gegebenen-

[13] vgl. vom Brocke (2006)
[14] vgl. Becker, Knackstedt, Beverungen et al. (2008)

falls mit seinem bereits existierenden Benutzerzugang anzumelden, was dazu führt, dass seine bereits erfassten Präferenzen automatisch geladen werden oder er kann sich als neuer Kunde registrieren. Als dritte und weitere Möglichkeit kann auch ein Gastzugang ausgewählt werden, so dass der Kunde die Registrierung umgehen kann, hierdurch jedoch in Kauf nimmt, dass seine Daten nicht gespeichert werden. Nach der ersten Anmeldung eines Kunden hat dieser die Möglichkeit, seine Präferenzen für z.B. die Lieferzeit der zu konfigurierenden Maschine, des Preises oder etwa der mit der Maschine einhergehenden Bedienkomplexität anzugeben. Nach Angabe der Wichtigkeit der Präferenzen bekommt der Kunde im nächsten Schritt zusätzlich die Möglichkeit, den ausgewählten Präferenzen genauere Präferenzwerte in Form von Gewichten zuzuordnen. Nach der Erfassung der Präferenzen und Ermittlung einer detaillierteren Bewertung der Benutzereingaben werden die zur Verfügung stehenden konfigurierbaren Leistungen kundenspezifisch nach Präferenzangaben geordnet.

Zusätzlich zu den vorgegebenen Konfigurationsoptionen erhält der Kunde die Möglichkeit, eventuelle Sonderwünsche anzugeben, die daraufhin auf Anbieterseite ausgewertet werden können, um ein kundenindividuelles Angebot zu erstellen. Des Weiteren wird der Kunde während des Konfigurationsprozesses durch ein sogenanntes Recommender-System unterstützt, welches Vorschläge für mögliche Leistungskombinationen unterbreitet. Im Falle, dass ein Kunde bezüglich der zu konfigurierenden Leistungen indifferent ist, kann er außerdem die Möglichkeit nutzen, die Konfiguration automatisiert anhand seiner zuvor bestimmten Präferenzen durchführen zu lassen.

Auch die monetären Konsequenzen einer Konfiguration werden direkt im Leistungskonfigurator visualisiert. Der Kunde erhält anhand seiner aus den zur Verfügung stehenden Modulen spezifizierten Leistungen eine tabellarische Auflistung der entstehenden Zahlungsfolgen im Lebenszyklus. Die anfallenden Kosten können bei Bedarf in einen vollständigen Finanzplan exportiert werden, anhand dessen die Zahlungsfolge der Investition und die daraus entstehenden Total Cost of Ownership (TCO) einfach abgelesen werden können.

2.3 Zahlungsbereitschaftstool (3a)

Momentan wird ein Tool zur Ermittlung von Zahlungsbereitschaften für hybride Leistungsbündel erstellt, das als sog. Plug-In in die Modellierungsumgebung eingebunden ist. Es handelt sich also nicht um ein zusätzliches Tool, sondern stellt seine Funktionalität im Rahmen der Gesamtarchitektur zur Verfügung. Die Entwicklung des Zahlungsbereitschaftstools wird voraussichtlich im Sommer 2008 abgeschlossen sein.

Durch die systematische Ermittlung der Zahlungsbereitschaft des Kunden für einzelne Eigenschaften hybrider Leistungsbündel mit der HILCA (Hierarchische Individualisierte Limit Conjoint-Analyse) kann eine Preisobergrenze für hybride Leistungsbündel ermittelt werden. Ein Vergleich mit den Kosten des Anbieters kann so den zur Verfügung stehenden Preisspielraum für das hybride Leistungsbündel aufzeigen, der beiden Seiten gerecht wird: Einerseits sollen die Kosten des Anbieters min-

destens gedeckt werden, andererseits ist die Zahlungsbereitschaft des Kunden als Obergrenze des Preisspielraums zu beachten.

2.4 Kostenkalkulationstool (3b)

Um Entscheidungsunterstützung bereits während der Entwurfsphase eines hybriden Leistungsbündels zu gewährleisten, wurde im Rahmen des Projektseminars zur Entwicklung von **H2-ServPay** die H2-Toolset-Solution um ein Plug-In zur Kostenkalkulation von Leistungen ergänzt. Dieses speziell für monetäre Berechnungen entwickelte Softwaretool unterstützt die Anbieter dabei, berichtigte Herstellkosten[15] für diverse Leistungen zu ermitteln. Die Berechnung der Kosten kann gegebenenfalls als Basis für die Preisgestaltung der Produktpalette dienen und lässt sich bis zur kleinsten enthaltenen Leistungseinheit schlüsseln.

Die Einsatzzwecke dieses zusätzlichen Werkzeuges sind vielfältig. Einerseits dient es als Controllinginstrument zur ex post Kontrolle der mit der Leistungserstellung und -erbringung einhergehenden Kosten. Andererseits dient es bereits in der Gestaltung eines hybriden Leistungsbündels der Überwachung der Kosten und Zuordnung der kostenverursachenden Stellen und Ressourcen zu Kostenstellen. Eine Eintragung der Kosten während der Modellierung und spätere Aggregation der Kosten bis auf eine beliebig feingranulare frei zu wählende Ebene verschafft einen effizienten Überblick über potentielle Kostentreiber und gegebenenfalls daraus resultierende Einsparungspotentiale.

Der Einstieg in das Tool erfolgt über die Auswahl einer Leistung aus einem bestimmten Modul und darauf folgende weitere Verfeinerungen der Aufschlüsselung hinterlegter Daten. So können die Kosten und die Dauer für einzelne Lebenszyklusphasen oder gar für Kombinationen unterschiedlicher Lebenszyklusphasen ermittelt werden. Kosten verschiedener Aktivitäten können tabellarisch dargestellt werden, um so z.B. die von einer bestimmten Stelle oder Organisationseinheit verursachten Kosten zu analysieren. Auch Materialkosten, die für bestimmte Ressourcen angefallen sind können inklusive ihrer verbrauchten Mengeneinheiten und Einzelkosten angezeigt werden. Generell bietet dieses Tool eine feingranulare Darstellungsmöglichkeit zur Ermittlung potentieller Kostentreiber und liefert somit Anhaltspunkte zur Optimierung und Ermittlung einer kurzfristigen Preisuntergrenze. Als kurzfristige Preisuntergrenze dienen hierzu die Selbstkosten, die die Summe aller durch den Leistungsprozess eines Betriebes entstandenen Kosten für einen Kostenträger darstellen. Dies sind sowohl Kosten in Form von Material- und Fertigungskosten, Sondereinzelkosten der Fertigung, welche in Form von Zuschlägen festgehalten werden können und die Herstellkosten bezeichnen, als auch eventuell zu erfassende Verwaltungs- Vertriebs- und Entwicklungskosten respektive Kosten der Forschung und Entwicklung.

Die Berechnung der reellen Kosten läuft für den Anwender unsichtbar im Programmcode ab. Um neben den nominalen Kosten auch die Vergleichbarkeit verschiedener Investitionen bzw. hybriden

[15] Herstellkosten im Sinne einer relativen Einzelkostenrechnung

Leistungsbündeln mit unterschiedlichen Laufzeiten und Zinssätzen zu gewährleisten, wird das Barwertkonzept aus der Finanzmathematik angewandt. Ein Zinssatz für eine Leistung kann ebenso wie die Dauer eines Intervalls einer Leistung als Attribut angegeben werden. Die Dauer muss hierbei als numerischer Wert angegeben werden und die Regelmäßigkeit sollte einen der Werte *Tage*, *Monate* oder *Jahre* erhalten. In einem spezifischen Intervall hinterlegte Kosten pro Mengeneinheit einer Ressource oder Stundensätze verschiedener Stellen werden dann nach folgender Formel auf den Barwert abgezinst:

$$\mathrm{PV}(Z_T) = \frac{Z_T}{(1 + r_T)^{\frac{T}{360}}}$$

Abb. 2.2: Gegenwartswert des hybriden Leistungsbündels

Durch diesen Wert, auch Gegenwartswert (*PV* = *Present Value*) genannt, lassen sich die reellen Zahlungen der angelegten Intervalle zum aktuellen Zeitpunkt bestimmen. Dabei wird der Zeitpunkt *T* eines Intervalls bei der Kostenkalkulation durch den Mittelwert des betrachteten Zeitraumes folgendermaßen berechnet:

$$\frac{\text{Startzeitpunkt} + \text{Endzeitpunkt}}{2}$$

Abb. 2.3: Gemittelter Zeitpunkt T eines Intervalls

Neben der direkten Anzeige der Kosten im Modell können die Daten auch aggregiert in tabellarischer Form dargestellt werden. So kann eine Kontrolle der in verschiedenen Lebenszyklusphasen anfallenden Kosten bis in höchste Detailtiefe durchgeführt werden.

2.5 Analysefunktionalität (4a, 4b, 4c)

Abgesehen von der Modellierung, Konfiguration und Preisbestimmung hybrider Leistungsbündel wird Anbietern auch eine vielfältige Analysefunktionalität bereitgestellt, um auf der Basis der gewonnenen Ergebnisse ihr Leistungsangebot zu optimieren. Diese ist bereits jetzt als Plug-Ins realisiert und befindet sich in der Testphase. Somit stehen folgende Komponenten zur Verfügung, die momentan noch erweitert und evaluiert werden und daher in einem späteren Arbeitsbericht vorgestellt werden:

- Clusteranalyse (4a): Erstellte Modelle können nach festzulegenden Kriterien automatisch geclustert werden. Beispielsweise kann eine Clusteranalyse der durch Kunden konfigurierten Leistungsbündel Hinweise darauf geben, welche Standardvarianten besonders nachgefragt wären und daher evtl. angeboten werden sollten. Das Clustering erfolgt dabei halbautomatisch durch das Tool, wird jedoch durch den Benutzer gezielt gesteuert.

- Modelldiskussion (4b): Im Rahmen eines Diskussionsprozesses können die erstellten Modelle mithilfe von Delphi-Studien, Umfragen und Kommentaren gemeinschaftlich weiterent-

wickelt werden. Insbesondere, wenn Module hybrider Leistungsbündel von mehreren Unternehmen oder Geschäftsbereichen zusammengestellt werden, kann so eine systematische und lückenlos dokumentierte Abstimmung des Leistungsangebotes erfolgen. Es ist geplant, dieses Tool auch in Standardisierungsverfahren mit dem DIN zu nutzen (in Standardisierungsverfahren des Forschungsprojekts *FlexNet*).

- Modellcontrolling (4c): Die im Modellierungstool erstellten Modelle hybrider Leistungsbündel können systematisch analysiert werden. Dazu können verschiedene Berichte angelegt werden, die bspw. ermitteln können, welche Leistung wie häufig an den Modellen vorkommt oder wie lange die Erstellung bestimmter Modelle gedauert hat. Angewandt auf die durch Kunden konfigurierten Leistungsbündel kann so festgestellt werden, welche Leistungen besonders häufig nachgefragt werden und welche kaum nachgefragt werden. Dies soll Anbieter bei der in der Praxis häufig fehlenden Optimierung des eigenen Leistungsangebotes unterstützen.

3 Fachkonzept

Aufgrund des großen Umfangs der Sprachdefinition zur Modellierung hybrider Leistungsbündel wird die gesamte Sprachdefinition in diesem Kapitel in einzelne Konstrukte zerlegt[16]. Diese werden zunächst einzeln vorgestellt und später in einem Gesamtzusammenhang dargestellt. Kapitel 3.1 beinhaltet die Teile des Gesamtmodells, die der Anbietersicht auf ein hybrides Leistungsbündel entsprechen. Kapitel 3.2 umfasst die Kundensicht auf ein hybrides Leistungsbündel. Das komplette Datenmodell, sowie die komplette H2-Sprachdefinition befinden sich im Anhang.

3.1 Anbietersicht

3.1.1 Hybride Leistungsbündel auf Typebene

Einstiegspunkt für die Modellierung hybrider Leistungsbündel ist das in Abb. 3.1 visualisierte Konstrukt *Hybrides Leistungsbündel auf Typebene*, im Folgenden als *HLBAT* bezeichnet. In der H2-Toolset-Solution ist der dazugehörige Kontext auf der höchsten Ebene angesiedelt.

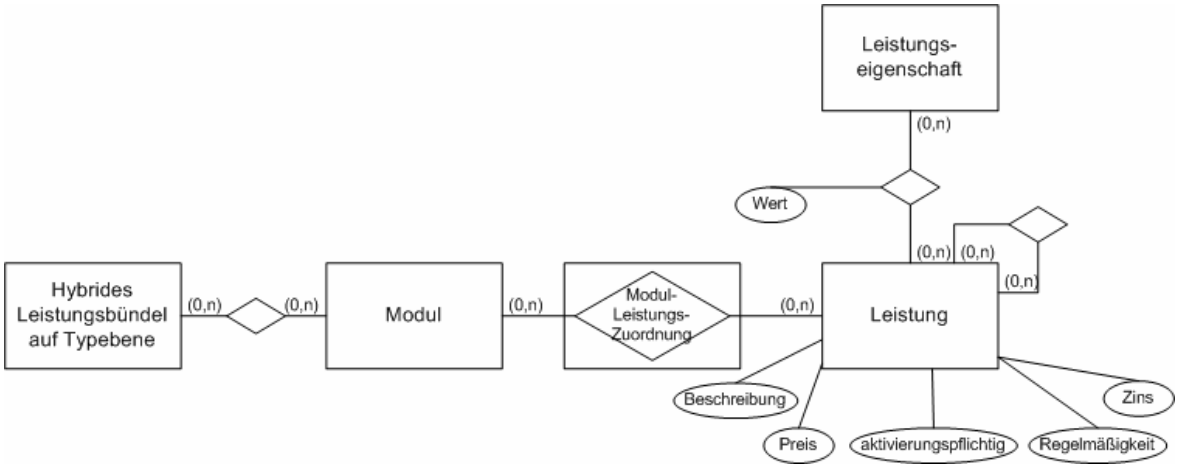


Abb. 3.1: Hybrides Leistungsbündel auf Typebene

Ein *HLBAT* enthält alle möglichen Konfigurationen, d.h. die Menge aller ggf. enthaltenen Leistungen, eines hybriden Leistungsbündels. Grundsätzlich wird ein *HLBAT* durch die in ihm enthaltenen *Module* definiert. *Module* ermöglichen eine Kategorisierung von Leistungen, sowie eine automatische Generierung von Konfigurationsschritten durch den Konfigurator[17].

Jedes *Modul* ist 0 bis n *HLBAT* zugeordnet, um eine spätere Einbindung bereits erstellter Module zu erlauben. Durch die Wiederverwendung bereits erstellter Teilmodelle werden Modelle von Leistungsbündeln einfach und redundanzfrei gehalten. Jedes *Modul* enthält 0 bis n *Leistungen*, wobei

16 Eine ganzheitliche Übersicht der Sprache findet sich in Abschnitt 0 des Anhangs

17 Jedes Objekt vom Typ Modul wird vom Konfigurator beim Laden des H2-Modells als einzelner Konfigurationsschritt erkannt und in den Konfigurationsprozess eingebunden.

jeweils eine Leistung pro Modul vom Kunden im Konfigurationsprozess ausgewählt werden kann[18]. Jede *Leistung* ist 0 bis n *Modulen* zugeordnet. Auch hier ist eine Erstellung ohne eine direkte Zuordnung möglich.

Eine Leistung kann andere Leistungen beinhalten. Deshalb erfolgt die Abbildung einer Leistung in Form einer Struktur (Leistungsstruktur). Ist dies der Fall, stellt die Leistung selbst ein hybrides Leistungsbündel dar. Es wird keine Trennung in Sach- und Dienstleistungen vorgenommen. Diese Entscheidung begründet sich neben den in Kapitel 1 erwähnten Gesichtspunkten wie folgt: Eine Trennung von Sach- und Dienstleistungen würde die Modellierung in der H2-Toolset-Solution wesentlich verkomplizieren, da mindestens ein weiterer Kontext benötigt werden würde. Dies würde auch das Einlesen der hybriden Leistungsbündel auf Typebene im Konfigurator erschweren. Diesen Nachteilen gegenüber steht die Möglichkeit, sach- und dienstleistungsspezifische Eigenschaften im Referenzmodell zu spezifizieren. Da das Referenzmodell möglichst generisch gehalten werden soll, wäre dafür eine komplette Erfassung aller ggf. relevanten Eigenschaften für jede denkbare Leistung notwendig. Dies ist nicht realistisch und würde den Umfang der Sprache in der H2-Toolset-Solution aufgrund der fehlenden Referenzmodellierungsfunktionalität in den meisten Fällen auch ineffizient groß gestalten.

Im Normalfall erfolgt die Beschreibung von Objekten in der H2-Toolset-Solution über Attribute. Die Beschreibung von Leistungen erfolgt jedoch über die Zuweisung von 0 bis n *Leistungseigenschaften* zu jeder *Leistung*. Diese ungewöhnliche Modellierung von Attributen einer Leistung ergibt sich aus der Struktur der H2-Toolset-Solution, welche eine Definition von Attributen eines Objektes auf Modellebene nicht erlaubt. Die in der H2-Toolset-Solution im Kontext *Leistung* angelegten *Leistungseigenschaften* sind keine Referenzen auf eine Leistungseigenschaft, sondern Instanzen eines im Kontext *Leistungseigenschaft* definierten Objekttyps. Im Referenzmodell sind für eine Leistung die Attribute Beschreibung, Preis, Zins, Regelmäßigkeit und aktivierungspflichtig vorgesehen. Das Attribut Beschreibung ermöglicht eine Detaillierung einer Leistung im Fließtext. Dieser Fließtext wird im Konfigurator angezeigt und kann somit z.B. der Bewerbung von Leistungen dienen. Der Preis ist für die Berechnung des Gesamtpreises eines hybriden Leistungsbündels auf Instanzebene im Konfigurator notwendig. Das Attribut Zins dient der Kalkulation innerhalb des anbieterseitig eingesetzten Tools zur Kostenkalkulation. Ein Zinssatz wird für eine komplette Leistung und nicht für einzelne Lebenszyklusphasen oder Intervalle definiert.

Abb. 3.2 zeigt die Kontexte für hybride Leistungsbündel auf Typebene, Module und Leistungen in der H2-Toolset-Solution. Die in den Kontexten enthaltenen Konstrukte weisen verschiedene Schriftfarben auf. Blaue Konstrukte sind frei definierbar, lila Konstrukte sind *explicit* und schwarze Kon-

[18] Aus diesem Sachverhalt und der Speicherung der vom Kunden konfigurierten Modelle als Instanzen des Entitytyps *hybride Leistungsbündel auf Instanzebene* bedingt sich die Uminterpretation des Relationshiptyps „Modul-Leistungs-Zuordnung“in einen Entitytyp. Dies wird in Abschnitt 3.2.1 detailliert erklärt.

strukte stammen aus einem anderen Kontext. Dieser Zusammenhang wird im Arbeitsbericht Nr. 115 „H2 for Reporting“ näher erläutert.

Der Kontext hybrides Leistungsbündel auf Typebene umfasst das hybride Leistungsbündel auf Typebene mit den untergeordneten Bestandteilen Modul, Lebenszyklusphasen und Regeln. Die Lebenszyklusphase wird in Kapitel 3.1.2 und das Regelkonstrukt in Kapitel 3.1.3 detailliert erläutert. Der zweite Kontext umfasst die Definition des Moduls. Einem Modul sind dabei wie oben erläutert, Leistungen untergeordnet. Der dritte Kontext mit der Sprachdefinition einer Leistung zeigt die enthaltenen Konstrukte auf. Leistungen enthalten Lebenszyklusphasen, Leistungen, Leistungseigenschaften und Präferenzen. Das Konstrukt der Präferenzen wird in Kapitel 3.2.3 erläutert.

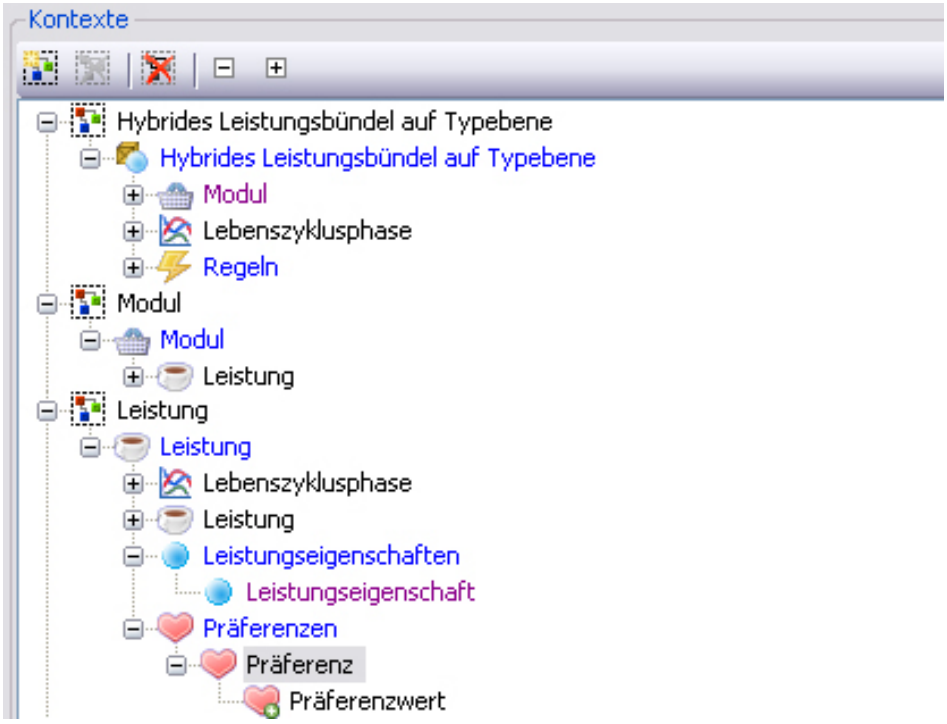


Abb. 3.2: H2-Sprachdefinition Hybrides Leistungsbündel auf Typebene

3.1.2 Lebenszyklusphasen

Zur Modellierung der Lebensdauer einer Leistung aus Anbietersicht dient das Konstrukt Lebenszyklusphase. Dazu wird die Lebensdauer einer Leistung in verschiedene Phasen unterteilt. Der zugehörige Ausschnitt aus dem Gesamtmodell ist in Abb. 3.3 dargestellt und wird im Folgenden erläutert.

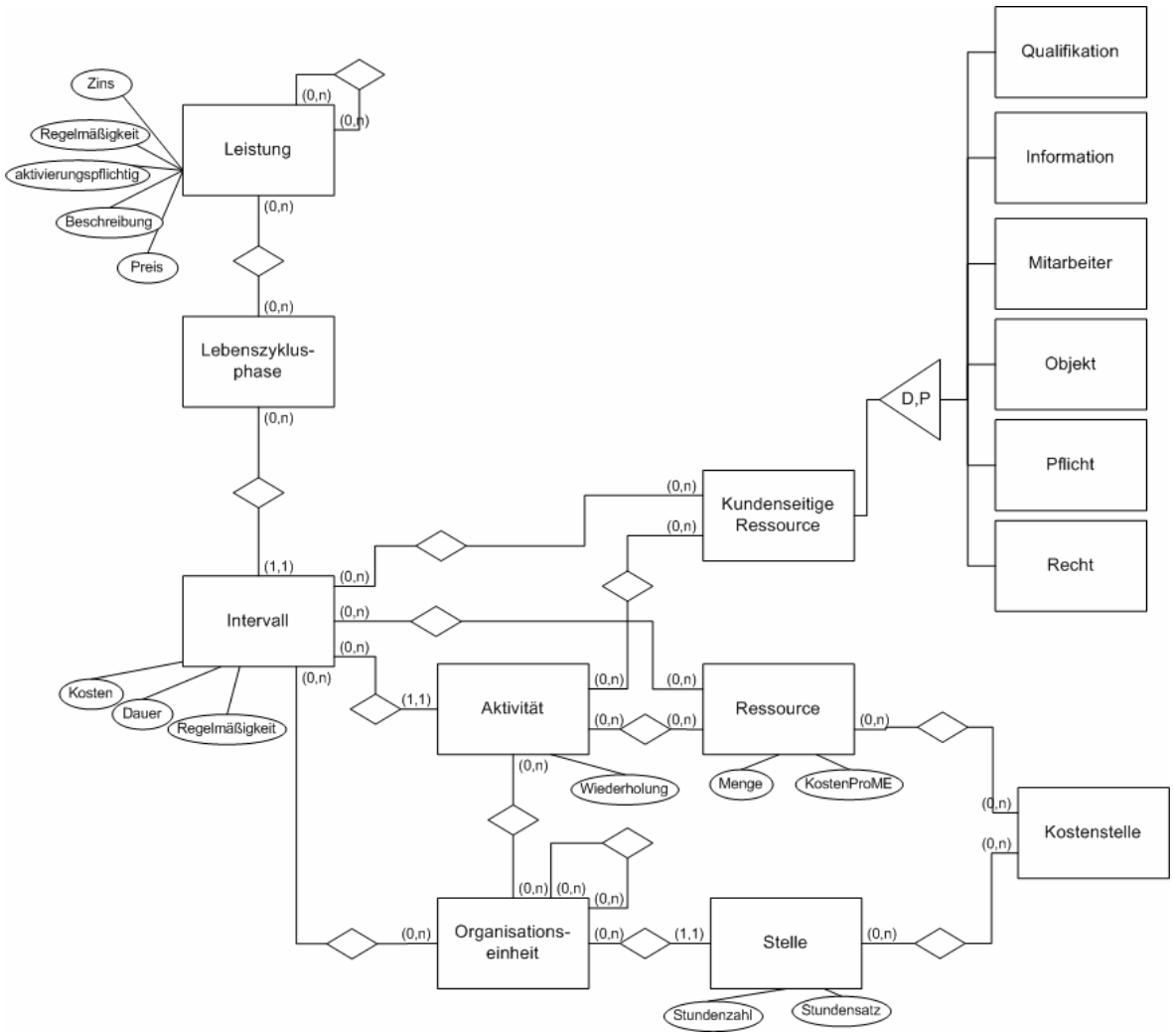


Abb. 3.3: Lebenszyklusphasen (Anbietersicht)

Jeder *Leistung* sind 0 bis n *Lebenszyklusphasen* zugeordnet. Im Konfigurator und Kostenkalkulationstool werden beispielhaft folgende Phasen verwendet: Entwurf, Fertigung, Kauf, Nutzung und Nachnutzung. Jedes *Intervall gehört genau einer Lebenszyklusphase an. Jeder Lebenszyklusphase* kann eine oder mehreren *Leistung(en) zugeordnet sein*. Das Intervall bildet den zeitlichen Rahmen einer Lebenszyklusphase ab. Zu diesem Zweck existieren die Attribute: *Kosten*, *Dauer* und *Regelmäßigkeit*. Die *Regelmäßigkeit* eines Intervalls gibt die Zeiteinheit an, in der das Intervall modelliert wird. Als Wertebereich sind hier Tage, Monate und Jahre vorgegeben. Die *Dauer* gibt die Anzahl der über die *Regelmäßigkeit* definierten Zeiteinheiten als numerischen Wert an. Ein Intervall mit der Zeitspanne von einem Jahr könnte z.B. mit den Attributpaaren (*Regelmäßigkeit*: Jahre, *Dauer*: 1), (*Regelmäßigkeit*: Monate, *Dauer*: 12) oder (*Regelmäßigkeit*: Tage, *Dauer*: 360) modelliert werden.

Das Attribut *Kosten* bildet anbieterseitig anfallende Zahlungen ab. Sie werden modelliert, wenn sich die *Kosten* für eine Leistung nicht aus den modellierten Aktivitäten, Ressourcen und Organisationseinheiten berechnen lassen. So dient dieses Attribut z.B. der Verrechnung von Gemeinkosten oder kann genutzt werden, falls eine detaillierte Modellierung nicht gewünscht wird. *Kosten* werden im Konfigurator nicht angezeigt, sondern nur im anbieterseitig eingesetzten Kostenkalkulationstool.

Ein *Intervall* kann *Ressourcen, Kundenseitige Ressourcen, Aktivitäten* und *Organisationseinheiten* beinhalten. Die *Aktivität* beschreibt eine im Intervall ausgeführte Tätigkeit und hat das Attribut *Wiederholung*, welches angibt, wie oft diese durchgeführt wird. Über *Aktivitäten* werden die einzelnen Schritte eines Arbeitsablaufs dargestellt. Diese können beispielsweise Prototypentwurf, Prototypentwicklung und Prototypfertigung innerhalb eines Intervalls sein, in dem ein Prototyp der T-800-Fräsmaschine entwickelt wird. Die Aktivität wird von mindestens einer Organisationseinheit ausgeführt. Bei der Modellierung von Intervallen ist es möglich auf Aktivitäten zu verzichten, wenn beispielsweise keine verfeinerte Darstellung nötig ist. Dies wird ermöglicht, indem sich die Organisationseinheiten und Ressourcen auch direkt dem Intervall zuordnen lassen.

Ressourcen werden bei der Durchführung einer Aktivität verbraucht oder gebraucht. Verbrauchsgüter sind Ressourcen wie beispielsweise Material (Schrauben) die im Leistungserstellungsprozess untergehen. Gebrauchsgüter sind Ressourcen wie beispielsweise Informationen (Berichte, Auskünfte) oder Maschinen die in den Leistungserstellungsprozess eingehen. Damit die Kosten für Ressourcen berücksichtigt werden können, existieren die Attribute *Menge* und *Kosten* pro Mengeneinheit. Im Gegensatz zur *Aktivität* sind *Ressourcen* globale Objekte. Sie können in mehreren *Intervallen* vorhanden sein.

Kundenseitige Ressourcen enthalten Ressourcen die vom Kunden selbst in den Erstellungsprozess eines hybriden Leistungsbündels eingebracht werden können oder gar müssen. Dies können beispielsweise sein: Qualifikationen, Informationen, Patente, Mitarbeiter, Objekte, Pflichten und/oder Rechte.

Die *Organisationseinheit* kann mehrere *Stellen* gruppieren und jede *Stelle* ist genau einer *Organisationseinheit* zugeordnet. Eine Organisationseinheit kann andere Organisationseinheiten enthalten. Um dies zu berücksichtigen wird auf das Konstrukt der (Organisations-)Struktur zurückgegriffen. Die Attribute *Stundenzahl* und *Stundensatz* an der Stelle ermöglichen eine Erfassung der durch die Stelle verursachten Kosten.

Es ist zu berücksichtigen, dass die Attribute *Menge* und *Stundenanzahl* im Kontext *Leistung* jeweils leistungsbezogen eingegeben werden, Kosten pro Mengeneinheit (KostenproME) und Stundensatz jedoch nur einmal im Kontext *Ressource* bzw. *Organisationseinheit*. Weiterhin ist zu beachten, dass *Aktivitäten*, genau wie das *Intervall* identische Bezeichnungen haben können, es sich aber um unterschiedliche Objekte handelt. Dies begründet sich in der Struktur der H2-Toolset-Solution.

Die H2-Sprachdefinition der Lebenszyklusphase im Kontext Leistung ist in Abb. 3.4 dargestellt. Unter dem Kontext Lebenszyklusphase sind die enthaltenen Konstrukte hierarchisch geordnet. Ein Intervall enthält in der nächsten Ebene Organisationseinheiten, Aktivitäten, Ressourcen und Kundenseitige Ressourcen. Organisationseinheit, Ressource und Kundenseitige Ressource sind wiederum der Aktivität untergeordnet. Somit hat der Modellierer die oben erläuterte Möglichkeit Organisationseinheiten, Ressourcen und Kundenseitige Ressourcen direkt in ein Intervall zu modellieren, um so auf eine detaillierte Modellierung verzichten zu können.

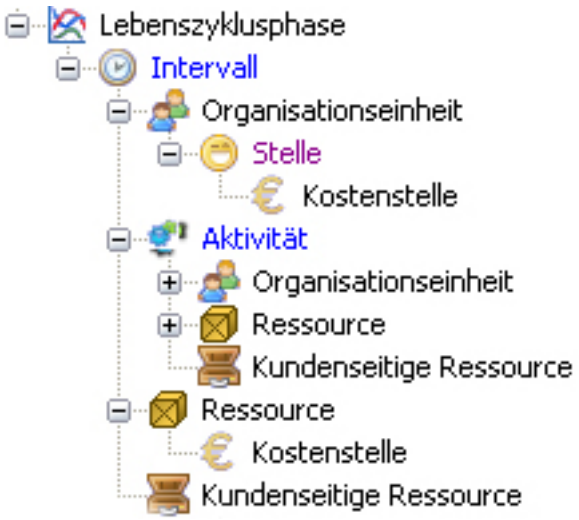


Abb. 3.4: H2-Sprachdefinition Lebenszyklusphase

3.1.3 Regelkonstrukte

Für die Modellierung von Regeln in der H2-Toolset-Solution stehen, aufgrund der im fachkonzeptionellen Referenzmodell formulierten Vorgaben, zwei sich gegenseitig ausschließende Regelkonstrukte zur Verfügung. Bei diesen handelt es sich um die in Abb. 3.5 dargestellten Regeln, die Schließt-Aus-Regel und die Bedingt-Regel.

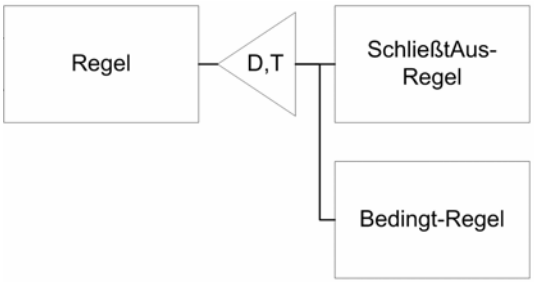


Abb. 3.5: Regelkonstrukte

Eine *Regel* steht relational mit einem *Auslöser* und einer *Konsequenz* in Beziehung. Jede Regel hat genau einen *Auslöser* und eine *Konsequenz*. Eine bestimmte *Leistung* oder ein bestimmtes *Modul* können sowohl *Auslöser* als auch *Konsequenz* einer Regelanwendung sein. Dabei werden mehrere Objekte jeweils durch den UND-Operator verknüpft. Wird ein Modul als Auslöser gewählt wird zusätzlich zu einer im Modul enthaltenen Leistung eine Leistungseigenschaft benötigt. Außerdem müssen zusätzlich die Attribute *Wert*, *Operator* spezifiziert werden. Wird eine Leistung zugeordnet, wird die Zuweisung der Attribute nicht benötigt, da sich die Regel nicht auf eine Leistungseigenschaft, sondern auf die Auswahl der Leistung selbst bezieht. Ein Ausschluss einer Leistung in einem Auslöser kann über den NICHT-Operator abgebildet werden. Dieser Sachverhalt ist zur Verdeutlichung in Abb. 3.6 dargestellt.

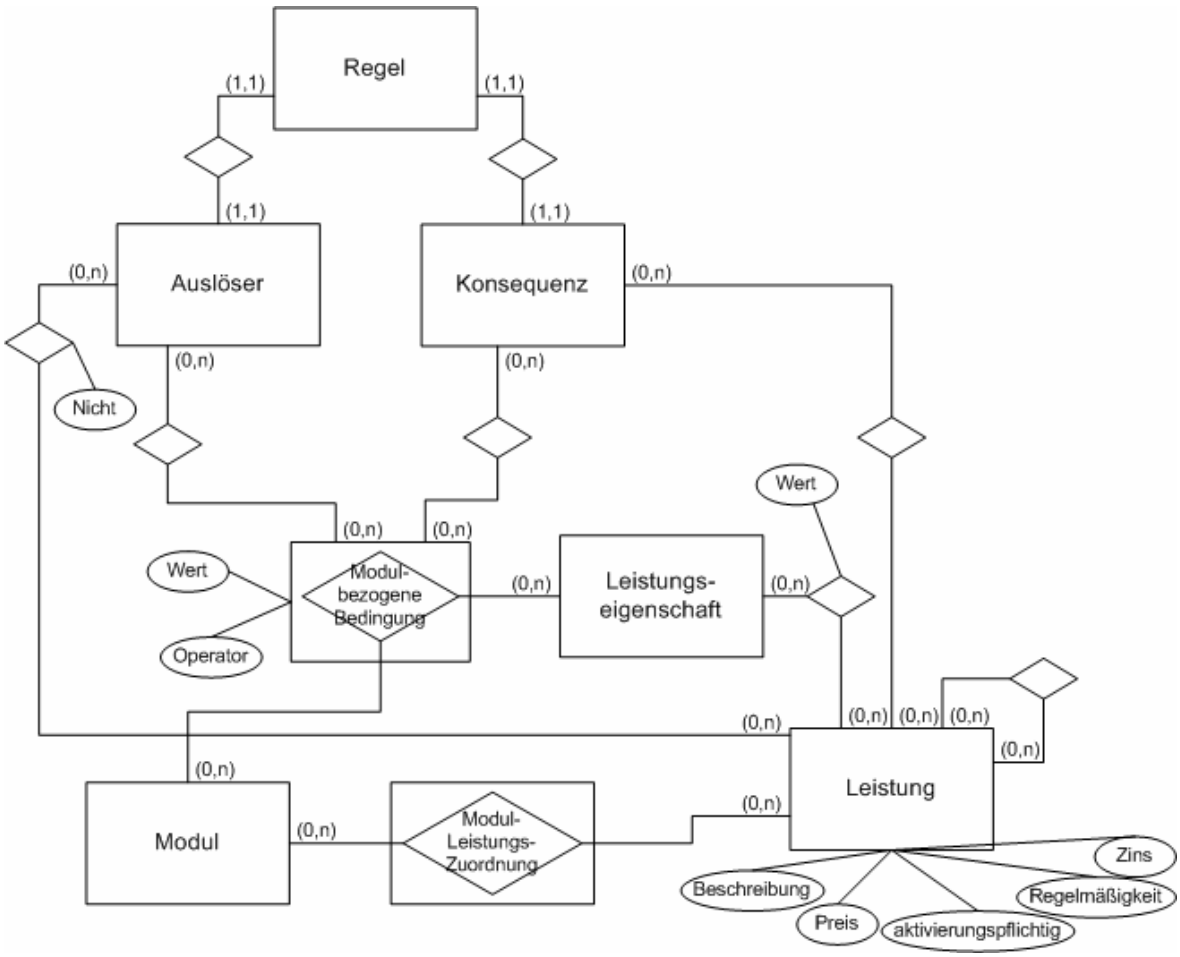


Abb. 3.6: Regelkonstrukte Auslöser-Konsequenz

Abb. 3.7 stellt das Regelkonstrukt als H2-Sprachdefinition dar. Regeln werden in einem Regelcontainer abgelegt. Dieser dient im HLBAT dazu, die Übersichtlichkeit zu erhalten. Auslöser und Konsequenz einer Regel befinden sich immer direkt eine Ebene tiefer als die Regel selbst. Wie oben erläutert wurde, können sowohl ein Modul als auch eine Leistung als Auslöser und Konsequenz eingesetzt werden. Wird ein Modul als Auslöser oder Konsequenz verwendet, so ist neben der Leistung auch eine Leistungseigenschaft anzugeben.

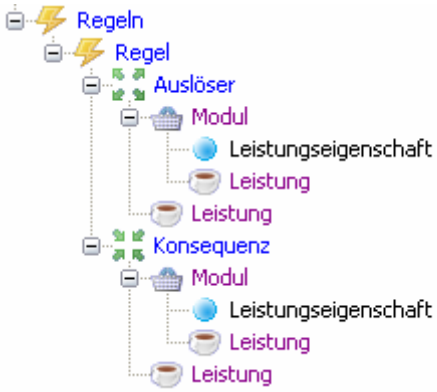


Abb. 3.7: H2-Sprachdefinition Regelkonstrukt

Als Operatoren können die in Abb. 3.8 dargestellten Vergleichsoperatoren genutzt werden.

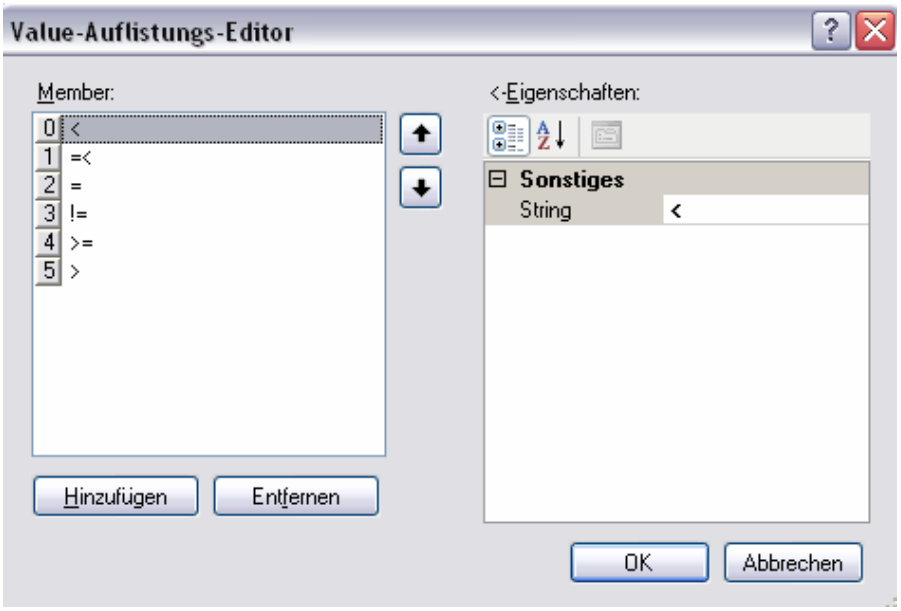


Abb. 3.8: Value-Auflistung des Attributs Operator

Erstellte Regeln werden nicht während der Modellierung in der H2-Toolset-Solution geprüft und durchgesetzt, sondern erst im Laufe des Konfigurationsprozesses im Konfigurator, um somit nur zulässige Konfigurationen zu erlauben. Der Typ einer Regel sowie der NICHT-Operator für eine Leistung werden über die Zuweisung von entsprechenden Kantentypen modelliert.

3.2 Kundensicht

3.2.1 Hybrides Leistungsbündel auf Instanzebene

Das hybride Leistungsbündel auf Instanzebene (HLBAI) ist ein Konstrukt, das ermöglicht vom Kunden erstellte Konfigurationen, die auf einem HLBAT basieren, zu speichern. Das dient zum einen dem Kunden in der Hinsicht, dass einmal erstellte Konfigurationen geladen werden können und nicht erneut konfiguriert werden müssen. Zum Anderen ist der Anbieter dadurch in der Lage, Analysen auf den gesammelten Daten durchzuführen. Letzteres ist gleichzeitig eine Vorraussetzung um ein Recommender-System mit gemeinschaftlichem Filtern realisieren zu können.

Die Daten im HLBAI sind eigenständig und stehen nicht in direkter Verbindung mit den restlichen Daten des HLBAT. Es gibt also keine DefiningContext- oder Explicit-Angaben bei den Objekten des HLBAI in der H2-Sprachdefinition. Dadurch wird gewährleistet, dass einmal konfigurierte Produkte nicht geändert werden, wenn Änderungen im HLBAT vorgenommen werden. Somit ist es möglich historische Konfigurationen der Kunden vorzuhalten, die so nicht mehr konfiguriert werden können (Abb. 3.9).

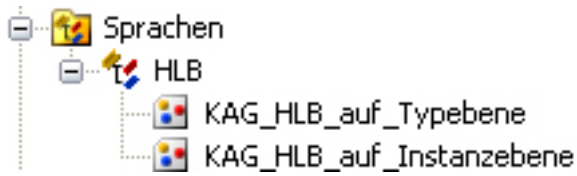


Abb. 3.9: H2 Modelle für Typ- und Instanzebene

Im Modell KAG_HLB_auf_Instanzebene können nun die Konfigurationen im Kontext HLBAI untergebracht werden. Neben dem HLBAI-Kontext und dem Assoziationsregelkontext erfolgt keine weitere Betrachtung der anderen Kontexte. Die Möglichkeit das Modell auf zwei einzelne Modelle zu verteilen dient der Strukturierung.

Für jeden Kunden wird im Kontext HLBAI ein Objekt angelegt, unter dem mehre Konfigurationen (HLBAI) gespeichert werden können (siehe Abb. 3.10). Die Speicherung geschieht automatisiert. Der Anbieter braucht diesen Kontext nur zu betrachten und muss selber keine Modellierung darin vornehmen. Für jedes HLBAI wird dann eine Kombination aus Leistung und Modul gespeichert, um festzuhalten welche Leistung in dem Modul vom Kunden ausgewählt worden ist.

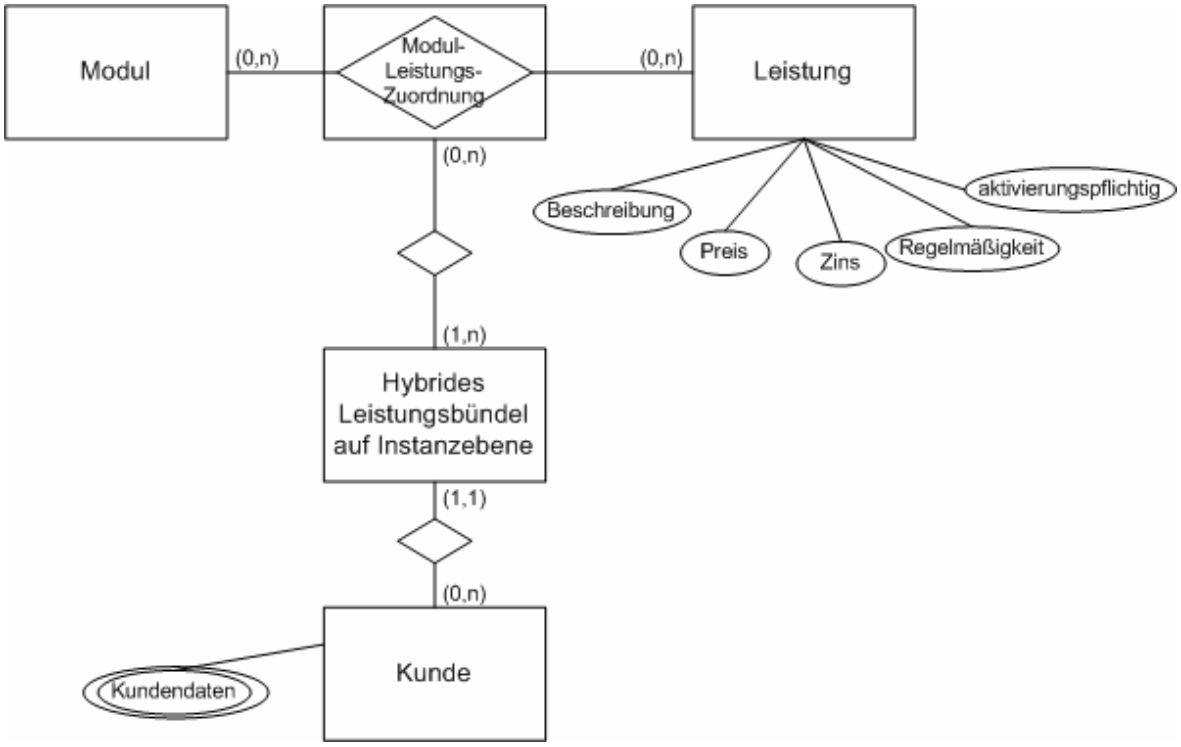


Abb. 3.10: HLB auf Instanzebene

In der H2-Sprachdefinition (vgl. Abb. 3.11) ist dies dadurch gelöst, dass einem Kunden mehrere HLBAI untergeordnet werden können und diesen wiederum Module mit einer Leistung.

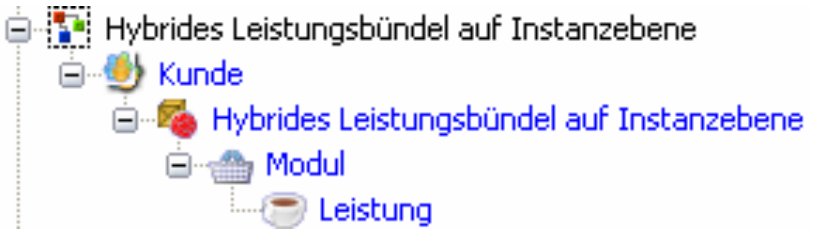


Abb. 3.11: H2-Sprachdefinition HLB auf Instanzebene

Im Gegensatz zum HLBAT darf hier jeweils nur eine Leistung unter jedem Modul angesiedelt sein. Das wird durch das automatische Füllen dieses Kontextes bewirkt. Der Bezug zu den Kundendaten, die im Kundenkontext des anderen Modells festgehalten ist erfolgt über Namensgleichheit der Kunden.

3.2.2 Lebenszyklusphasen

In Kapitel 3.1.2 wurden bereits Lebenszyklusphasen aus Anbietersicht erläutert. Neben der Anbietersicht auf ein HLBAT existiert eine Kundensicht. Lebenszyklusphasen dienen in der Kundensicht zur Erfassung aller anfallenden Zahlungen die ein Kunde über die gesamte Lebensdauer eines HLBAT leistet. Dazu ist es nötig Leistungen innerhalb der Lebenszyklusphasen abzubilden. Diese Perspektive auf die Lebenszyklusphasen ist demnach genau umgekehrt zu der Anbietersicht. In der Anbietersicht enthalten Leistungen Lebenszyklusphasen und in der Kundensicht enthalten Lebenszyklusphasen Leistungen. Intervalle werden analog zur Anbietersicht verwendet.

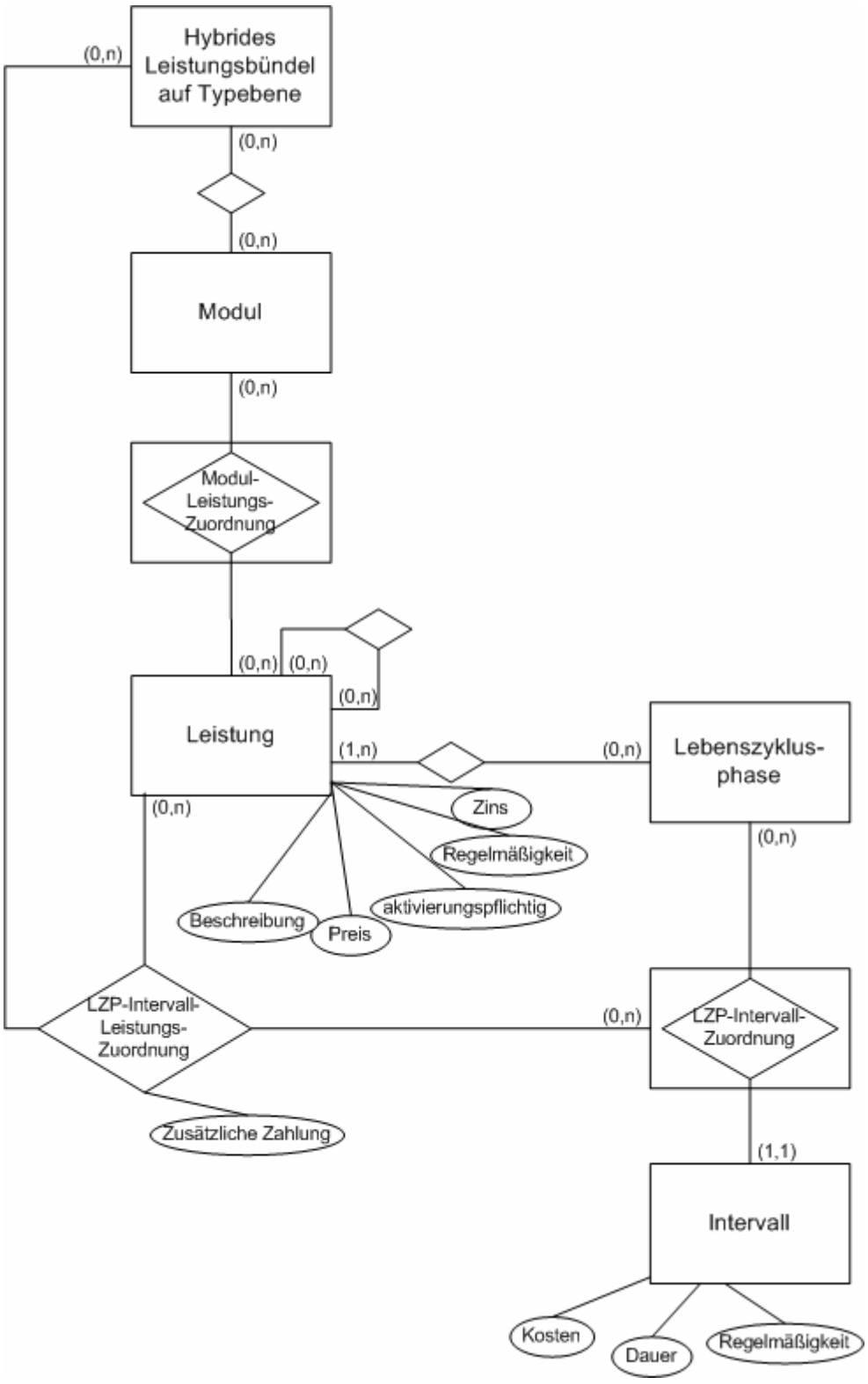


Abb. 3.12: Lebenszyklusphasen (Kundensicht)

Die Zuordnung von Leistungen zu Lebenszyklusphasen wird durch das Konstrukt *LZP-Intervallzuordnung* ermöglicht (vgl. Abb. 3.12). Dadurch können 0 bis n Leistungen Intervallen zugeordnet werden, um somit die gesamten anfallenden Zahlungen, bzw. den Gesamtpreis eines HLBAI zu ermitteln. Der Gesamtpreis wird wie folgt bestimmt:

Zunächst wird über die Attribute *Preis* und *Regelmäßigkeit* einer Leistung der Preis einer Leistung auf einen Tag herunter gerechnet. Die Berechnung des Preises pro Tag geschieht intern. Durch die Attribute *Dauer* und *Regelmäßigkeit* lässt sich die Länge eines Intervalls ebenfalls auf Tage herunter rechnen. Die Zahlung pro Intervall ermittelt sich aus dem Preis der Leistung pro Tag und der Dauer eines Intervalls in Tagen. Zusätzlich zu der ermittelten Zahlung pro Intervall lässt sich durch das Attribut *Zusätzliche Zahlung* am Konstrukt *LZP-Intervall-Leistungszuordnung* eine zusätzlich einmalig anfallende Zahlung für jede Leistung in einem Intervall erfassen**.**

Abb. 3.13 zeigt die H2-Sprachdefinition für Lebenszyklusphasen aus Kundensicht. Die Lebenszyklusphasen können einem HLBAT direkt auf der nächsten Ebene zugeordnet werden. Unterhalb der Lebenszyklusphasen werden immer Intervalle angelegt. Innerhalb der Intervalle befinden sich immer die Leistungen, die für die jeweilige Lebenszyklusphase für Zahlungen aus Sicht des Kunden sorgen können.

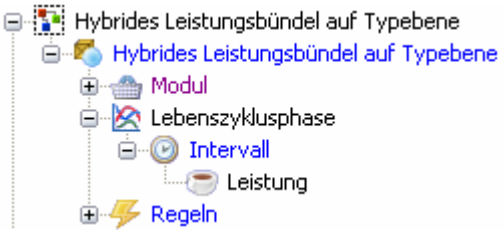


Abb. 3.13: H2-Sprachdefinition Lebenszyklusphasen (Kundensicht)

3.2.3 Präferenzen und Präferenzwerte

Das Präferenzkonstrukt dient dem wissensbasierten Verfahren zur Vorschlagsgenerierung im Recommender System. In diesem werden Präferenzen mit dazugehörigen Präferenzwerten benötigt, die vom Kunden bewertet werden. Jeder Präferenz sind 0 bis n Präferenzwerte und jedem Präferenzwert sind 0 bis n Präferenzen zugeordnet. Jedem Kunden und jeder Leistung werden 0 bis n Präferenzen und Präferenzwerte zugeordnet. Ist das Bezugsobjekt eine Präferenz, dann ist das Attribut Wichtigkeit zu detaillieren. Ist das Bezugsobjekt hingegen ein Präferenzwert, so ist das Attribut Anforderung zu spezifizieren. Dieser Zusammenhang ist in Abb. 3.14 genauer verdeutlicht.

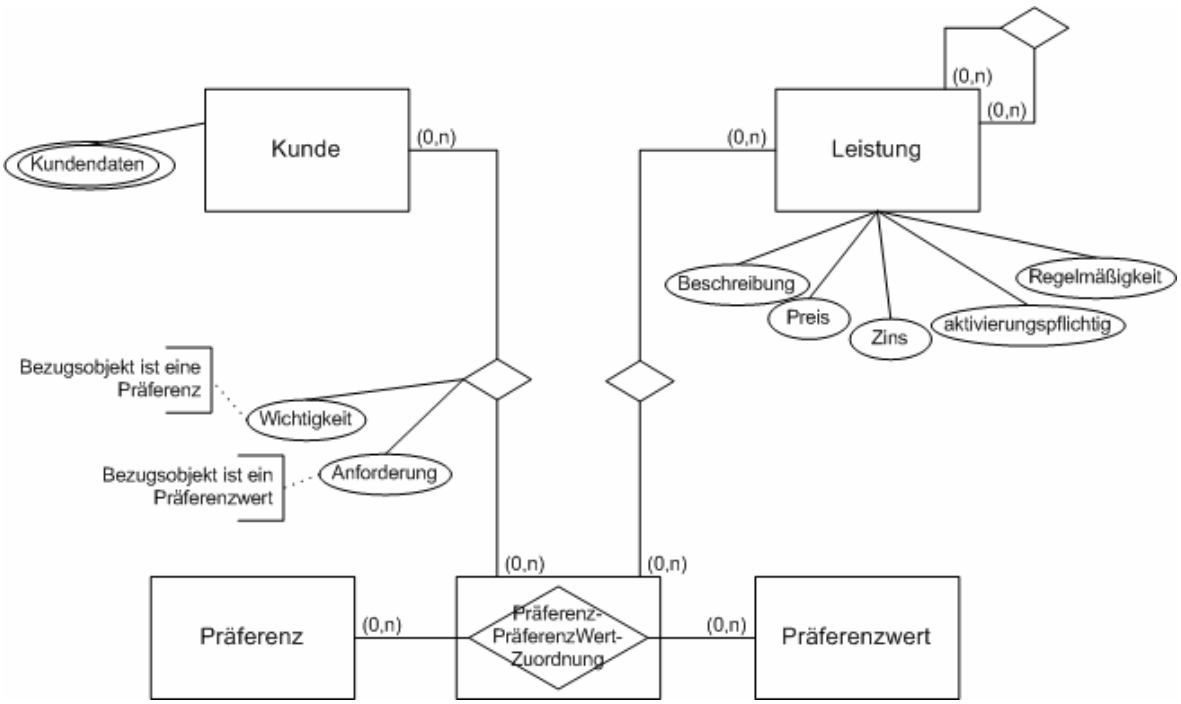


Abb. 3.14: Präferenzen und Präferenzwerte

Abb. 3.15 visualisiert die Sprachdefinition des Präferenzkonstrukts. Dabei werden Präferenzen und Präferenzwerte einem Kunden untergeordnet. Jede Präferenz enthält auf einer Ebene tiefer Präferenzwerte.

Abb. 3.15: H2-Sprachdefinition Präferenzkonstrukt

3.2.4 Assoziationsregeln

Assoziationsregeln werden für das Verfahren des gemeinschaftlichen Filterns im Recommender System benötigt. Eine *Assoziationsregel* steht mit einem *Auslöser (AR)* und einer *Konsequenz (AR)* in Beziehung. Auslöser einer Assoziationsregelanwendung können eine oder mehrere Leistungen in dazugehörigen Modulen sein. Dabei werden mehrere Objekte jeweils durch den UND-Operator verknüpft. Die Konsequenz hingegen kann genau eine Leistung eines bestimmten Moduls sein. Eine Assoziationsregel verfügt über die Attribute *Support* und *Confidence*. Im Attribut Support wird dabei angegeben welcher Anteil aller Transaktionen für eine bestimmte Regel gilt. Im Attribut *Confidence*

wird hinterlegt, für welchen Anteil der Transaktionen in denen ein bestimmtes Item vorkommt, auch das andere Item vorkommt. Das Assoziationsregelkonstrukt ist in Abb. 3.16 dargestellt.[19]

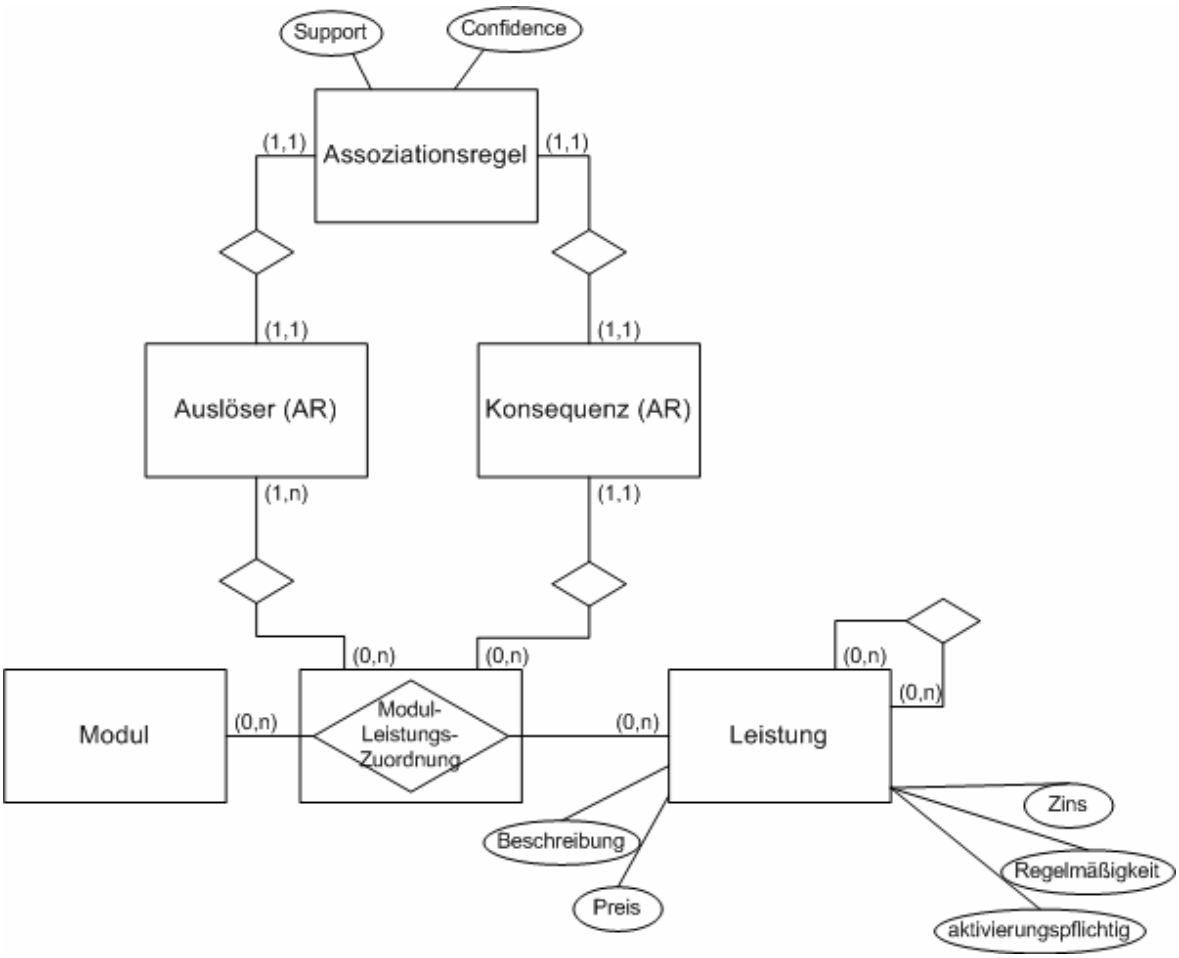


Abb. 3.16: Assoziationsregeln

Abb. 3.17 stellt das Assoziationsregelkonstrukt als H2-Sprachdefinition dar. Auslöser und Konsequenz einer Assoziationsregelregel befinden sich immer direkt eine Ebene tiefer als die Assoziationsregel selbst. Auslöser und Konsequenz sind immer Leistungen mit dem zugehörigen Modul.

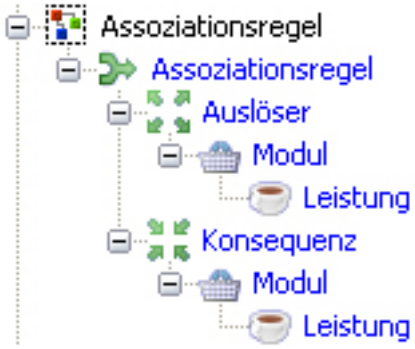


Abb. 3.17: H2-Sprachdefinition Assoziationsregeln

[19] Zum Konzept der Assoziationsregeln und ihrer Berechnung vgl. Agrawal, Imielinski, Swami (2003)

4 Vorgehensmodelle und Anleitungen

4.1 Modellierung anhand der HLB-Sprache

Im vorherigen Kapitel wurde die Sprache zur Modellierung hybrider Leistungsbündel vorgestellt. In diesem Kapitel soll veranschaulicht werden, wie mit der vorgestellten Sprache und der H2-Toolset-Solution ein hybrides Leistungsbündel modelliert werden kann. Aufgrund des großen Umfangs des Gesamtmodells, welches innerhalb dieses Projektseminars entstanden ist, wird bei der Vorstellung der Modellkonstrukte lediglich auf Teile des Gesamtmodells zurückgegriffen, damit die Übersichtlichkeit dieser Anleitung gewährleistet ist. Dazu werden einzelne Konstrukte im Modelleditor erstellt. Die vorgestellten Beispiele können auf weitere Leistungen, Ressourcen etc. übertragen werden.

Das im Folgenden vorgestellte Vorgehensmodell zur Modellierung hybrider Leistungsbündel stellt lediglich eine Empfehlung an den Modellierer dar. Generell lässt sich die Modellierung auch auf anderem Wege durchführen. Auf jeden Fall sollten wichtige Hinweise beachtet werden, damit am Ende des Modellierungsprozesses ein korrektes Modell eines hybriden Leistungsbündels vorliegt. Falls anhand dieses Vorgehensmodells modelliert wird, ist es zunächst wichtig, eine strukturierte Übersicht der zu modellierenden Leistungen und der Unternehmensstruktur vorliegen zu haben. Dies ist deshalb von Vorteil, damit nicht immer wieder zurück in einen anderen Kontext gewechselt werden muss, um eventuell vergessene oder ausgelassene Objekte nachzumodellieren. Das Vorgehensmodell ist in fünf Schritte unterteilt. Abb. 4.1 stellt das Schema des Modellierungsprozesses zusammengefasst dar:

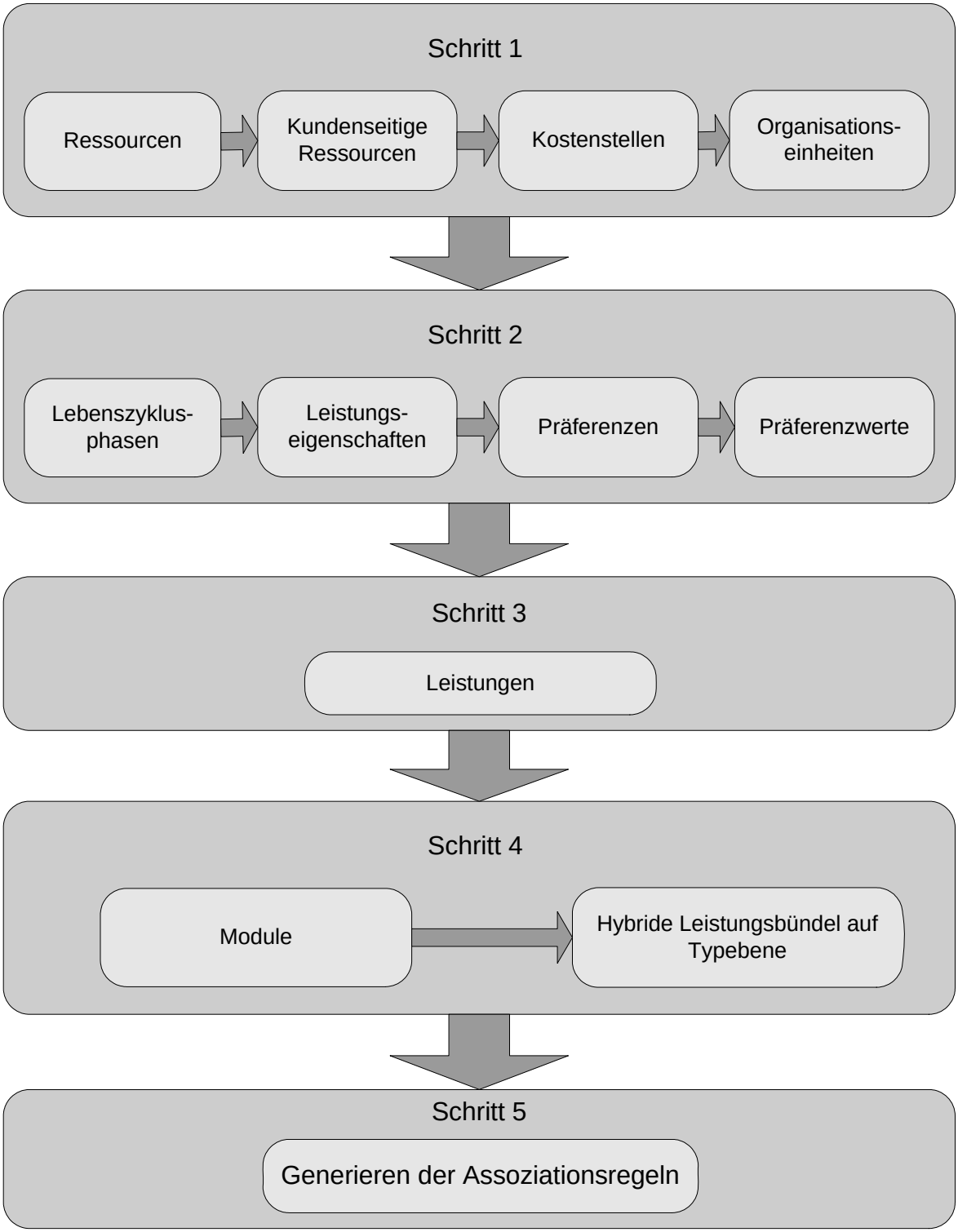


Abb. 4.1: Vorgehensmodell der Modellierung

Bevor der eigentliche Modellierungsprozess gestartet werden kann, ist es notwendig, ein neues Modell anzulegen. Das Erstellen eines neuen Modells erfolgt über einen Rechtsklick auf den Eintrag Modelle in der jeweiligen Projektmappe in der H2-Toolset-Solution. Es öffnet sich ein Fenster, in dem der Name des Modells einzutragen ist. Neben dem Eintragen des Modellnamens ist die Sprache, mit der das Modell erstellt werden soll, auszuwählen. Hier ist die Sprache „HLB“ auszuwählen (vgl. Abb. 4.2).

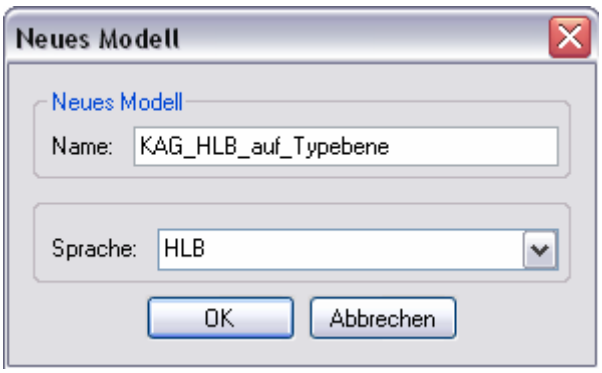


Abb. 4.2: Anlegen eines neuen Modells

4.1.1 (Kundenseitige) Ressourcen, Kostenstellen, Organisationseinheiten

In diesem Kapitel wird erläutert, wie die Modellierung von Ressourcen, Kundenseitigen Ressourcen, Kostenstellen und Organisationseinheiten durchzuführen ist.

Ressourcen

Die Modellierung einer Ressource erfolgt über das Anlegen der Ressource im Kontext Ressource. Abb. 4.3 visualisiert den Modelleditor mit zwei bereits angelegten Ressourcen „Werkzeugkit“ und „Aluschiene“.

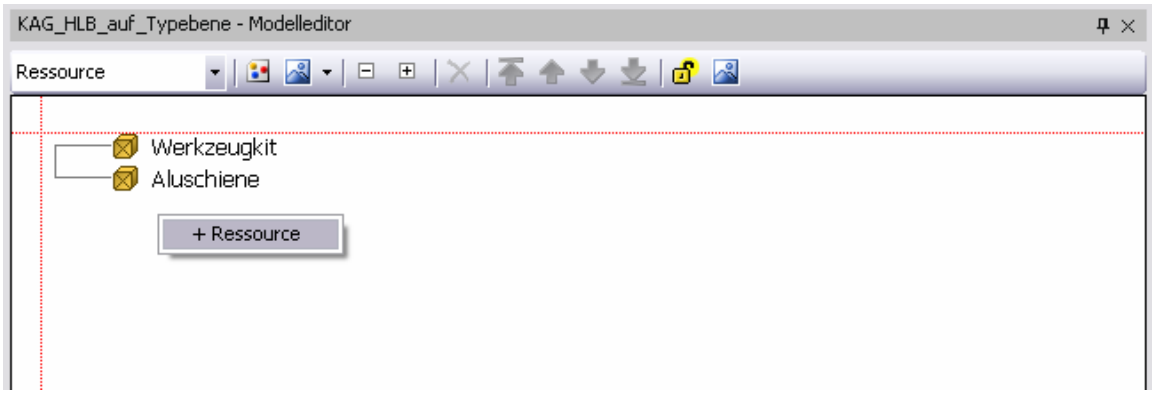


Abb. 4.3: Hinzufügen einer Ressource

Über einen Rechtsklick auf die jeweilige Ressource lassen sich Attribute anlegen. Wird die Option „Attributtypgruppe hinzufügen“ gewählt, werden direkt alle Attribute für eine Ressource angelegt. Abb. 4.4 zeigt das Anlegen von Attributen im Kontext Ressourcen.

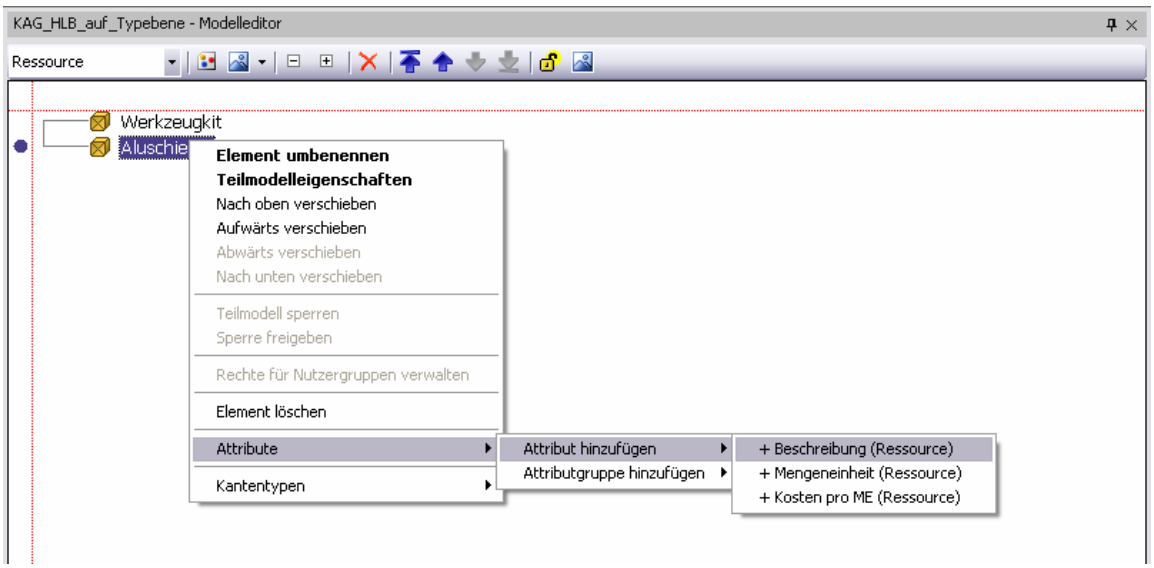


Abb. 4.4: Hinzufügen von Attributen (Ressourcen)

Für Ressourcen existieren drei Attribute: „Beschreibung“, „Mengeneinheit“ und „Kosten pro ME“. Besonders bei dem Attribut „Mengeneinheit“ sollte auf eine einheitliche Verwendung geachtet werden, damit eine Vergleichbarkeit gewährleistet ist. Nachdem die Attribute angelegt worden sind, können diese gefüllt werden. Abb. 4.5 zeigt beispielhaft Attributwerte für die Ressource „Aluschiene“. Besonders wichtig ist, dass Attribute direkt nach dem Anlegen auch mit Inhalten gefüllt werden, da es bei großen Modellen schnell zum versehentlichen Auslassen kommen kann.

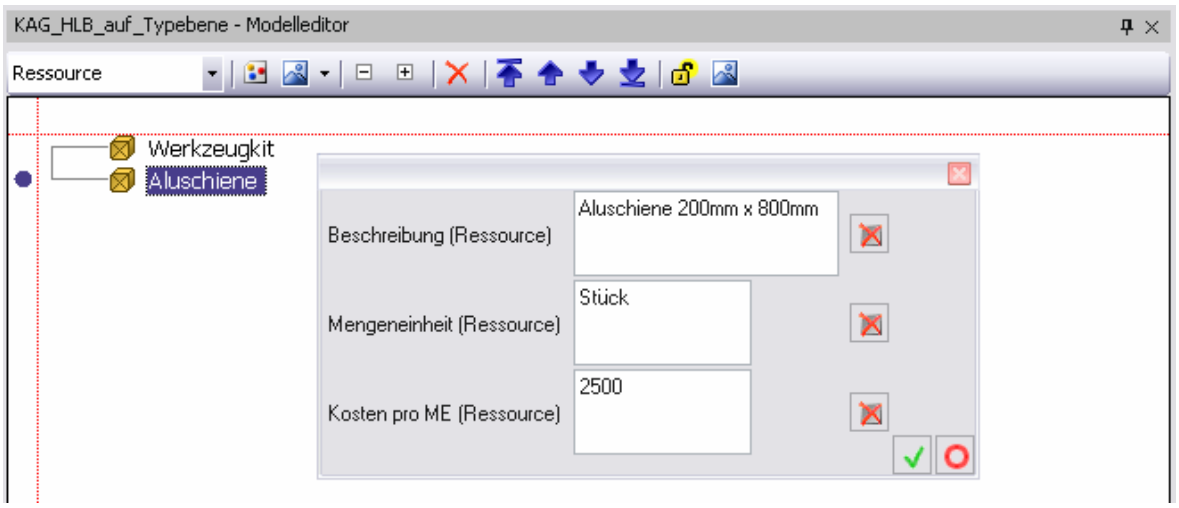


Abb. 4.5: Attributwerte bei der Ressource „Aluschiene“

Kostenstellen

Die Modellierung von Kostenstellen erfolgt analog der Modellierung von Ressourcen. Dazu werden nach Auswahl des Kontextes Kostenstelle per Rechtsklick neue Kostenstellen angelegt. In Abb. 4.6 wurden die fünf Kostenstellen Controlling, Fertigung, Montage, Vertrieb und Verwaltung angelegt. Für Kostenstellen existiert das Attribut Beschreibung. Mit diesem kann die jeweilige Kostenstelle näher erläutert werden.

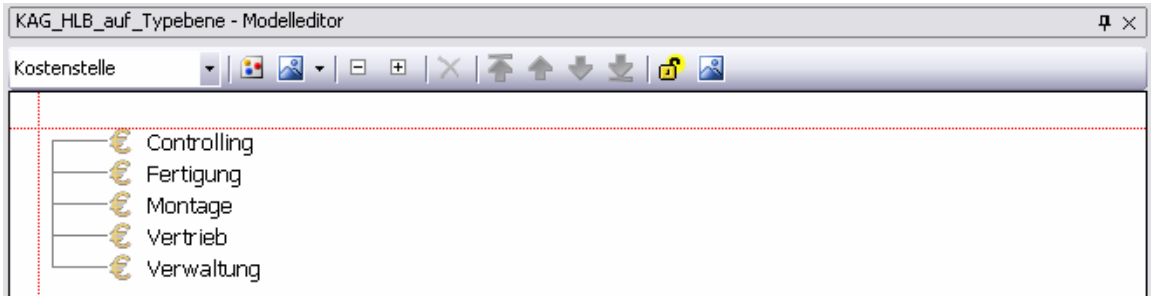


Abb. 4.6: Hinzufügen von Kostenstellen

Kundenseitige Ressource

Zur Modellierung Kundenseitiger Ressourcen ist der Kontext Kundenseitige Ressource im Modelleditor auszuwählen. In Abb. 4.7 wurden exemplarisch die Kundenseitigen Ressourcen Information, Mitarbeiter und Standort angelegt.

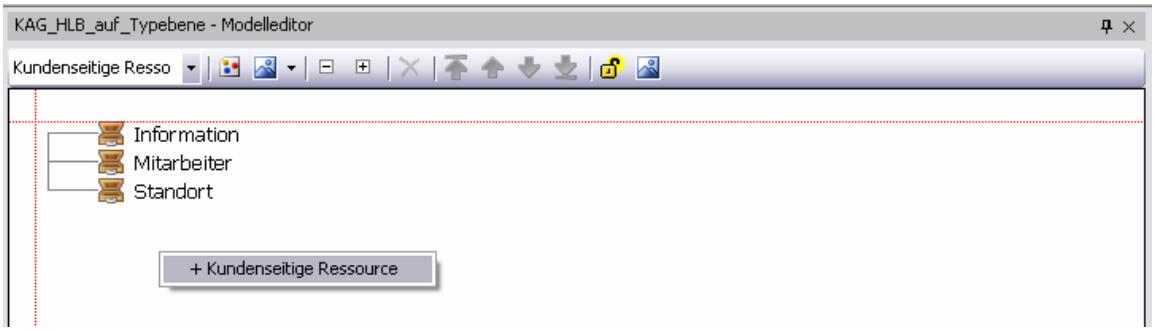


Abb. 4.7: Hinzufügen von Kundenseitigen Ressourcen

Kundenseitige Ressourcen verfügen wie die Kostenstellen über das Attribut Beschreibung. In diesem lässt sich die Kundenseitige Ressource näher spezifizieren. In Abb. 4.8 wurde die Kundenseitige Ressource „Information“ beschrieben. Dabei kann es sich beispielsweise um Informationen handeln, die vom Kunden mit in den Leistungserstellungsprozess eingebracht werden.

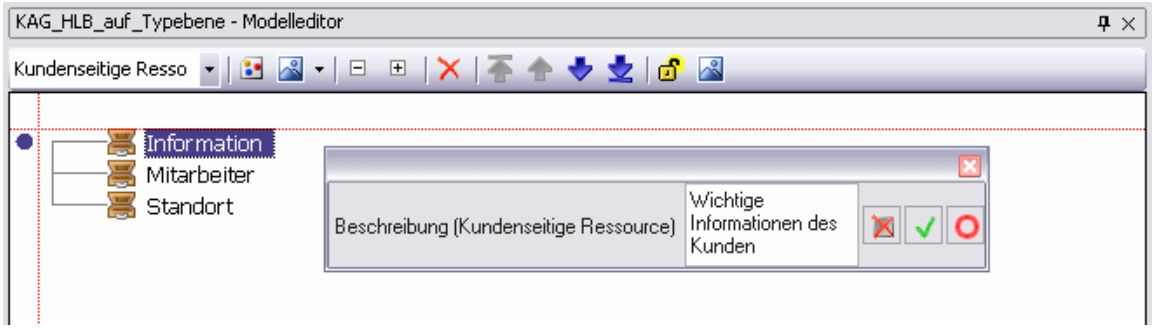


Abb. 4.8: Beschreibung einer Kundenseitigen Ressource

Organisationseinheit

Zur Modellierung einer Organisationseinheit muss zuerst die eigentliche Organisationseinheit im Kontext Organisationseinheit angelegt werden. In Abb. 4.9 wurden die vier Organisationseinheiten Montageabteilung, Forschung, Entwicklung und Controlling hinzugefügt. Eine Organisationseinheit

verfügt über das Attribut Beschreibung, hier kann sie näher erläutert werden. Auf eine zusätzliche Abbildung zur Beschreibung soll hier verzichtet werden, da sie analog zu der Beschreibung einer Kundenseitigen Ressource erfolgt.

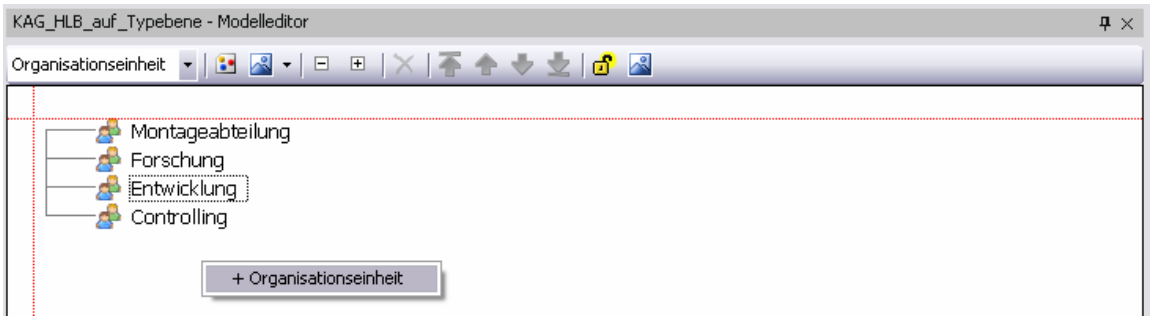


Abb. 4.9: Hinzufügen von Organisationseinheiten

Nachdem die Organisationseinheiten festgelegt wurden, können diese durch das Hinzufügen von Stellen weiter untergliedert werden. Dies geschieht durch einen Rechtsklick auf die jeweilige Organisationseinheit unter dem Menüpunkt „+Stelle“. Abb. 4.10 zeigt die angelegten Stellen für die Organisationseinheit Entwicklung. Hier wurden die Stellen Maschinenbauingenieur und Designer angelegt.

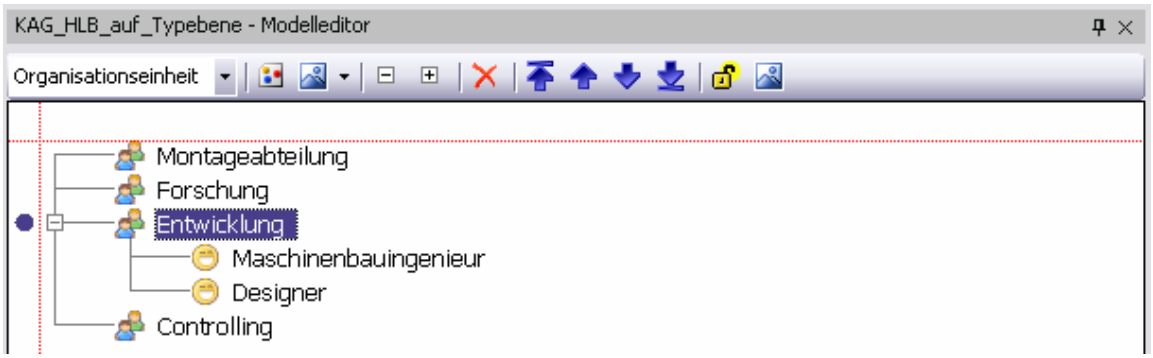


Abb. 4.10: Hinzufügen von Stellen

Stellen verfügen über das Attribut Stundensatz, was eine Erfassung der durch die Stelle verursachten Kosten ermöglicht. Das Attribut Stundensatz kann für jede Stelle einmal im Kontext Organisationseinheit hinterlegt werden. In Abb. 4.11 wurde das Attribut exemplarisch für die Stelle Maschinenbauingenieur hinterlegt.

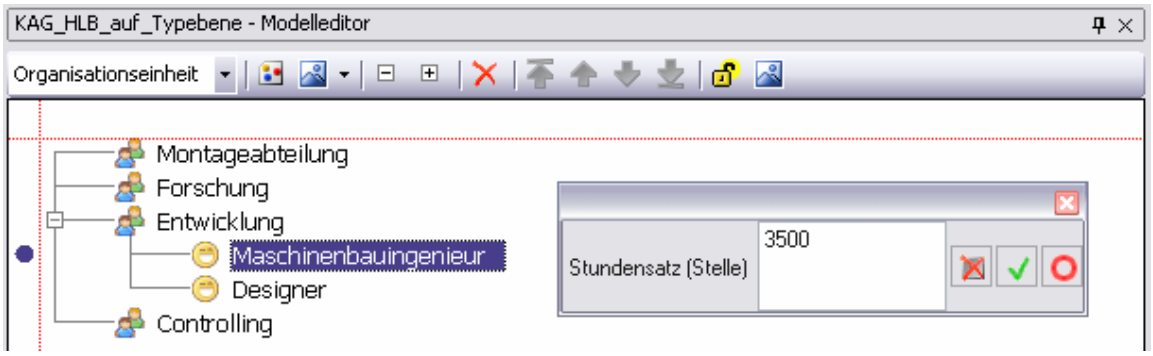


Abb. 4.11: Attributwerte bei der Stelle „Maschinenbauingenieur“

In einem nächsten Schritt können den angelegten Stellen Kostenstellen zugeordnet werden. Dazu muss ein zweiter Modelleditor mit dem Kontext Kostenstellen geöffnet werden. Aus dem Kostenstellen-Kontext werden die benötigten Kostenstellen dann per „drag und drop“ auf die jeweilige Stelle gezogen. In Abb. 4.12 wurde der Stelle Maschinenbauingenieur die Kostenstelle Forschung & Entwicklung zugeordnet.

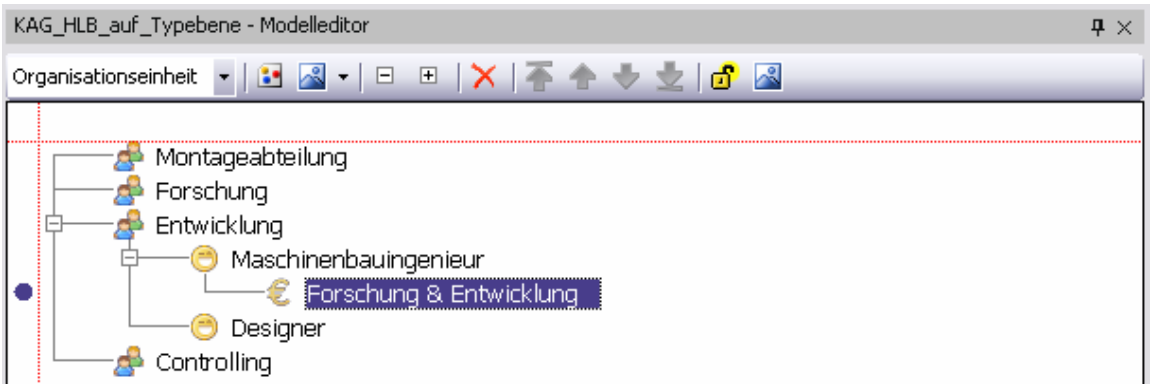


Abb. 4.12: Einfügen von Kostenstellen unter Stellen

4.1.2 Lebenszyklusphasen, Leistungseigenschaften, Präferenzen, Präferenzwerte

Lebenszyklusphasen

Damit die Lebensdauer einer Leistung berücksichtigt werden kann, wird im folgenden Schritt verdeutlicht, wie Lebenszyklusphasen festgelegt werden können. Dazu wird der Kontext Lebenszyklusphase im Modelleditor ausgewählt. Mittels Rechtsklick können verschiedene Phasen eines Lebenszyklus erstellt werden. Abb. 4.13 zeigt fünf angelegte Lebenszyklusphasen. Bei diesen handelt es sich um Entwurf, Fertigung, Kauf, Nutzung und Nachnutzung. Die weitere Detaillierung und Nutzung der angelegten Lebenszyklusphasen wird im Schritt „Modellierung der Leistungen“ erläutert.

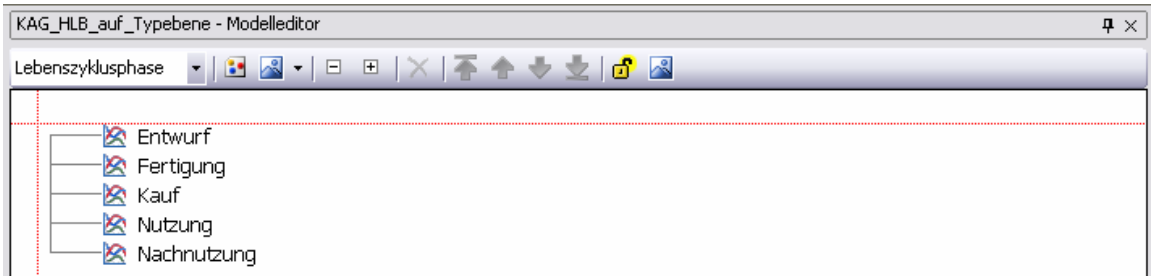


Abb. 4.13: Hinzufügen von Lebenszyklusphasen

Leistungseigenschaft

Über den Kontext Leistungseigenschaften lassen sich Leistungen detailliert beschreiben. Dazu können unter diesem Kontext verschiedene Leistungseigenschaften mittels eines Rechtsklicks angelegt werden. Exemplarisch wurden in Abb. 4.14 die Leistungseigenschaften Gewicht, Raumbedarf und Automatisierungsgrad angelegt.

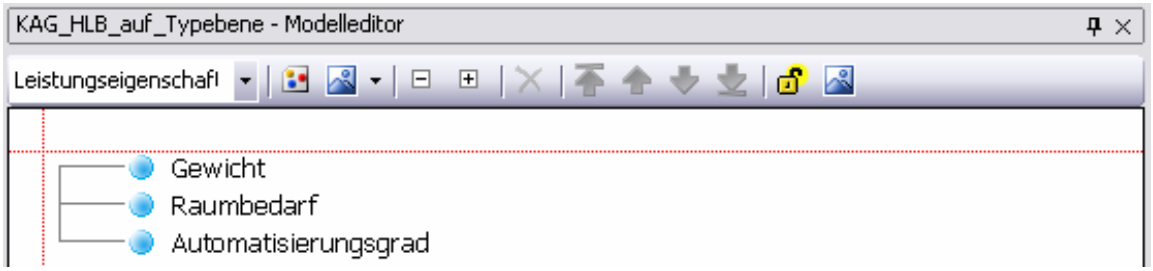


Abb. 4.14: Hinzufügen von Leistungseigenschaften

Präferenzen und Präferenzwerte

Das im Konfigurator zum Einsatz kommende Recommender System besteht zum einen aus einem Wissensbasierenden Verfahren und zum anderen aus dem Kollaborativen Filtern. Damit mit dem Verfahren zur Wissensbasierten Vorschlagsgenerierung gearbeitet werden kann, sind Präferenzen und dazugehörige Präferenzwerte nötig, die vom Kunden bewertet werden. In einem ersten Schritt werden dazu unter dem Kontext Präferenz mittels Rechtsklick neue Präferenzen angelegt. In Abb. 4.15 sind die drei Präferenzen Preis, Bedienkomplexität und Automatisierungsgrad erstellt worden. Über das Attribut Beschreibung können die einzelnen Präferenzen im Fließtext erläutert werden. Diese Beschreibung wird im Konfigurator zur Erläuterung angezeigt.

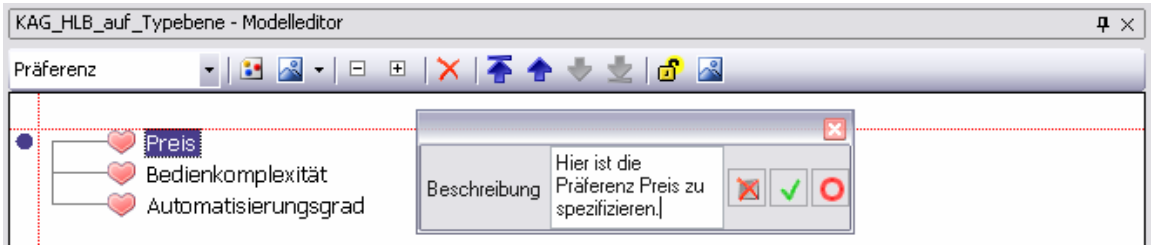


Abb. 4.15: Hinzufügen von Präferenzen

Als nächstes werden für die angelegten Präferenzen passende Präferenzwerte erstellt. Dazu wird im Modelleditor der Kontext Präferenzwerte ausgewählt. Abb. 4.16 zeigt exemplarisch Präferenzwerte für die Präferenz Preis. Hier sind als mögliche Präferenzwerte „Oberes Preissegment“, „Mittleres Preissegment“, „ Unteres Preissegment“ und „kostenlos“ angelegt worden.

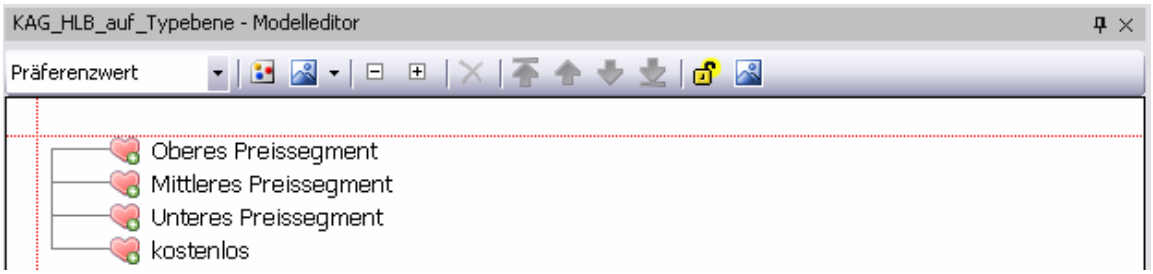


Abb. 4.16: Hinzufügen von Präferenzwerten

In einem nächsten Schritt werden die erstellten Präferenzwerte in die dazugehörigen Präferenzen eingefügt. Dazu können die Präferenzwerte in einem Modelleditorfenster markiert per „drag and

drop“ in das andere Modelleditorfenster in die dazugehörige Präferenz gezogen werden. Abb. 4.17 zeigt die Präferenz Preis nach dem Einfügen der zuvor erstellten Präferenzwerte.

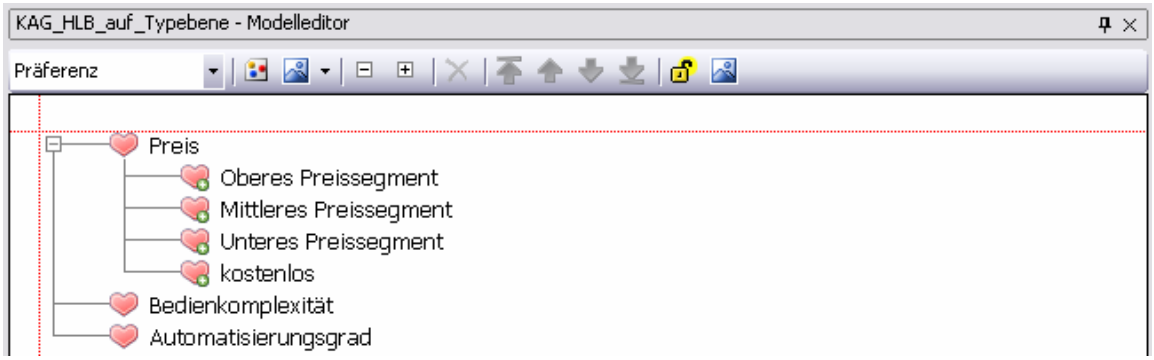


Abb. 4.17: Einfügen der Präferenzwerte unter die Präferenzen

4.1.3 Leistungen

Die Hauptkomponente hybrider Leistungsbündel sind Leistungen. Diese gilt es im nächsten Schritt zu erstellen. Damit eine Leistung vollständig modelliert werden kann, muss zunächst die eigentliche Leistung im Kontext Leistung angelegt werden. In Abb. 4.18 wurden die drei Fräsmaschinen der T-Serie (T-800, T-1000, T-2000) und drei Dienstleistungen (Wartung bronze, Wartung silber, Wartung gold) erstellt.

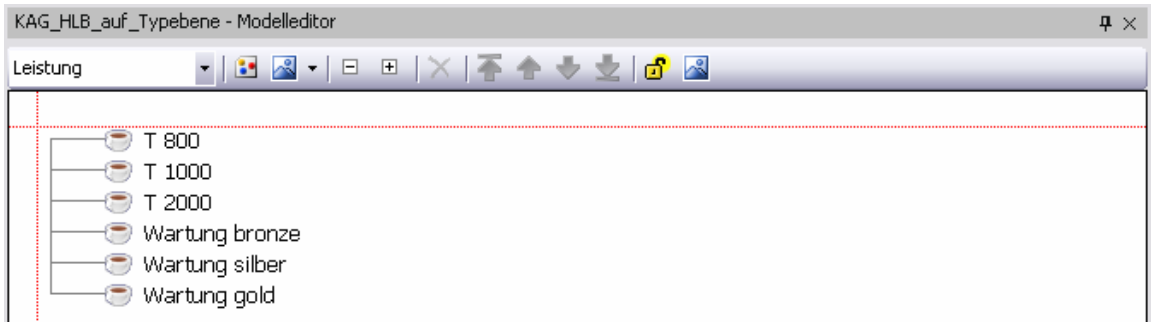


Abb. 4.18: Hinzufügen von Leistungen

Nachdem die Leistungen angelegt worden sind, können diese weiter detailliert werden. Nach der H2-Sprachdefinition kann eine Leistung Lebenszyklusphasen, andere Leistungen, Leistungseigenschaften und Präferenzen enthalten. Als erstes werden Leistungseigenschaften hinzugefügt. Dies erfolgt durch einen Rechtsklick auf die jeweilige Leistung (+Leistungseigenschaften). Der angelegte Punkt Leistungseigenschaften unter der Leistung kann nun per „drag and drop“ mit Leistungseigenschaften gefüllt werden. Die Fräsmaschine T-800 wurde in Abb. 4.19 mit den Leistungseigenschaften Gewicht, Raumbedarf und Automatisierungsgrad versehen.

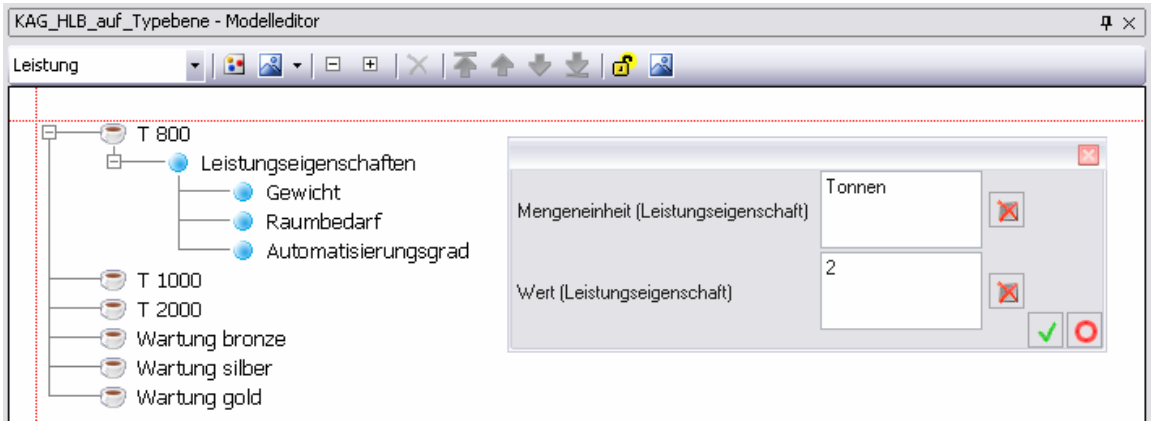


Abb. 4.19: Hinzufügen von Leistungseigenschaften zur Leistung T-800

Über die Attribute der einzelnen Leistungseigenschaften kann die Fräsmaschine T-800 nun genauer spezifiziert werden. Hier im Beispiel wird die Leistungseigenschaft Gewicht mit dem Wert 2 und der Mengeneinheit Tonnen versehen. Die anderen Attribute können analog ergänzt werden. Besonders zu beachten ist, dass die Mengeneinheiten einheitlich gewählt werden, da sie im Konfigurator angezeigt werden und auch bei der Regelanwendung vergleichbar sein müssen.

In einem nächsten Schritt erfolgt das Hinzufügen von Lebenszyklusphasen zu einer Leistung. Für die Fräsmaschine T-800 wurden die vier Lebenszyklusphasen Entwurf, Fertigung, Kauf und Nutzung aus dem Kontext Lebenszyklusphase hinzugefügt. Abb. 4.20 zeigt die Leistung T-800 mit den bisher angelegten Leistungseigenschaften und Lebenszyklusphasen.

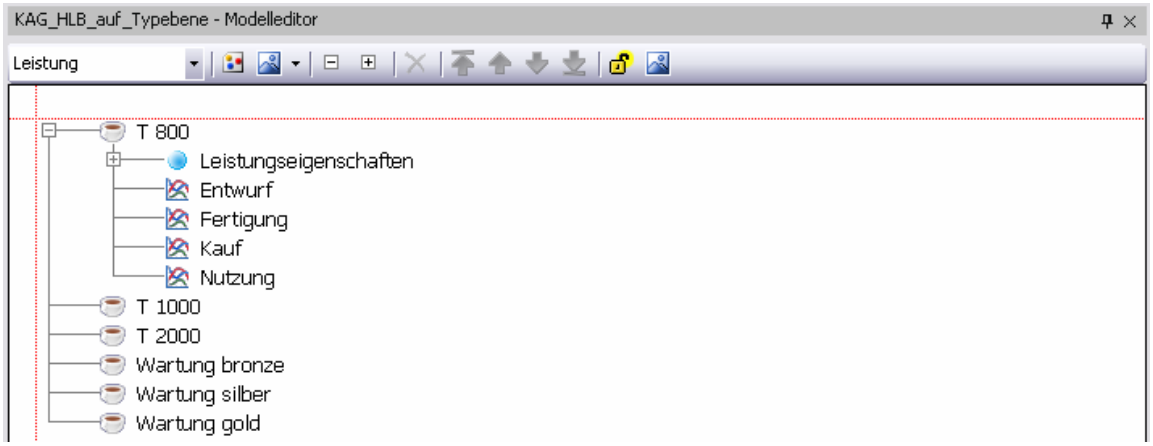


Abb. 4.20: Hinzufügen von Lebenszyklusphasen

Nach dem Anlegen der Lebenszyklusphasen können diese nun mit Intervallen feiner untergliedert werden. Intervalle werden durch einen Rechtsklick auf die entsprechende Lebenszyklusphase neu erstellt. In Abb. 4.21 wurden für die Lebenszyklusphase Entwurf zwei Intervalle angelegt. Die Intervalle „Planung und „Prototyping“ unterteilen die gesamte Entwurfsphase in zwei Abschnitte. Intervalle werden chronologisch nach ihrer Anordnung ausgewertet. Sie sollten deshalb nach der tatsächlichen zeitlichen Abfolge angeordnet sein.

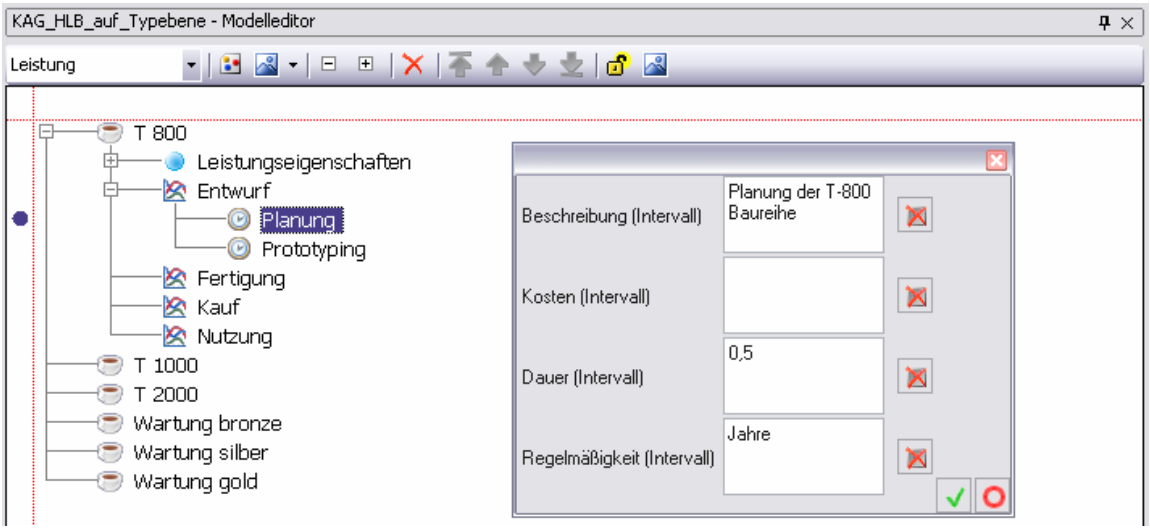


Abb. 4.21: Hinzufügen von Intervallen und Attributen

Intervalle verfügen über die Attribute Beschreibung, Kosten, Dauer und Regelmäßigkeit. Über das Attribut Beschreibung wurde das Intervall Planung genauer detailliert. Über das Attribut Kosten können anfallende Kosten, welche das Intervall betreffen, hinterlegt werden. Das Attribut Dauer gibt die Länge des Intervalls an. In diesem Beispiel hat das Intervall die Länge 0,5 mit einer Regelmäßigkeit Jahre.

Nach dem Anlegen der Intervalle können diese weiter untergliedert werden. Dazu können unterhalb der Intervalle Aktivitäten eingefügt werden. Dies geschieht mittels eines Rechtsklicks auf das entsprechende Intervall unter dem Menüpunkt „+Aktivität“. In Abb. 4.22 ist die Aktivität „Prototypentwicklung“ eingefügt worden. Dies bedeutet, dass im Intervall „Prototyping“ die Aktivität „Prototypentwicklung“ vom Anbieter geleistet wird, für die Kosten entstehen. Aktivitäten lassen sich wieder durch Attribute detaillieren. Die hier möglichen Attribute sind eine Beschreibung der Aktivität und die Anzahl der Wiederholungen innerhalb des zutreffenden Intervalls.

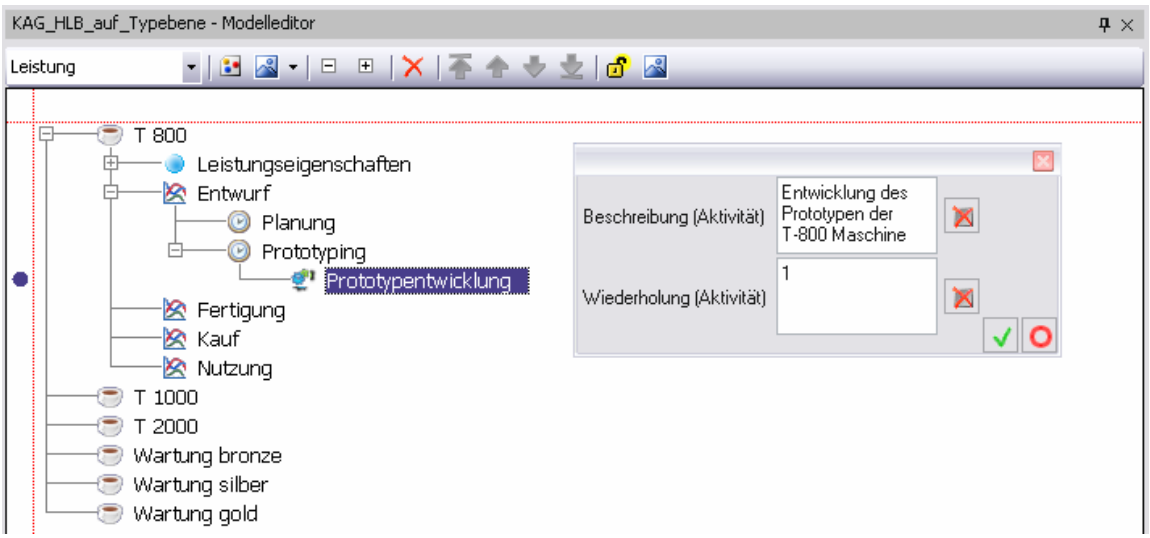


Abb. 4.22: Hinzufügen von Aktivitäten und Attributen

Aktivitäten eines Anbieters werden von zuständigen Organisationseinheiten ausgeführt. Diese können im nächsten Schritt aus dem Kontext Organisationseinheit zugeordnet werden. Beispielhaft wurde in Abb. 4.23 als ausführende Organisationseinheit der Aktivität Prototypentwicklung, die Organisationseinheit Entwicklung mit der dazugehörigen Stelle Maschinenbauingenieur und der zugeordneten Kostenstelle Forschung & Entwicklung eingefügt. Über das Attribut Stundenzahl der Stelle Maschinenbauingenieur kann die benötigte Stundenanzahl der ausführenden Stelle leistungsspezifisch hinterlegt werden. In diesem Fall benötigt der Maschinenbauingenieur für die Prototypentwicklung 140 Stunden.

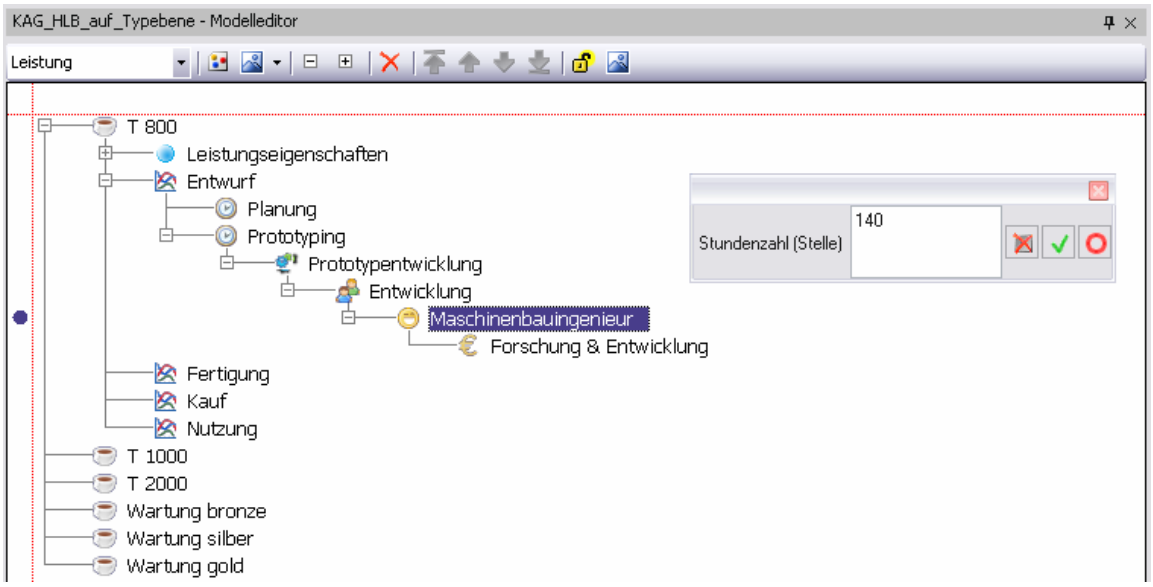


Abb. 4.23: Hinzufügen einer Organisationseinheit zu einer Aktivität

Neben Aktivitäten lassen sich in einem Intervall auch Ressourcen hinterlegen. Dies können beispielsweise Ressourcen sein, die bei der Ausführung einer Aktivität benötigt werden. In Abb. 4.24 wurde die Ressource „Aluschiene“ eingefügt. Über die Attribute Menge und Kosten pro ME Aufschlag lassen sich leistungsspezifisch die Kosten hinterlegen. Im Intervall „Prototyping“ entstehen durch die Aluschiene zusätzliche Kosten in Höhe von 2500 Euro.

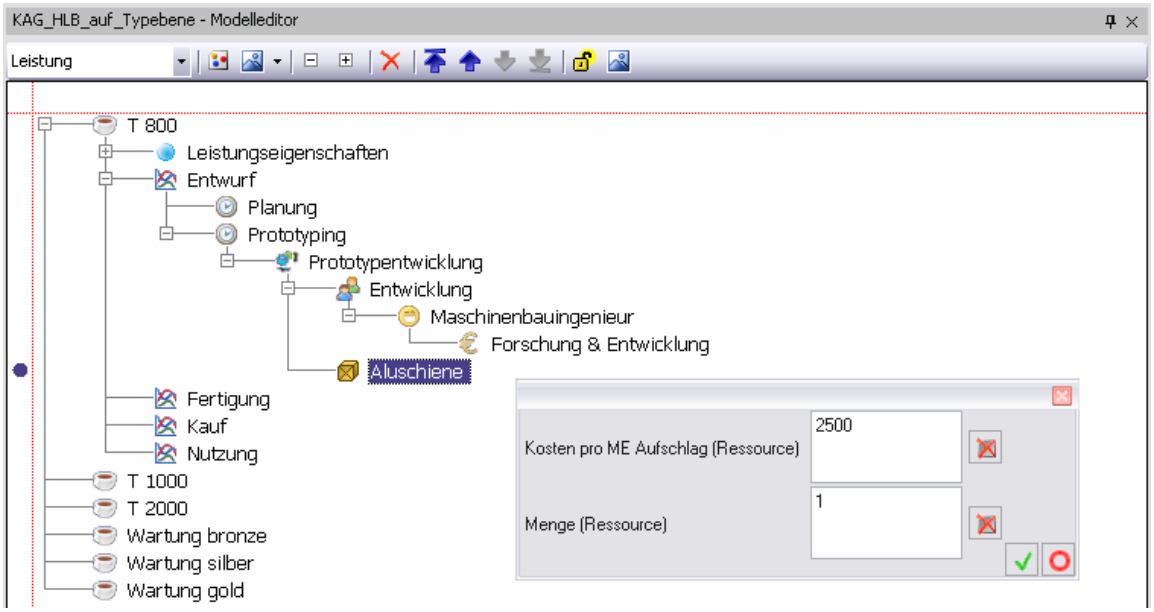


Abb. 4.24: Hinzufügen einer Ressource und Attributen

Wenn alle Lebenszyklusphasen dementsprechend gefüllt worden sind, kann im nächsten Schritt die Modellierung der Präferenzen mit einem dazugehörigen Präferenzwert unterhalb der Leistung erfolgen. Dazu wird mittels eines Rechtsklicks auf die jeweilige Leistung der Knoten Präferenzen erzeugt (+Präferenzen). In diesen lassen sich die gewünschten Präferenzen, sowie Präferenzwerte einfügen. In Abb. 4.25 wurde exemplarisch die Präferenz Preis mit dem Präferenzwert Unteres Preissegment eingefügt. Zu beachten ist, dass immer nur ein Präferenzwert pro Präferenz einer Leistung eingefügt werden darf.

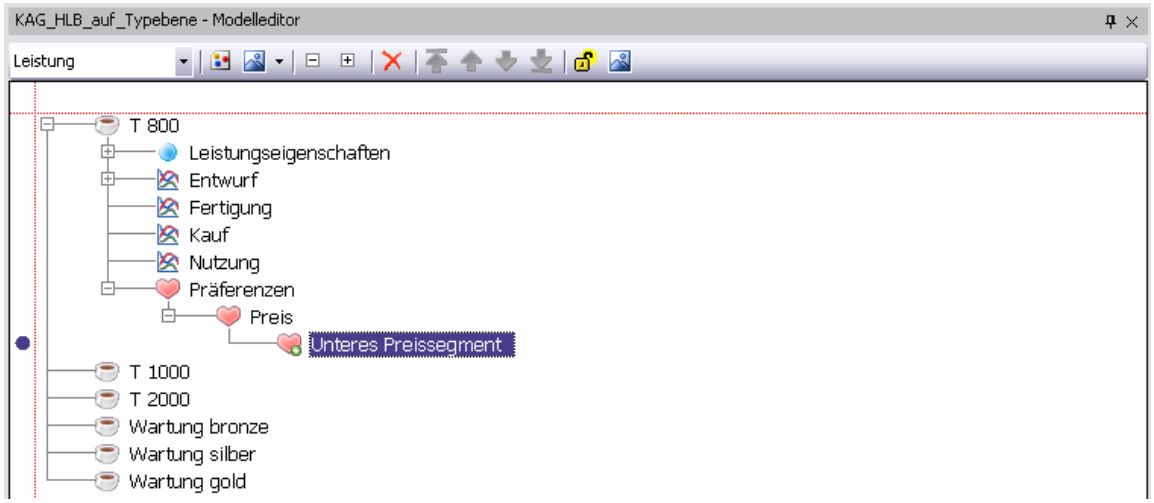


Abb. 4.25: Hinzufügen von Präferenzen zur Leistung T-800

Als Letztes können Leistungen wie oben bereits erläutert wurde, andere Leistungen enthalten (Leistungsstruktur). Dies kann per „drag und drop“ im Leistungskontext selbst stattfinden. Exemplarisch wurde die Leistung Maschinenbett erstellt und in die Leistung T-800 übertragen (vgl. Abb. 4.26).

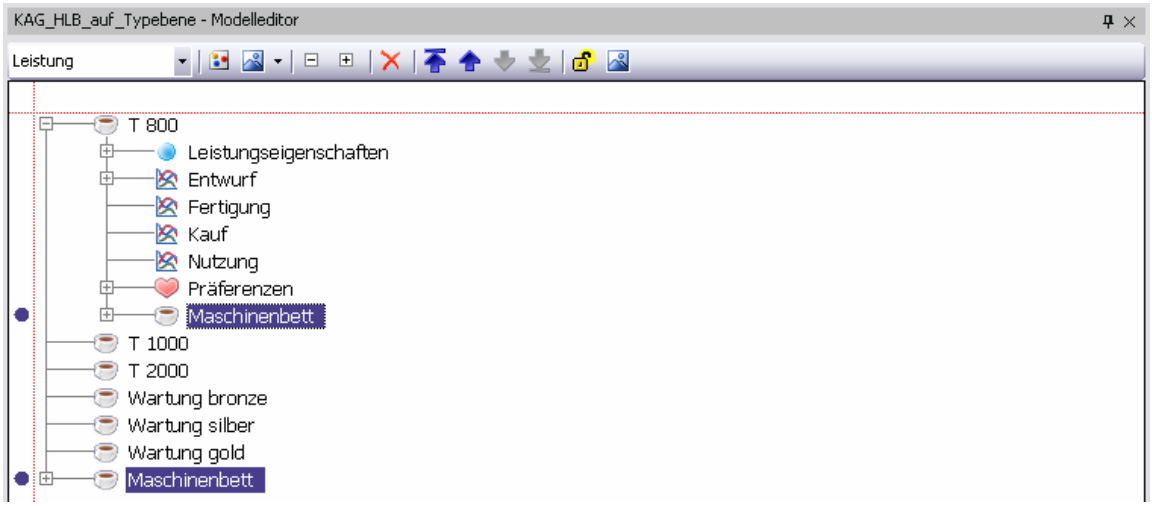


Abb. 4.26: Hinzufügen einer Leistung zu einer Leistung

4.1.4 Module und Hybride Leistungsbündel auf Typebene

Wie bereits erläutert wurde, wird ein hybrides Leistungsbündel durch die in ihm enthaltenen Module gegliedert. Deshalb wird in diesem Abschnitt zunächst erläutert, wie Module modelliert werden, um diese im nächsten Schritt in einem Hybriden Leistungsbündel auf Typebene anzulegen.

Module

Module ermöglichen eine Kategorisierung von Leistungen sowie eine automatische Generierung von Konfigurationsschritten durch den Konfigurator. Innerhalb eines Moduls sind immer genau die Leistungen enthalten, die später im Konfigurator Alternativen zueinander darstellen. Beispielsweise sind im Modul Maschinentyp nur Maschinentypen, die in einem Konfigurationsschritt ausgewählt werden können. In Abb. 4.27 wurden die beiden Module Maschinentyp und Wartung im Kontext Module angelegt und mit den entsprechenden Leistungen gefüllt. Dazu werden die entsprechenden Leistungen aus dem Kontext Leistungen per „drag und drop“ in die jeweiligen Module gezogen.

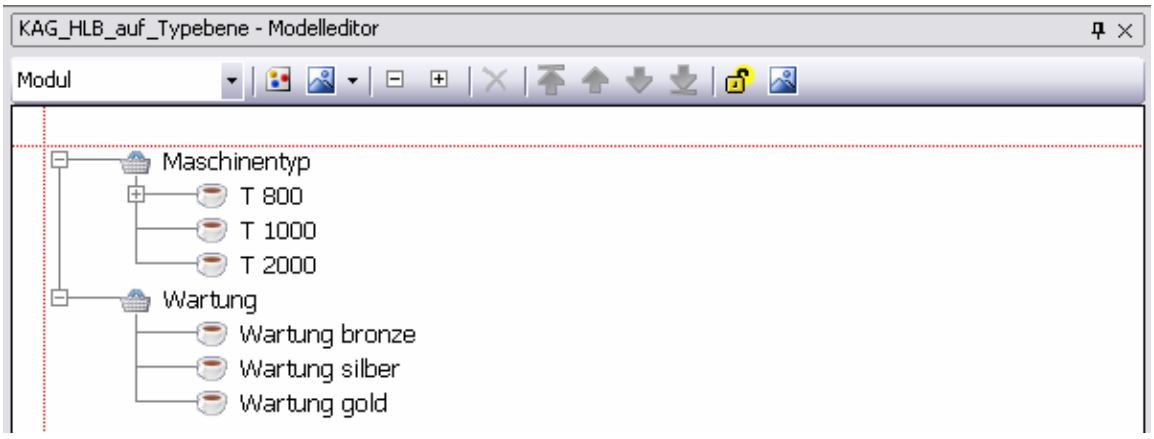


Abb. 4.27: Erstellen von Modulen

Bei der Erstellung von Modulen sollte besonders darauf geachtet werden, dass Leistungen innerhalb eines Moduls über die gleichen Leistungseigenschaften verfügen. Dies ist nötig, damit ein korrekter

Vergleich der jeweiligen Alternativen im Konfigurator durchgeführt werden kann. Dies gilt ebenfalls für Präferenzen von Leistungen innerhalb eines Moduls. Abb. 4.28 visualisiert die drei Maschinen mit jeweils gleichen Leistungseigenschaften. Auf Präferenzwerte wurde aufgrund der Übersichtlichkeit verzichtet.

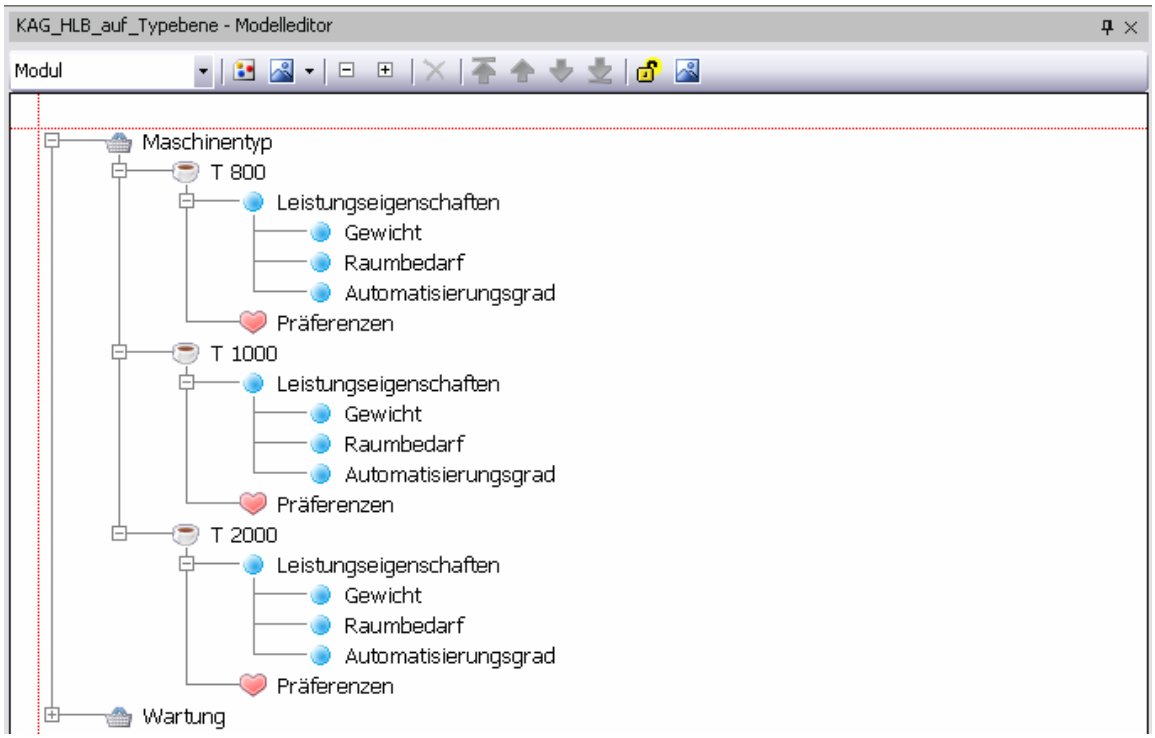


Abb. 4.28: Gleiche Leistungseigenschaften und Präferenzen

Hybrides Leistungsbündel auf Typebene

Wie in Kapitel 2 bereits erläutert wurde, enthält ein HLBAT alle möglichen Konfigurationen eines hybriden Leistungsbündels. Um ein neues HLBAT anzulegen, ist im Modelleditor in den Kontext Hybrides Leistungsbündel auf Typebene zu wechseln. Mit einem Rechtsklick kann unter dem Menüpunkt „+ Hybrides Leistungsbündel auf Typebene“ ein neues HLBAT angelegt werden. Bei der Wahl des Namens ist ein möglichst „überordnender“ Name zu wählen, denn dieser kennzeichnet das gesamte Leistungsbündel. Im Beispielszenario wurde dem HLBAT der Name „T-Serie“ gegeben. Über das Attribut Beschreibung lässt sich in Textform eine nähere Beschreibung des Leistungsbündels festhalten. Nach der Sprachdefinition eines HLBAT aus Kapitel 2 kann ein HLBAT Module, Lebenszyklusphasen und Regeln enthalten. In einem ersten Schritt müssen zuerst Module in das HLBAT eingefügt werden. In Abb. 4.29 wurden beispielhaft die beiden Module Maschinentyp und Wartung aus dem Kontext Module in das HLBAT „T-Serie“ im Kontext Hybrides Leistungsbündel auf Typebene eingefügt. Die beiden Module enthalten die oben eingefügten Leistungen, die wie bereits erläutert wurde, Alternativen zueinander darstellen.

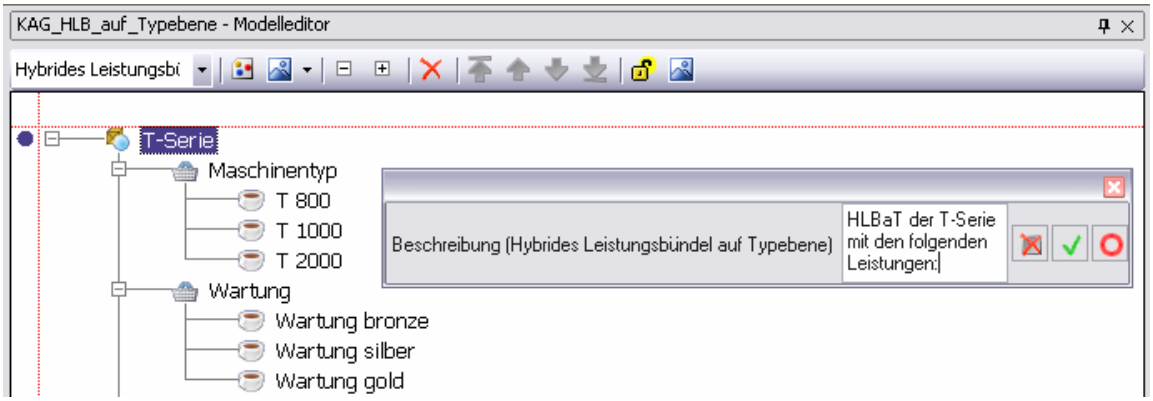


Abb. 4.29: HLBAT und Attribut Beschreibung

Nach dem Einfügen der Module erfolgt das Anlegen der Lebenszyklusphasen aus dem Kontext Lebenszyklusphasen. Die Lebenszyklusphasen die unter einem HLBAT angelegt werden können, sind aus Sicht des Kunden zu sehen. In einem ersten Schritt lassen sich die Lebenszyklusphasen wieder in Intervalle untergliedern. Innerhalb der Intervalle stehen sämtliche Leistungen eines hybriden Leistungsbündels, die für die jeweilige Phase Kosten verursachen können. Die Intervalle Entwurf und Fertigung werden vom Konfigurator werden ignoriert.

Lebenszyklusphasen werden aus dem Kontext Lebenszyklusphasen per „drag and drop“ hinzugefügt. In Abbildung 4.30 wurden im HLBAT T-Serie die drei Lebenszyklusphasen Kauf, Nutzung und Nachnutzung angelegt. Die Nutzungsphase besteht aus den Intervallen „1. Jahr“, „2. Jahr“ und „3. Jahr“. In diesem Intervall wurden die drei Leistungen „Wartung bronze“, „Wartung silber“ und „Wartung gold“ hinzugefügt. Aus Kundensicht und innerhalb dieses Beispiels können genau diese drei Leistungen für Auszahlungen innerhalb dieses Intervalls sorgen. Falls es nötig ist, eine neben den Kosten einer Leistung existierende Auszahlung zu berücksichtigen, so kann dies über das Attribut „zusätzlicheAuszahlungProIntervall“ hinterlegt werden. In Abb. 4.30 wurde exemplarisch eine zusätzliche Auszahlung in Höhe von 2000 bei der Leistung „Wartung silber“ hinterlegt.

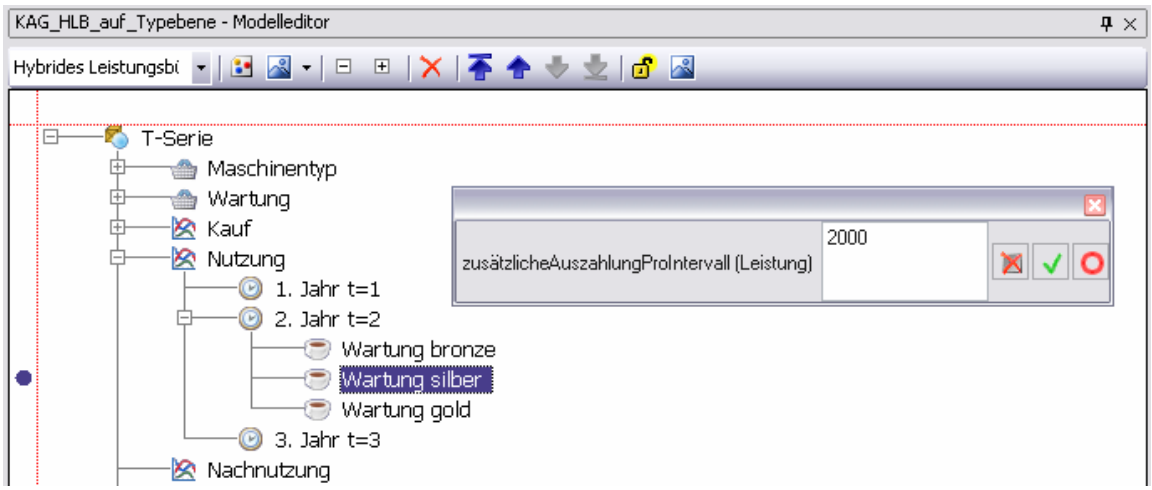


Abb. 4.30: Lebenszyklusphasen und Intervalle im HLBAT

Letztendlich hat der Modellierer die Möglichkeit Regeln innerhalb eines HLBAT zu modellieren. Wie in Kapitel 2 erläutert wurde, kann der Modellierer zwischen zwei Regeln wählen (Bedingt- und

Schließt aus –Regel). Die Modellierung einer Regel erfolgt durch einen Rechtsklick auf das Hybride Leistungsbündel unter dem Menüpunkt „+Regeln“. Unter diesem neu angelegten Regelcontainer können nun benötigte Regeln angelegt werden. Dies erfolgt wiederum durch einen Rechtsklick auf den Regelcontainer, Menüpunkt „+Regeln“. Nachdem die Regel mit einem Namen versehen worden ist, muss einer der beiden Kantentyp zugewiesen werden. Durch einen Rechtsklick auf die Regel, kann im Menüpunkt Kantentyp zwischen zwei Kantentypen für die beiden Regelarten wählen. In einem nächsten Schritt können Auslöser und Konsequenz der Regel festegelegt werden. Das Hinzufügen von Auslöser- und Konsequenzknoten erfolgt durch einen Rechtsklick auf die Regel unter dem Menüpunkt „+Auslöser“ bzw. „+Konsequenz“. Jede Regel muss jeweils genau einen dieser Container enthalten. Nach der Sprachdefinition des Regelkonstruktes hat der Modellierer die Möglichkeit entweder Leistungen direkt als Auslöser bzw. Konsequenz zu wählen, oder entsprechende Module mit speziellen Leistungseigenschaften zu hinterlegen.

In Abb. 4.31 ist eine Bedingt-Regel visualisiert. Auslöser der Regel ist die Wahl des Maschinentyps T-2000. Konsequenz für die Wahl dieses Maschinentyps ist die Leistung Wartung gold. Vereinfacht bedeutet dies, dass wenn der Kunde die Leistung T-2000 im Konfigurator auswählt, die Wartung gold automatisch benötigt und damit auch hinzugefügt wird. Eine Schließt aus Regel lässt sich analog zur Bedingt-Regel modellieren.

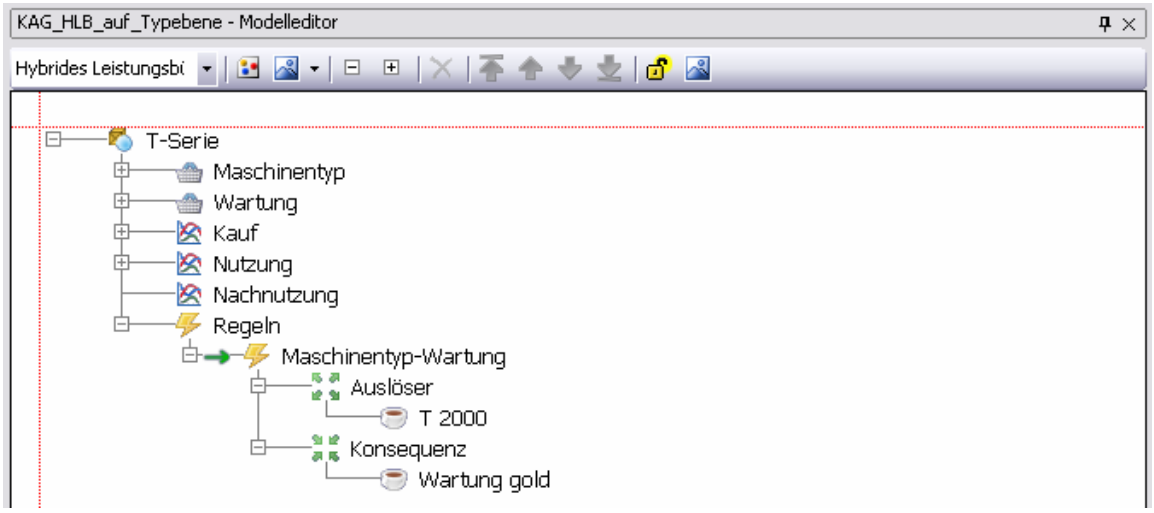


Abb. 4.31: Regelkonstrukt Maschinentyp-Wartung

Neben Leistungen können nach der H2-Sprachdefinition auch Leistungseigenschaften innerhalb eines Moduls aus Auslöser genutzt werden. In Abb. 4.32 wurde eine neue Regel mit dem Namen „Automatisierungsgrad-Wartung“ angelegt. Auslöser dieser Regel ist die Leistungseigenschaft Automatisierungsgrad. Als Konsequenz wurde Wartung silber gewählt. Die Bedeutung dieser Regel lässt sich beispielsweise wie folgt zusammenfassen: Verfügt der ausgewählte Maschinentyp über die Leistungseigenschaft „hoher Automatisierungsgrad“, so ist immer die Wartung silber die Konsequenz.

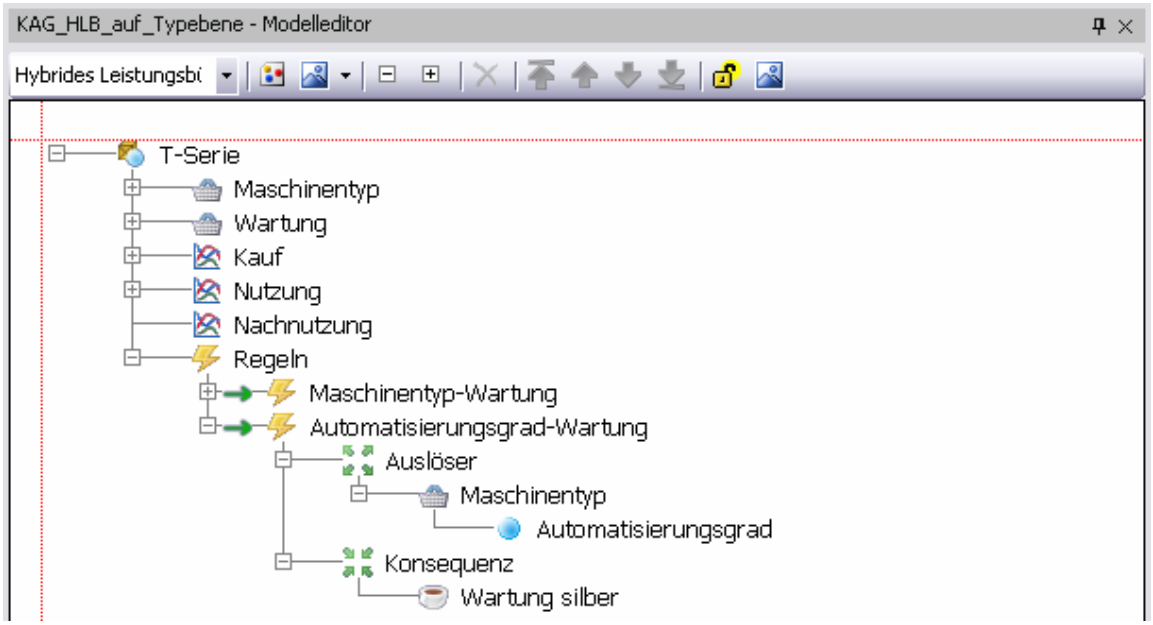


Abb. 4.32: Regelkonstrukt Automatisierungsgrad-Wartung

4.1.5 Assoziationsregeln

Für das Gemeinschaftliche Filtern werden Assoziationsregeln benötigt. Diese werden im HLBAI angelegt und sollen zukünftig automatisch generiert werden. Um eine neue Assoziationsregel manuell zu erstellen ist im Modelleditor der Kontext Assoziationsregeln auszuwählen. Durch einen Rechtsklick lässt sich über den Menüpunkt „+Assoziationsregel“ eine neue Regel erzeugen. In Abb. 4.33 wurde die Assoziationsregel Wartungsbedarf erzeugt. Durch einen Rechtsklick auf die erzeugte Regel müssen im nächsten Schritt ein Auslöser und eine Konsequenz für die Regel festgelegt werden. Ein Auslöser (Leistung) befindet sich immer innerhalb eines Moduls. Deshalb wird zuerst das entsprechende Modul angegeben, in dem sich die auslösende Leistung befindet. In einem nächsten Schritt wird die auslösende Leistung innerhalb des Moduls hinzugefügt. Analog erfolgt das Anlegen einer Konsequenz. Jede Assoziationsregel hat genau eine Leistung als Konsequenz, kann aber mehrere Auslöser haben. Über einen Rechtsklick auf die Assoziationsregel lassen sich die beiden Attribute Confidence und Support anlegen. Diese wurden im Beispiel mit den Werten 0,5 bzw. 0,3 versehen.

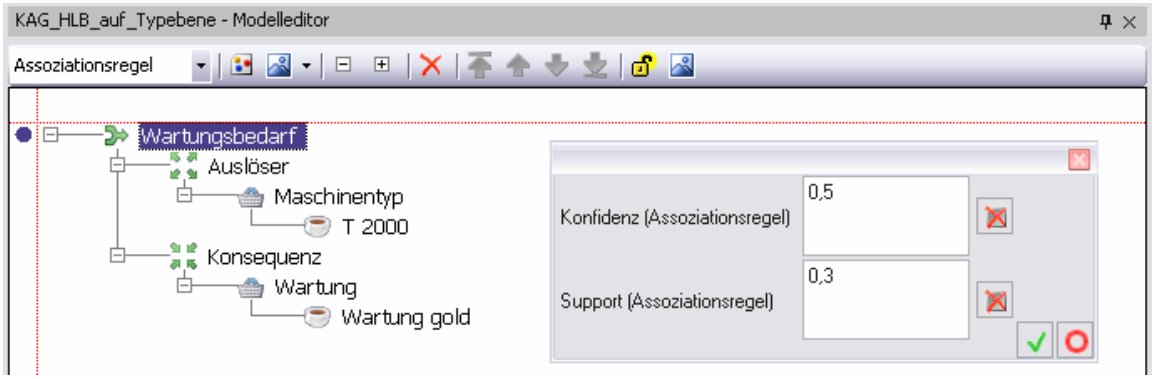


Abb. 4.33: Assoziationsregeln

4.2 Anleitung zur Nutzung des Kostenkalkulationstools

Der Aufruf des Kostenkalkulationstools erfolgt, wie in Abb. 4.34 dargestellt, über einen Rechtsklick auf das zu analysierende Modell.

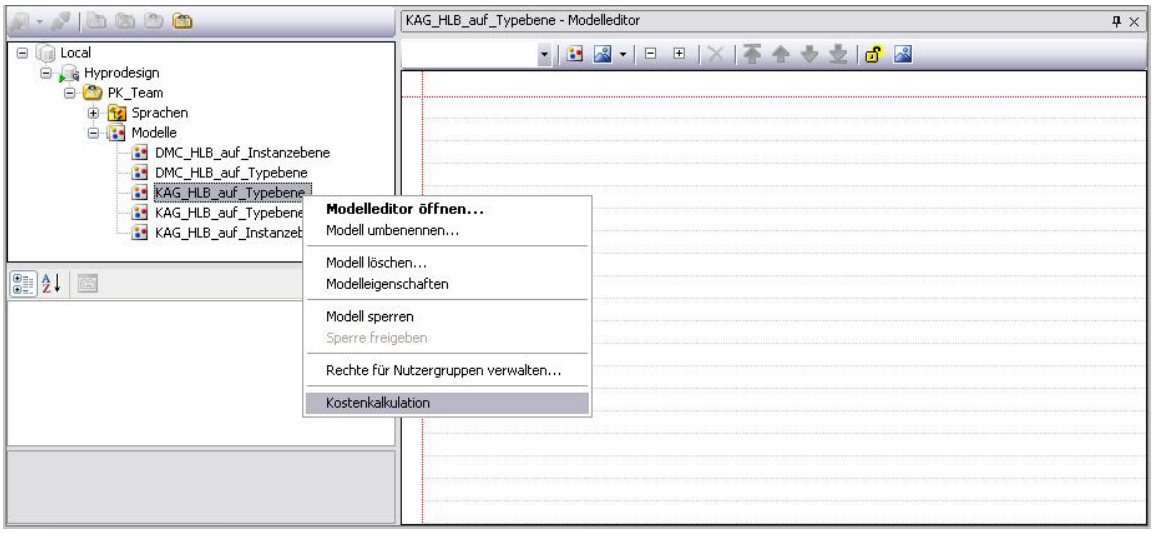


Abb. 4.34: Kostenkalkulation starten

Anschließend muss zunächst ein Modul als Träger verschiedener Leistungen ausgewählt werden.

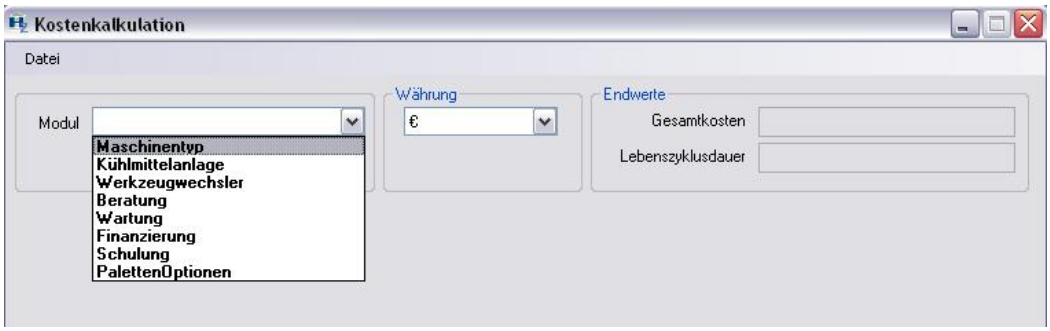


Abb. 4.35: Auswahl eines Moduls

Nachdem ein Modul ausgewählt wurde, wird im nächsten Schritt die gewünschte zu analysierende Leistung ausgewählt, wie in Abb. 4.36 dargestellt.

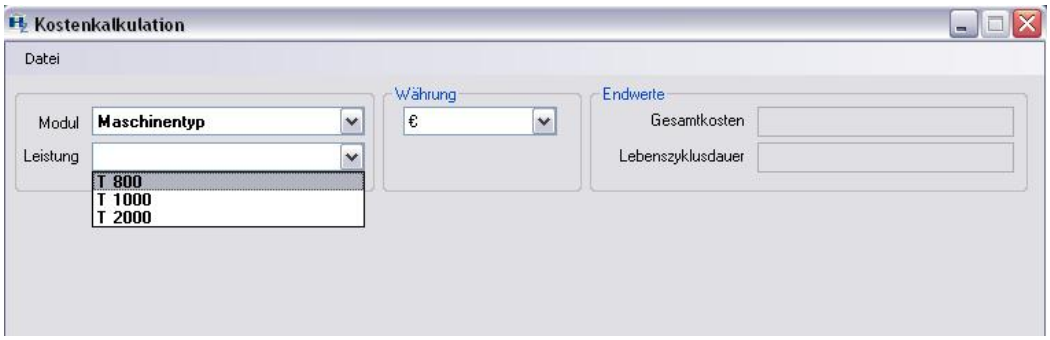


Abb. 4.36: Auswahl einer Leistung

Nach der Auswahl einer Leistung erscheint eine detaillierte Übersicht, anhand welcher der Anwender weitere Informationen über die durch die Leistungserstellung verursachten Kosten erfahren kann.

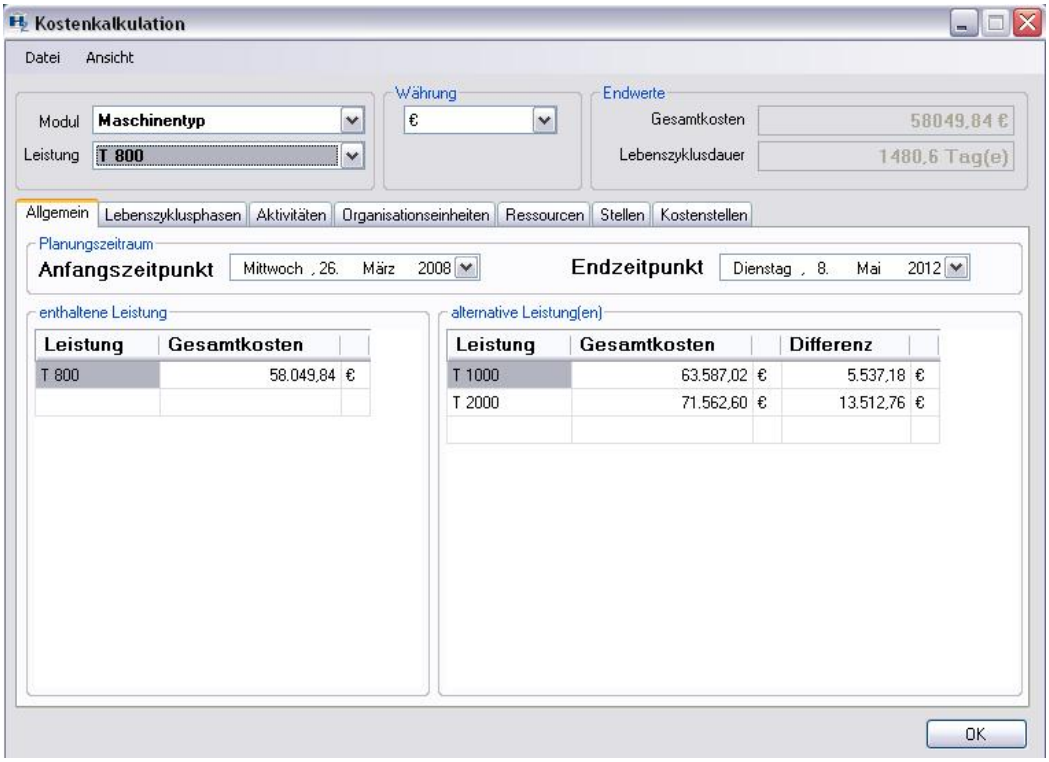


Abb. 4.37: Gesamtansicht Kostenkalkulation

Auf den ersten Blick wird ersichtlich, welche alternativen Leistungen in dem ausgewählten Modul enthalten sind und welche Kosten diese verursachen würden. Zwecks besserer Planbarkeit ist außerdem wie in Abb. 4.37 ersichtlich, der Planungszeitraum visualisiert. Dort kann der geplante Anfangszeitpunkt der Investition angegeben werden und es wird automatisch der Endzeitpunkt ermittelt.

Die Nutzung eines DateTimePickers ermöglicht die Auswahl eines Anfangszeitpunktes oder Endzeitpunktes der Investition, wie in Abb. 4.38 angezeigt.

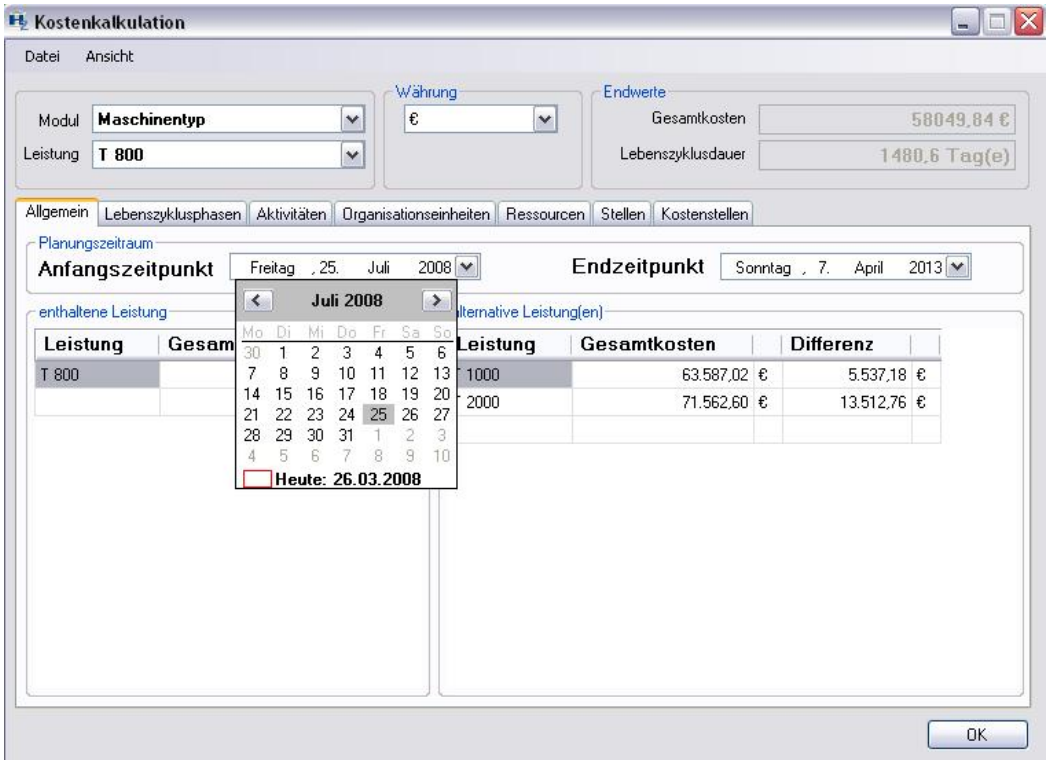


Abb. 4.38: Angabe des Planungszeitraumes einer Leistung

Unter dem Reiter Lebenszyklusphasen können die Kosten einzeln auswählbarer Lebenszyklusphasen angezeigt werden und es besteht des Weiteren die Möglichkeit zwischen der Anzeige der Kosten und der Anzeige der Dauer der Lebenszyklusphasen umzuschalten.

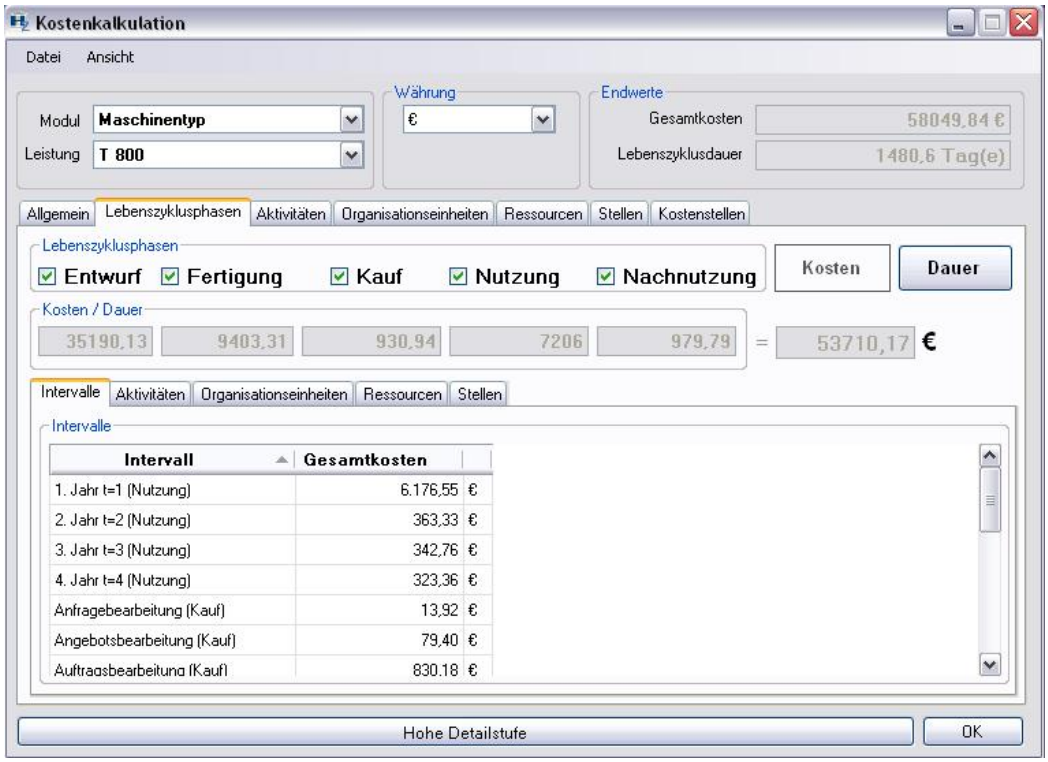


Abb. 4.39: Kosten und Dauer der Lebenszyklusphasen einer Leistung

In der unteren Hälfte des Fensters des Kostenkalkulationstools können wahlweise die Kosten der Intervalle, der Aktivitäten, der Organisationseinheiten, der Ressourcen oder der Stellen angezeigt

werden. So kann für jede beliebige Kombination an Lebenszyklusphasen schnell und übersichtlich nachvollzogen werden, welche Kosten wofür anfallen.

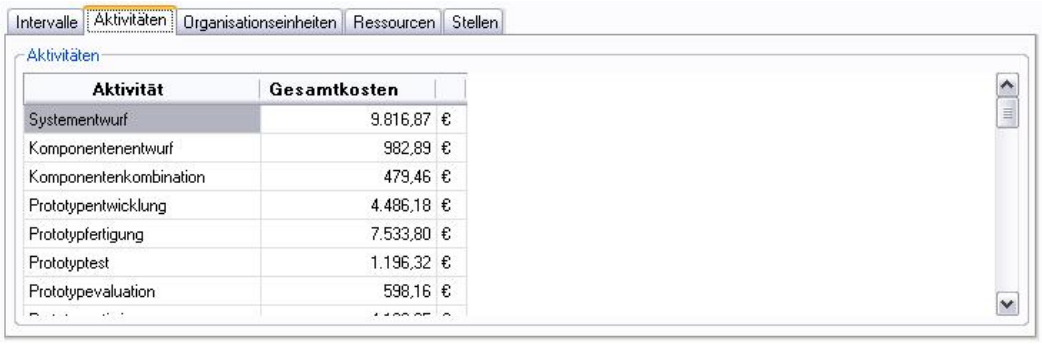


Abb. 4.40: Untergliederung von Lebenszyklusphasen

Neben der in Abb. 4.40 möglichen Ansicht kann zusätzlich auf eine höhere Detailstufe umgeschaltet werden, wie in Abb. 4.41 dargestellt.

Abb. 4.41: Wechsel der Detailstufe

In der niedrigen Detailstufe, in Abb. 4.40 dargestellt, werden die Aktivitäten aufgelistet, die in der ausgewählten Leistung enthalten sind. In der hohen Detailstufe hingegen, welche durch den Button am unteren Rand des Applikationsfensters ausgewählt werden kann, werden die Einzel- und Gesamtkosten aller in den Aktivitäten enthaltenen Ressourcen und Stellen angezeigt.

Außer den zuvor vorgestellten Lebenszyklusphasen und der allgemeinen Darstellung, existieren weiterhin auswählbare Reiter für Aktivitäten, Organisationseinheiten, Ressourcen, Stellen und Kostenstellen. Diese Darstellung ist unabhängig von den ausgewählten Lebenszyklusphasen, siehe Abb. 4.42, Auswahl Aktivitäten.

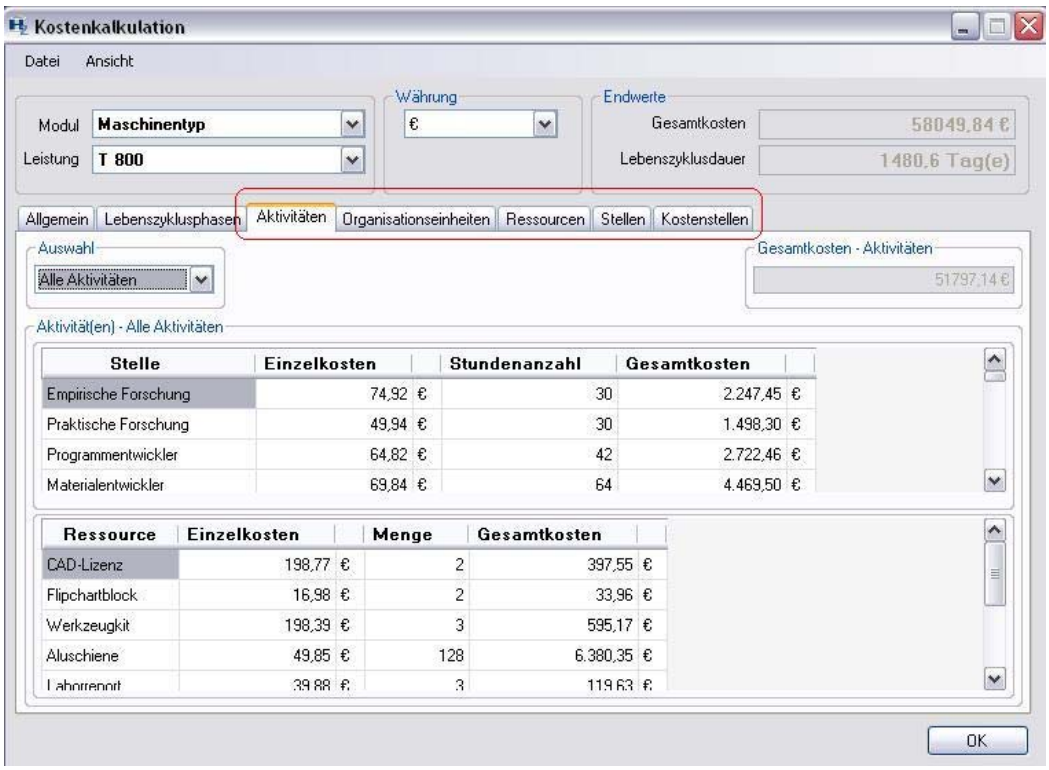


Abb. 4.42: Auswahl lebenszyklusunabhängiger Kostenkalkulationsstufen

Unter den in Abb. 4.43 rot eingerahmten Reitern kann jeweils eine Auswahl vorgenommen werden, welche Aktivitäten, Organisationseinheiten oder Ressourcen etc. bezüglich der Kosten gelistet werden sollen.

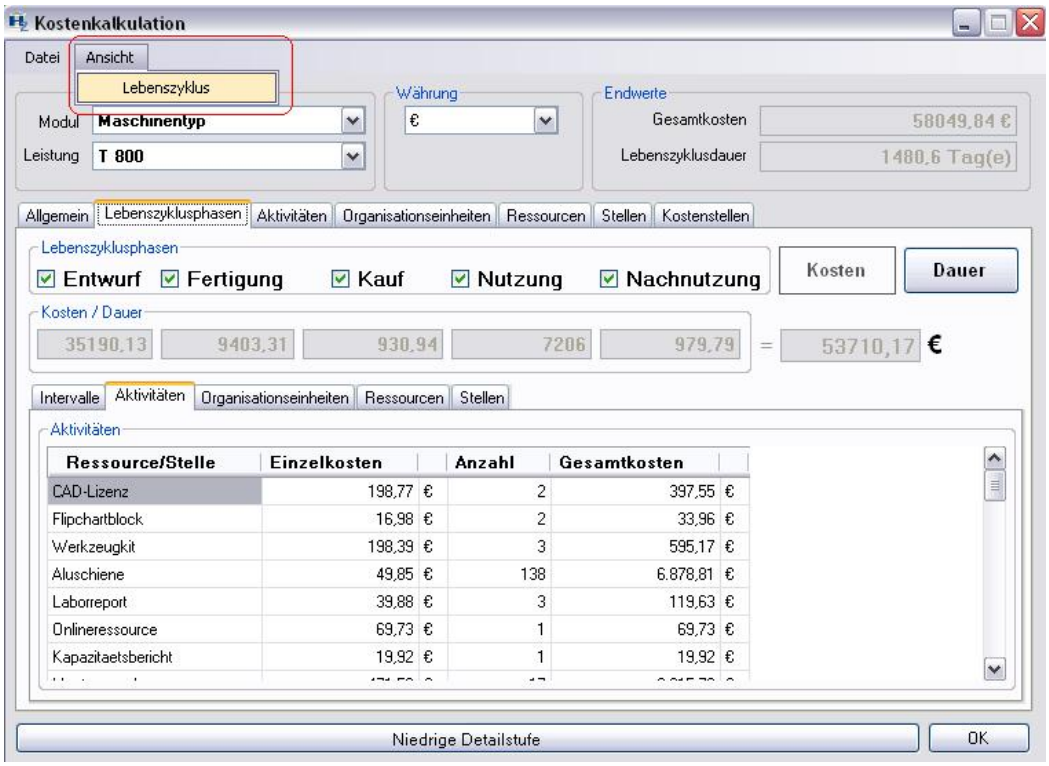


Abb. 4.43: Lebenszyklusansicht

Abb. 4.44 zeigt, wie der Anwender die Ansicht auf die Lebenszyklusansicht, in welcher eine detaillierte tabellarische Darstellung lebenszyklusphasenbezogener Kosten möglich ist, wechseln kann.

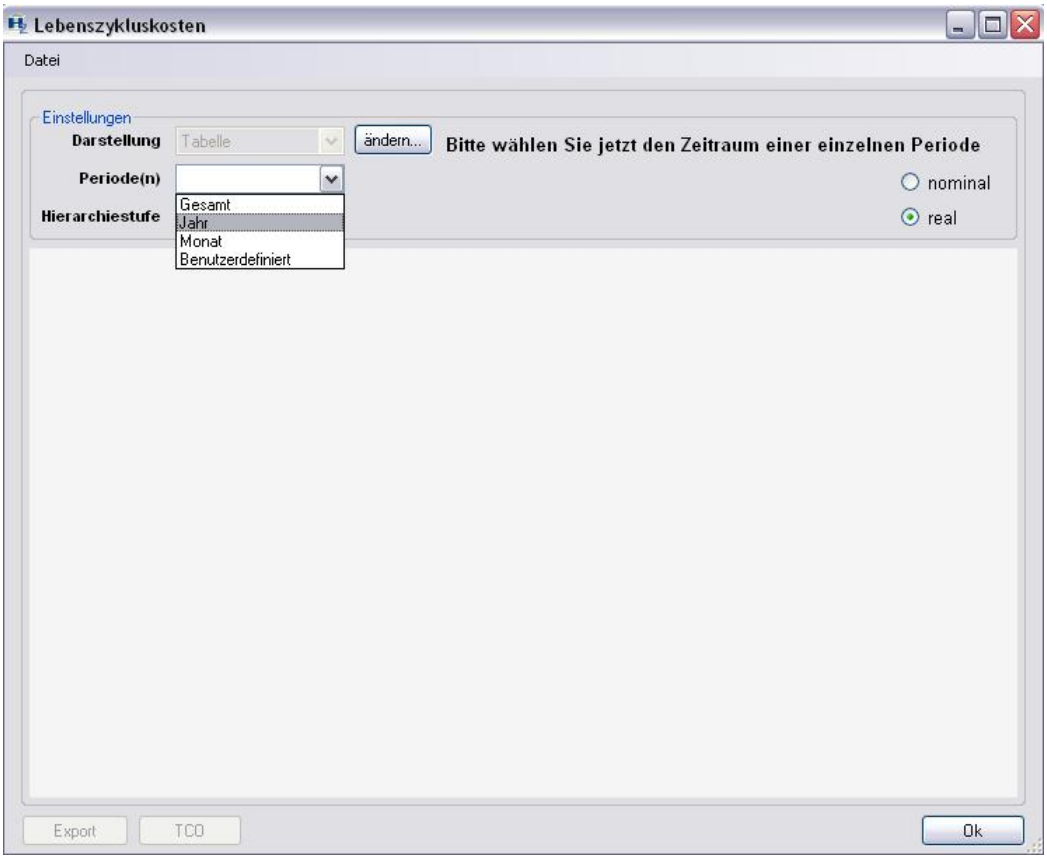


Abb. 4.44: Auswahl der Perioden

Wie in Abb. 4.44 dargestellt, können verschiedene Perioden für die Darstellung der Lebenszykluskosten ausgewählt werden. Entweder wird die Totalperiode als *Gesamt* ausgewählt oder eine Unterteilung der Perioden nach Jahren, Monaten oder nach benutzerdefinierter Eingabe.

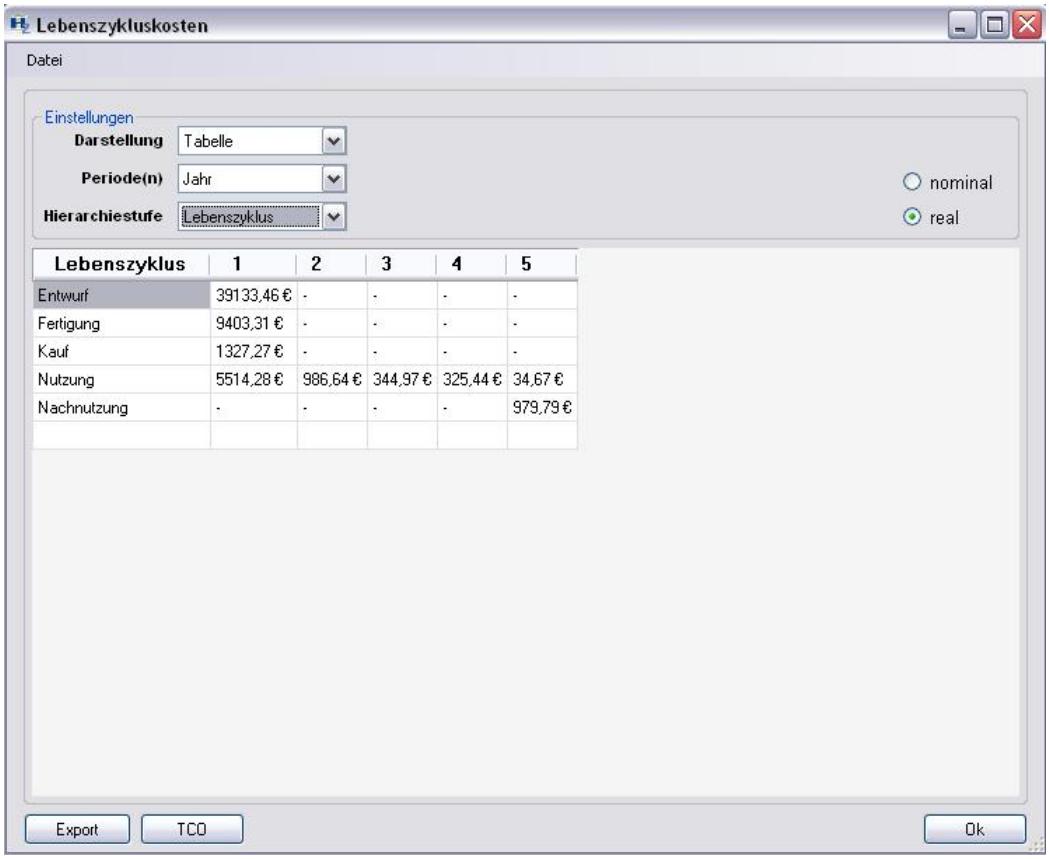


Abb. 4.45: Auswahl der Hierarchiestufe

Nach Angabe der gewünschten Periode(n) wird eine Hierarchiestufe ausgewählt. Zu unterscheiden ist zwischen Leistung, Lebenszyklus, Intervall oder Stelle/Ressourcen. Je nach Auswahl wird die tabellarische Darstellung geändert. Im angegebenen Beispiel in Abb. 4.45 ist als Hierarchiestufe der Lebenszyklus ausgewählt, welcher in der Tabelle gegliedert nach Entwurf, Fertigung, Kauf Nutzung und Nachnutzung aufgeteilt wird.

Des Weiteren ist ein Export nach Excel möglich. Einerseits kann die tabellarische Darstellung durch den Export-Button exportiert werden und andererseits besteht die Möglichkeit, diese Daten direkt in einen vollständigen Finanzplan für die Analyse der Total Cost of Ownership übertragen zu lassen.

4.3 Anleitung zur Nutzung des Leistungskonfigurators

Beim Aufruf der Webseite wird dem Kunden die Startseite des Konfigurators mit einem entsprechenden Text des Anbieters präsentiert.

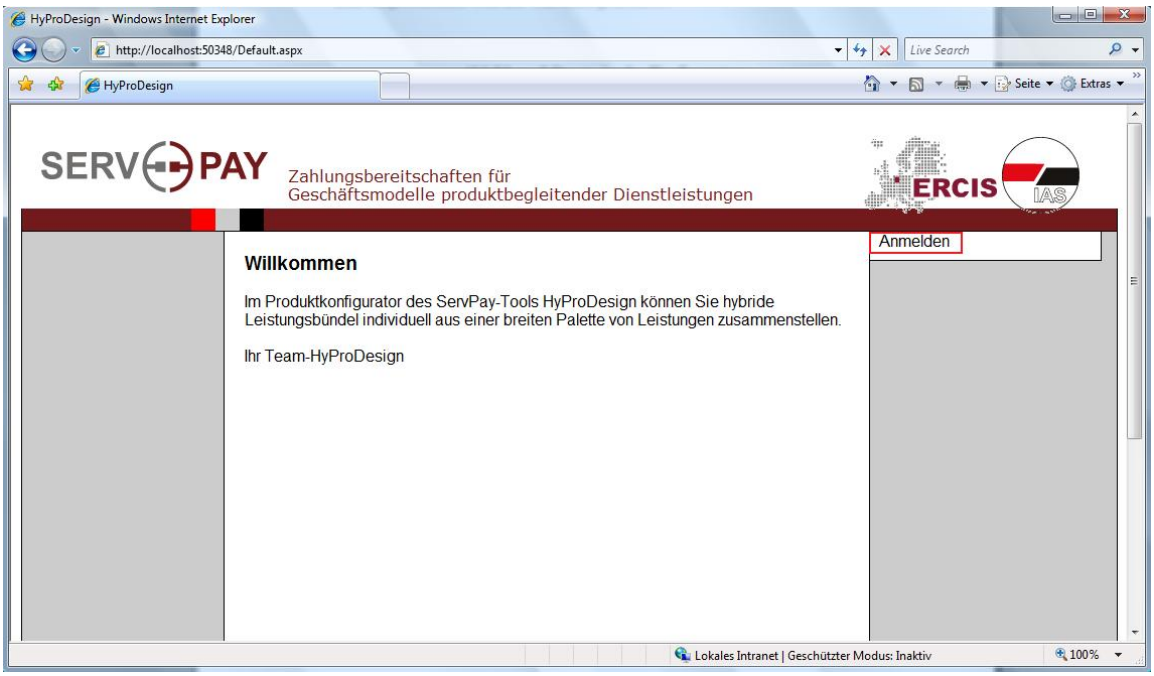


Abb. 4.46: Startseite des Konfigurators

Über die Anmelden-Schaltfläche gelangt der Kunde zur Login-Seite. Auch bei einem Timeout auf dem Webserver wird der Kunde auf diese Seite weitergeleitet.

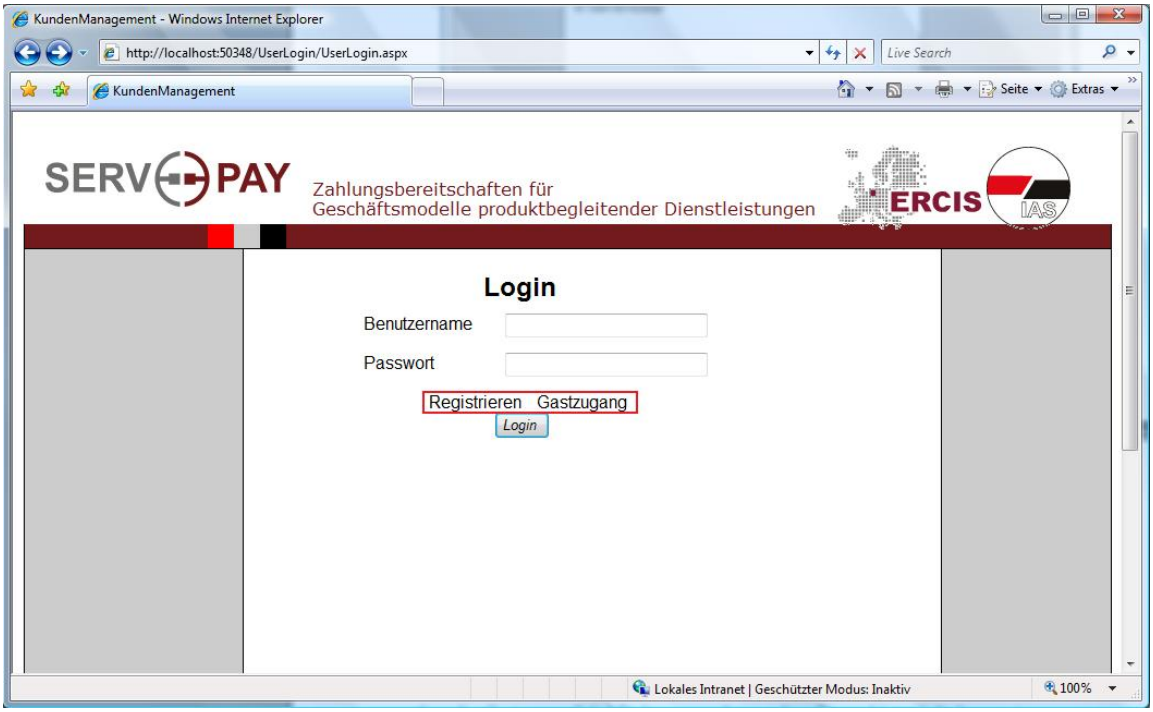


Abb. 4.47: Login-Seite des Konfigurators

Hier können sich bereits registrierte Nutzer mit ihrem Benutzernamen und ihrem Passwort anmelden. Alternativ kann für die Konfiguration auch ein ggf. vorhandener Gastzugang über die Gastzugang-Schaltfläche verwendet werden. Dieser bietet jedoch nur eingeschränkten Zugriff auf die Funktionen des Konfigurators und sollte nur genutzt werden, wenn ein Kunde sich einen ersten Eindruck verschaffen will, ohne bereits persönliche Daten anzugeben. Über die Registrieren-Schaltfläche

kann ein neuer Account angelegt werden. Der Kunde wird dafür auf die im Folgenden beschriebene Seite weitergeleitet, auf der seine persönlichen Daten erfasst werden.

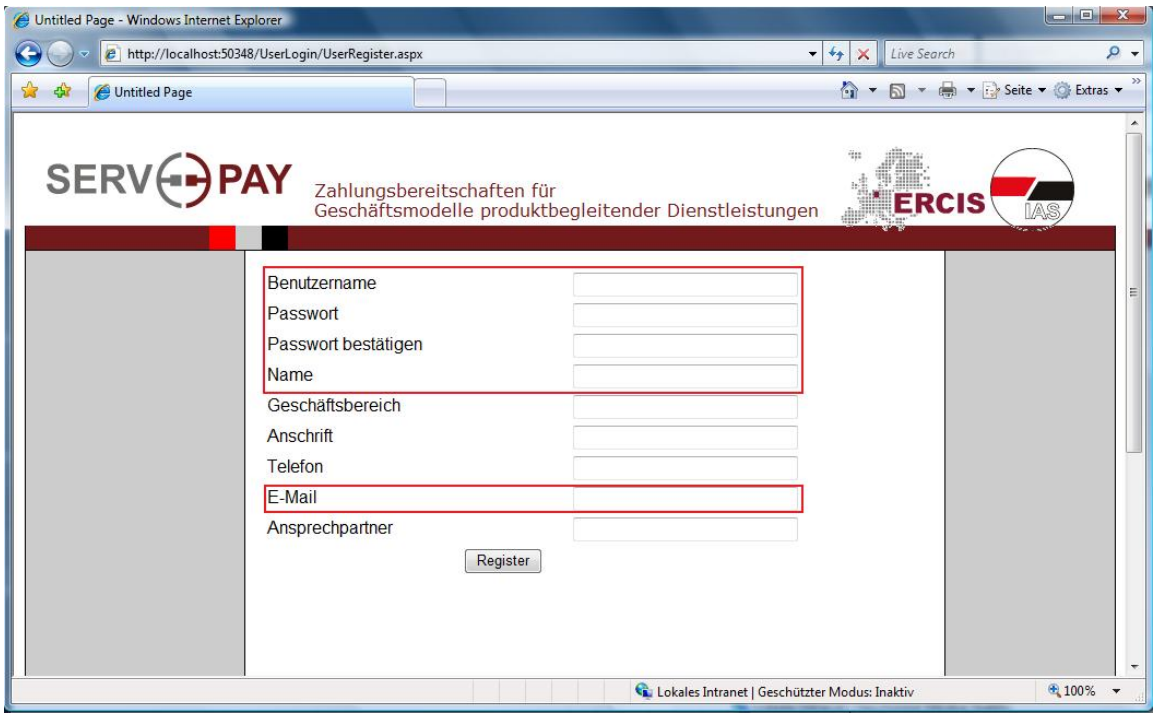


Abb. 4.48: Registrierung im Konfigurator

Notwendig für eine Registrierung sind neben dem Nutzernamen und einem Passwort der Name des Kunden und eine E-Mail-Adresse. Nutzername und Passwort dürfen nur alphanumerische Zeichen enthalten und müssen mindestens 1 bzw. 5 Zeichen und maximal 20 bzw. 16 Zeichen lang sein. Eine Überprüfung der E-Mail-Adresse oder eine Aktivierung des Accounts per E-Mail erfolgt nicht[20]. Bei Eingaben, die die Konventionen nicht einhalten oder weiteren Fehlern bei der Registrierung, z.B. wenn ein Nutzername bereits vergeben ist, wird eine entsprechende Fehlermeldung auf der Seite angezeigt. Die eingegebenen Daten bleiben dabei erhalten und können dann angepasst werden. Zusätzlich zu den oben genannten erforderlichen Daten können noch die Adresse, der Geschäftsbereich, die Telefonnummer sowie Daten zu einem Ansprechpartner eingegeben werden, um eine möglichst effiziente Kommunikation zwischen Kunde und Anbieter zu ermöglichen.

Nach erfolgreicher Registrierung oder Anmeldung wird der Kunde auf seine Profilseite weitergeleitet. Diese bietet die Möglichkeit, sowohl die persönlichen als auch die accountbezogenen Daten zu ändern. Dafür ist immer die Eingabe des Passwortes erforderlich. Des Weiteren kann aus einer Liste der vom Kunden hinterlegten Konfigurationen eine bestimmte Konfiguration geladen werden. Dies wird in einem späteren Abschnitt ausführlich beschrieben. Über einen Klick auf einen vom Anbieter bereitgestellten Link, im Beispiel das ServPay-Logo im Header, gelangt der Kunde zum Konfigurator für das im Link angegebene Leistungsbündel.

[20] Grund hierfür ist das geringe Gefährdungspotenzial, welches von einem böswilligen Nutzer ausgeht, da kein Kontakt zu anderen Nutzern aufgebaut wird und keine Manipulationsmöglichkeiten bezüglich der Daten im H2 Toolset Server bestehen.

Beim ersten Aufruf des Konfigurators durch den Nutzer erfolgt vor der eigentlichen Konfiguration die Erhebung der individuellen Präferenzen des Kunden. Diese dienen als Datenbasis für die wissensbasierte Vorschlagsgenerierung im Konfigurator. Für jede in den Leistungen des HLBAT vorhandene Präferenz wird die Wichtigkeit für den Kunden und seine Anforderung an die Präferenz erfasst. Zuerst wird dabei die Wichtigkeit abgefragt.

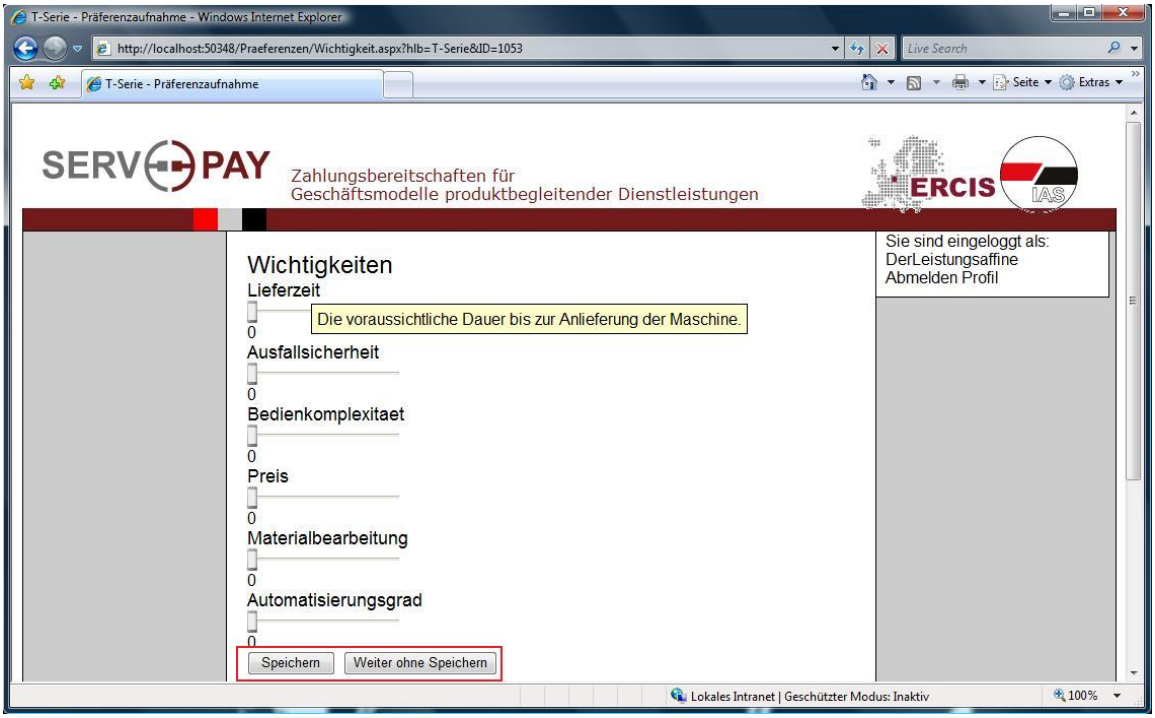


Abb. 4.49: Aufnahme der individuellen Wichtigkeiten

Die Eingabe der individuellen Wichtigkeit erfolgt über einen Schieberegler. Der Wertebereich für jede Präferenz geht von 0 (nicht wichtig) bis 100 (sehr wichtig). Zu jeder Präferenz wird eine Beschreibung angezeigt, wenn sich die Maus über dem Namen der Präferenz befindet. In Abb. 4.49 wird dies für die Präferenz „Lieferzeit“ angezeigt. Die Eingabe der Wichtigkeiten wird über die Speichern-Schaltfläche abgeschlossen. Es ist auch möglich, die Aufnahme der Wichtigkeiten über die WeiterOhneSpeichern-Schaltfläche abzubrechen. Dann werden ggf. vorhandene bereits gespeicherte Werte und ansonsten Wichtigkeiten von 0 verwendet. Danach erfolgt die Aufnahme der Anforderungswerte für die einzelnen Präferenzen.

Abb. 4.50: Aufnahme der individuellen Präferenzanforderungen

Für jede Präferenz werden nun die Anforderungen für die vom Anbieter modellierten Präferenzwerte gesetzt. Dabei sollte die Summe der den Präferenzwerten zugeordneten Werte 100 ergeben, um eine möglichst passende Vorschlagsgenerierung zu erreichen. Der jeweils verbleibende Rest wird für jeden Präferenzwert angezeigt. Die Gewichtung der einzelnen Werte erfolgt dabei relativ zueinander. Ein ggf. vorhandener Rest verfällt. Eine Anforderung von 0 bedeutet hierbei, dass es dem Kunden egal ist, wenn eine Leistung den Präferenzwert erfüllt, ein Wert von 100 drückt aus, dass ausschließlich dieser Präferenzwert vom Kunden gewünscht wird. Zu jedem Präferenzwert wird bei Bedarf eine Beschreibung angezeigt, wenn sich die Maus über dem Namen des Präferenzwertes befindet. In Abb. 4.50 wird dies für den Präferenzwert „kurzfristige Lieferung“ gezeigt. Die Wichtigkeit der jeweiligen Präferenz wird ebenfalls angezeigt (blau markiert) und kann über einen Linksklick nochmals geändert werden. Über die Speichern- bzw. WeiterOhneSpeichern-Schaltfläche gelangt der Kunde dann zur eigentlichen Konfiguration.

Abb. 4.51: Module und Leistungen im Konfigurator

Der Konfigurator ist wie folgt aufgebaut: Eine Abmeldung oder die Rückkehr zum Profil ist über die Schaltflächen in der Status-Box auf der rechten Seite möglich. Das Menü auf der linken Seite dient der Navigation zwischen den Modulen des hybriden Leistungsbündels. In der Mitte der Seite werden die im aktuell betrachteten Modul, in Abb. 4.51 das Modul „Maschinentyp“, enthaltenen Leistungen in einer Liste angezeigt.

Abb. 4.52: Leistung im Konfigurator

Für jede Leistung, hier die Maschine „T 2000“, werden neben dem Namen und einem Bild der Beschreibungstext, der Preis (blau markiert) und die Leistungseigenschaften (grün markiert) angezeigt. Die Beschreibung ist ein vom Anbieter eingegebener Text und kann auch durch HTML-Tags weiter formatiert werden, um den Text besser zu strukturieren oder ansprechender zu gestalten. Der Preis wird bei einmalig anfallenden Leistungen, die beim Kauf gezahlt werden, - in der Regel Sachleistungen wie der Maschine - angegeben. Bei Leistungen, für die Zahlungen über den Lebenszyklus verteilt anfallen, z.B. bei Dienstleistungen wie der Wartung, wird kein Preis angezeigt, sondern auf die Lebenszyklusdarstellung verwiesen. Die Leistungseigenschaften bestehen aus dem Namen der Eigenschaft, einem Wert und dessen Mengeneinheit, z.B. ein Gewicht von 6 Tonnen. Sie dienen der Vergleichbarkeit der Leistungen anhand von kategorisierten und gleich skalierten Eigenschaften. Alle Leistungen in einem Modul sollten die gleichen Leistungseigenschaften aufweisen, um die Entscheidung für eine Leistung zu vereinfachen. Neben dieser Vergleichmöglichkeit wird auf die anfangs erhobenen individuellen Präferenzen des Kunden zurückgegriffen, um Leistungen zur Auswahl vorzuschlagen. Jede Leistung erhält eine Bewertung mittels eines Algorithmus aus dem Bereich der wissensbasierten Vorschlagsgenerierung[21]. Anhand dieser Bewertung werden die Leistungen sortiert, wobei immer die Leistung mit der höchsten Bewertung an erster Stelle aufgelistet wird.

[21] Eine detaillierte Beschreibung des Algorithmus und sowie eine Diskussion alternativer Algorithmen befindet sich in Abschnitt 0 des Anhangs

Abb. 4.53: Bewertung einer Leistung

Die Bewertung setzt sich aus den Teilbewertungen der von der Leistung erfüllten Präferenzwerte, in Abb. 4.53 ist dies unter anderem die Präferenz „Lieferzeit“, zusammen. Die Teilbewertungen werden angezeigt, wenn sich die Maus über der Bewertung befindet. Eine Teilbewertung von 1 würde bedeuten, dass die Leistung die Anforderungen des Kunden an die Präferenzwerte für die entsprechenden Präferenz genau erfüllt, eine Teilbewertung von 0 bildet analog eine Nichterfüllung der Anforderungen ab. Je näher der Wert der Teilbewertung also bei 1 liegt, desto besser werden die Anforderungen des Kunden für diese Präferenz von der Leistung erfüllt. Die Teilbewertungen werden dann mit Hilfe der erhobenen Wichtigkeiten zur Bewertung verdichtet. Für den Wertebereich dieser gilt wiederum, je näher der Wert an 1 liegt, desto besser passt die Leistung zu den Präferenzen des Kunden.

Zusätzlich wird die Entscheidungsfindung des Kunden durch den Einsatz von Assoziationsregeln unterstützt. Diese Technik aus dem Bereich der gemeinschaftlichen Vorschlagsgenerierung analysiert alle gespeicherten Konfigurationen aller Kunden und identifiziert Kombinationen von Leistungen, die besonders häufig zusammen gewählt werden. Diese werden dem Kunden abhängig von den von ihm gewählten Leistungen angezeigt. Abb. 4.54 zeigt die angezeigten Assoziationsregeln bei der Auswahl der Maschine „T 2000“.

Abb. 4.54: Assoziationsregeln im Konfigurator

Die beiden Assoziationsregeln schlagen dem Kunden basierend auf seiner Entscheidung für die „T 2000“ und der bekannten Entscheidungen anderer Kunden, die ebenfalls diese Maschine gewählt haben, nun die Auswahl einer Intensivschulung und die Auswahl eines Roboterarmes als Werkzeugwechsler vor. Die Sortierung der angezeigten Assoziationsregeln ergibt sich aus ihrer Bewertung, welche über eine Auswertung der relativen und absoluten Häufigkeiten der Kombinationen errechnet wird[22]. Über einen Linksklick auf die in den Assoziationsregeln enthaltenen Leistungen gelangt der Kunde zu den Modulen, welche die vorgeschlagenen Leistungen enthalten. Neben den Assoziationsregeln werden auch vom Anbieter festgelegte Regeln ausgewertet.

22 Zum Konzept der Assoziationsregeln und ihrer Berechnung vgl. Agrawal, Imielinski, Swami (2003)

Abb. 4.55: Regelauswertung im Konfigurator

Diese Regeln bieten dem Anbieter die Möglichkeit, die Leistungsauswahl zu beeinflussen, z.B. um die Konfiguration nicht realisierbarer Leistungsbündel auszuschließen. Zu diesem Zweck können Kombinationen von Leistungen ausgeschlossen oder erzwungen werden. Regeln werden immer nach der Auswahl einer Leistung ausgewertet und ggf. die Konsequenzen ausgeführt. Diese werden im betroffenen Modul in roter Schrift angezeigt. Abb. 4.55 zeigt die Auswertung einer Regel, welche bedingt, dass in Kombination mit der Maschine „T 2000“ auch die Leistung „Vorverkaufsberatung“ gekauft werden muss.

Neben der Auswahl einer angebotenen Leistung kann der Kunde auch einen Sonderwunsch für jedes Modul äußern. Dies ist z.B. sinnvoll, wenn die vom Anbieter angebotenen Leistungen in kleinerem Umfang angepasst werden müssen, um speziellen Anforderungen des Kunden zu entsprechen, oder eine für den Kunden passende Leistung im Angebot nicht enthalten ist. Die Eingabe erfolgt über das Sonderwunsch-Feld, welches immer das letzte Element in der Liste der Leistungen eines Moduls ist.

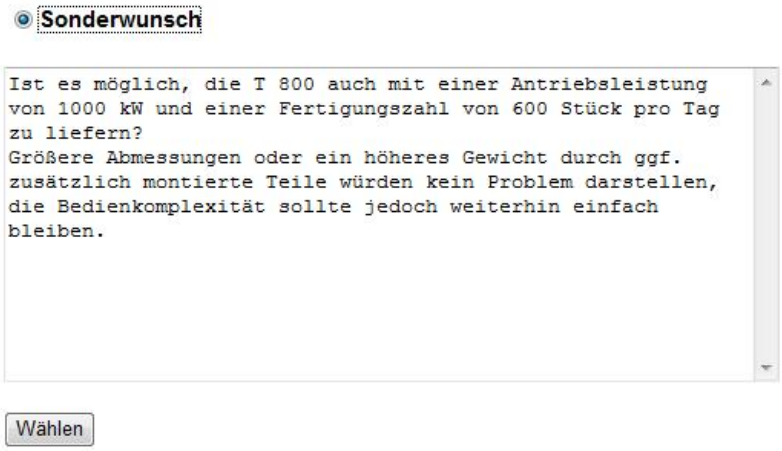


Abb. 4.56: Eingabe eines Sonderwunsches

Die Eingabe des Sonderwunsches erfolgt als einfacher Text. Eine weitergehende Formatierung ist nicht möglich. Die Auswahl einer Leistung oder eines Sonderwunsches erfolgt über das Setzen der Optionsschaltfläche und der Bestätigung über die Wählen-Schaltfläche.

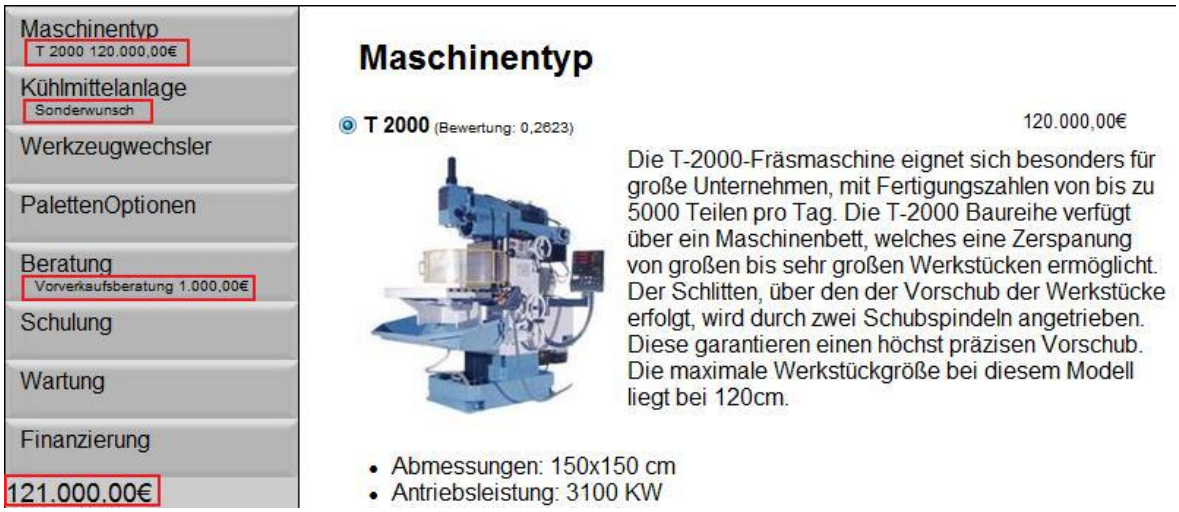


Abb. 4.57: Übersicht über die ausgewählten Leistungen im Konfigurator

Nach der Auswahl wird die entsprechende Leistung im linken Menü unter dem Modulnamen mit dem dazugehörigen Preis angezeigt. Abb. 4.57 zeigt dies für die Leistungen „T 2000“ (Maschinentyp), „Vorverkaufsberatung“ (Beratung) und einem Sonderwunsch (Kühlmittelanlage). Neben der manuellen Auswahl von Leistungen in allen Modulen ist es auch möglich, die Konfiguration automatisch durchführen oder beenden zu lassen. Dies geschieht auf der Basis der erfassten Präferenzen und unter Berücksichtigung der ggf. vorhandenen vom Anbieter formulierten Regeln über die AutomatischKonfigurieren-Schaltfläche. Die automatisch generierte Konfiguration kann weiter angepasst werden und wird nicht automatisch gespeichert. Eine neue Konfiguration kann über die Schaltfläche Neue Konfiguration erstellt werden.

Unter dem Navigationsmenü erscheint der Gesamtpreis der derzeitig ausgewählten Leistungen. Dabei werden nur Leistungen mit in die Rechnung einbezogen, für die direkt ein Preis angezeigt

wird. Ausgenommen sind Sonderwünsche und Leistungen, für die die Zahlungen nicht einmalig beim Kauf, sondern verteilt über den Lebenszyklus anfallen. Letztere werden wie folgt dargestellt.

Abb. 4.58: Lebenszyklusdarstellung für eine Leistung

Die Zahlungen im Lebenszyklus einer Leistung werden eingeblendet, wenn die Maus sich auf dem Namen der Leistung befindet. Die angezeigte Tabelle zeigt dann die Zahlungen der bisher gewählten Leistungen und der aktuell betrachteten Leistung, welche zur besseren Übersicht beige hinterlegt wird. Abb. 4.58 zeigt dies für die Leistung „Wartung silber“. Die Summe unter den Spalten zeigt die jeweils in einer Periode anfallenden Zahlungen, für die Periode 0, die Kaufphase, entspricht in den meisten Fällen dem unter dem Navigationsmenü angezeigten Gesamtpreis. Ausnahmen stellen z.B. Leistungsbündel mit Finanzierungsdienstleistungen dar. Die Dauer einer Periode beträgt ein Jahr, die Anzahl der Perioden errechnet sich aus dem Maximum der Lebenszyklusdauern aller im HLBAT enthaltenen Leistungen, nicht der aktuell ausgewählten Leistungen. Um eine Vergleichbarkeit der Leistungen innerhalb eines Moduls herzustellen, wird in jedem Modul eine Lebenszykluskostentabelle mit allen ausgewählten Leistungen und allen Leistungen im Modul unterhalb der Auswahlliste angezeigt.

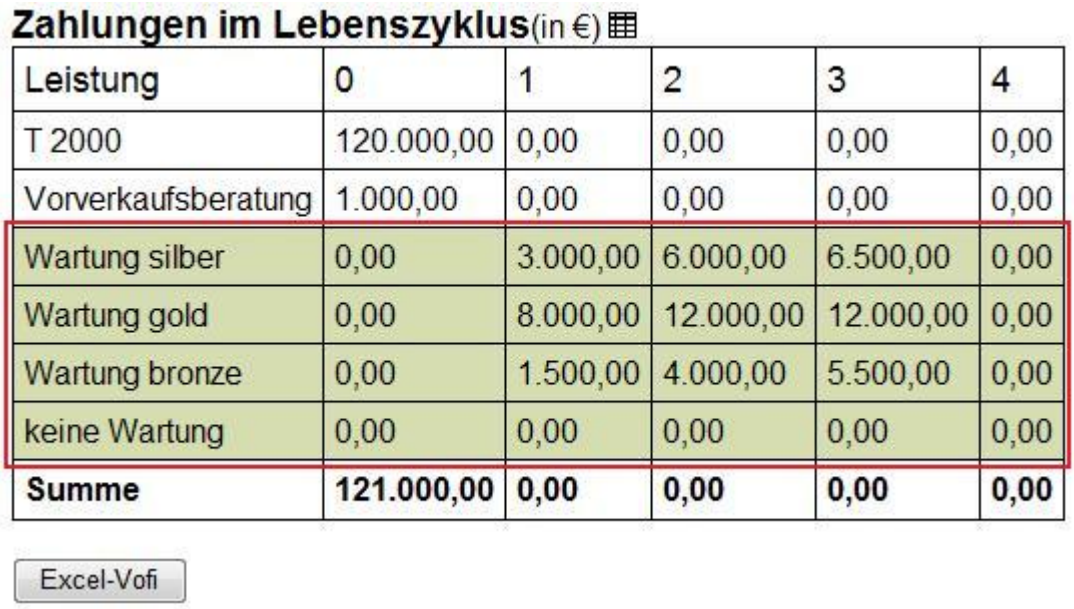
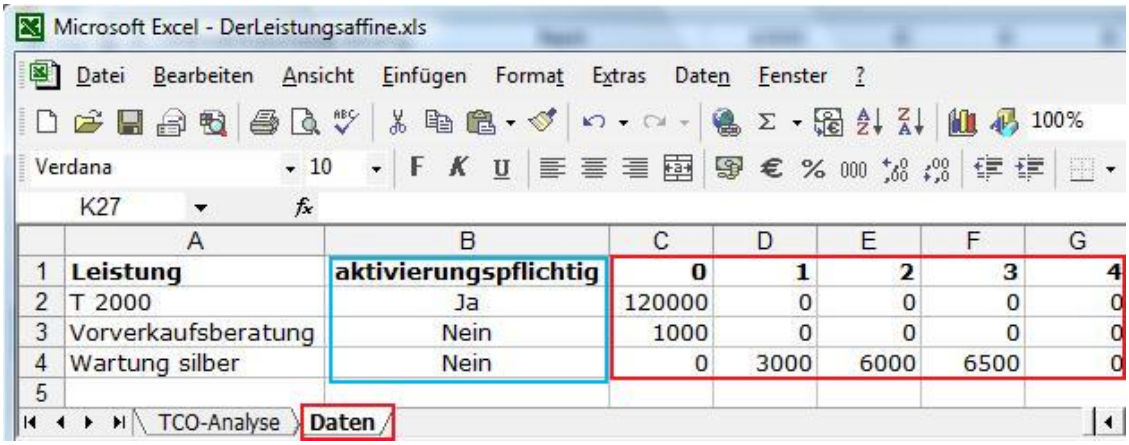


Abb. 4.59: Lebenszyklusdarstellung für ein Modul

Die im aktuell betrachteten Modul enthaltenen Leistungen werden wie bei der Darstellung für eine einzelne Leistung beige hinterlegt, um sie von den bereits ausgewählten Leistungen abzugrenzen. Abb. 4.59 zeigt dieses für die Leistungen des Moduls „Wartung“. Eine ausgewählte Leistung im aktuellen Modul wird blau hinterlegt. Auch die Option, keine Wartungsleistung zu wählen, wird hier mit Zahlungen von 0 mit angezeigt. Eine analoge Darstellung der Sonderwünsche findet nicht statt, da die Option, keine Leistung auszuwählen nicht immer gegeben sein muss und sie somit im Gegensatz zu Sonderwünschen als normale Leistung behandelt wird. Über die Excel-Schaltfläche besteht die Möglichkeit, die Zahlungsfolgen der ausgewählten Leistungen für eine TotalCostofOwnership-Analyse mit Hilfe eines vollständigen Finanzplanes (VOFI) in ein Excel Dokument zu exportieren.

Abb. 4.60: Download des TCO-VOFI

Dieses wird dem Kunden zum Download oder direkten Öffnen angeboten und bietet die Möglichkeit, detaillierte Analysen der Lebenszykluskosten auch durch die Eingabe weiterer eigener Daten zu erstellen. Das Excel-Dokument ist in zwei Tabellenblätter aufgeteilt: Das Blatt Daten enthält die Zahlungsfolgen aus dem Konfigurator.



	A	B	C	D	E	F	G
1	**Leistung**	**aktivierungspflichtig**	**0**	**1**	**2**	**3**	**4**
2	T 2000	Ja	120000	0	0	0	0
3	Vorverkaufsberatung	Nein	1000	0	0	0	0
4	Wartung silber	Nein	0	3000	6000	6500	0
5							

Abb. 4.61: Zahlungsfolgen im TCO-VOFI

Neben dem Namen der Leistung wird zusätzlich zu den Zahlungsfolgen noch das Attribut „aktivierungspflichtig“ angezeigt. Dieses legt fest, ob die Zahlungen der Leistung abschreibungsrelevant sind oder nicht. Es wird vom Anbieter gesetzt und stellt eine grundlegende Empfehlung dar, welche

bei Bedarf geändert werden kann und sollte. Eingaben außer „Ja“ und „Nein“ sind in diesem Feld nicht zulässig. Die Zahlungsfolgen können beliebig geändert werden, eine Erweiterung der Lebenszyklusdauer über das Hinzufügen von Perioden ist ebenfalls möglich. Die eigentliche TCO-Analyse[23] auf Basis dieser Zahlungsfolgen wird im Tabellenblatt „TCO-Analyse“ durchgeführt.

	A	B	C	D	E	F	G
1							
2	**Eigenkapital**	10.000					
3							
4	**Kapitalaufnahme**						
5	Kredit mit Endtilgung						
6	Nominalwert	15.000					
7	Sollzinsfuß	9%					
8	Disagio	0%					
9	Laufzeit	3					
10							
11	Kontokorrentkredit						
12	Sollzinsfuß	10%					
13							
14	**Kapitalanalage**						
15	Habenzinsfuß	10%					
16							
17	Opportunitätskostensatz	10%					
18							
19	**Außerordentliche Eträge**						
20	Zeitpunkt	**0**	**1**	**2**	**3**	**4**	**S**
21	Einzahlungen						
22							
23	**Steuern und Abschreibungen**						
24	Ertragsteuermultifaktor	50,00%					
25							
26	Abschreibungsmethode	linear					
27	Abschreibungsdauer	4					
28	Restbuchwert	80000					
29							

Abb. 4.62: Eingabeparameter für die TCO-Analyse

Im oberen Bereich des Tabellenblattes können die Parameter der TCO-Analyse gesetzt werden. Die Gestaltung dieser Parameter ist abhängig von der Situation des Kunden und sollte deshalb immer vorgenommen werden, bevor aus den Berechnungen Entscheidungen abgeleitet werden. Dateneingaben sind in allen grau hinterlegten Feldern möglich. Die Parameter und ihre Auswirkungen auf die TCO-Analyse sind:

- Eigenkapital: Die dem Kunden zur Finanzierung des hybriden Leistungsbündels zur Verfügung stehenden eigenen Mittel. Je nach Höhe des Eigenkapitals muss Kapital am Markt besorgt werden, um die Investition zu finanzieren.

- Kapitalaufnahme: In diesem Bereich werden die dem Kunden zur Auswahl stehenden Möglichkeiten der Kapitalaufnahme modelliert. Dabei stehen ein Kredit mit Endtilgung und ein Kontokorrentkredit zur Verfügung. Kann die notwendige Finanzierung der Investition nicht durch den Kredit mit Endtilgung geleistet werden, wird auf den Kontokorrentkredit zurückgegriffen.

[23] Eine genaue methodische Beschreibung des Konzepts der vollständigen Finanzpläne findet sich bei GROB (2006), eine detaillierte Beschreibung der TCO-Analyse auf Basis eines VOFI findet sich bei Grob, Lahme (2004)

- Kapitalanlage: Hier kann der Zins für die Anlage von Kapital über die Dauer des Lebenszyklus des hybriden Leistungsbündels eingegeben werden. Eine solche Anlage kann z.B. durch das Vorliegen von Eigenkapital von höherem Betrag als die Auszahlungen des hybriden Leistungsbündels oder die Inanspruchnahme von Finanzierungsleistungen möglich werden. Des Weiteren kann der Opportunitätskostensatz, der Zins für die Anlage des Eigenkapitals, wenn das hybride Leistungsbündel nicht gekauft wird, eingegeben werden.

- Außerordentliche Erträge: In dieser Zeile können außerordentliche Erträge[24], die durch den Kauf des hybriden Leistungsbündels entstehen, eingetragen werden.

- Steuern und Abschreibungen: Neben dem Ertragsteuermultifaktor, welcher den kumulierten Steuersatz darstellt, können hier die Abschreibungsmethode, -dauer und der Restbuchwert des hybriden Leistungsbündels eingegeben werden. Bei Wahl der linearen Abschreibungsmethode werden die Abschreibungen automatisch berechnet, bei der manuellen Abschreibung müssen alle Abschreibungen direkt eingegeben werden. Ist die Abschreibungsdauer länger als die Dauer des Lebenszyklus, müssen die Daten ebenfalls manuell eingegeben werden. Der Restbuchwert darf nicht unter 0 liegen. Fehlerhafte Parameter oder Eingaben, die eine manuelle Eingabe erfordern, werden durch eine rote Warnmeldung neben den Eingabefeldern angezeigt.

Unterhalb des Eingabebereiches befindet sich die Werteberechnung für das hybride Leistungsbündel. Sie ist durch Überschriften in dunklem Türkis gekennzeichnet und in ein Haupttableau und zwei Tableaus für Nebenrechnungen untergliedert. Alle Werte im Haupttableau werden automatisch über Formeln berechnet. Eine manuelle Eingabe kann die Formelbzüge innerhalb der Tabelle aufheben und zu Fehlern führen. Deshalb sollte eine solche Eingabe an dieser Stelle vermieden werden.

30							
31	**VOFI der Investition**						
32	Zeitpunkt	0	1	2	3	4	S
33	Einzahlungen	0	0	0	0	0	0
34	- aktivierungspflichtige Auszah	120.000	0	0	0	0	120.000
35	- nicht aktivierungspflichtige A	1.000	3.000	6.000	6.500	0	16.500
36	**Zahlungsfolge**	-121.000	-3.000	-6.000	-6.500	0	
37	**Eigenkapital**	10.000					
38	**Kredit mit**						
39	**Endtilgung**						
40	+ Aufnahme	15.000,00					
41	- Tilgung	0,00	0,00	0,00	15.000,00	0,00	
42	- Sollzinsen						
43	(inkl. Disagio)	0,00	1.350,00	1.350,00	1.350,00	0,00	4.050,00
44	**Kontokorrentkredit**						
45	+ Aufnahme	96.000,00	1.975,00	3.573,75	19.002,44	1.027,56	
46	- Tilgung		0,00	0,00	0,00	0,00	
47	- Sollzinsen		9.600,00	9.797,50	10.154,88	12.055,12	41.607,49
48	**Standardanlage**						
49	- Anlage	0,00	0,00	0,00	0,00	0,00	
50	+ Auflösung	0,00	0,00	0,00	0,00	0,00	
51	+ Habenzinsen	0,00	0,00	0,00	0,00	0,00	
52	**Steuerzahlungen**						
53	- Auszahlungen	0,00	0,00	0,00	0,00	0,00	
54	+ Erstattung	0,00	11.975,00	13.573,75	14.002,44	11.027,56	
55	Finanzierungssaldo	0,00	0,00	0,00	0,00	0,00	
56	**Bestandsgrößen**						
57	Kredit mit Endtilgung	15.000,00	15.000,00	15.000,00	0,00	0,00	
58	Kontokorrentkredit	96.000,00	97.975,00	101.548,75	120.551,19	121.578,75	
59	Guthabenstand	0,00	0,00	0,00	0,00	0,00	
60	**Bestandssaldo**	**-111.000,00**	**-112.975,00**	**-116.548,75**	**-120.551,19**	**-121.578,75**	
61							

Abb. 4.63: Zahlungsfolgen im Haupttableau des TCO-VOFI

24 Diese stellen nicht die Erträge durch eine Erstellung von Produkten dar. Eine genaue Definition findet sich bei Grob, Lahme (2004), S. 3 ff.

Der obere Teil des Haupttableaus enthält die Zahlungsfolgen der Leistungen aus dem Tabellenblatt „Daten“. Dabei werden aktivierungspflichtige und nicht aktivierungspflichtige Leistungen getrennt ausgewiesen, bevor sie in der Zeile „Zahlungsfolge“ zusammengefasst werden. Die letzte Spalte, gekennzeichnet mit „S“, enthält jeweils die Zeilensumme.

30							
31	**VOFI der Investition**						
32	Zeitpunkt	0	1	2	3	4	S
33	Einzahlungen	0	0	0	0	0	0
34	- aktivierungspflichtige Auszah	120.000	0	0	0	0	120.000
35	- nicht aktivierungspflichtige A	1.000	3.000	6.000	6.500	0	16.500
36	**Zahlungsfolge**	-121.000	-3.000	-6.000	-6.500	0	
37	**Eigenkapital**	10.000					
38	**Kredit mit**						
39	**Endtilgung**						
40	+ Aufnahme	15.000,00					
41	- Tilgung	0,00	0,00	0,00	15.000,00	0,00	
42	- Sollzinsen						
43	(inkl. Disagio)	0,00	1.350,00	1.350,00	1.350,00	0,00	4.050,00
44	**Kontokorrentkredit**						
45	+ Aufnahme	96.000,00	1.975,00	3.573,75	19.002,44	1.027,56	
46	- Tilgung		0,00	0,00	0,00	0,00	
47	- Sollzinsen		9.600,00	9.797,50	10.154,88	12.055,12	41.607,49
48	**Standardanlage**						
49	- Anlage	0,00	0,00	0,00	0,00	0,00	
50	+ Auflösung	0,00	0,00	0,00	0,00	0,00	
51	+ Habenzinsen	0,00	0,00	0,00	0,00	0,00	
52	**Steuerzahlungen**						
53	- Auszahlungen	0,00	0,00	0,00	0,00	0,00	
54	+ Erstattung	0,00	11.975,00	13.573,75	14.002,44	11.027,56	
55	Finanzierungssaldo	0,00	0,00	0,00	0,00	0,00	
56	**Bestandsgrößen**						
57	Kredit mit Endtilgung	15.000,00	15.000,00	15.000,00	0,00	0,00	
58	Kontokorrentkredit	96.000,00	97.975,00	101.548,75	120.551,19	121.578,75	
59	Guthabenstand	0,00	0,00	0,00	0,00	0,00	
60	**Bestandssaldo**	**-111.000,00**	**-112.975,00**	**-116.548,75**	**-120.551,19**	**-121.578,75**	
61							

Abb. 4.64: Kapitalaufnahme und -anlage im Haupttableau des TCO-VOFI

Der in Abb. 4.64 gezeigte Abschnitt des Haupttableaus enthält die Berechnungen für die Kapitalaufnahme und –anlage. Für die Kredite wird eine Berechnung von Zinsen, ggf. inklusive Disagio, und bestmöglicher Tilgung, für die Kapitalanlage analog mit der jeweils höchstmöglichen Anlage durchgeführt. In den meisten Fällen wird eine Kapitalanlage jedoch nicht möglich sein. Die Berechnung der Steuerzahlungen wird mit Hilfe der beiden Nebenrechnungstableaus durchgeführt.

62						
63	**Berechnung der Steuerzahlungen für den VOFI der Investition**					
64	Zeitpunkt	1	2	3	4	S
65	Ertragsüberschuss	-3.000,00	-6.000,00	-6.500,00	0,00	
66	- Zinsaufwand	10.950,00	11.147,50	11.504,88	12.055,12	
67	+ Zinsertrag	0,00	0,00	0,00	0,00	
68	- Disagio	0,00	0,00	0,00	0,00	
69	- Abschreibungen	10.000,00	10.000,00	10.000,00	10.000,00	40.000,00
70	Steuerbemessungsgrundlage	-23.950,00	-27.147,50	-28.004,88	-22.055,12	
71	Erstattung	11.975,00	13.573,75	14.002,44	11.027,56	
72	Auszahlung	0,00	0,00	0,00	0,00	
73						
74						
75	**Berechnung der Abschreibungen der Investition**					
76	Zeitpunkt	1	2	3	4	
77	Buchwert zu Beginn des Jahres	120.000,00	110.000,00	100.000,00	90.000,00	
78	- Abschreibungen	10.000,00	10.000,00	10.000,00	10.000,00	
79	- zusaetzliche Abschreibungen	0,00	0,00	0,00	0,00	
80	Buchwert zum Ende des Jahres	110.000,00	100.000,00	90.000,00	80.000,00	
81						

Abb. 4.65: Nebenrechnungstableaus im TCO-VOFI

Die Steuerbemessungsgrundlage setzt sich aus mehren Komponenten zusammen, die anhand des jeweils nebenstehenden Operators in die Rechnung mit einbezogen werden. Der Ertragsüberschuss

ergibt sich aus den negativen Zahlungen der nicht aktivierungspflichtigen Leistungen der vorangegangenen Periode. Zinsaufwand, -ertrag und Disagio aus den Berechnungen für die Kapitalaufnahme und –anlage im Haupttableau. Die Abschreibungen werden im zweiten Nebentableau bestimmt.

62						
63	**Berechnung der Steuerzahlungen für den VOFI der Investition**					
64	Zeitpunkt	1	2	3	4	S
65	Ertragsüberschuss	-3.000,00	-6.000,00	-6.500,00	0,00	
66	- Zinsaufwand	10.950,00	11.147,50	11.504,88	12.055,12	
67	+ Zinsertrag	0,00	0,00	0,00	0,00	
68	- Disagio	0,00	0,00	0,00	0,00	
69	- Abschreibungen	10.000,00	10.000,00	10.000,00	10.000,00	40.000,00
70	Steuerbemessungsgrundlage	-23.950,00	-27.147,50	-28.004,88	-22.055,12	
71	Erstattung	11.975,00	13.573,75	14.002,44	11.027,56	
72	Auszahlung	0,00	0,00	0,00	0,00	
73						
74						
75	**Berechnung der Abschreibungen der Investition**					
76	Zeitpunkt	1	2	3	4	
77	Buchwert zu Beginn des Jahres	120.000,00	110.000,00	100.000,00	90.000,00	
78	- Abschreibungen	10.000,00	10.000,00	10.000,00	10.000,00	
79	- zusaetzliche Abschreibungen	0,00	0,00	0,00	0,00	
80	Buchwert zum Ende des Jahres	110.000,00	100.000,00	90.000,00	80.000,00	
81						

Abb. 4.66: Abschreibungsberechnung im TCO-VOFI

Die Höhe der jährlichen Abschreibung ergibt sich aus dem Buchwert zu Beginn der Berechnungen, der Abschreibungsdauer und dem Restbuchwert. Der anfängliche Buchwert entspricht dabei den aktivierungspflichtigen Abschreibungen in der Periode 0. Die resultierenden Werte werden automatisch in das Tableau eingetragen. Bei einer manuellen Eingabe der Abschreibungen müssen alle benötigten Werte hier eingegeben werden. Die Abschreibungen pro Periode sind der letzte benötigte Parameter für die Berechnung der Steuerbemessungsgrundlage.

62						
63	**Berechnung der Steuerzahlungen für den VOFI der Investition**					
64	Zeitpunkt	1	2	3	4	S
65	Ertragsüberschuss	-3.000,00	-6.000,00	-6.500,00	0,00	
66	- Zinsaufwand	10.950,00	11.147,50	11.504,88	12.055,12	
67	+ Zinsertrag	0,00	0,00	0,00	0,00	
68	- Disagio	0,00	0,00	0,00	0,00	
69	- Abschreibungen	10.000,00	10.000,00	10.000,00	10.000,00	40.000,00
70	Steuerbemessungsgrundlage	-23.950,00	-27.147,50	-28.004,88	-22.055,12	
71	Erstattung	11.975,00	13.573,75	14.002,44	11.027,56	
72	Auszahlung	0,00	0,00	0,00	0,00	
73						
74						
75	**Berechnung der Abschreibungen der Investition**					
76	Zeitpunkt	1	2	3	4	
77	Buchwert zu Beginn des Jahres	120.000,00	110.000,00	100.000,00	90.000,00	
78	- Abschreibungen	10.000,00	10.000,00	10.000,00	10.000,00	
79	- zusaetzliche Abschreibungen	0,00	0,00	0,00	0,00	
80	Buchwert zum Ende des Jahres	110.000,00	100.000,00	90.000,00	80.000,00	
81						

Abb. 4.67: Berechnung der Steuerzahlungen im TCO-VOFI

Nach der Berechnung der Steuerbemessungsgrundlage wird die für die Periode anfallende Steuerauszahlung oder -erstattung mit Hilfe des Ertragsteuermultifaktors ermittelt. In den meisten Fällen werden nur Steuererstattungen bei dieser Art von TCO-Analyse eines hybriden Leistungsbündels anfallen. Die Steuerzahlungen werden dann im Haupttableau verwendet.

30							
31	**VOFI der Investition**						
32	Zeitpunkt	0	1	2	3	4	S
33	Einzahlungen	0	0	0	0	0	0
34	- aktivierungspflichtige Auszah	120.000	0	0	0	0	120.000
35	- nicht aktivierungspflichtige A	1.000	3.000	6.000	6.500	0	16.500
36	**Zahlungsfolge**	-121.000	-3.000	-6.000	-6.500	0	
37	**Eigenkapital**	10.000					
38	**Kredit mit**						
39	**Endtilgung**						
40	+ Aufnahme	15.000,00					
41	- Tilgung	0,00	0,00	0,00	15.000,00	0,00	
42	- Sollzinsen						
43	(inkl. Disagio)	0,00	1.350,00	1.350,00	1.350,00	0,00	4.050,00
44	**Kontokorrentkredit**						
45	+ Aufnahme	96.000,00	1.975,00	3.573,75	19.002,44	1.027,56	
46	- Tilgung		0,00	0,00	0,00	0,00	
47	- Sollzinsen		9.600,00	9.797,50	10.154,88	12.055,12	41.607,49
48	**Standardanlage**						
49	- Anlage	0,00	0,00	0,00	0,00	0,00	
50	+ Auflösung	0,00	0,00	0,00	0,00	0,00	
51	+ Habenzinsen	0,00	0,00	0,00	0,00	0,00	
52	**Steuerzahlungen**						
53	- Auszahlungen	0,00	0,00	0,00	0,00	0,00	
54	+ Erstattung	0,00	11.975,00	13.573,75	14.002,44	11.027,56	
55	Finanzierungssaldo	0,00	0,00	0,00	0,00	0,00	
56	**Bestandsgrößen**						
57	Kredit mit Endtilgung	15.000,00	15.000,00	15.000,00	0,00	0,00	
58	Kontokorrentkredit	96.000,00	97.975,00	101.548,75	120.551,19	121.578,75	
59	Guthabenstand	0,00	0,00	0,00	0,00	0,00	
60	**Bestandssaldo**	**-111.000,00**	**-112.975,00**	**-116.548,75**	**-120.551,19**	**-121.578,75**	
61							

Abb. 4.68: Bestandssaldoberechnung im TCO-VOFI

Der Finanzierungssaldo rechnet die Beträge der Kapitalaufnahmen und -anlagen gegeneinander auf. Er hat bei korrekter Berechnung den Wert 0 und stellt somit bei manuellen Eingaben eine Kontrollmöglichkeit dar. Darunter werden die Bestände der Kapitalanlage und –aufnahme einzeln aufgeführt und schließlich zum Bestandssaldo zusammengerechnet.

Die Alternative zum Kauf des hybriden Leistungsbündels, stellt in dieser Form der TCO-Analyse die Anlage des Eigenkapitals zum Opportunitätskostensatz dar. Für diese werden die relevanten Werte in gleicher Form wie für das hybride Leistungsbündel berechnet. Die benötigten Tableaus sind durch Überschriften in hellem Türkis gekennzeichnet. Zahlungsfolgen, Abschreibungen und eine Kapitalaufnahme existieren bei dieser reinen Kapitalanlage nicht. Der dazugehörige VOFI ist daher weniger umfangreich als der des hybriden Leistungsbündels. Abschließend wird zur Entscheidungsfindung eine TCO-Analyse auf Basis der berechneten Werte durchgeführt.

105		
106	**TCO-Analyse**	
107	**Erlöse**	0
108	+ außerordentliche Erträge	0
109	- Abschreibungen	40.000,00
110	= pagatorischer Gewinnvor Zinsen, Ergänzungs- und Reinvestitionserträgen	-40.000,00
111	+ Ergänzungs- und Reinvestitionserträge	0,00
112	- Zinsaufwand (inkl. Disagio)	45.657,49
113	= pagatorischer Gewinn (vor Ertragsteuern)	-85.657,49
114	+ Ertragssteuererstattung	50.578,75
115	= pagatorischer Gewinn (nach Ertragsteuern)	-35.078,75
116	- kalkulatorische Zinsen	4.310,13
117	+ Steuerkorrektur der kalkulatorischen Zinsen	2155,0625
118	= kalkulatorischer Totalgewinn	-37.233,81
119	**= TCO**	37.233,81
120		

Abb. 4.69: Berechnung der TCO-Kennzahl

Die TCO-Kennzahl wird über die in Abb. 4.69 gezeigte Formel berechnet und stellt „einen Sonderfall der Analyse des Totalgewinnes“ [25] dar. Obwohl vom Betrag her negativ, wird die Kennzahl als positiver Betrag angegeben. Sie stellt dar, welche Kosten für den Kunden beim Kauf des hybriden Leistungsbündels unter Bezug auf eine alternative Anlage des zur Verfügung stehenden Eigenkapitals über den gesamten Lebenszyklus verteilt anfallen.

Zusammengestellte Konfigurationen, für die ggf. zum detaillierten Vergleich jeweils TCO-Analysen durchgeführt wurden, können im Konfigurator über die Speichern-Schaltfläche gespeichert werden. Dabei wird die aktuelle Konfiguration bei mehrmaligem Speichern überschrieben. Zusätzlich wird die letzte im Konfigurator angezeigte Konfiguration automatisch gespeichert, um z.B. bei einem Timeout einen Datenverlust zu vermeiden. Gespeicherte Konfigurationen werden im Profil unter den persönlichen Daten in einer Liste angezeigt.

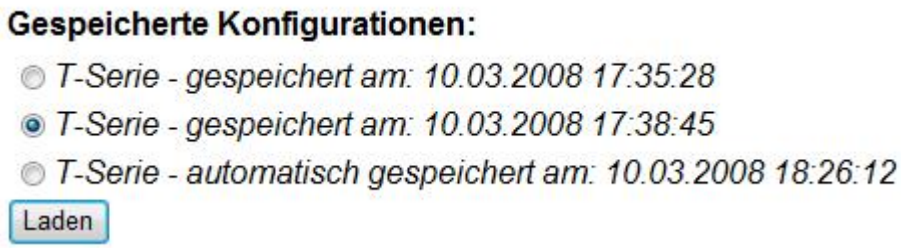


Abb. 4.70: Liste gespeicherter Konfigurationen

Für jede Konfiguration werden der Name des hybriden Leistungsbündels und das Datum der Speicherung angezeigt. Die automatisch gespeicherte Konfiguration wird durch den Zusatz „automatisch“ gekennzeichnet. Das Laden einer Konfiguration erfolgt über die Laden-Schaltfläche. Ein erneutes Speichern bewirkt ein Überschreiben der geladenen Konfiguration.

Soll eine gespeicherte Konfiguration als Anfrage an den Anbieter geschickt werden, um eine Kontaktaufnahme und ggf. folgende Angebotsverhandlungen anzustoßen, geschieht dies über die Auswahl einer gespeicherten Konfiguration aus der Liste und Betätigung der AnfrageSenden-Schaltfläche.

[25] Grob, Lahme (2004) , S.6

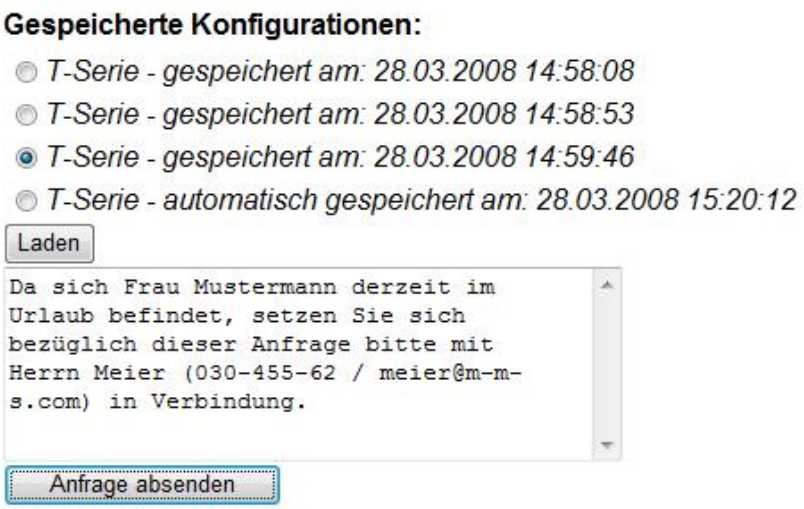


Abb. 4.71: Absenden einer Anfrage

In diesem Vorgang kann der Kunde noch Anmerkungen zur Anfrage über das Textfeld unter der Laden-Schaltfläche eingeben. In Abb. 4.71 wird dies für eine kurzfristige Änderung der Kontaktdaten des Kunden gezeigt. Nach erfolgreichem Absenden der Anfrage wird dies über eine Textbox angezeigt und die Konfiguration in der Liste der gespeicherten Konfigurationen mit dem Zusatz „(Anfrage gesendet)“ gekennzeichnet.

Abb. 4.72: gesendete Anfragen im Konfigurator

5 Implementierung

Die folgenden Ausführungen enthalten eine kurze Beschreibung der Struktur der in Kostenkalkulationstool und Konfigurator verwendeten Klassen und Beziehungen zwischen diesen. Eine Darstellung der einzelnen Funktionen auf der Ebene des Programmcodes findet nicht statt, da diese sich nicht signifikant von den im Code befindlichen Kommentaren unterscheiden würde. Besonderheiten in der Implementierung werden im Folgenden beschrieben und erklärt.

5.1 Kostenkalkulationstool

Die Implementierung des Plugins erfolgt in der Klasse H2CalculatingTool. Diese erbt von schon implementierter Klasse PlugInBase und überschreibt die Methode OnDatabaseExplorerContextMenuPopup() (Abb. 5.1). Die Überschreibung bewirkt, dass zu vorhandenen Auswahlmöglichkeiten eine weitere, Start des Kostenkalkulationstools, hinzugefügt wird.

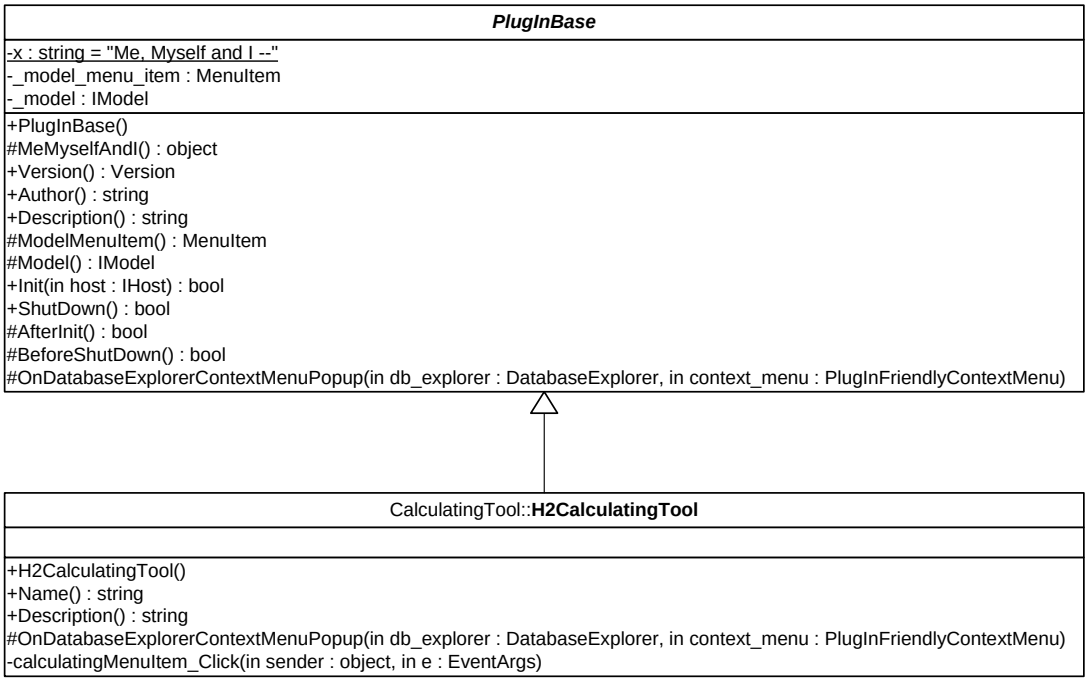


Abb. 5.1: Einbindung des Plugins

Grundlagen

Die Grundlage aller Berechnungen und Auslesoperationen wird in der Gruppe der Basisklassen geschaffen (siehe Abb. 5.2). Die Basisklassen beinhalten alle Methoden, die zur Auslese der benötigten Informationen nötig sind und trennen diese nach ihren Ausgabewerten. So befinden sich in der Klasse Ressource alle Methoden, deren Ausgabewert in Verbindung mit der Ressource gebracht werden kann (z.B. die Kosten je Mengeneinheit einer Ressource). Die getRessourcenBy…()–Methoden liefern immer eine ArrayList, die IDs der gefundenen Ressourcen beinhaltet. Allein die Methode getRessourcen(int leistungID) (in der Klasse Stelle – getStellen(int leistungID)) liefert ein

zweidimensionales Object-Array in dem alle benötigten Informationen gespeichert sind (siehe. Tab. 5.1).

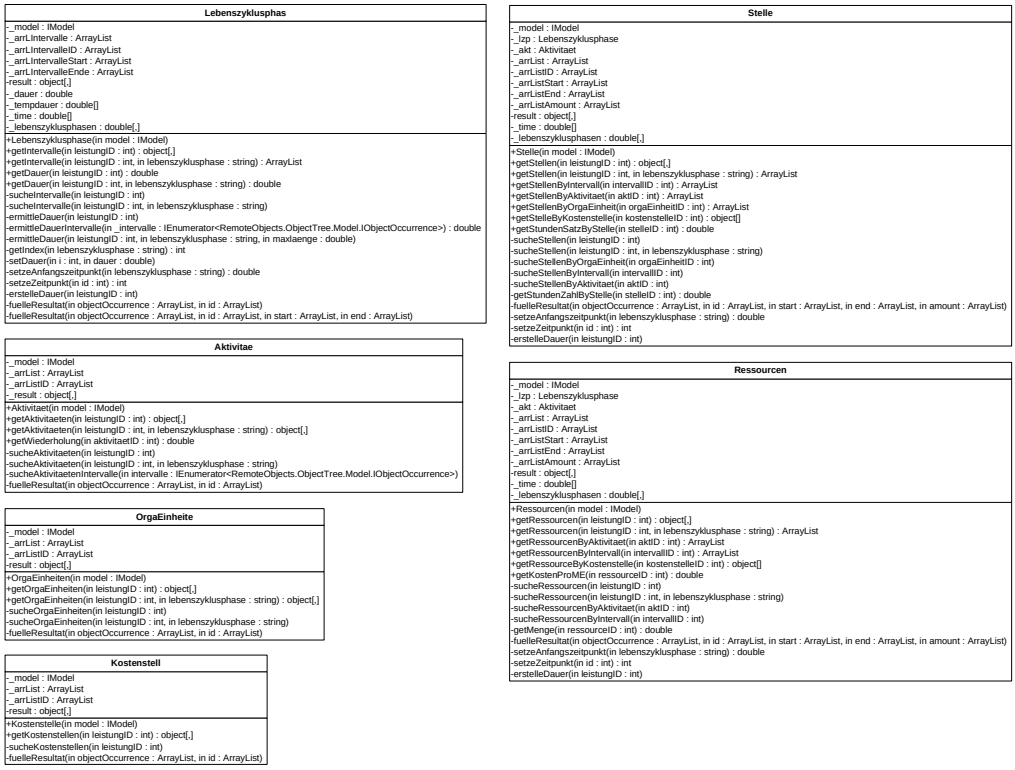


Abb. 5.2: Basisklassen

Name	ID	Start	Ende	Menge*

* - Die Spalte entfällt bei getIntervalle(int leistungID)

Tab. 5.1: Ausgabearray der „Grundmethoden“

Kostenkalkulation - Hauptfenster

Beim Starten des Kostenkalkulationstools werden alle benötigten Grundtabellen (Intervalle, Ressourcen und Stellen) erstellt. Das soll die Geschwindigkeit und die Reaktionszeit des Tools verbessern, da keine weiteren größeren Operationen mehr notwendig sind um Kosten zu ermitteln. Die Tabellen sollen auch nur dann aktualisiert werden, wenn der entsprechende Tab gerade sichtbar ist. Der Nachteil dieser Methode ist, dass bei jeder Änderung des Sichtfensters weitere Operationen, falls noch nicht geschehen, durchgeführt werden müssen um Anzeige der Kosten zu aktualisieren. Ein Vorteil liegt darin, dass das Startfenster wesentlich schneller erscheint und dadurch auch unnötige Daten, die den Anbieter nicht interessieren, gar nicht erst berechnet werden müssen.

Nachdem das Modul und die Leistung ausgewählt worden sind, führt das Programm, außer der Erstellung der Grundtabellen, auf jeden Fall die Berechnung der Gesamtdauer einer Leistung sowie

der Gesamtkosten. Für die Berechnung der Gesamtkosten läuft die Implementierung alle Zeilen der Grundtabellen durch und addiert dabei den berechneten Wert zum bereits vorhandenem. Bei der Berechnung der Dauer ist zu beachten, dass eine Lebenszyklusphase wiederholt auftreten und somit auch keine einfache Addition der Intervalle stattfinden kann. Die Annahme hier ist: Ist eine Lebenszyklusphase mehr als einmal in der Leistung vorhanden, so können die Aktivitäten in den Lebenszyklusphase parallel gestartet und durchgeführt werden. D.h. Die Bearbeitung der Aktivitäten startet sofort am Anfang einer Lebenszyklusphase und wartet nicht bis die Aktivitäten einer Lebenszyklusphase abgearbeitet worden sind. Um das zu berücksichtigen reicht es nicht aus die Leistung einfach durchzulaufen und den gefundenen Ressourcen (Stellen, Intervallen) ihre Zeitpunkte zuzuweisen. Es muss vorher eine Berechnung der maximalen Dauer jeder Lebenszyklusphase durchgeführt und somit ein grober Plan mit Startzeitpunkten der Lebenszyklusphasen erstellt werden. Beim weiteren Durchlauf der Leistung liest man nun, wenn man auf eine Lebenszyklusphase trifft, ihren Startzeitpunkt aus der erstellten Tabelle aus und benutzt diesen als Referenzzeitpunkt.

Um die Übersicht zu steigern und nicht die ganze Implementierung in eine einzige Klasse packen zu müssen, sind einige internal Klassen implementiert, die sich um einzelne Tabs im Kostenkalkulationsfenster kümmert. Die gesamte Übersicht findet man in Abb. 5.3. Allein die Klasse Kosten ist für alle Tabs zuständig und beschäftigt sich mit der Ermittlung der Kosten. Eine detaillierte Darstellung der einzelnen Klassen und Methoden findet man im Anhang 0.

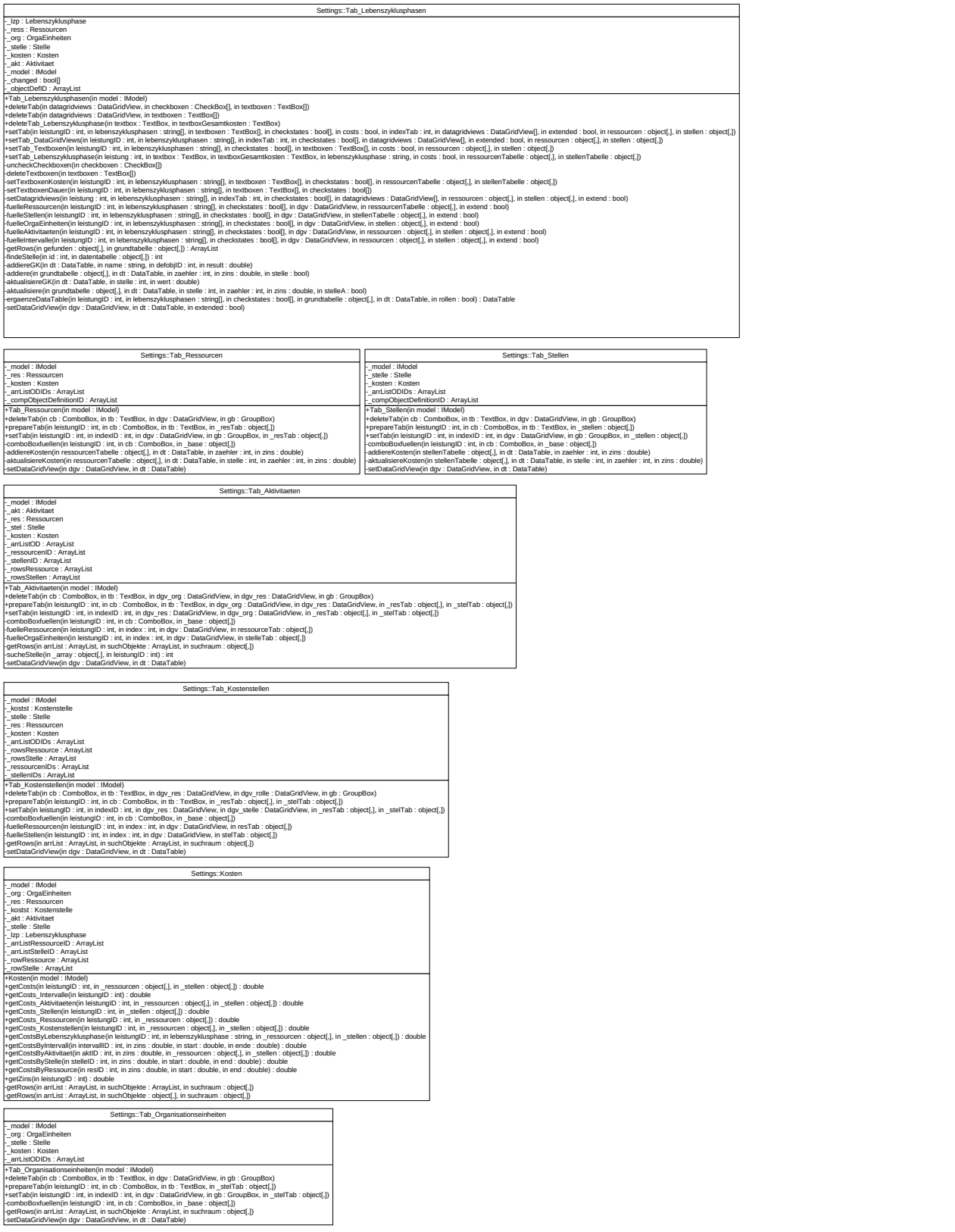


Abb. 5.3: Hilfsklassen

Kostenkalkulation - Lebenszykluskosten

Die Vorgehensweise beim Starten des Produktlebenszyklus ist für den Anbieter, dank der Führung durch die Implementierung, intuitiv.[26] Alle Berechnungen und Funktionen werden in der Klasse Produktlebenszyklus implementiert.

Während des gesamten Initialisierungsvorgangs sorgt die bool-Variable anfang dafür, dass noch keine Berechnungen durchgeführt werden, da noch nicht alle Informationen gesammelt werden konnten.Bei der Auswahl der Detailanzeige hat man momentan nur die Tabellensicht, weitere Darstellungen (z.B. Diagramme) sollten bzw. können bei der Weiterentwicklung hinzukommen. Die minimale Periodenzeitspanne ist auf ein Monat festgelegt. Die Untergrenze von einem Monat wurde deshalb gewählt, weil die Perioden unter einem Monat die Tabellen, abhängig von der gesamten Lebenszyklusdauer, recht unübersichtlich machen. Nach oben hin sind keine Grenzen gesetzt. Die größte Periodeneinheit ist „Jahr“, die Anzahl der Jahre ist nach oben hin offen. Abhängig von der ausgewählten Option wird der Variable periode eine Integer-Zahl, die die Anzahl von Tagen widergeben soll, zugewiesen. Ein Monat hat dabei 30, ein Jahr 360 und die Option „Gesamt“ hat die gesamte Leistungsdauer inkrementiert um 1.Die Detailstufenauswahl beinhaltet vier verschiedene Stufen. Dabei wird der Variable hierarchiestufe eine Kennzahl zugewiesen. Die niedrigste Detailstufe wäre die Leistungssicht (hierarchiestufe = 0), danach werden die Kosten immer detaillierter (Lebenszyklusphasen (1) < Intervalle (2) < Stellen/Ressourcen (3)) dargestellt. Bei allen diesen Ansichten werden die Höhe und der Zeitpunkt, zu dem die Kosten anfallen, visuell dargestellt. Zu beachten ist, dass die „Intervalle“-Sicht die höchstmögliche Stufe ist, bei der alle Kosten einer Leistung aufgezeigt werden.

Ist der Initialisierungsvorgang durchgeführt worden, wird die anfang-Variable auf false gesetzt und die Methode aktualisiere() gestartet. Unabhängig von den gewählten Optionen ist die Erstellung des Grundgerüstes immer identisch. Mit der Methode fuegeSpaltenein() werden die benötigten Spalten, deren Anzahl abhängig von der Gesamtleistungsdauer und der gewählten Periode ist, ermittelt und zum DataTable angefügt. Wenn die Grundtabelle nun steht sollen einzelne Positionen betrachtet werden. Als erstes werden die Gesamtkosten der Position mit der Methode getCostsBy…() aus der Klasse Kosten ermittelt. Danach wird überprüft, ob der Start- und der Endzeitpunkt in der gleichen Periode liegen. Ist das der Fall, werden die gesamten Kosten in die Spalte, ermittelt durch die Methode ermittleSpalte(), geschrieben. Ist das aber nicht der Fall, so können und müssen anteilige Kosten der Spalte hinzuaddiert werden. Dabei werden die Gesamtkosten mit dem Quotienten aus Tagen die zusammen noch zu einer Spalte gehören und der Gesamtdauer multipliziert. Als letztes werden allen Kosten in der Tabelle die Währungseinheit mit der Methode reloadTable() angefügt und dem DataGridView das DataTable als Source zugewiesen.

Ein Feature ist der Export der Zahlen durch Excel. Bei der Berechnung des TCO-VOFI[27] müssen erst die nominalen Kosten, falls der Zins nicht bei 0% liegt, berechnet werden und dann soll eine

[26] vgl. Kapitel 4.2

Vorlage, die im gleichen Ordner abgelegt ist, geöffnet und die Zahlen in das entsprechende Tabellenblatt reingeschrieben werden. Die Berechnung erfolgt automatisch durch die vorher bestimmten Formeln. Soll die Vorlage mal nicht gefunden werden, wird eine Exception ausgeworfen, der Export abgebrochen und das restliche Programm weiter ausgeführt.

Beim „normalen“ Export werden die Zahlen, so wie sie gerade angezeigt werden, in eine Excel-Tabelle exportiert. Der Aufruf der Excel Anwendung wird hier durch das sogenannte Late-Binding realisiert. Das heißt, es findet vorher kein Assembly-Verweis statt. Die Excel Bibliothek wird erst dann aufgerufen, wenn sie benötigt wird und auch dann ist es für die Implementierung nicht von Bedeutung welche Excel-Version auf dem Computer installiert ist.

Bei der Weiterentwicklung des Kalkulationstools ist zu beachten, dass die Implementierung sich nicht auf eine bestimmte Version des Office-Tools festlegen sollte, es sollte möglich sein mit unterschiedlichen Versionen einwandfrei zu arbeiten (Benutzung vom Late-Binding).

5.2 Leistungskonfigurator

Der Leistungskonfigurator ist als Webanwendung in ASP .NET implementiert. Der im Hintergrund ausgeführte Code ist in C# geschrieben. Zur Darstellung einiger Seiten wird auf AJAX Elemente zurückgegriffen, eine Installation der AJAX Extensions und des AJAX Control Toolkit für .NET ist somit notwendig. Der Webserver steht über .NET Remoting mit dem H2-Toolset-Server, welcher die Datenhaltung übernimmt, in Verbindung.

Als Webserver wurde im Rahmen dieses Projektes ein Microsoft Internet Information Services Server (IIS) verwendet. Alternativ könnte auch eine ASP .NET fähige Apache-Version verwendet werden. Die Konfiguration des IIS erfordert keine besonderen Einstellungen, allerdings erfordert der Download der Excel-Vorlage die Vergabe von Zugriffs- und Schreibberechtigungen auf den Ordner, in dem die Excel-Vorlage abgelegt wurde, sowie die Berechtigung zum Ausführen von Microsoft Excel für den Nutzer der Webseite. Standardmäßig ist dies der Nutzer „NetworkServices“.

Die Webanwendung ist in die beiden Packages BusinessLogic und Webfrontend aufgeteilt. Ersteres enthält die Verwaltungsklassen mit der Geschäftslogik, sowie die Datenhaltungsobjekte im Unterordner OwnObjects. Verwaltungsklassen sind durch den Anhang „Management“ im Namen gekennzeichnet, eine Ausnahme stellt die Klasse Connection dar, welche die Verbindung zum H2-Toolset-Server aufbaut. Die Datenhaltungsobjekte werden auf Basis der Daten aus dem H2-Toolset-Server bei Bedarf erstellt und ermöglichen einen effizienten Zugriff auf die Daten. Ein direkter Zugriff auf den ObjectTree hätte zum einen den Programmcode aufgrund der Baumstruktur verkompliziert und verlangsamt und zum anderen die Verbindung zum H2-Toolset-Server aufgrund einer Vielzahl von RemoteObject-Referenzen mit einem hohen Overhead belastet. Die einzigen dauerhaft auf dem

---

[27] vgl. Kapitel 4.3

Webserver gespeicherten RemoteObjects sind die Modelle in der Connection-Klasse, welche in einem späteren Abschnitt detailliert beschrieben wird. Die im Konfigurator angezeigten Bilder stellen eine Ausnahme bezüglich der Datenhaltung dar. Sie werden nicht auf dem H2-Toolset-Server gespeichert, es wird über das Attribut „BildURL“ einer Leistung eine URL als string ausgelesen und als Bildquelle verwendet. Dies dient ebenfalls der Verminderung des Datenaufkommens zwischen Webserver und H2-Toolset-Server. Im Rahmen dieses Projektes wurden alle Bilder auf dem Webserver im Unterordner „bilder“ vorgehalten.

Das Package Webfrontend beinhaltet die Präsentationsschicht der Anwendung. Die dabei verwendeten Seiten sind zur weiteren Strukturierung auf die drei Unterordner Configurator, UserLogin und Praeferenzen verteilt. Des Weiteren existieren die Seiten Default.aspx, die Startseite, und ErrorMessage.aspx, die zur Anzeige von ggf. bei schweren Fehlern auftretenden Exceptions dient. Ebenfalls im Package enthalten ist die Klasse global.asax. Sie enthält die EventHandler, für die während der Ausführung der Webanwendung ggf. auftretenden Events. Sie werden verwendet, um die Verbindung zu initialisieren, Fehler abzufangen und den Start und das Ende von Sessions zu behandeln. Das Layout der Seiten wird über das **H2-ServPay**.Master-Template im Unterordner Template bestimmt.

Die Konfigurationsdatei Web.config enthält alle Konfigurationseinstellungen für den Webserver. Neben den standardgemäß gesetzten Einstellungen wurden hier die Abschnitte sessionState und appSettings eingefügt. Die Einstellungen zum sessionState bestimmen die Dauer einer Session, aktuell 15 Minuten, den Namen des an den Benutzer vergebenen Cookie und verbieten eine Wiederverwendung abgelaufener SessionIDs. Letzteres ist notwendig, um in der Nutzerverwaltung eine mehrfache gleichzeitige Anmeldung mit einem Nutzernamen verhindern zu können. Die appSettings enthalten folgende Einstellungen.

- Zugriffsdaten zum H2-Toolset-Server: Hier werden die Serveradresse, der Port und ein Verbindungskanal, TCP und http werden unterstützt, gewählt. Zusätzlich muss ein Benutzername und ein Passwort zum Login beim H2-Toolset-Server angegeben werden, um eine Anmeldung zu ermöglichen. Die darunter folgenden Parameter setzen die Dauer bis zum Timeout der Verbindung bei Inaktivität und die Zeit zwischen der Aktualisierung der RemoteObjects im Cache fest.

- Logging-Konfiguration: Soll ein Log4Net Logger verwendet werden, um auftretende Fehler und Ereignisse zu dokumentieren, muss hier der Pfad einer entsprechenden Log4Net XML-Konfigurationsdatei angegeben werden.

- Vorschlagsgenerierung: In diesem Abschnitt kann bezüglich der wissensbasierten Vorschlagsgenerierung festgelegt werden, ob die Leistungen im Konfigurator anhand ihrer Bewertungen sortiert werden sollen. Bezüglich der Assoziationsregeln kann ein Mindestwert für Support und Confidence und die maximale Anzahl der angezeigten Regeln festgelegt werden.

- Zugriffsdaten für die H2-Modelle: Dieser Abschnitt enthält die Namen der vom Konfigurator ausgelesenen Modelle und der dazugehörigen Sprache, Projektmappe und Modelldatenbank.
- Datums- und Währungsformat: Enthält die Formatierungen für die im Konfigurator verwendete Darstellung.
- Excel-Vorlage für den TCO-VOFI: Enthält den Pfad der Excel-Vorlage für die TCO-Analyse.

Diese Parameter werden bei Bedarf aus der Web.config ausgelesen. Sie geben dem Anbieter die Möglichkeit, ohne Veränderungen am eigentlichen Code Einfluss auf den Konfigurator zu nehmen.

Verbindung zum H2-Toolset-Server

Der Webserver des Konfigurators ist über eine im Projektseminar implementierte modifizierte Variante der .NET Remoting Technologie an den H2-Toolset-Server angebunden. Das Verwalten des Verbindungsaufbaus wird von der Singleton-Klasse Connection übernommen. Diese wird beim Start des Webservers instanziiert und speichert die für den Konfigurator benötigten Modelle für das HLBAT und HLBAI in zwei privaten Attributen. Auf diese kann über entsprechende get()-Methoden zugegriffen werden. Um bei Inaktivität einen Timeout der dauerhaft aufrechterhalten Verbindung zu vermeiden, implementiert die Klasse einen SingleEventTimer, der bei Inaktivität die Verbindung trennt und die Referenzen auf die Modelle löscht. Das nachfolgende Sequenzdiagramm zeigt die Instanziierung der Klasse, welche beim Start des Webservers erfolgt.

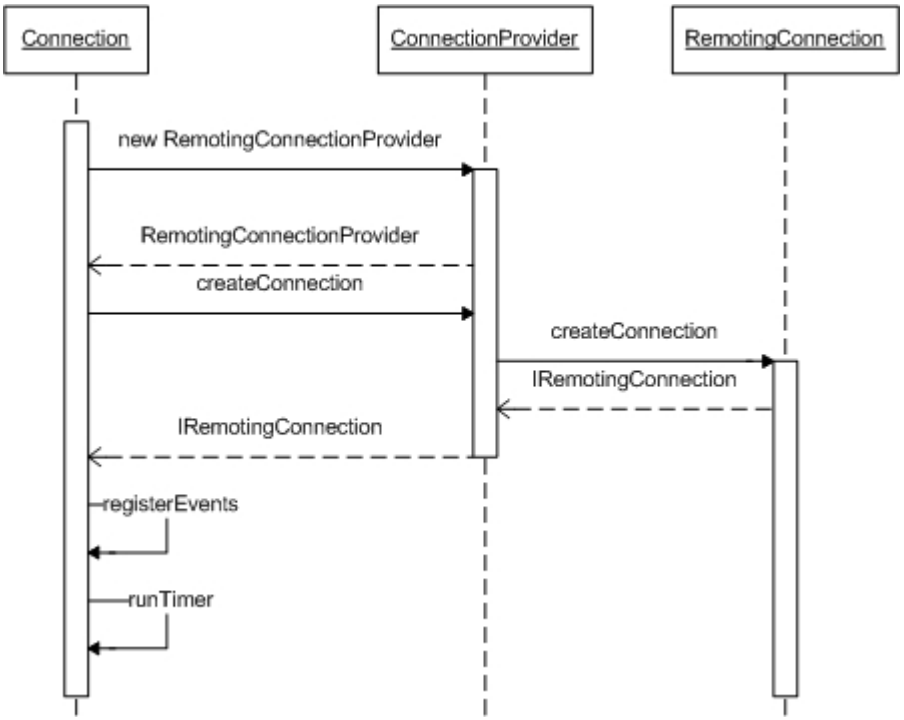


Abb. 5.4: Instanziierung der Connection-Klasse

Im Aufruf des Konstruktors der Connection-Klasse wird zuerst ein Objekt vom Typ RemotingConnectionProviders erzeugt. Anschließend werden die zu verwendenden Ports und das Impersonati-

ons-Level aus der Konfigurationsdatei Web.config ausgelesen und gesetzt. Danach wird die create-Connection()-Methode des ConnectionProviders aufgerufen. Sie liefert – wiederum mit den Parametern aus der Web.config, wie z.B. der IP-Adresse des Hosts und der Verbindungsart (TCP oder HTTP) – die eigentliche Verbindung in Form eines Objektes der IRemotingConnection-Klasse. Danach werden die Events für den Auf- und Abbau der Verbindung registriert und der Timer gestartet. Zu diesem Zeitpunkt ist die Verbindung noch inaktiv, es existiert keine Session für den Webserver am H2-Toolset-Server. Die Erstellung dieser zeigt das unten stehende Sequenzdiagramm.

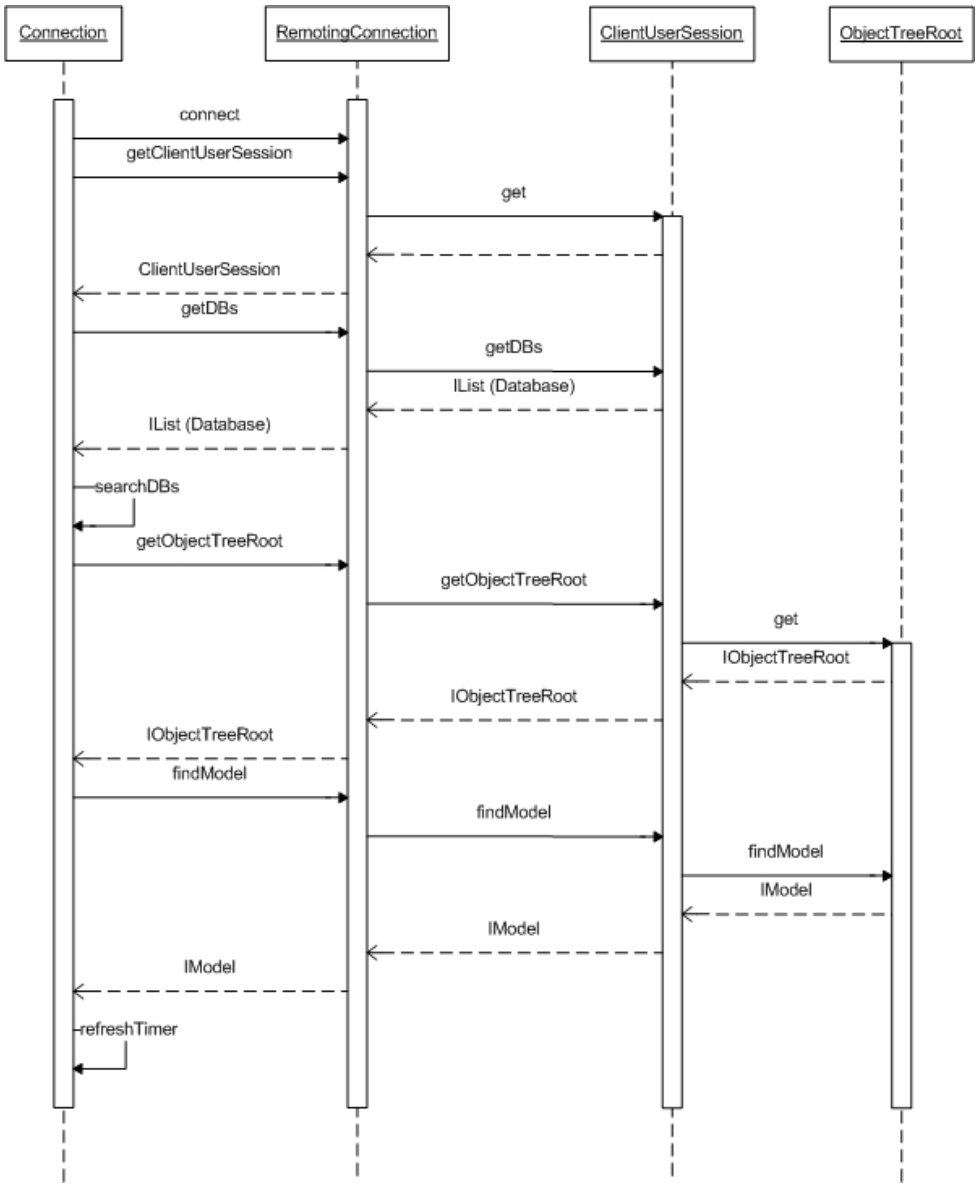


Abb. 5.5: Verbindungsaufbau und Laden eines Modells

Die Verbindung wird beim Zugriff auf Objekte auf dem H2-Toolset-Server aktiviert. Dies geschieht in der connect()-Methode und dem ConnectEventHandler. Zuerst wird die Verbindung über den Aufruf der connect()-Methode der RemotingConnection aktiviert, bevor eine ClientUserSession für den in der Web.config hinterlegten Benutzer, der den Webserver beim H2-Toolset-Server repräsentiert, angelegt wird. Um nun die für den Konfigurator benötigten Modelle - HLBAT und HLBAI – zu laden, wird über die ClientUserSession eine Auflistung aller vorhandenen Modelldatenbanken auf dem

Server angefordert. Aus dieser wird die benötigte Datenbank gesucht und die Wurzel des ObjectTree, ein Interface vom Typ IObjectTreeRoot geladen. Über die FindModel()-Methode dieses Interface kann dann das gesuchte Modell, eine Schnittstelle vom Typ IModel, anhand seines Namens geladen und in der Connection gespeichert werden. Die für die Suche nach Datenbank und Modell notwendigen Namen werden wiederum aus der Web.config ausgelesen. Da die Erfassung von Änderungen an den Modellen für die Konsistenz der Konfiguration durch den Kunden kritisch ist, werden die Modelle bei jedem Aufruf des ConnectEvents neu geladen, um eine möglichst aktuelle Datenbasis zu garantieren und gleichzeitig den laufenden Betrieb nicht unnötig zu unterbrechen. Abschließend werden die Timer für die Inaktivität der Verbindung und das Aktualisieren des Cache aktualisiert. Das Trennen der Verbindung zeigt das nachfolgende Sequenzdiagramm.

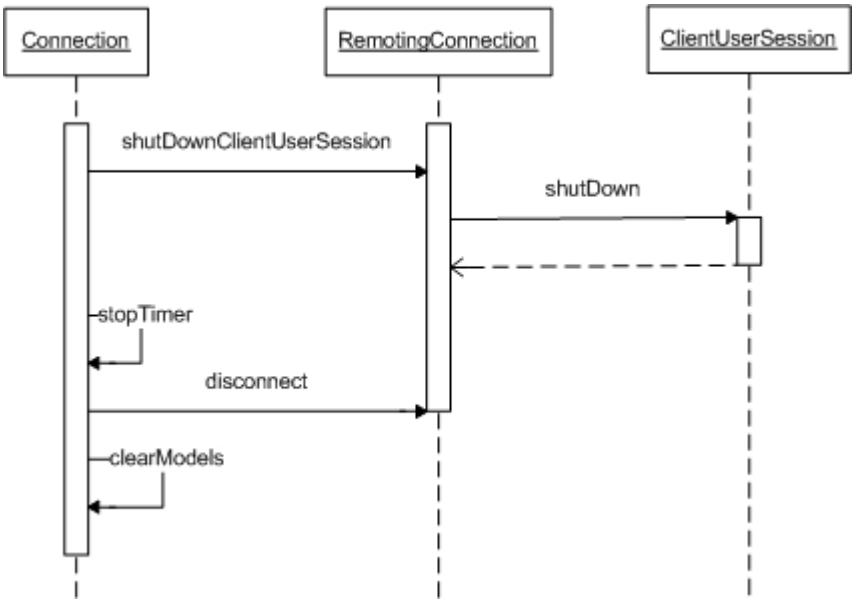


Abb. 5.6: Trennen der Verbindung und Zurücksetzen der Modelle

Bei längerer Inaktivität oder dem Beenden des Webservers wird die Verbindung zum H2-Toolset-Server getrennt. Dies erfolgt durch das Aufrufen der disconnect()-Methode und des dazugehörigen DisconnectEventHandlers. Die ClientUserSesssion wird dann über die shutDown()-Methode beendet, der Timer der Connection gestoppt und die Verbindung anschließend über die disconnect()-Methode der RemotingConnection deaktiviert. Es besteht dann keine Session auf dem H2-Toolset-Server mehr, die Verbindung ist inaktiv. Abschließend werden die gespeicherten Modelle in der Connection auf NULL gesetzt.

Nutzerverwaltung

Um eine Kommunikation zwischen dem Anbieter und den Kunden zu ermöglichen, werden von jedem Nutzer des Konfigurators im Rahmen einer Registrierung Kontaktdaten abgefragt. Die Speicherung dieser Daten erfolgt auf dem H2-Toolset-Server. Die Nutzer, die im Konfigurator angelegt werden, sind keine H2-Toolset-Server-Benutzer, d.h. beim H2-Toolset-Server wird immer nur ein Benutzer, der Webserver, angemeldet und es wird nur eine Verbindung zu diesem aufgebaut. Die Verwaltung der Nutzer des Konfigurators erfolgt ausschließlich über den Webserver. Das untenste-

hende Klassendiagramm gibt einen Überblick über die im Rahmen der Nutzerverwaltung verwendeten Klassen.

Abb. 5.7: Klassen für die Nutzerverwaltung

Die Klassen UserRegister, UserLogin und UserEdit im Package UserLogin sind die Webseiten, über die der Nutzer sich registrieren, anmelden und seine Daten ändern kann. Die Geschäftslogik wird von der Klasse UserManagement übernommen, die Daten des Kunden und Daten ggf. gespeicherter Konfigurationen werden in den Datenhaltungsklassen Customer, HLBAI und ConfiguredModules auf dem Webserver vorgehalten.

Die Einhaltung der Konventionen für den Nutzernamen und das Passwort wird in der Klasse UserRegister über Objekte vom Typ ASP .NET RegularExpressionValidator und entsprechende RegularExpressions sichergestellt. Der Nutzer wird über die Register()-Methode der UserManagement-Klasse als neue ObjectDefinition in der H2-Toolset-Solution im Kontext „Kunde“ im Modell HLBAT mit den angegebenen Daten angelegt. Zugleich wird auch ein neues Objekt des Typs Kunde im Modell HLBAI im Kontext „Hybrides Leistungsbündel auf Instanzebene“ angelegt, um das Speichern von Konfigurationen durch den Kunden zu ermöglichen.

Nach erfolgreicher Registrierung oder Anmeldung über die UserLogin-Seite wird ein Objekt der Klasse Customer erstellt und in der Sessionvariable user gespeichert. Dies ermöglicht einen direkten Zugriff der Webseiten-Klassen auf die Daten des angemeldeten Nutzers und eine schnelle Überprüfung, ob eine Anmeldung vorliegt. Zudem wird der Nutzer in ein als Attribut der Klasse Connection gespeichertes Dictionary eingetragen. Dieses Dictionary, ActiveUsers, enthält als Schlüssel die Nutzernamen aller aktuell angemeldeten Nutzer und als Wert jeweils die dazugehörige SessionID. So wird sichergestellt, dass mit einem Nutzernamen immer nur genau ein Kunde auf dem Webserver angemeldet ist. Diese Einschränkung ist notwendig, um Schreibkonflikte bei gleichzeitigem Zugriff auf den H2-Toolset-Server zu vermeiden. Eine Implementierung eines Transaktionsmanagements würde die ggf. auftretenden Probleme in technischer Hinsicht beheben, jedoch ein evtl.

unbeabsichtigtes Überschreiben von Konfigurationen nicht verhindern. Ein Verzicht auf das Überschreiben von gespeicherten Konfigurationen ist aufgrund des zu erwartenden hohen Anteils nicht vollständiger Konfigurationen, des steigenden Datenaufkommens und der sinkenden Qualität einer Analyse der von den Kunden hinterlegten Konfigurationen ebenfalls keine Alternative zu der umgesetzten Lösung. Zudem würde das Konzept der individuellen Präferenzerfassung[28] dadurch konterkariert. Bei Ablauf einer Session oder einer Abmeldung wird der Nutzername wieder aus dem Dictionary entfernt. Sollte ein Nutzer ggf. aufgrund eines technischen Defektes den Browser schließen, bleibt der Nutzer bis zum Timeout der Session im Dictionary. Bei einem erneuten Aufruf der Seite erfolgt durch die Wiederverwendung der SessionID eine automatische Anmeldung. Die Customer-Klasse enthält neben den Kundendaten auch die IDs der ObjectDefinition und ObjectOccurrence des Kundenobjektes im Modell HLBAT auf dem H2-Toolset-Server, um einen schnellen Zugriff über die modelleigenen get()-Methoden für weitere Schreib- und Lesezugriffe zu ermöglichen. Eine Änderung der Kundendaten ist über die UserEdit-Seite, auf welche der Nutzer bei einer erfolgreichen Registrierung oder Anmeldung weitergeleitet wird, möglich. Die Validierung der Eingaben wird von Objekten vom Typ ASP .NET RegularExpressionValidator und die edit()-Methode der UserManagement-Klasse geleistet. Neben den von den Kunden registrierbaren Nutzernamen gibt es mit dem Administrator- und dem Gastzugang zwei reservierte Nutzernamen. Der Administrator muss manuell im HLBAT als Kunde „Admin“ vom Anbieter mit einem frei wählbaren Passwort angelegt werden, der Gast als Kunde „Guest“ mit dem Passwort „guest“. Als Administrator erhält der Nutzer auf der Seite UserEdit Zugriff auf Funktionen zur Konfiguration des Webservers und dem manuellen Löschen des Cache auf dem Webserver. Letzteres setzt bis auf die Customer-Objekte alle von der Seite Configurator benötigten Objekte aus dem OwnObjects-Package auf NULL und lädt die Modelle in der Connection-Klasse neu. Der Gastzugang ermöglicht eine Konfiguration von hybriden Leistungsbündeln ohne Vorschlagsgenerierung und Speicherung der Konfigurationen und ist für Nutzer gedacht, die den Konfigurator primär als Informationsquelle über das Angebot des Anbieters nutzen wollen. Um eine Auswertung der über den Gastzugang getätigten Konfigurationen zu ermöglichen, wird die aktuelle Konfiguration beim Timeout der Session automatisch über die saveConfiguration()-Methode aus der ConfiguratorManagement-Klasse gespeichert.

Zusätzlich werden gespeicherte Konfigurationen hybrider Leistungsbündel des Kunden in einer ArrayList im entsprechenden Customer-Objekt gespeichert. Die in dieser Liste enthaltenen Objekte der Klasse HLBAI enthalten die für ein Laden der Daten zu einer gespeicherten Konfiguration notwendigen Informationen. Um diese Objekte zu erstellen, wird auf dem H2-Toolset-Server im Modell HLBAI im Kontext „Hybrides Leistungsbündel auf Instanzebene“ zuerst der Kunde anhand seines Nutzernamens gesucht. Die Kinder dieses Objektes, Objekte vom Typ Hybrides Leistungsbündel auf Instanzebene, werden dann durchlaufen und mit den enthaltenen Daten HLBAI-Objekte erstellt. Neben dem Namen und den IDs für die ObjectDefinition und ObjectOccurrence im Modell HLBAI wird die eine einem Attribut im Modell HLBAI hinterlegte ObjectDefinitionID des zu Grunde liegenden

[28] Zur Erfassung und Verwendung von Präferenzen siehe den Abschnitt Recommender System

HLBATs im Modell HLBAT erfasst. Diese ID wird verwendet, um bei einer Aktivierung einer gespeicherten Konfiguration das entsprechende HLBAT zu laden und dort die Daten der einzelnen Leistungen auszulesen. Sie stellt also die Verbindung zwischen den Modellen her. Des Weiteren wird geprüft, ob eine gespeicherte Konfiguration noch aktuell ist, d.h. ob das Datum der letzten Bearbeitung des hybriden Leistungsbündels auf Instanzebene[29] nach dem des hybriden Leistungsbündels auf Typebene liegt. Somit wird ein Laden von nicht mehr gültigen Konfigurationen, z.B. durch das Löschen von Leistungen oder die Änderung von Regeln, verhindert. Jedes HLBAI enthält noch eine ArrayList mit Objekten der Klasse ConfiguredModule. Diese enthalten die ausgewählte Leistung für ein Modul des HLBAI. Erfasst werden an dieser Stelle nur die IDs aus dem Modell HLBAI, sowie die Tatsache, ob ein Sonderwunsch vom Nutzer spezifiziert wurde. Die gefundenen und bezüglich des Bearbeitungsdatums zulässigen HLBAI-Objekte werden dem Nutzer in Form einer RadioButtonList mit dem Namen des hybriden Leistungsbündels und dem Datum der letzten Änderung zur Auswahl bereitgestellt. Bereitgestellt werden die HLBAIs über die loadConfigurations()-Methode aus der Klasse UserManagement. Das Laden der Daten zu einer gespeicherten Konfiguration und die Aktivierung dieser wird durch die Klasse ConfiguratorManagement übernommen.

Konfiguratorverwaltung

Der Konfigurator wird mit dem Post-Parameter hlb, der den Namen des HLBAT angibt im Browser aufgerufen (localhost/Configurator/Configurator.aspx?hlb=T-Serie). Dieser Name wird verwendet um das richtige HLBAT aus dem Kontext „Hybrides Leistungsbündel auf Typebene“ zu laden. Die in Abb. 5.8 gezeigte Klasse ConfiguratorManagement stellt dazu die Methode getHLB() bereit.

[29] Dieses Datum wird im H2 Toolset automatisch gespeichert und kann für jede ObjectOccurrence über die Methode getLatestChildOccurrenceModification abgerufen werden.

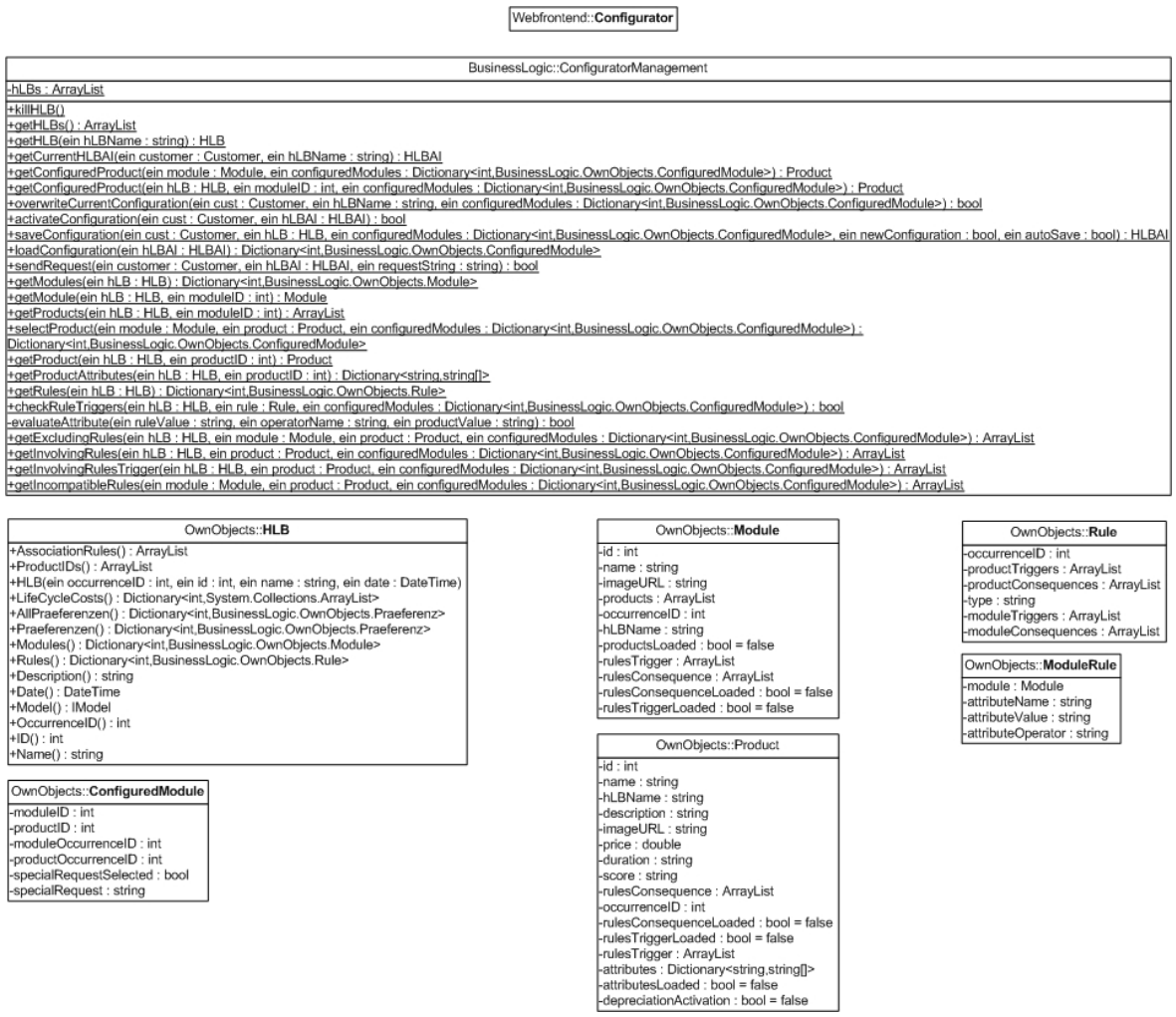


Abb. 5.8: Klassen zur Konfiguratorverwaltung

Beim ersten Zugriff werden die benötigten Daten vom H2-Toolset-Server angefordert und in einem Objekt vom Typ HLB gespeichert. Gleichzeitig wird dieses Objekt in die ArrayList hLBs der ConfiguratorManagement-Klasse aufgenommen und kann von hier aus nach einer Prüfung des letzten Änderungsdatums ohne weiteren Zugriff auf den H2-Toolset-Server verwendet werden.

Alle im HLB enthaltenen ArrayLists und Dictionaries werden beim ersten Zugriff geladen. So ruft der getter der Modules-Property beispielsweise einmalig die Methode getModules() der Klasse ConfiguratorManagement auf. Dies geschieht spätestens zu dem Zeitpunkt in dem die linke Navigationsleiste initialisiert wird. Genau so verhält sich der getter des Module-Objektes im Bezug auf die Leistungen. Eine bestimmte Leistung bzw. Modul kann über die Methode getProduct() bzw. getModule() aufgefunden werden.

Regeln

Alle Regeln werden einmal aus dem H2-Toolset-Server geladen und in der ArrayList rules im HLB Objekt gespeichert (Methode: getRules()). Jede Regel hat das Attribut „Typ“ in dem der Kantentyp der Regel gespeichert wird (bedingt/schliesst aus). Die vier ArrayLists enthalten die Auslöser und Konsequenzen der Regel in Form von Modulen bzw. Leistungen. Im Fall von Leistungen werden die

Objekte vom Typ Produkt direkt darin abgelegt. Bei Modulen werden Objekte vom Typ ModuleRule darin gespeichert, weil zusätzliche Informationen über die Leistungseigenschaften mit ihren Bedingungen (Operator und Wert) erforderlich sind (z.B. Preis >= 500000).

Jedes Produkt und Modul hat ein Attribut RulesTrigger und RulesConsequence. In RulesTrigger sind die Regeln aufgelistet, in denen die Leistung bzw. das Modul als Auslöser agiert. Bei den RulesConsequence-Regeln handelt es sich um die Konsequenz. Beide Listen werden gefüllt indem einmalig alle Regeln, die im HLB-Objekt hinterlegt sind durchsucht und ggf. referenziert werden.

Beim Aufrufen eines Moduls im Konfigurator werden die Regeln abgefragt. Dabei werden drei Fälle behandelt:

- Wird eine Leistung durch eine aktive Regel bedingt (Methode: getInvolvingRules())
- Ist eine Leistung durch eine aktive Regel ausgeschlossen (Methode: getExcludingRules())
- Gibt es Regeln, die bei Auswahl der Leistung aktiv werden würden und aufgrund der bisherigen Konfiguration nicht eingehalten werden können (Methode: getIncompatibleRules())

Im 1. Fall werden alle anderen Leistungen, die zu dem Modul gehören deaktiviert. Beim 2. und 3. Fall wird die Leistung selber deaktiviert. Deaktivierte Leistungen sind weiterhin sichtbar, können aber nicht ausgewählt werden.

Die Methode checkRuleTriggers() wird als Hilfsfunktion verwendet um zu prüfen ob eine Regel aktiv ist, also alle Auslöser durch die aktuelle Konfiguration erfüllt sind. Sofern Module in eine Regel mit einbezogen werden dient die Methode evaluateAttribute() dazu festzustellen, ob die Eigenschaft der Leistung der in der ModuleRule spezifizierten Anforderung einer Regel entspricht.

Gespeicherte Konfigurationen

Konfigurationen werden in ConfiguredModules gespeichert, welche in einer ArrayList im HLBAI-Objekt der Klasse Customer vorgehalten werden. Das Laden einer vorher gespeicherten Konfiguration, wie es in Abschnitt Nutzerverwaltung beschrieben wurde ist über die Methode activateConfiguration() möglich, die bei dem entsprechenden HLBAI das Attribut Current auf den Wert true setzt. Das aktuell geladene HLBAI kann dann über die Methode getCurrentHLBAI() anhand dieses Attributwertes ermittelt werden. Die Speicherung eines HLBAI wird über die Methode saveConfiguration() vorgenommen, die entweder durch einen Klick auf die Schaltfläche „Konfiguration speichern“ oder bei einem Session-Timeout als als AutoSave-Konfiguration ausgelöst wird. Im ersten Fall wird eine vorher geladene Konfiguration dabei überschrieben. Für jeden Kunden steht eine Autosave-Konfiguration bereit, die bei jedem Session-Timeout ebenfalls überschrieben wird. Eine Ausnahme bilden dabei die Konfigurationen über das Gast-Konto. Hier werden alle Konfigurationen gespeichert. Eine bereits gespeicherte Konfiguration kann über die sendRequest()-Methode zu einer Anfrage an den Anbieter umgeschrieben werden. Dies wird über das Attribut isRequest in der HLBAI-

Klasse und die zwei Attribute, „Anfrage“ und „AnfrageAnmerkungen“ im Modell HLBAI realisiert. Anfragen werden nicht überschrieben, für jede gespeicherte Konfiguration existiert maximal eine Anfrage.

Wie bereits erwähnt werden die Konfigurationen in Form von ConfiguredModules gespeichert. Darin wird einem Modul eine Leistung oder ein Sonderwunsch zugeordnet. Eine vollständige Konfiguration wird durch ein Dictionary von ConfiguredModules im HLBAI-Objekt beschrieben. Sobald im Konfigurator eine Leistung über die „wählen“ Schaltfläche ausgewählt wurde wird dem Dictionary ein entsprechendes ConfiguredModule mit dem Key ModuleID hinzugefügt oder das bestehende angepasst. Diese Aufgabe übernimmt die Methode overwriteCurrentConfiguration(). Gleichzeitig werden alle durch diese Leistung bedingten Leistungen mit ausgewählt. Die entsprechenden Regeln werden über die Methode getInvolvingRulesTrigger() abgerufen. Das Ergebnis ist ein neues Dictionary von ConfiguredModules, welches durch overwriteCurrentConfiguration() zur aktuellen Konfiguration gemacht wird.

Lebenszykluskosten und TCO-VOFI

Neben den oben genannten Daten werden dem Kunden die über die gesamte (vom Anbieter erwartete) Lebensdauer einer Leistung anfallenden Auszahlungen, aus denen sich die Lebenszykluskosten ergeben, in Form einer Tabelle dargestellt. Die Berechnung dieser Zahlungsreihen für ein HLBAT wird in der Klasse LifeCycleCostManagement ausgeführt.

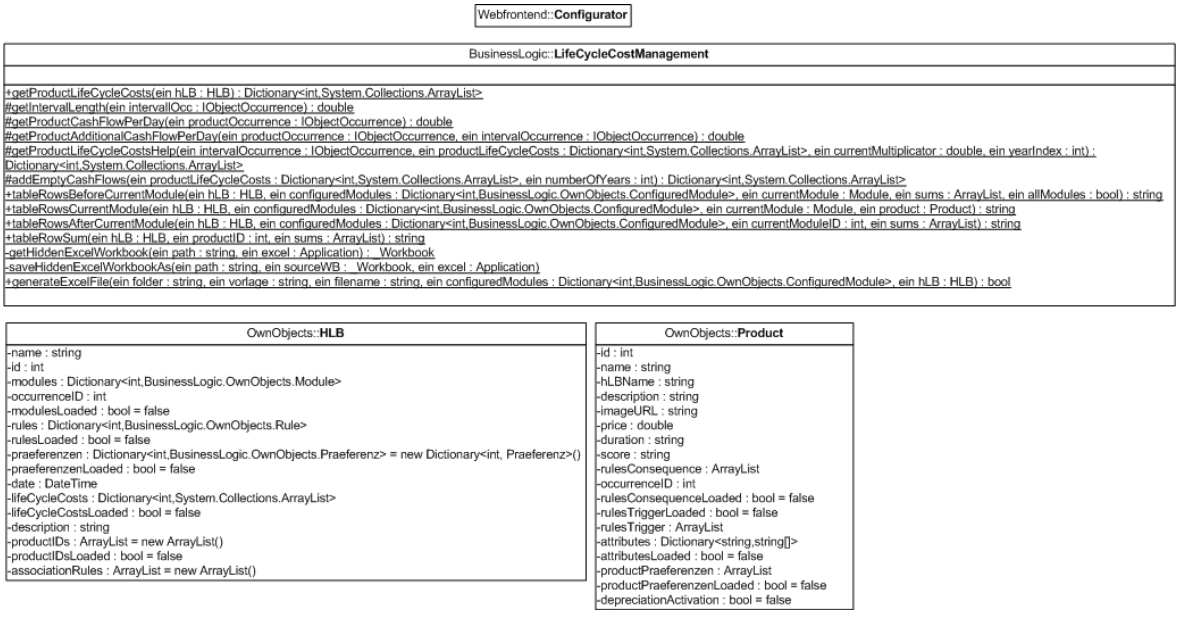


Abb. 5.9: Klassen zur Lebenszyklusdarstellung

Beim Erstellen des HLB-Objektes durch die ConfiguratorManagement-Klasse wird die Methode getProductLifeCycleCosts() aufgerufen. Diese gibt ein Dictionary mit der ID der im HLB enthaltenen Product-Objekte als Key und einer ArrayList mit double Werten als Value zurück. Die ArrayList enthält die für jede Periode – gemessen in Jahren - die Zahlung aus Sicht des Kunden. Zur Berechnung wird wie folgt vorgegangen. Im Kontext „Hybrides Leistungsbündel auf Typebene“ im Modell

HLBAT werden alle relevanten Lebenszyklusphasen, d.h. mit „Kauf“, „Nutzung“ oder „Nachnutzung“ benannte Phasen, durchlaufen. Andere Lebenszyklusphasen sind für die Lebenszykluskosten aus Kundensicht nicht relevant und werden ignoriert. Für jedes Intervall innerhalb einer Lebenszyklusphase wird nun die Länge in Tagen mit der Methode getIntervalLength() bestimmt. Dies geschieht durch das Auslesen der Attribute „Dauer“ und „Regelmäßigkeit“ und einer einfachen Umrechnung[30]. Ist eines der Attribute nicht gesetzt, wird eine Dauer von 0 Tagen angesetzt. Anschließend wird geprüft, ob das aktuell betrachtete Intervall in die aktuelle Periode der Lebenszykluskosten passt oder ob die Zahlungen mit einem Multiplikator, currentMultiplicator, aufgeteilt werden müssen. Da die Perioden eine Dauer von einem Jahr haben, wird restOfCurrentYear, die Prüfvariable, mit 360 initialisiert. Der Zähler für die aktuelle Periode, yearIndex, wird hingegen mit 0 initialisiert. Es ergeben sich für die Überprüfung folgende Möglichkeiten.

- Die Dauer des aktuellen Intervalls ist kürzer als die restliche Dauer der aktuellen Periode. In diesem Fall ist keine Aufteilung der Zahlungen innerhalb des Intervalls notwendig, der currentMultiplicator ist gleich Intervalldauer. Die Zahlungen werden für jede Leistung im Intervall berechnet und das Dictionary aktualisiert. Der yearIndex bleibt unverändert, die Dauer des Intervalls wird von der restlichen Dauer der Periode abgezogen.

- Die Dauer des aktuellen Intervalls ist gleich der restlichen Dauer der aktuellen Periode. Wiederum ist keine Aufteilung der Zahlungen innerhalb des Intervalls notwendig, der currentMultiplicator ist gleich der Intervalldauer. Die Zahlungen werden für jede Leistung im Intervall berechnet und das Dictionary aktualisiert. Der yearIndex wird um 1 erhöht, da die aktuelle Periode beendet ist, und restOfCurrentYear wird wieder auf 360 gesetzt.

- Die Dauer des aktuellen Intervalls ist größer der restlichen Dauer der aktuellen Periode. Eine Aufteilung der Zahlungen innerhalb des Intervalls ist notwendig, der currentMultiplicator ist gleich der restlichen Dauer der aktuellen Periode. Die Zahlungen werden für jede Leistung im Intervall berechnet und das Dictionary aktualisiert. Der yearIndex wird um 1 erhöht, da die aktuelle Periode beendet ist, und restOfCurrentYear wird wieder auf 360 gesetzt. Die Intervalldauer wird um die restliche Dauer der aktuellen Periode reduziert. Zudem wird über das Setzen der boolschen Variable intervalNotFinished sichergestellt, dass die Schleife nochmals über das aktuelle Intervall läuft, bis einer der beiden oben geschilderten Fälle eintritt.

- Einen Sonderfall stellt die Lebenszyklusphase „Kauf“ dar. Um sicherzustellen, dass die enthaltenen Leistungen gemäß der Konventionen der gewählten Lebenszykluskostendarstellung in einem TCO-VOFI als Einmalzahlungen erfasst werden, ist der currentMultiplicator gleich 1. Die modellierte Intervalldauer wird hier also nicht beachtet. Der yearIndex wird um 1 erhöht, da die aktuelle Periode beendet ist, und restOfCurrentYear wird wieder auf 360 gesetzt.

[30] Ein Jahr entspricht dabei 360 Tagen, ein Monat 30 Tagen.

Die in den aufgezählten Fällen erwähnte Berechnung der Zahlungen für jede Leistung im Intervall wird von der Methode getProductLifeCycleCostsHelp() ausgeführt. Dabei wird zuerst die ArrayList mit den bisherigen Zahlungen der Leistung aus dem Dictionary geladen oder ggf. neu erstellt. Ist Letzteres der Fall werden bis zur aktuellen Periode alle Perioden mit dem Wert 0 aufgefüllt. Danach wird die Zahlung der Leistung pro Tag über die Methode getProductCashFlowPerDay() bestimmt, welche die Attribute „Preis“ und „Regelmäßigkeit“ aus dem Kontext „Leistung“ ausliest und auf die Dauer eines Tages umrechnet. Eine nicht gesetzte Regelmäßigkeit führt zu keiner Umrechnung, es wird also eine Regelmäßigkeit von Tagen angenommen[31]. Eine ggf. vorliegende zusätzliche Zahlung im Intervall wird von der Methode getAdditionalProductCashFlowPerDay() über das Auslesen des entsprechenden Attributes und die Umrechnung über die Intervalldauer berechnet. Ein nicht gesetztes Attribut wird als zusätzliche Zahlung vom Betrag 0 interpretiert. Schließlich wird die für die Leistung im Intervall anfallende Zahlung, currentCashFlow, wie folgt berechnet:

```
currentCashFlow = (currentProductCashFlowPerDay *
                   currentMultiplicator) +
                   (currentProductAdditionalCashFlowInInterval * currentMultiplicator)
```

Dieser Betrag wird dann in die ArrayList mit den Zahlungen der Leistung eingetragen – bei schon bestehenden Zahlungen für die aktuelle Periode wird der Betrag hinzuaddiert - und diese wieder in das Dictionary geschrieben. So wird mit allen Leistungen des Intervalls verfahren.

Sind alle Lebenszyklusphasen durchlaufen, werden im Dictionary vor der Rückgabe noch nicht gesetzte Intervalle für alle Leistungen mit Hilfe der Funktion addEmptyCashFlows() mit dem Wert 0 aufgefüllt. Dadurch werden alle ArrayLists im Dictionary auf die gleiche Länge, die der gesamten modellierten Lebensdauer entspricht, normiert.

Zur Darstellung der in der HLB-Klasse gespeicherten Lebenszykluskosten in der Configurator-Seite als HTML-Tabelle dienen die Funktionen tableRowsBeforeCurrentModule(), tableRowsCurrentModule() und tableRowsAfterCurrentModule(). Erstere liest die gewählten Leistungen aus den ConfiguredModules-Objekten und erstellt daraus die Zeilen der Tabelle vor der Auswahl einer Leistung im betrachteten Modul. Die zweite Funktion ergänzt die Tabelle um die Zeilen für die zur Auswahl stehenden Leistungen im betrachteten Modul und kennzeichnet diese farbig. Letztere markiert die Zeile der gewählten Leistung in der Tabelle nochmals in einer anderen Farbe. Die Funktion tableRowSum() wird von der Configurator-Seite aufgerufen, um die Zeile der Tabelle zu berechnen, welche die Spaltensummen der angezeigten Leistungen enthält.

Zur weiteren Analyse der Lebenszykluskosten dient der Download eines TCO-VOFI in Form eines Excel-Dokumentes. Dieses basiert auf einem auf dem Webserver liegenden Excel-Dokument, in welches die Lebenszykluskosten der aktuellen Konfiguration eingetragen werden. Das Tabellenblatt „Daten“ dient der Eintragung der Zahlungen der Leistungen, im Tabellenblatt „TCO-Analyse“ wird

[31] Diese Annahme führt im Falle von vergessenen Attributen bei der Modellierung zu sehr hohen Auszahlungen, die ein Bemerken und Lokalisieren des Fehlers innerhalb kurzer Zeit wahrscheinlich machen

die eigentliche Berechnung[32] ausgeführt. Diese erfolgt ausschließlich über die Verwendung von Formeln, um Sicherheitsrisiken und Kompatibilitätsprobleme zu vermeiden.

Die Bereitstellung des TCO-VOFI als Excel-Dokument wird ebenfalls von der Klasse LifeCycleCostManagement übernommen. Hierfür wird die Funktion generateExcelFile() verwendet. Dieser werden als Argumente der Pfad der Excel-Vorlage auf dem Server, ein Dictionary mit ConfiguredModules-Objekten und ein HLB-Objekt übergeben. Mit Angabe des Pfades wird mit der getHiddenExcelWorkbook()-Methode im Hintergrund eine neue Excel-Datei anhand der Vorlage erstellt. Daraufhin werden alle ConfiguredModules-Objekte durchlaufen und für jede enthaltene Leistung die im HLB-Objekt gespeicherten Lebenszykluskosten in eine Zeile im Tabellenblatt „Daten“ eingetragen. Zusätzlich zu den Zahlungen wir noch der Name der Leistung und der Wert des Attributs „aktivierungspflichtig“ - dieser ist notwendig zur korrekten Abschreibungsberechnung - festgehalten. Anschließend werden die Periodennummern eingetragen und das Excel-Dokument mit Hilfe der saveHiddenExcelWorkbookAs()-Methode unter dem Namen des aktuellen Nutzers auf dem Server im Ordner der Vorlage gespeichert. Gegebenenfalls bereits vorhandene Dateien mit dem gleichen Namen werden dabei überschrieben. Diese Datei wir dann von der Configurator-Seite über die HTTP Response dem Nutzer zum Download angeboten.

[32] Die Berechnung erfolgt anhand des Schemas von GROB, vgl. Grob, Lahme (2004), S. 3-7

6 Fazit und Ausblick

Mit der in diesem Projektseminar entworfenen Modellierungssprache **H2-ServPay** wurde gezeigt, wie sich hybride Leistungsbündel strukturiert und unter Berücksichtigung der spezifischen Charakteristika von Leistungen (bestehend aus Leistungen, die Sach- und/oder Dienstleistungen sein können) über ihre gesamte Lebensdauer hinweg abbilden lassen. Die entwickelte Metasprache ermöglicht aufgrund ihres generischen Charakters die Modellierung beliebiger Varianten von hybriden Leistungsbündeln. Daher ist die Toolunterstützung vor dem Hintergrund vielfältigster Geschäftsmodelle und Leistungen einsetzbar.

Das parallel entwickelte Kostenkalkulationstool ermöglicht dem Anbieter die im Modell hinterlegten Kosten auszulesen, um anhand dieser die berichtigten Herstellkosten der im hybriden Leistungsbündel enthaltenen Leistungen zu ermitteln. Neben dieser Berechnung kann das Tool zur ex post Kontrolle der mit der Leistungserstellung und -erbringung einhergehenden Kosten auf beliebiger Detailstufe eingesetzt werden. Auch eine stetige Überwachung der Kosten während des Gestaltungsprozesses des hybriden Leistungsbündels ist möglich. Durch den konzipierten webbasierten Leistungskonfigurator wird eine kundenseitige Konfiguration hybrider Leistungsbündel ermöglicht. Der Kunde hat dadurch die Möglichkeit, sich hybride Leistungsbündel nach seinen Bedürfnissen und den sprachlich vorgegebenen Konfigurationsparametern zu erstellen. Die automatisierte Unterstützung des Kunden im Konfigurationsprozess verbessert die Passgenauigkeit der individuell zusammengestellten hybriden Leistungsbündel und verhindert in diesem Zusammenhang auch den Abbruch von Konfigurationen aufgrund von Überforderung oder Indifferenz. Zusätzliche Transparenz und Hilfe bei der Entscheidungsfindung wird über die Möglichkeit einer TCO-Analyse des zusammengestellten Leistungsbündels geschaffen. Des Weiteren werden umfangreiche Daten über die Kunden sowie die durchgeführten Konfigurationen im Konfigurator erfasst. Diese bilden die Basis für umfangreiche Analysemethoden, welche durch das ebenfalls im Rahmen dieses Projektes entwickelte Modellanalyse-Plugin unterstützt werden.

Eine anbieterseitig nutzbare graphische Benutzerschnittstelle (GUI), welche die Modellierung von hybriden Leistungsbündeln mit der **H2-ServPay**-Sprache automatisiert unterstützt und somit komfortabler gestaltet, könnte im Rahmen von zukünftigen Entwicklungen realisiert werden. Eine solche GUI würde zum einen dafür sorgen, dass sprachliche Modellierungskonventionen eingehalten werden und somit korrekte Modelle erstellt werden. Dies ist insbesondere für unerfahrene Nutzer der H2-Toolset-Solution sinnvoll. Zum anderen könnte die GUI über eine Integration von Funktionen aus Kostenkalkulationstool und Leistungskonfigurator die Auswirkungen der aktuellen Modellierung bezüglich der Anbieter- und der Kundensicht, z.B. über eine Vorschau der Zahlungen im Lebenszyklus auf Anbieter- und Kundenseite, anzeigen und somit die Qualität der Modellierung steigern.

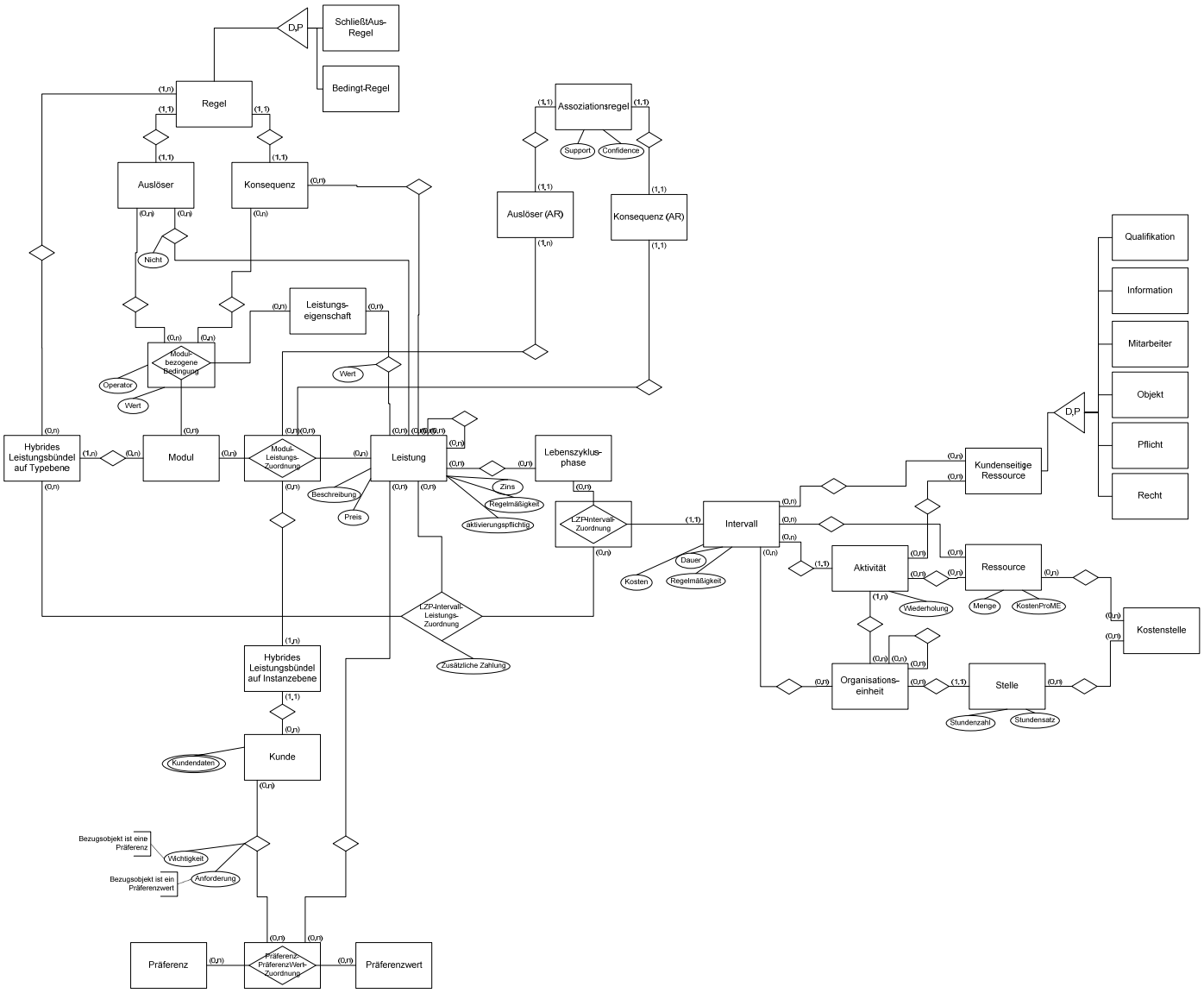
Des Weiteren wird die Erfassung der kundenseitigen Zahlungsbereitschaften im Rahmen des ServPay-Projektes weiter verfolgt. Aufbauend auf der entwickelten **H2-ServPay**-Sprache, ist es in diesem Teilprojekt das Ziel, über eine ConJoint-Analyse genaue Aussagen über die Zahlungsbereitschaft für einzelne Leistungen zu erfassen. Die Generierung von Stimuli erfolgt dabei auf Basis der

modellierten hybriden Leistungsbündel. Die Ergebnisse der Analyse könnten dann z.B. verwendet werden, um die Preisgestaltung in Kombination mit den Daten aus dem Kostenkalkulationstool zu optimieren.

Anhang

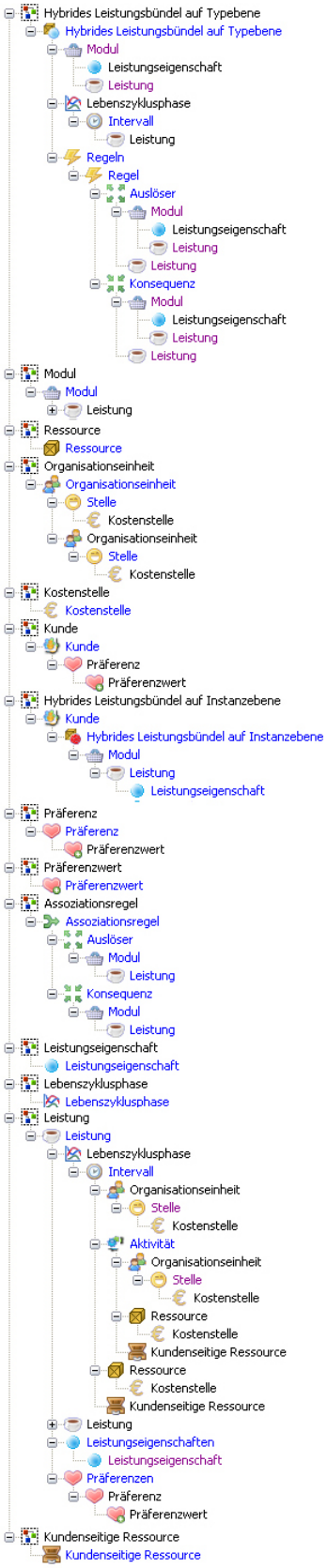
A H2 Sprachdefinition

.a Ganzheitliches Datenmodell



Anh. 1: Ganzheitliches Datenmodell

.b Ganzheitliche H2-Sprachdefinition



Anh. 2: Ganzheitliche H2-Sprachdefinition

B Wissensbasierte Vorschlagsgenerierung nach Ardissono und Goy

Dieser Ansatz wurde 2001 von Ardissono und Goy in ihrem Artikel „Tailoring the Interaction With Users in Web stores"[33] vorgestellt. Zur Bestimmung der Kundenpräferenzen werden stereotypische Nutzermodelle verwendet, denen die Nutzer anhand ihrer demografischen Daten zugeordnet werden. Da die Nutzer des in diesem Projekt entwickelten Leistungskonfigurators aus dem Geschäftskundenbereich stammen und die Aussagekraft der unternehmensbezogenen Daten geringer einzuschätzen ist als die personenbezogener demografischer Daten, wird zur Bestimmung der Kundenpräferenzen ein kompositionelles Verfahren herangezogen. Die mit dieser Methode bestimmten Präferenzen entsprechen den ursprünglich von Ardissono und Goy vorgestellten Kundendaten. Ein Beispiel für die Präferenz Lieferzeit ist im Folgenden dargestellt.

Lieferzeit Wichtigkeit ($imp_{Lieferzeit}$): 85%

Präferenzwert ($p_{Lieferzeit}$):

Kurz (2 bis 4 Wochen)	65%
Mittel (4 bis 8 Wochen)	25%
Lang (mehr als 8 aber maximal 14 Wochen)	10%

Der Präferenz Lieferzeit wird eine Wichtigkeit von 85% zugeordnet. Diese stellt die Gewichtung der Präferenz bei der Berechnung des Scores für eine Leistung dar. Die Summe der Gewichte muss dabei nicht 100% ergeben. Den drei Präferenzwerten kurz, mittel und lang werden vom Kunden dann Werte zugeordnet, welche ausdrücken, in welchem Maß die Präferenzwerte seine Präferenz erfüllen. Im Beispiel würde eine Leistung mit kurzer Lieferzeit die Qualitätsanforderungen des Kunden zu einem Grad von 65 % erfüllen.

Die Berechnung des Scores einer Leistung erfolgt in zwei Schritten. Zuerst wird für jede Präferenz eine Präferenzscore nach der Formel

$$score_A = imp_A * p_A + (1 - imp_A)$$

berechnet. Daraus ergeben sie Teilergebnisse im Wertebereich [0;1]. Die einzelnen Präferenzscores werden dann in einem zweiten Schritt wie folgt

$$SCORE(score_A, score_B) = \frac{score_A * score_B}{score_A + score_B - score_A * score_B}$$

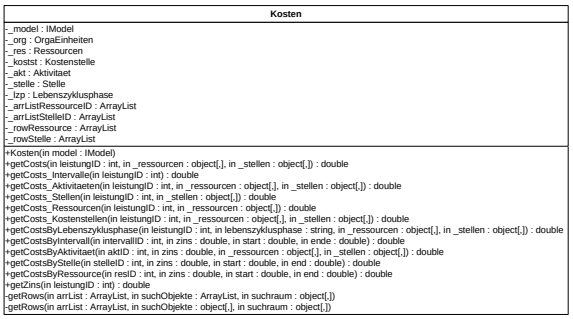
[33] An dieser Stelle soll nur eine kurze Einführung in die Berechnung der Bewertungen und die vorgenommenen Änderungen bezüglich der Präferenzaufnahme gegeben werden, für weitergehende Ausführungen insbesondere zur Herleitung der Berechnungsformeln finden sich bei Ardissono, Goy (2000), S. 251

verdichtet. Bei mehr als zwei Präferenzen erfolgt die Berechnung nach dem Schema:

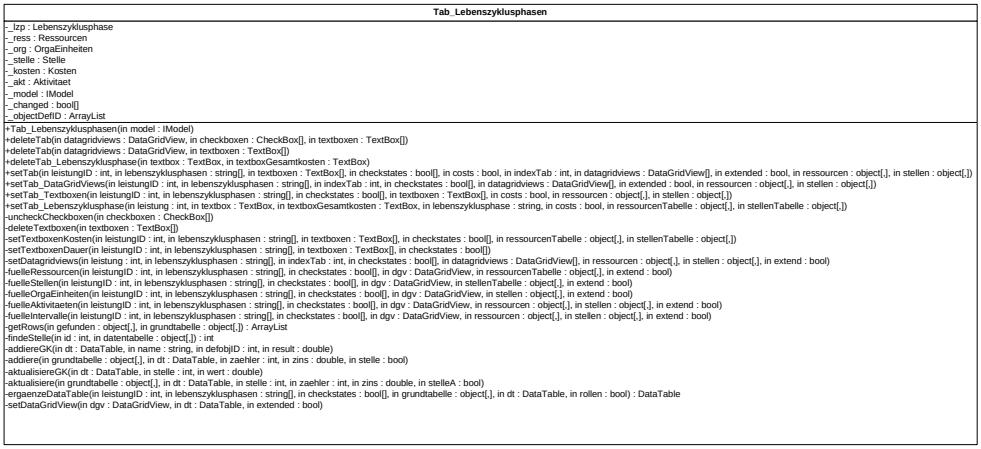
$$SCORE(score_A, score_B, score_C) = \frac{SCORE(score_A, score_B) * score_C}{SCORE(score_A, score_B) + score_B - SCORE(score_A, score_B) score_B}$$

Daraus ergeben sich ebenfalls Ergebnisse im Wertebereich [0;1]. Sind für eine Präferenz nicht für alle Leistungen Präferenzwerte zugeordnet worden, wird diese Präferenz bei der Berechnung nicht betrachtet.

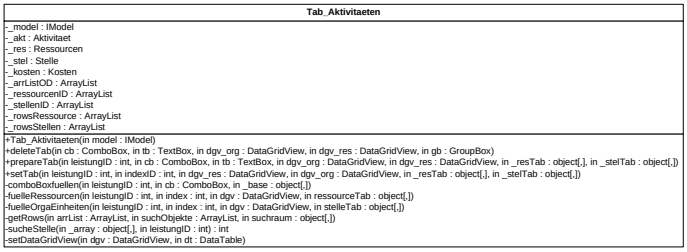
C Detaillierte Darstellung der Hilfsklassen



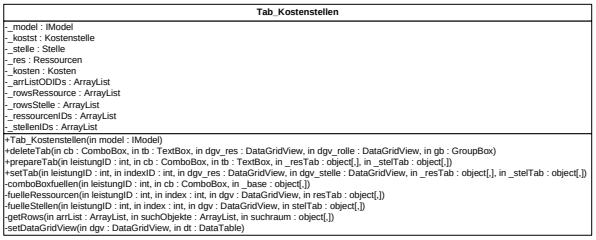
Anh. 3: Klasse Kosten



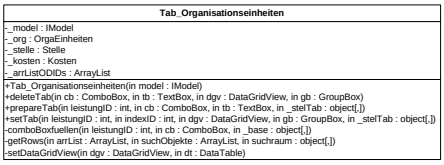
Anh. 4: Klasse Tab_Lebenszyklusphase



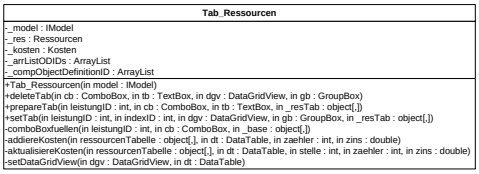
Anh. 5: Klasse Tab_Aktivitaeten



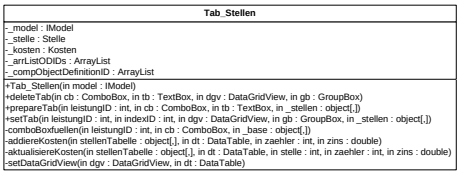
Anh. 6: Klasse Tab_Kostenstellen



Anh. 7: Klasse Tab_Organisationseinheiten



Anh. 8: Klasse Tab_Ressourcen



Anh. 9: Klasse Tab_Stellen

Literaturverzeichnis

AGRAWAL, R./IMIELINSKI, T./SWAMI, A. (1993), Mining association rules between sets of items in large databases. In Proceedings of the ACM SIGMOD International Conference on Management of Data, pp. 207-216, Washington D.C., May 1993.

BACKHAUS, K./WEDDELING, M. (2007), Neue Herausforderungen an die Vermarktung von produktbegleitenden Dienstleistungen, in: Belz, C./Schögel, M./ Tomczak, T. (Hrsg.): Innovation Driven Marketing - Vom Trend zur innovativen Marketinglösung, S. 84-86.

BECKER, J., BEVERUNGEN, D., KNACKSTEDT, R., MÜLLER, O. (2008a), Konzeption einer Modellierungssprache zur tool-unterstützten Modellierung, Konfiguration und Bewertung hybrider Leistungsbündel. GI-Tagung Modellierung, Workshop Dienstleistungsmodellierung. Berlin, 2008. S. 45-62.

BECKER, J., BEVERUNGEN, D., KNACKSTEDT, R., MÜLLER, O. (2008b), Modeling, Customer-Specific Configuration and Calculation of Value Bundles. In: Proceedings of the 14th Americas Conference on Information Systems (AMCIS). Toronto, Canada, 2008.

BEIER, MARIO (2006), Service Engineering - Ermittlung der Ressourcen- und Kostenstruktur eines neuen Dienstleistungsproduktes. Saarbrücken.

ENGELHARDT, W. H.; KLEINALTENKAMP, M.; RECKENFELDERBÄUMER, M. (1993), Leistungsbündel als Absatzobjekte. Ein Ansatz zur Überwindung der Dichotomie zwischen Sach- und Dienstleistungen. In: Zeitschrift für betriebswirtschaftliche Forschung, 45 (1993) 5; S. 395–426.

GROB, HEINZ L. (2001), Einführung in die Investitionsrechnung. 4. Auflage, Vahlen, München.

GROB, HEINZ L. (2006), Einführung in die Investitionsrechnung. 6. Auflage, Vahlen, München.

GROB, HEINZ L.; LAHME, NORMAN (2004), Total Cost Of Ownership-Analyse mit VOFI. In: Arbeitsberichte des Institut für Wirtschaftsinformatik, Nr. 20. Münster.

HILKE, W. (1989), Dienstleistungs-Marketing. Wiesbaden 1989.

SCHEER, AUGUST-WILHELM; GRIEBLE, OLIVER; KLEIN, RALF (2006), Modellbasiertes Dienstleistungsmanagement. In: Bullinger, Hans J.; Scheer, August-Wilhelm (Eds.): Service Engineering. Springer-Verlag, Berlin u.a..

SCHMID, MICHAEL (2005), Service Engineering Innovationsmanagement für Industrie und Dienstleister. Stuttgart.

VOM BROCKE, JAN (2006), Serviceorientiertes Prozesscontrolling. Gestaltung von Organisations und Informationssystemen bei Serviceorientierten Architekturen. Westfälische Wilhelms-Universität Münster, Münster.

VARGO, S.L.; LUSCH, R.F. (2004), The Four Service Marketing Myths. Remnants of a Goods-Based, Manufacturing Model. In: Journal of Service Research, Volume 6, No. 4, May 2004, S. 324-335.

ServPay
European Research Center for Information Systems (ERCIS)
Prof. Dr. Jörg Becker
Leonardo-Campus 3 | 48149 Münster
Tel: +49 (251) 83 - 3 81 00 | Fax: +49 (251) 83 - 2 80 62
info@servpay.de | http://www.servpay.de