Fakultät für Wirtschaftswissenschaften
Wismar Business School

Uwe Lämmel, Anatoli Beifert, Marcel Brätz,
Stefan Brandenburg, Matthias Buse, Christian Höhn,
Gert Mannheimer, Michael Rehfeld, Alexander Richter,
Stefan Wissuwa

Business Rules
Die Wissensverarbeitung erreicht die Betriebswirtschaft
Einsatzmöglichkeiten und Marktübersicht

Heft 05 / 2007

W D P

Wismarer Diskussionspapiere / Wismar Discussion Papers

Die Fakultät für Wirtschaftswissenschaften der Hochschule Wismar, University of Technology, Business and Design bietet die Präsenzstudiengänge Betriebswirtschaft, Management sozialer Dienstleistungen, Wirtschaftsinformatik und Wirtschaftsrecht sowie die Fernstudiengänge Betriebswirtschaft, Business Consulting, Business Systems, Facility Management, Quality Management, Sales and Marketing und Wirtschaftsinformatik an. Gegenstand der Ausbildung sind die verschiedenen Aspekte des Wirtschaftens in der Unternehmung, der modernen Verwaltungstätigkeit im sozialen Bereich, der Verbindung von angewandter Informatik und Wirtschaftswissenschaften sowie des Rechts im Bereich der Wirtschaft.
Nähere Informationen zu Studienangebot, Forschung und Ansprechpartnern finden Sie auf unserer Homepage im World Wide Web (WWW): http://www.wi.hs-wismar.de/.

Herausgeber: Prof. Dr. Jost W. Kramer
Fakultät für Wirtschaftswissenschaften
Hochschule Wismar
University of Technology, Business and Design
Philipp-Müller-Straße
Postfach 12 10
D – 23966 Wismar
Telefon: ++49/(0)3841/753 441
Fax: ++49/(0)3841/753 131
E-Mail: j.kramer@wi.hs-wismar.de

Vertrieb: HWS-Hochschule Wismar Service GmbH
Phillipp-Müller-Straße
Postfach 12 10
23952 Wismar
Telefon:++49/(0)3841/753-574
Fax: ++49/(0) 3841/753-575
E-Mail: info@hws-wismar.de
Homepage: http://cms.hws-wismar.de/service/wismarer-diskussions-brpapiere.html

ISSN 1612-0884
ISBN 978-3-939159-20-9
JEL-Klassifikation C65, C81, C60, C80

Printed in Germany

Inhaltsverzeichnis

1. Vorwort

Das Thema „*Business Rules*“ rückt seit einer Reihe von Jahren zunehmend in den Mittelpunkt des Interesses von IT-Anwendern, so dass es nahe liegt, sich diesem Ansatz verstärkt zuzuwenden. Mittels Business Rules kann betriebliche Anwendungssoftware und damit letztlich das Unternehmen schneller auf sich verändernde Bedingungen reagieren:

„Der Business-Rules-Ansatz schafft agile Unternehmen, die sich schnell und kostengünstig an die sich stets verändernde Umwelt anpassen können“ (Schacher/Grässler 2006: 1).

Durch Business Rules wird es den Anwendern ermöglicht, das Wissen über Abläufe und Bedingungen im Unternehmen explizit darzustellen und dieses Wissen einer automatisierten Verarbeitung zugänglich zu machen.

Die vorliegende Arbeit geht auf zwei Ideen zurück. Zum einen wenden sich IT-Unternehmen auch im regionalen Umfeld verstärkt dieser Thematik zu und versuchen, ihre IT-Systeme durch die Integration von Business-Rules-Komponenten flexibler zu gestalten. Zum anderen liegt es nahe, das Thema der Business Rules in einer Lehrveranstaltung „Wissensbasierte Systeme“ zu behandeln. Im Wintersemester 2006/2007 standen daher die Business Rules im Mittelpunkt der entsprechenden Lehrveranstaltung „Wissensbasierte Systeme“, die von Studenten des Master- sowie des Diplomstudiengangs Wirtschaftsinformatik besucht wurde.

Da den Studenten das Prinzip regelbasierter Wissensdarstellung und -verarbeitung aus dem Modul „Künstliche Intelligenz“ vertraut ist, wird ein Vergleich verschiedener Business-Rule-Systeme in den Mittelpunkt gestellt. Dazu wurden Anbieter von Business-Rule-Lösungen recherchiert und deren Produkte bewertet. Die Ergebnisse dieser kleinen Marktstudie werden in diesem Heft vorgelegt.

Um auch der eigentlichen Zielgruppe für den Einsatz von Business Rules - den Anwendern im Unternehmen - einen Einstieg in das Gebiet zu ermöglichen, werden das Konzept und die Einsatzmöglichkeiten von Business Rules in eigenen Kapiteln der Marktanalyse vorangestellt. Damit kann das vorliegende Heft zum Einstieg in die Thematik des Geschäftsregel-Ansatzes bzw. des Business-Rules-Ansatzes dienen. Darüber hinaus liefert es Informationen zu existierenden Systemen, so genannten Rules Engines, und kann Hilfestellung bei der Auswahl eines Systems geben.

Die Technologie der Geschäftsregeln ist eine Form des Einsatzes von Techniken der Künstlichen Intelligenz in betriebswirtschaftlichen Anwendungen. Einen Überblick über die Einsatzmöglichkeiten von Techniken der Künstlichen Intelligenz in betriebswirtschaftlichen Anwendungen wird in Haas (2006) gegeben.

Leider hat sich bisher die deutsche Übersetzung „Geschäftsregel“ noch

nicht durchsetzen können, so dass wir in diesem Heft immer wieder auf den englischen Begriff zurückgreifen, um Missverständnisse zu vermeiden.

2. Business Rules

Für den Informatiker kann der Business-Rules-Ansatz leicht definiert werden: Es ist die regelbasierte Wissensdarstellung und -verarbeitung für betriebswirtschaftliche Prozesse. Die Wissensdarstellung in Form von Regeln ist eine in der Künstlichen Intelligenz seit langem bekannte und gut verstandene Methode. Eine Regel stellt das Wissen immer in einer Wenn-Dann-Form dar, die unterschiedlich interpretiert werden kann:

WENN	Situation	DANN	Aktion
WENN	Bedingung	DANN	Ergebnis
WENN	Voraussetzung	DANN	Folgerung

Eine derartige Wissensdarstellung hat viele Vorteile. Regeln werden im Tagtäglichen häufig verwendet, sie sind intuitiv verständlich, entsprechen der menschlichen Denkweise und können als Handlungsanleitung oder auch zum Nachweis eines bestimmten Ergebnisses herangezogen werden.

Im folgenden Abschnitt wird das Konzept der Business Rules vorgestellt, welches die regelbasierte Wissensverarbeitung in betriebswirtschaftliche Anwendungen einführt.

2.1. Motivation

Die Unternehmen sind sich immer schneller ändernden Rahmenbedingungen ausgesetzt. Zum einen erfordert eine erfolgreiche Positionierung auf dem Markt ein schnelles Anpassen an das Marktgeschehen. Kürzere Zyklen in der Entwicklung, Produktion und Vermarktung von Produkten und Dienstleistungen sind notwendig. Diese dynamischen Prozesse sind von den IT-Systemen in hoher Qualität zu unterstützen. Eine Anpassung der IT-Systeme kann jedoch langwierig und kostenintensiv sein. Hier sind sehr flexible Lösungen erforderlich. Die Dynamik wird nicht nur durch die Märkte, sondern auch durch sich ständig ändernde Regelungen erhöht.

In jedem Unternehmen existieren Regelungen für die ablaufenden Prozesse: für die Beschaffung, für die Vermarktung, für die Kundenbeziehungen, für die Abrechnung, für das Personal, und so weiter. Diese Geschäftsregeln sind vorhanden, sie sind jedoch oft nicht einheitlich formuliert, sondern verbal in staatlichen Regelungen, in Betriebsvereinbarungen, in Arbeitsanweisungen festgeschrieben oder auch in IT-Systemen implementiert.

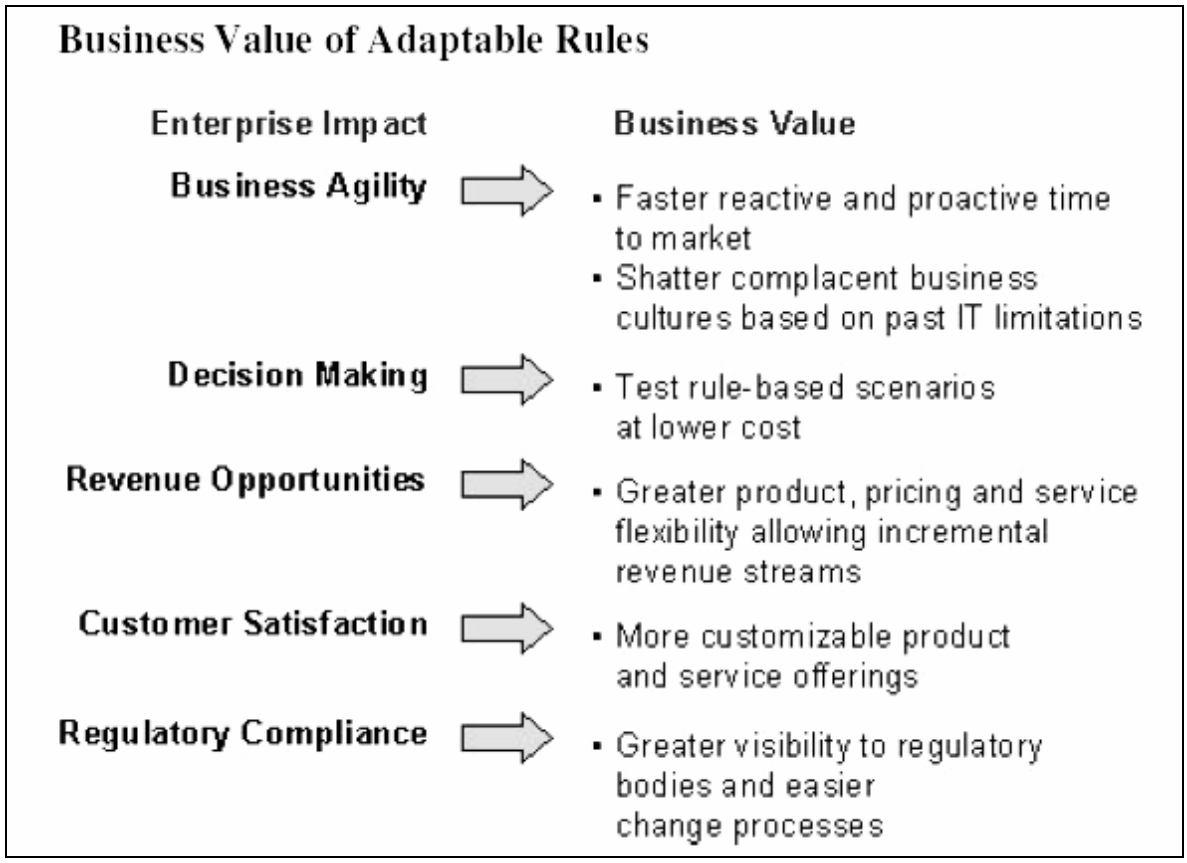
Der Business-Rule-Ansatz ist geeignet, die Flexibilität der IT-Systeme und damit auch der Unternehmen insgesamt zu erhöhen, indem die vorhandenen Geschäftsregeln (Business Rules) extrahiert, explizit dokumentiert und nach

Möglichkeit einer rechentechnischen Verwaltung und Verarbeitung zugänglich gemacht werden. In Schacher/Grässler (2006: 1) werden die Vorteile des Business-Rule-Ansatzes in drei Punkten zusammengefasst:

1. Höhere Transparenz durch explizite Darstellung von fachlichen Begriffen, auf denen die Geschäftsregeln aufbauen, und den Geschäftsregeln selbst,
2. höhere Flexibilität durch Geschäftsregeln, da explizit dargestellte Regeln leichter geändert werden können,
3. höhere Effizienz durch automatisierte Verarbeitung der Regeln.

Die Gartner-Group[1] sieht die Vorteile des Einsatzes von Geschäftsregeln in mehreren Punkten, siehe Abbildung 1.

Abbildung 1: Wirtschaftlicher Vorteil des Einsatzes von Geschäftsregeln



Quelle: Gartner Research
(Sinur, J.: Rules: Adding Intelligence to the Enterprise Architecture, in: http://www.gartner.com/reprints/fairisaac/108391.html, Zugriff 26.01.2007).

Die Transparenz ist ein entscheidender Vorteil, denn die explizite Formulierung von Regeln zwingt zum Durchdenken der Prozesse und hilft dabei, überflüssige Festlegungen zu eliminieren. Die Transparenz bezieht sich auch auf

[1] Sinur, J.: Rules: Adding Intelligence to the Enterprise Architecture, in: http://www.gartner.com/reprints/fairisaac/108391.html, Zugriff 26.01.2007.

die IT-Systeme: Unter Einsatz von Business-Rule-Engines funktionieren die Systeme so, wie die Geschäftsregeln es beschreiben, da genau diese Geschäftsregeln im System verarbeitet werden.

Transparenz selbst erhöht die Flexibilität: Änderungen und deren Auswirkungen können leichter nachvollzogen werden. Zudem sind Geschäftsregeln wesentlich leichter modifizierbar als fest implementierte Regelungen. Da neue Regelungen nicht langwierig eingeführt und umgesetzt werden, sondern bei einer automatisierten Vorgehensweise sofort durch die Rule-Engine berücksichtigt werden, ist eine hohe Effizienz des Gesamtsystems zu erwarten.

2.2. Konzept

Geschäftsregeln sind in jedem Unternehmen vorhanden. Die vorhandenen Regelungen werden explizit gemacht und einer automatisierten Verarbeitung zugeführt. Der Business-Rule-Ansatz geht davon aus, dass Festlegungen in Form von Wenn-Dann-Regeln beschrieben werden:

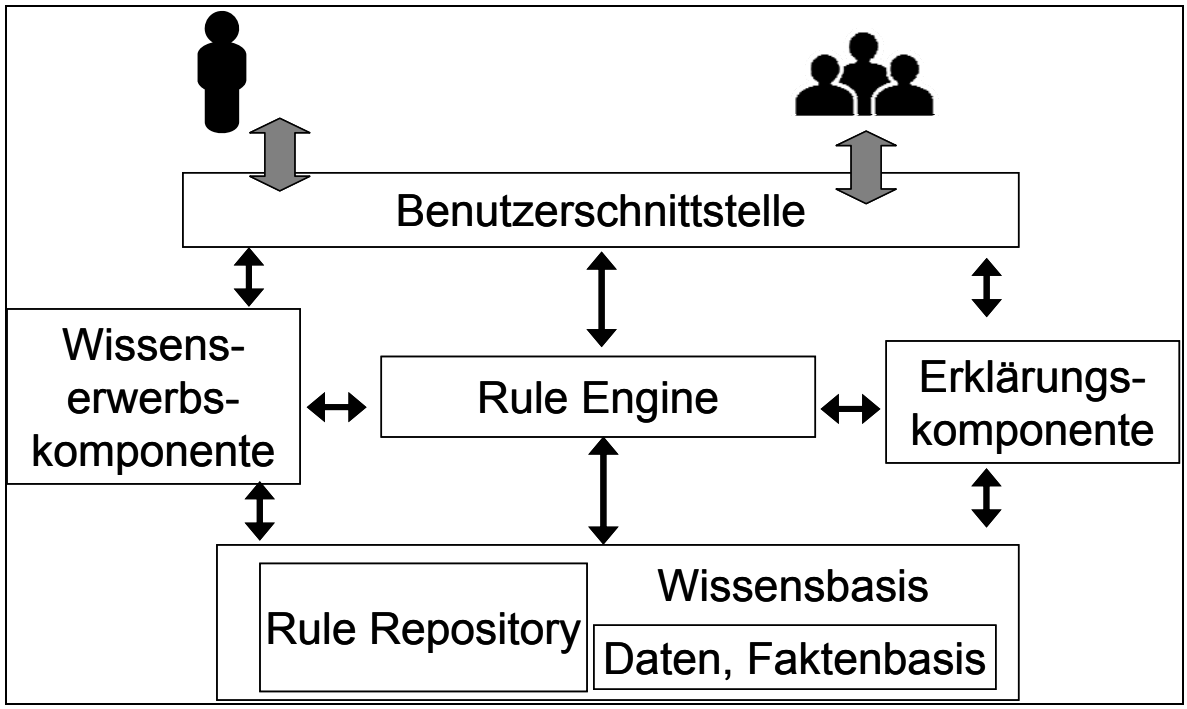
WENN Kunde	DANN auf Rechnung versenden.
WENN noch kein Kunde	DANN mit Nachnahme versenden.
WENN guter Kunde	DANN Rabatt gewähren.

Bereits dieses einfache Beispiel wirft einige Fragen zu den verwendeten Begriffen auf: Wer ist ein Kunde? Wer ist ein guter Kunde? Damit wird deutlich, dass es nicht ausreicht, Regeln zu formulieren, sondern es erforderlich ist, die Grundbegriffe, die Fachbegriffe, ganz allgemein das Vokabular, zu definieren. Im folgenden Beispiel wird man den Begriff Kunde als Grundbegriff erläutern, die Eigenschaft guter Kunde kann eventuell durch eine eigene Geschäftsregel festgelegt werden:

WENN Umsatz im Jahr > guter Umsatz DANN guter Kunde.

Es ist somit eine Menge von Grundbegriffen zu erarbeiten, auf denen die Regeln aufgebaut werden können. Grundbegriffe, insbesondere solche, die mit Daten verknüpft sind, werden üblicherweise bereits in den Firmendatenbanken verwaltet. Ein Business-Rules-Management-System ermöglicht es, die Business Rules zu verwalten. Dazu gehören Komponenten zum Zugriff auf vorhandene Datenbanken, eine Benutzerschnittstelle sowohl für den Wissensingenieur als auch den Anwender, eine Komponente zur Integration neuer Regelen (neues Wissen) sowie eine Erklärungskomponente. Die Verarbeitungskomponente - die Rule Engine - verknüpft die Regeln und kann somit Prozesse steuern oder Entscheidungen vorbereiten. Insofern ist die Struktur eines Business-Rules-Management-Systems ähnlich der Struktur eines Expertensystems, siehe zum Beispiel Lämmel/Cleve (2004).

Abbildung 2: Struktur eines wissensbasierten Systems auf der Basis von Regeln



Quelle: Uwe Lämmel.

Der Schwerpunkt des Business-Rule-Ansatzes hat sich gegenüber den Expertensystementwicklung der 1980er Jahre deutlich verlagert. Waren es damals Beratungs- oder Diagnosesysteme für spezielle Gebiete, so greift der Business-Rule-Ansatz in das gesamte Unternehmensgeschehen ein. Im Business-Rule-Manifest[2] werden wichtige Eigenschaften und Forderungen zusammengestellt, von denen an dieser Stelle einige angesprochen werden:

1. Die Anforderungen (des Unternehmens) stehen an erster Stelle. Diese werden essentiell durch Regeln ausgedrückt. Regeln sind unverzichtbarer Teil für das Geschäfts- und Technologie-Modell.
2. Geschäftsregeln sind explizit formuliert und nicht implizit in anderen Prozessbeschreibungen enthalten. Es gibt ein einheitliches, zusammenhängendes Regelwerk für die Unternehmensaktivitäten und es wird konsistent angewandt.
3. Wissen wird bewusst und explizit formuliert. Das Wissen wird in Form von Regeln dargestellt, die auf Fakten basieren.
4. Durch die Formulierung des Wissens steht das WAS (deklarativ) im Mittelpunkt und nicht das WIE (prozedural)
5. Regelgesteuerte Programmierung erlaubt eine neue Flexibilität der IT-Systeme, ausnahmengesteuerte Programmierung wird verdrängt.

[2] Siehe http://www.businessrulesgroup.org.

6. Es wird die Geschäftslogik verwaltet, anstatt viel Kraft auf das Management von Software- und Hardwareplattformen zu vergeuden.

Diese und weitere Forderungen machen deutlich, dass die explizite Formulierung des Wissens eines Unternehmens in Form von Regeln, die auf die Geschäftsprozesse des Unternehmens ausgerichtet sind, der zentrale Ansatz ist. Für die Durchsetzung stehen hierzu Business-Rules-Management-Systeme zur Verfügung.

Beispiel[3]

Die Einsatzmöglichkeiten für Geschäftsregeln in der Wirtschaft sind vielfältig. Zum einen bietet es sich an, bei der Validierung von Daten auf Business Rules zurückzugreifen. So ist es denkbar, dass Angaben, die ein Kunde in ein Formular eingegeben hat, von einer Rule Engine unmittelbar überprüft und eingeordnet werden. Dies kann zum Beispiel bei der Eingabe einer Telefonnummer geschehen. Anhand des Aufbaus der Nummer lässt sich eine Unterscheidung in Festnetz- und Mobilfunkanschluss treffen. Ist durch eine entsprechende Regel eine Handynummer identifiziert

WENN Nummer_beginnt_mit_0160 DANN Mobilfunkanschluss,

kann diese Information für etwaige Marketingaktivitäten im Unternehmen genutzt werden. Ein weiterer Anwendungsfall für Geschäftsregeln kann in der Beschreibung von ganzen Workflows im Unternehmen liegen. So ist es denkbar, dass der gesamte betriebliche Arbeitsfluss mit seinen Arbeitsanweisungen durch Business Rules beschrieben wird. Eine solche Regel kann beispielsweise wie folgt lauten:

WENN Dokument_unterschrieben_von_A DANN hole_Unterschrift_ein_von_B.

Ähnlich ist es bei Entscheidungsunterstützenden Systemen. Hier kann der Nutzer anhand vorher definierter Regeln hinsichtlich seiner zu treffenden Entscheidung durch entsprechende Informationen unterstützt werden. Ein Beispiel hierfür ist ein System zur Bewertung der Kreditwürdigkeit eines Kunden. Dabei können unterschiedliche Bedingungen für die Beurteilung angeführt werden:

WENN kein_Einkommen DANN nicht_kreditwuerdig

oder

WENN geringe_Schulden UND sicheres_Einkommen DANN kreditwuerdig

Ein weiterer Anwendungsfall für den Einsatz von Business Rules ist die Beschreibung von Kundenbeziehungen im Unternehmen. Dies soll im Folgenden an einem ausführlicheren Beispiel dargestellt werden.

---

[3] Abschnitt erstellt von Gert Mannheimer.

Ausgangspunkt unserer Überlegungen sei der Onlineshop einer Buchhandlung. Anforderung sei, dass mittels definierter Geschäftsregeln der Rechnungsbetrag einer Bestellung ermittelt werden soll. Gerade im Hinblick auf die Herausforderung, ständig auf Veränderungen am Markt reagieren zu müssen, ist es von hoher Bedeutung, flexibel in das Marktgeschehen eingreifen zu können. Dies kann durch individuelle Angebote und Rabatte geschehen, die den Kunden angeboten werden und in der Rechnung berücksichtigt werden müssen. Dabei ist es möglich, neue Angebote einfach einzupflegen und alte ebenso leicht zu entfernen. Dabei ist darauf zu achten, dass sich die unterschiedlichen Regeln nicht widersprechen. Gerade bei einer komplexen Geschäftslogik stellt dies eine große Herausforderung dar. Hier ein Blick auf die Geschäftsregeln unseres gewählten Beispiels:

1. WENN KundenDB.Bestellanzahl≤10 DANN Kunde=Neukunde
2. WENN KundenDB.Bestellanzahl>10 DANN Kunde=Stammkunde
3. WENN KundenDB.Bestellanzahl>100 DANN Kunde=Prämiumkunde
4. WENN Kunde=Stammkunde UND Lieferkosten>3,95€
   DANN Lieferkosten=3,95€
5. WENN Kunde=Prämiumkunde UND Lieferkosten>0,00€
   DANN Lieferkosten=0,00€
6. WENN Kunde=Neukunde UND Warenkorb≤20€ DANN Lieferkosten=5,95€
7. WENN Warenkorb>20€ UND Lieferkosten>3,95€ DANN Lieferkosten=3,95€
8. WENN Warenkorb>50€ UND Lieferkosten>0,00€ DANN Lieferkosten=0,00€
9. WENN Tag.heute=Freitag UND Lieferkosten>3,95€
   DANN Lieferkosten=3,95€
10. WENN Warenkorb>100€ DANN Kundenrabatt=+5%
11. WENN Tag.heute=Montag DANN Kundenrabatt=+5%
12. WENN Tag.heute=Samstag
    ODER Tag.heute=Sonntag UND Warengruppe_in_Warenkorb=CD
    DANN Rabatt=+5%
13. WENN KundenDB.Geburtstag=heute DANN Kundenabatt=+5%
14. WENN 24h-Lieferung=true DANN Zuschlag=+6,00€
15. WENN Zahlungsart=Nachnahme DANN Zuschlag=+2,50€
16. WENN offeneRechnung_in_KundenDB>2 DANN Bestellung_nicht_möglich
17. WENN KundenDB.Alter<18 UND Ware_in_Warenkorb.FSK18=true
    DANN Bestellung_nicht_möglich
18. WENN Bestellung DANN
    Kosten=Warenkorb*Kunderabatt/100+Lieferkosten+Zuschlag
    UND KundenDB.Bestellanzahl=+1

Die ersten drei Regeln kategorisieren einen Kunden anhand der Anzahl seiner in der Vergangenheit getätigten Bestellungen. Regel Vier bis Sechs legen auf Grundlage dieser Kategorisierung die zu zahlenden Lieferkosten fest.

Die folgenden drei Regeln befassen sich ebenfalls mit der Höhe der Lieferkosten. Je nach Größe des Warenkorbs bzw. des aktuellen Wochentags werden die Zustellgebühren verändert. Da sich, wie oben erwähnt, die einzelnen Regeln nicht widersprechen dürfen, sind als zusätzliche Bedingung die aktuellen Lieferkosten aufgenommen worden, so dass verhindert wird, dass bereits minimierte Kosten wieder erhöht werden.

Die nächsten Regeln definieren den gewährten Kundenrabatt, der aufsummiert wird. An ihnen ist aber auch die Flexibilität ersichtlich, die die Business Rules mit sich bringen. Auf einfachste Art und Weise lässt sich die Höhe des Rabattes genauso variieren wie die Bedingung, die erfüllt sein muss, damit dieser gewährt wird. So ist es beispielsweise denkbar, dass statt montags am Wochenende ein Preisnachlass angeboten wird. Ebenso lassen sich weitere Gründe definieren, die einen Rabatt rechtfertigen.

Zwei weitere Regeln, die bei vom Kunden gewünschten Zusatzleistungen zu befolgen sind, definieren einen dafür berechneten Zuschlag.

Ebenfalls denkbar sind Abbruchkriterien. Wie im Beispiel können das mehr als zwei offene Rechnungen in der Kundendatenbank sein. Auch hier können leicht weitere Regeln bestimmt werden.

Als letztes lässt sich durch eine Geschäftsregel auch die Berechnung des zu zahlenden Endbetrages definieren. Hier setzt er sich aus der Summe des Warenkorbwertes abzüglich der gewährten Rabatte sowie der Lieferkosten und berechneter Zuschläge zusammen. Außerdem wird mit Abschluss der Bestellung auch die Bestellanzahl des Kunden in der Datenbank erhöht.

Die gezeigten Beispiele können stets nur einen kleinen Einblick geben. Die Vorteile einer Beschreibung der Geschäftslogik mittels entsprechender Geschäftsregeln zeigen sich erst an komplexeren Problemstellungen, die nicht mehr mit einem Blick erfasst werden können. Weitere Einsatzmöglichkeiten von Geschäftsregeln finden sich im Bereich von Diagnose-, Beratungs- oder Planungssystemen sowie im Workflow-Management.

2.3. Standard

Dieser Abschnitt wendet sich einem mehr technischen Aspekt zu und kann übergangen werden, ohne dass das Verständnis der nachfolgenden Abschnitte und Kapitel leidet. Der Begriff „Standard“ wird in der Informationstechnik leider sehr inflationär und selten seiner Bedeutung gemäß verwendet. Ähnlich zum Beispiel der Sprache SQL für relationale Datenbanken, ist eine einheitliche Darstellung von Geschäftsregeln, eine standardisierte Notationsform, wünschenswert. Im folgenden Abschnitt wird auf den aktuellen Stand der Standardisierungsbemühungen auf der Basis von XML eingegangen. RuleML

stellt einen Ansatz zur standardisierten Beschreibung von Regeln dar. Zunächst wird die Einordnung von RuleML in das System der XML-basierten Beschreibungssprachen vorgenommen. Im Anschluss wird anhand von Beispielen beschrieben, wie die konkrete Implementierung von Regeln in RuleML erfolgen kann

2.3.1. Die XML-Sprachfamilie und RuleML

XML (Extensible Markup Language) ist eine Beschreibungssprache, die geeignet ist, beliebige Daten strukturiert zu beschreiben. XML wurde entworfen, um HTML zu ersetzen. XML basiert genauso wie HTML auf SGML und kann damit als funktionale Untergruppe (ein Profil) von SGML bezeichnet werden.

XML definiert Dokumentklassen durch DTDs (Document Type Definition) und XML-Schemata. Damit stehen bei XML nicht das Dokument, sondern die Daten im Mittelpunkt. XML eignet sich daher für B2B-Anwendungen, die Datenaustausch gegenüber dem Dokumentaustausch bevorzugen.

RDF (Resource Description Framework) wurde entworfen, um META-Daten in WWW-Anwendungen zu verwenden. Als Urheber gelten Tim Berners-Lee (W3C-NOTE 29-October-1997 on DataFormats), IBM und Microsoft (Juli 1998, W3C) mit der DCD (Document Content Description) für XML. DCD ist hierbei dazu gedacht, ein Vokabular zu schaffen und Randbedingungen und Beschränkungen in XML für diesen Zweck festzulegen.

Beispiele hierfür sind XML-basierte Dokumente der beiden Softwarehersteller, Webseiten, Suchmaschinendatensammlungen, digitale Bibliotheken und so weiter. RDF bietet Mittel zur Veröffentlichung von sowohl durch Menschen, als auch von Maschinen lesbaren Definitionen von Attributlisten, so genannten *Property-Sets*.

RDF kommt beispielsweise in RSS1[4] zur Anwendung. Weiterführende Informationen finden sich auch zu den XSL-Transformationen XSLT.[5]

MathML ist eine auf XML basierende Beschreibungssprache, mit der es möglich ist, mathematische Notation und mathematischen Inhalt zu beschreiben. Etwa 180 vordefinierte Tags ermöglichen Notationsbeschreibungen sowie die Beschreibung des abhängigen Inhalts nebst seiner Bedeutung. Die aktuelle Version ist 3.0. Näheres dazu kann unter www.w3.org/Math/ nachgelesen werden.

RuleML ist seinerseits eine Basis des WSDF (Web Service Description Frameworks). Die aktuelle Version von RuleML ist 0.91. Entwickler ist die Rule Markup Initiative[6]. RuleML bietet Elemente zur Implementierung von

---

[4] Siehe web.resource.org/rss/1.0/ (letzter Zugriff am 19.01.2007).

[5] Siehe www.w3.org/TR/xslt (letzter Zugriff am 19.01.2007) bzw. dret.net/glossary/xslt oder www.w3.org/2005/08/online_xslt/.

[6] Siehe hierzu http://www.ruleml.org (letzter Zugriff am 19.01.2007).

verschiedenen Logik- und Regelsystemen. Somit ist RuleML als Transfermedium zwischen Rule-Engines und anderen Systemen geeignet. Im Folgenden werden Beispiele für Regeln aus der logischen Programmiersprache Prolog gezeigt.

2.3.2. Syntax-Beispiele

Bei der Beschreibung von Fakten und Regeln ist es möglich, sich weitgehend an Prolog zu orientieren. Um einen Ausdruck in RDF-artigen Rollentags darzustellen, werden innerhalb des Hauptelements Unterelemente definiert.

```
<person>
 <_vorname><String>Max</String></_vorname>
 <_nachname><String>Mustermann</String></_nachname>
</person>
```

Mit RuleML lassen sich auf einfache Weise Fakten definieren. Hierbei kann man sich an der in Prolog gebräuchlichen Syntax orientieren: Für „*Marcel ist Student*“ notiert man in Prolog *student(Marcel).*

```
<cterm>
  <_opc><ctor>Student</ctor></opc>
  <ind>Marcel</ind>
</cterm>
```

cterm ist hier das Rollenelement mit dem Subelement _opc für den Konstruktor für das Argument, in diesem Fall eines Namens. Marcel ist somit Argument für Student. Das folgende Beispiel beschreibt die Relation: *Ein Student studiert an einer Hochschule*. In Prolog notiert man: *studiert(Marcel, Hochschule).*

```
<atom>
  <_opr><rel>studiert</rel></opr>
  <ind>Student</ind>
  <ind>Hochschule</ind>
</atom>
```

Solche Strukturen können nun ihrerseits in andere Strukturen eingefügt werden. Letztendlich ist diese Schreibweise eine Baumnotation.

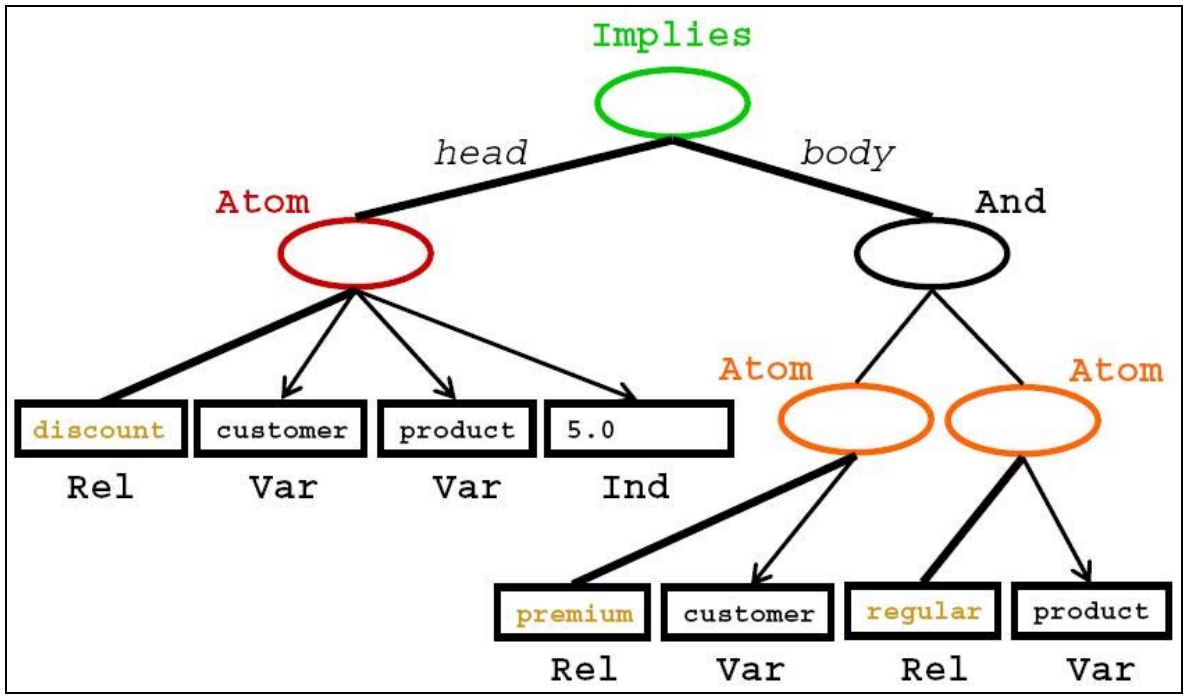
```
<fact>
  <_head>
   <atom>
   <_opr><rel>studiert</rel></opr>
   <ind>Student</ind>
   <ind>Hochschule</ind>
   </atom>
   <atom>
```

```
      ...
      </atom>
    </_head>
  </fact>
```

Ohne weiteres lassen sich Wenn-Dann-Regeln (Implikationen) mit RuleML implementieren. Das Beispiel[7] der Notation einer Regel zeigt die Baumstruktur der Darstellung. Es wird die folgende Geschäftsregel in RuleML abgebildet, siehe Abbildung 3:

"*The discount for a customer buying a product is 5 percent if the customer is premium and the product is regular.*"

Abbildung 3: Baumdarstellung der RuleML-Notation einer Geschäftsregel



Quelle: Boley, Harold in 2006.ruleml.org/slides/RuleML-Family-PPSWR06-talk-up.pdf.

Die RuleML-Notation für die Regel hat die folgende Gestalt:

[7] Siehe Harold Boley in 2006.ruleml.org/slides/RuleML-Family-PPSWR06-talk-up.pdf, (letzter Zugriff 19.01.2007).

```
<Implies>
  <_head>
    <Atom>
      <Rel>discount</Rel>
      <Var>customer</Var>
      <Var>product</Var>
      <Ind>5.0</Ind>
    </Atom>
  </_head>
  <_body>
    <And>
      <Atom>
        <Rel>premium</Rel>
        <Var>customer</Var>
      </Atom>
      <Atom>
        <Rel>regular</Rel>
        <Var>product</Var>
      </Atom>
    </And>
  </_body>
</Implies>
```

Im Folgenden wird beispielhaft eine Regelbasis für die oben aufgeführten Beispiele als XML-Schema definiert. *rulebase* wird als EBNF-artiges SGML-subset festgelegt:

```
<!ELEMENT rulebase ((imp | fact)*)>
```

Die Definition des Schemas in XML sieht nun wie folgt aus:

```
<xsd:schema xmlns:xsd=„http://www.w3.org/XML/Schema“>
  <xsd:element name=„rulebase“>
    <xsd:complexType>
      <xsd:choice minOccurs=„0“ maxOccurs=„unbounded“>
      <xsd:element name=„imp“ type=„impType“/>
       <xsd:element name=„fact“ type=„factType“/>
      </xsd:choice>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Wenn-Dann-Regeln werden als Konstrukte mit dem Schlüsselwort imp definiert:

```
<!ELEMENT imp (_head | _body)>
```

Die Struktur einer Regel wird durch folgende SGML-Notation definiert:

```
<xsd:schema xmlns:xsd=„http://www.w3.org/XML/Schema“>
  <xsd:complexType name=„impType“>
    <xsd:all>
      <xsd:element name=„_head“ type=„headType“
                                        minOccurs=„1“ maxOccurs=„1“ />
      <xsd:element name=„_body“ type=„bodyType“
                                        minOccurs=„1“ maxOccurs=„1“ />
    </xsd:all>
```

```
    </xsd:complexType>
  </xsd:schema>
```

Auf diese Weise fährt man fort, den vollständigen Namensraum zu implementieren, so wie dieser in den genannten Beispielen verwendet wurde:

```
<!ELEMENT _head (atom)>
<!ELEMENT _body (atom|and)>
<!ELEMENT and (atom*)>
<!ELEMENT atom ((_opr,(ind|var)*)|((ind|var)+,_opr))>
<!ELEMENT cterm ((_opc,(ind|var|cterm)*)|((ind|var|cterm)+, _opc))>
```

2.3.3. Einsatz von RuleML

RuleML wird von einigen Rule-Engines, wie zum Beispiel Mandarax, unterstützt. Besonders offene Anwendungen legen Wert auf transparente Datenformate. Der Einsatz von RuleML ist sinnvoll für die Modellierung von Regelsystemen, da sie einen effektiven und formalen Rahmen für die Implementierung bieten. Ferner ist die Verwendung von RuleML eine mögliche Basis für die Entwicklung von Import- und Export-Formaten zwischen verschiedenen Systemen und somit ein Ansatz für eine standardisierte Speicherung von Geschäftsregeln.

Es existieren mehrere Alternativen zum eher content-neutralen RuleML. Dies sind:

- *BRML (Business Rule Markup Language)* ist ein in XML beschriebenes Business-Rule-Engine-neutrales Daten-Austauschformat für Regeln, hervorgegangen aus den Common-Rules-Spezfikationen für E-Commerce der IBM (xml.coverpages.org/brml.html, Zugriff 09.01.2007).
- *DAML (Defense Advanced Research Projects Agency Agent Markup Language)* ist eine XML-Spezifikation, die sich mit der Einführung von Regelanweisungen als Tags in XML-Dokumenten beschäftigt, die sich dann von einer Business-Rule-Engine lesen lassen (www.daml.org/, Zugriff 07.01.2007).
- *FPML (Financial Products Markup Language)* wurde speziell für Finanzprodukte entwickelt. Auch diese basiert auf einer XML-Darstellung (www.fpml.org/, 09.01.2007).
- Dem Java-Umfeld entstammt das *JSR-94 (Java Specification Request 94)*, welches im Rahmen der Standardisierungsbemühungen für Java entwickelt wurde. Die dort verwendeten Klassen javax.rules und javax.rules.admin sollen Teil des Java Development Kit (JDK) werden und eine einheitliche Programmierschnittstelle für die Einbindung einer Business Rule Engine bieten (jcp.org/aboutJava/communityprocess/review/jsr094/, Zugriff 26.12.2006).

Das Industriekonsortium OMG (Object Management Group) bietet ebenfalls

Konzepte an (www.omg.org/, Zugriff 22.12.2006):

- *BSBR (Business Semantics of Business Rules)* stellt eine einheitliche Business-Semantik zur Formulierung von Regeln dar, die unabhängig von Implementierungen sein soll.
- *Production Rules*, welches seinerseits ähnlich angelegt ist wie RuleML

Es wird deutlich, dass es noch ein weiter Weg bis zu einem allgemein akzeptierten Standard ist. Um die Austauschbarkeit von einmal formuliertem Wissen zu gewährleisten ist somit darauf zu achten, dass ein eingesetztes Business-Rule-System Möglichkeiten zum Export dieses Wissens bietet.

3. Einsatzmöglichkeiten

Auf den Seiten des Business-Rule-Portals[8] finden sich einige Anwendungsfälle. Interessante Einsatzgebiete können auch den Web-Seiten der Anbieter von Business-Rules-Management-Systemen entnommen werden. Die Angaben zu den Kunden bzw. Projekten der Anbieter zeigen die Vielfalt der Anwendungsmöglichkeiten auf:

1. http://www.ilog.com/products/jrules/customers.cfm,
2. http://www.fairisaac.com/Fairisaac/Solutions/Enterprise+Decision+Management/Business+rules/,
3. http://www.visual-rules.de/063_customers/successstories.html,
4. http://www.yasutech.com/downloads/casestudies/index.htm.

Es lassen sich sowohl ganze Arbeitsabläufe (Workflows), die Geschäftslogik (Business Logic), die Kundenbeziehungen als auch relativ enge Anwendungen zur Datenüberprüfung mittels Business Rules beschreiben:

1. Telefongesellschaften setzen Geschäftsregeln zur Beschreibung ihrer Tarife ein.
2. Banken nutzen Geschäftsregeln für die Bewertung von Kunden unter Beachtung der BASEl-II-Vorgaben.
3. In Banken wird die Anlageberatung regelbasiert durchgeführt.
4. Steuerungsaufgaben werden durch Business Rules übernommen.

Insbesondere der erste Punkt steht im Zentrum der Geschäftsregeln zur Beschreibung der Geschäftslogik: Unternehmen mit einer hohen Anzahl an Kunden und vielen kundenspezifischen Tarif-Modellen können durch den Einsatz von Geschäftsregeln ihre Flexibilität erhöhen. Tarife können wesentlich flexibler und kundenspezifischer gestaltet sowie schneller in das vorhandene Modell integriert werden.

[8] http://www.brportal.org.

4. Business-Rule-Systeme im Vergleich

Als Grundlage des im nachfolgenden Abschnitt behandelten Vergleichs dient ein Kriterienkatalog. Bei der Zusammenstellung der Kriterien wurden zum einen Anforderungen berücksichtigt, wie sich diese generell aus der Anwendung, der Darstellung und der Verarbeitung von regelbasiertem Wissen ergeben. Andererseits sind ebenso Forderungen eingeflossen, die sich aus einem Praxiseinsatz ergeben. Im Abschnitt 4.1. werden die Kriterien vorgestellt und anschließend ein Bewertungsmaßstab nebst Interpretationen angegeben.

Abschnitt 4.2. stellt das Ergebnis der Untersuchungen vor. Es wird eine Übersicht über die in die Analyse einbezogenen Anbieter von Business-Rules-Systemen gegeben. Es schließt sich eine Tabelle an, in der die Systeme mit den Bewertungen gegenüber gestellt werden.

4.1. Kriterien

Es kann zwischen zwei Arten von Kriterien unterschieden werden: *anwendungsunabhängigen* und *anwendungsbezogenen* Kriterien.

Die *anwendungsunabhängigen* Kriterien umfassen Forderungen, die an jedes Software-Produkt gestellt werden können: Aufgabenangemessenheit, Korrektheit, Flexibilität, Verständlichkeit, Stabilität sowie Effizienz. Die *anwendungsbezogenen* Kriterien beschreiben die gewünschte Funktionalität einer Software, hier die Arbeit mit Geschäftsregeln.

Zu den *anwendungsunabhängigen* Kriterien zählen die Kosten für Lizenzen, relevante Eigenschaften der Laufzeitumgebung und Performance, der Ressourcenbedarf und die Usability. Je nach Business-Rule-System können sich das Angebot und somit auch die Kosten für Entwicklungslizenzen, Produktionslizenzen oder Lizenzen für eventuelle Zusatzprogramme unterscheiden. Angaben zu Kosten für zusätzliche Services und Supports werden zusammengefasst. Die Laufzeitumgebung unterteilt sich in Plattformunabhängigkeit, Language Bindings, Applikationsserver und Datenbank-Anbindungen. Die Gliederung betrachtet die Business-Rule-Systeme in Bezug auf ihre unterstützten Schnittstellen und Betriebssysteme, wie zum Beispiel Linux, Windows oder Mac, ihrem Umgang mit verschiedenen Programmiersprachen, den zu benutzenden Applikationsservern, welche die Business Rules ausführen, und die möglichen Datenbanken, in denen die relevante Datenbasis für die Business Rules hinterlegt wird. Die Performance betrachtet Informationen zur Messbarkeit, ausgedrückt durch Benchmarks, Geschwindigkeit, evaluiert durch die Benchmark-Ergebnisse und Skalierbarkeit. Der Ressourcenbedarf betrachtet die notwendigen Hardwarevoraussetzungen, wie zum Beispiel die Größe des Arbeitsspeichers, die notwendigen CPU-Eigenschaften und den Speicherbedarf einer Installation. Als abschließenden Punkt der anwendungsunabhängigen Kriterien werden Informationen zur Usability berücksichtigt.

Diese betreffen die Administration und den Betrieb während der Anwendung.

Die *anwendungsbezogenen* Kriterien betrachten verwendbare Algorithmen zur Regel-Verarbeitung, Anbindung bzw. direkte Einbindung in Programmiersprachen, Datenbanken, Entwicklungsunterstützung, Deployment von Business-Rules, Regeln und die Sicherheit. Bei den Algorithmen wird nach der Art der Verkettung (vorwärts, rückwärts) sowie der Verwendung spezieller Algorithmen (z. B. RETE) gefragt. Im Abschnitt der Entwicklungsunterstützung werden die Eigenschaften des Regel-Editors hinsichtlich seiner grafischen oder textorientierten Regelerstellung, die Funktionen zur Codeüberprüfung und Test auf unerreichbare Codeabschnitte, Plausibilitäts- und Konsistenzprüfung, sowie Validierung und Identifizierung von Fakten bewertet.

Nach dem Deployment von Business-Rules wird im Punkt „Regeln“ näher auf die sprachliche Unterstützung eingegangen. Ist die Verwendung einer natürlichen Sprache möglich? Welcher Sprachstandard kann vom Business-Rule-System benutzt werden? Welche Schnittstellen zum Import und Export von Regeln bietet das Produkt?

Im Punkt „Sicherheit“ wird die Fehlerbehandlung und die Zugriffssicherheit betrachtet. Dazu gehören Authentifizierung, Autorisierung und die Art der Verschlüsselung. Die Persistenz bezeichnet die dauerhafte Speicherung von Regeln auf Speichermedien. Informationen zur Nachvollziehbarkeit, Revisions- und Abwärtskompatibilität sowie zur Versionierung der Regeln werden in den letzten Teilabschnitten einbezogen und bewertet.

Der Vergleich benötigt, um einheitlich bewertet werden zu können, außer einem Kriterienkatalog auch eine Bewertungsstruktur, auf die zurückgegriffen wird. Mit der unterstehenden Tabelle ist eine solche gegeben.

Tabelle 1: Verwendete Interpretationen für den Bewertungsmaßstab

Wert	**Interpretation A** **vorhanden:**	**Interpretation B** **passt…**	**Interpretation C**
0	Nein	… nicht	Ungenügend
1	Teilweise	… teilweise	Genügend
2	Ja	… ja	Befriedigend
3	Ja, teilweise gut	… teilweise gut	Gut
4	Ja – überwiegend gut	… überwiegend gut	Sehr gut
5	Ja – immer gut	… perfekt	Ausgezeichnet

Quelle: Eigene Darstellung.

4.2. Business-Rule-Systeme: Anbieter und Vergleich

Bei der Recherche nach Anbietern von Business-Rule-Systemen sind die Autoren auf die Arbeit von Endl (2004: 266ff) aus dem Jahre 2004 aufmerksam

geworden, die eine Zusammenstellung sowohl kommerzieller als auch nicht-kommerzieller Systeme enthält. Diese Liste wurde aktualisiert und ergänzt. Im Ergebnis wurde sich auf die in Tabelle 2 Anbieter konzentriert.

Tabelle 2: Anbieter von Business-Rule-Systmen

Anbieter	Logo	Web-Seite
Blaze Advisor	FairIsaac.	www.fairisaac.com/Fairisaac/Solutions/
ILOG	ILOG Changing the rules of business	www.ilog.de
JBoss	JBoss a division of Red Hat	labs.jboss.com/portal/jbossrules
Mandarax	SOURCEFORGE.net Create Participate Evaluate	www.mandarax.org, mandarax.sourceforge.net
Oracle	ORACLE	www.oracle.com/appserver/rules.html
QuickRules	YASU TECHNOLOGIES	www.yasutech.com/
Versata	versata	www.versata.com/company/germany.htm
Visual Rules	visual rules® rule your business logic	www.visual-rules.de

Quelle: Eigene Darstellung.

Die Regel-Komponente von Versata wurde nicht mit in die Bewertung mit aufgenommen, da nicht ausreichend Informationen über die Rule-Engine ermittelt werden konnte. Die Liste erhebt keinen Anspruch auf Vollständigkeit. Eine Zusammenstellung nicht-kommerzieller Regel-Systeme, die unter Nutzung von Java entwickelt wurden, kann der Web-Seite www.manageability.org/blog/stuff/rule_engines/view (Zugriff 26.01.2007) entnommen werden. Im Anschluss an die Vorstellung der Regel-Systeme aus Tabelle 2 werden die auf dieser Web-Seite aufgeführten Systeme kurz charakterisiert (Abschnitt 5.9).

Tabelle 3: Bewertung der Business-Rule-Systeme

Kriterien / Software	Blaze Advisor	ILOG	JBOSS Rules	Mandarax	Oracle	QuickRules	Visual Rules
Anwendungsunabhängig							
Lizenzierung / Kosten	0	4	5	5	2	1	3
Laufzeitumgebung	5	5	5	5	4	5	5
Performance	5	5	4	3	4	4	5
Ressourcenbedarf	3	5	4	4	4	5	4
Usability	4	3	3	3	4	4	4
Kompatibilität/ Austauschbarkeit	0	4	2	3	2	2	0
Anwendungsbezogen							
Regel-Darstellung	4	4	3	3	4	3	5
Regel-Verarbeitung	4	4	4	4	4	4	4
Faktenbasis	5	5	5	4	5	5	3
Entwicklungsunterstützung	3	5	3	2	3	3	4
Deployment von Regeln	5	4	2	1	3	4	4
Sicherheit	5	5	2	1	3	4	4
	43	**53**	**42**	**38**	**42**	**44**	**45**

Quelle: Eigene Darstellung.

Die Bewertungen in der Tabelle können nur einen groben Anhaltspunkt geben, da sie mit einem subjektiven Faktor versehen sind. Trotz der Probleme bei der Installation des ILOG-Business-Rule-Systems schneidet dieses System insgesamt am besten ab. Mandarax und JBOSS schneiden erwartungsgemäß etwas schlechter ab, da beide Systeme keine vollständigen Business-Rules-Management-Systeme darstellen. Aus Sicht der Lehre an Hochschulen sind insbesondere die kostenfreien Systeme Mandarax oder JBOSS-Rules interessant.

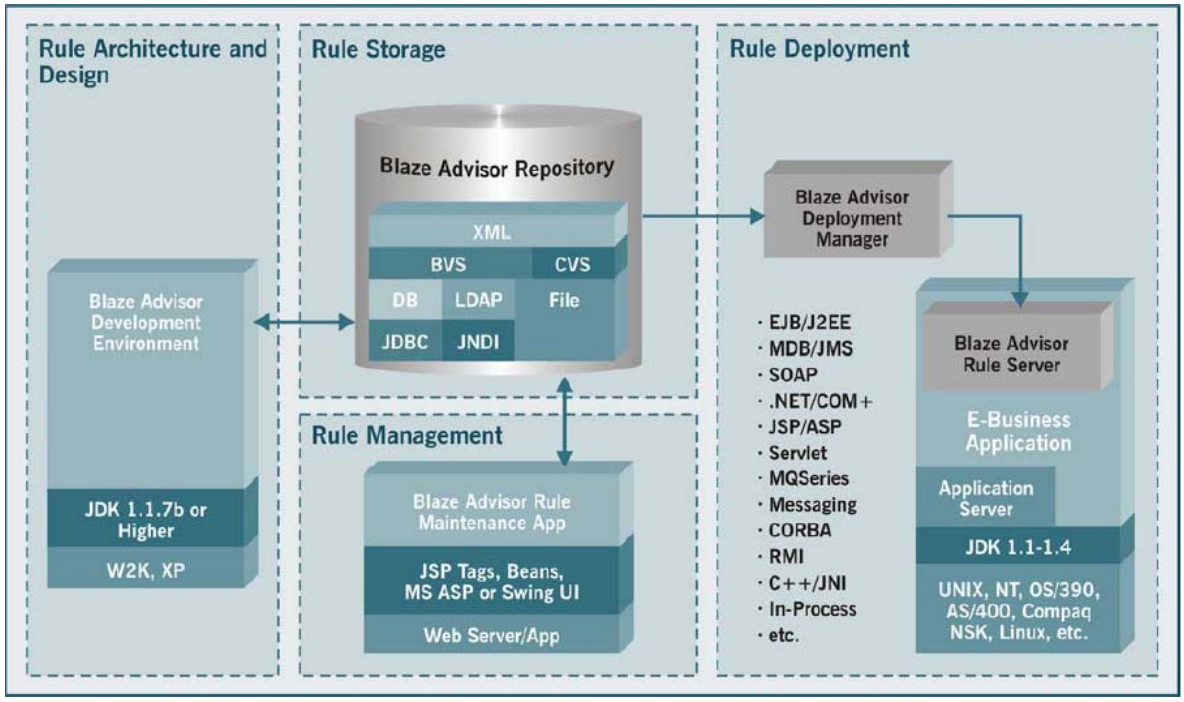
5. Business-Rule-Systeme

In diesem Kapitel werden die Systeme im Einzelnen vorgestellt. Nach einer kurzen Charakterisierung sowie einigen technischen Daten wird nach Möglichkeit auf ein Beispiel eingegangen. Daran werden die Möglichkeiten der Regel-Darstellung gezeigt. Am Ende steht jeweils die Tabelle der Bewertungen der Krieterien.

5.1. Blaze Advisor

Blaze Advisor wird von einigen großen Unternehmen verwendet und ist die erste Rules-Engine, die eine Abbildung von ein und derselben Regel in die Sprachen Java, .NET oder COBOL unterstützt. Blaze Advisor bietet Möglichkeiten zum Erstellen und Testen von Geschäftsregeln (Business Rules) sowie deren Verwendung und Integration in unternehmensspezifischen Anwendungen, insbesondere bei denen, wo Datenübertragungen und operative Prozesse im Mittelpunkt stehen. Regeln werden in einer Syntax, die dem Englischen ähnlich ist, beschrieben.

Abbildung 4: Komponenten des Blaze Advisor Rule Management Systems



Quelle: Fair Isaac Cooperation.

Blaze Advisor bietet eine Entwicklungsumgebung zur Erstellung von Regel-Modellen sowie deren Prüfung und Simulation, bevor es automatisch Integrationsdienste für den Einsatz in Anwendungen generiert. Blaze Advisor ist ein vollständiges Business-Rule-Management-System (BRMS) und eignet sich für

umfangreiche oder hochwertige Unternehmensanwendungen, insbesondere auch für Echtzeit-Anwendungen.

Die Architektur von Blaze Advisor setzt sich aus Dateneingabe, Umgebung zur Entwicklung von Modellen für Data Mining und Datenanalyse, Umgebung für die Entwicklung von Business Rules, einer Umgebung für Strategy Design unter Verwendung von Regeln und Modellen und einer Umgebung für das Entscheidungsmanagement zusammen. Aus systemspezifischer Sicht kommen diese Funktionalitäten in drei Modulen zum Einsatz, die Erstellung, Pflege und Verwaltung der Geschäftsregeln unterstützen: Builder, Rule Server und Repository.

5.1.1. Technische Daten

Hersteller:
Blaze Advisor ist ein Produkt von *Fair Isaac Corporation*[9]. Fair Isaac Corporation (NYSE:FIC) mit seinem Hauptgeschäftssitz in Minneapolis (Minnesota, USA) beschäftigt sich hauptsächlich mit Enterprise-Decision-Management, Scoring und Predictive-Analytics zur Unterstützung intelligenter Geschäftsentscheidungen. Das Unternehmen wurde 1956 gegründet und unterstützt mit seinen Produkten heute jährlich Hunderttausende von Entscheidungen in den Bereichen Finanzdienstleistungen, Einzelhandel, Versicherungen, Telekommunikation, Markenartikeln, Gesundheitswesen und im öffentlichen Dienst.

Kosten:
Die Preise für Blaze Advisor basieren auf folgenden Modulen:
- Entwicklungstools – pro Entwicklersitz.
- Rule Maintenance Applications – keine zusätzlichen Kosten.
- Produktionsserverlizenzen – je nach Anwendungsumfang.
- Test- und QA-Serverlizenzen – keine zusätzlichen Kosten.

Die Lizenzerteilung ist unbefristet, wobei mit Zusatzkosten in Höhe von etwa 18% für Wartung und vollen Support gerechnet werden muss. Die Kosten für eine typische Projektlizenz liegen im Bereich von US $ 250.000, beginnen bei US $ 50.000 und reichen bis zu US $ 1.000.000 und höher für Abteilungs- oder Unternehmenslizenzen.

Blaze Advisor für Java ist ein reines Java-Produkt und kann auf allen Plattformen ausgeführt werden, die eine Java Virtual Machine unterstützen. Die Entwicklungsumgebung wird von Windows 2000/XP unterstützt. Der Blaze Advisor Rule Server wurde erfolgreich auf folgenden Plattformen getestet:
- Windows NT 4.0 / 2000 / 2003 / XP
- Sun Solaris 2.6 bis 2.8
- IBM AIX 4.3 bis 5.1

---

[9] Siehe http://www.fairisaac.com/rules.

- IBM OS/390 Version 2.6 bis 2.8
- IBM z/OS 1.1 und 1.2
- IBM OS/400 V4R4 bis V5RI
- HP/Compaq NSK (Tandem)
- HP/Compaq Tru64 Unix
- HP-UX 11.0 und Linux (Intel oder IBM E-Series

Blaze Advisor unterstützt plattformübergreifende Einsatzkonzept: es bietet die Einsatzmöglichkeiten derselben Regeln in Java-, .NET- und COBOL-Umgebungen. Mehrere schnelle Deployer-Dienstprogramme erstellen für jede Umgebung einen Integrationscode, um die Integration mit unterstützten Plattformen zu beschleunigen.

Fair Isaac hat vor kurzem RulesPower für sein RETE-III-Execution Engine erworben und voll in Blaze Advisor integriert. Laut eigener Aussage ist die Leistung dieses neuen RETE-III-Algorithmus um 300% höher als die Leistung anderer Rule Engines. Je mehr Regeln, Bedingungen und Objekte im Entscheidungsprozess eingebunden sind, desto größer wird der Geschwindigkeitsvorteil.

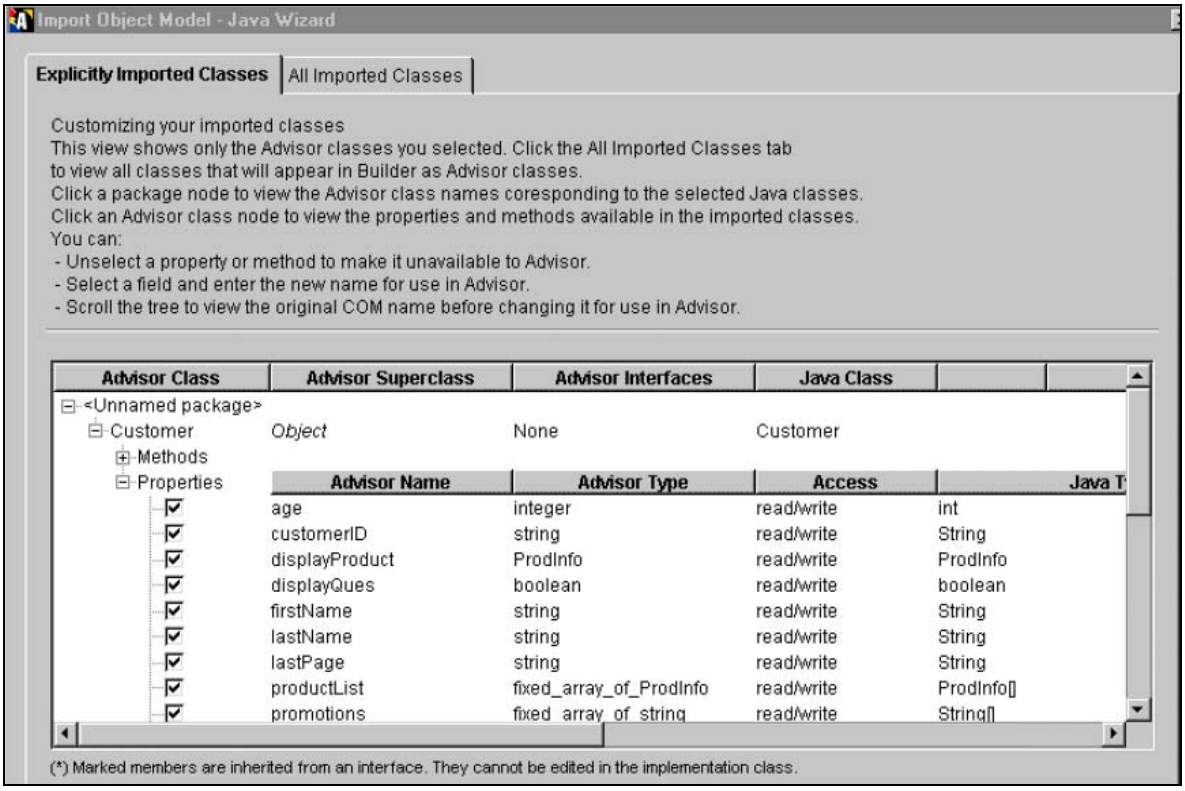
Eine automatische Skalierbarkeit für Tausende von Client-Sitzungen ist möglich. Die Skalierung erfolgt linear gemäß der zunehmenden Anzahl von CPUs. Laut Fair Isaac kann Blaze Advisor eine Regel-Datenbank mit mehr als 150.000 Regeln kompilieren und ausführen und wird in Umgebungen mit 128 CPUs eingesetzt.

5.1.2. Regel-Editor

Blaze Advisor verwendet eine eigene Regelsprache, die intuitive Lesbarkeit mit hoher Flexibilität verbindet und bietet eine Reihe von Tools für die Implementierung und das Debugging von Regeln.

Intern erstellt Blaze Advisor Referenzen zu Daten in einem Objektmodell analog zu Java. Zum Beispiel könnte ein Kundenobjekt mit Eigenschaften wie Kontonummer, Vorname, Nachname, Alter, Kontostand usw. definiert werden. Wie im objektorientierten Ansatz können Objekte auch die Eigenschaften von Elternteilobjekten erhalten (erben), so dass z.B. auf Grundlage eines Kundenobjekts ein VIP-Kunde definiert werden kann, der alle Eigenschaften des Kunden automatisch übernimmt, inklusive der Eigenschaften des persönlichen Vertreters, der Kreditlinie, usw. Da schon Datendefinitionen (auch bekannt als Objektmodelle oder Datenmodelle) im System vorhanden sind, macht Blaze Advisor es leicht, jene Definitionen zu durchsuchen und erstellt danach automatisch entsprechende Objekt-Referenzen. Blaze Advisor enthält einen Business Objekt Model Adapter (BOMA), der die externe Datenquellen ausliest, eine Objekt-/Eigenschaftsvertretung schafft und automatische Verbindungen zur externen Datenquelle aufbaut.

Abbildung 5: Object Model Import Wizards von Blaze Advisor.



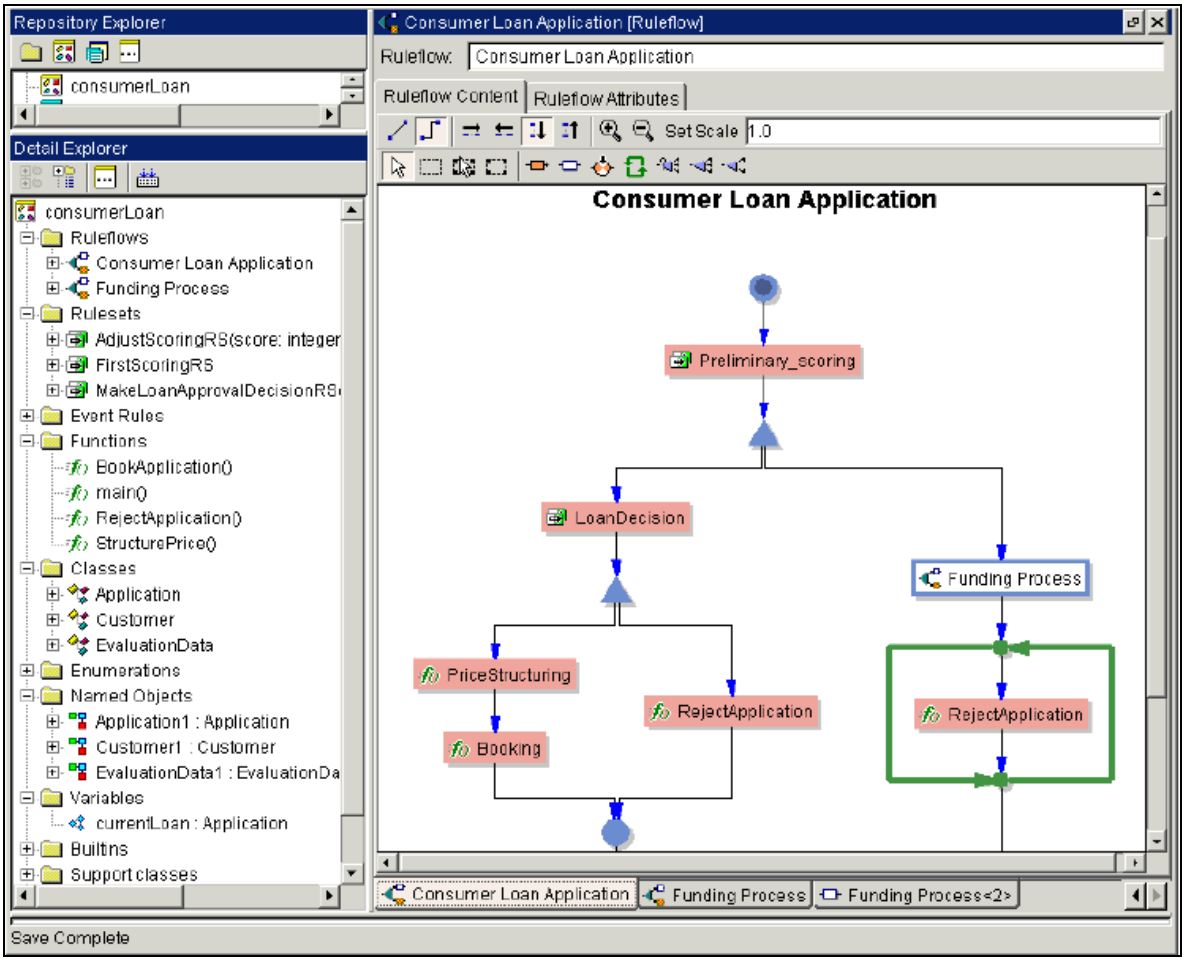
Quelle: „Fair Isaac Blaze Advisor: How it works“, April 2006.

Regelgruppen können sowohl in flexiblen Tabellen oder Baumstrukturen als auch im Code definiert, gepflegt und dargestellt werden.

Blaze Advisor Customised Rules Maintenance Application ist eine Anwendung zur Regelpflege auf Geschäftsebene, mit der befehlsgesteuerte End-to-End-Internet-Anwendungen erstellt werden können, die Benutzern Zugriff auf Regeln der Bereiche Sicherheit, benutzerdefinierte Anzeigen, Versionsverwaltung und Versionsführung, Historie und Auditierung bieten.

Ruleflows in Blaze Advisor definieren in graphischer Darstellung die Reihenfolge der Durchführungsschritte in einem Entscheidungsprozess unter Einsatz von Schleifen (loops), Verzweigungen (branches) und Aufgaben (tasks). Jede Aufgabe im *Ruleflow* wird durch eine Regelmenge (ruleset), eine Entscheidungstabelle, eine Funktion oder einen untergeordneten Ruleflow definiert. Ruleflows sind ebenso wie die anderen hier erwähnten Komponenten mehrfach verwendbar. Die Ruleflows von Blaze Advisor basieren auf XML können in einem graphischen Regeleditor verwaltet werden.

Abbildung 6: Ruleflow von Blaze Advizor

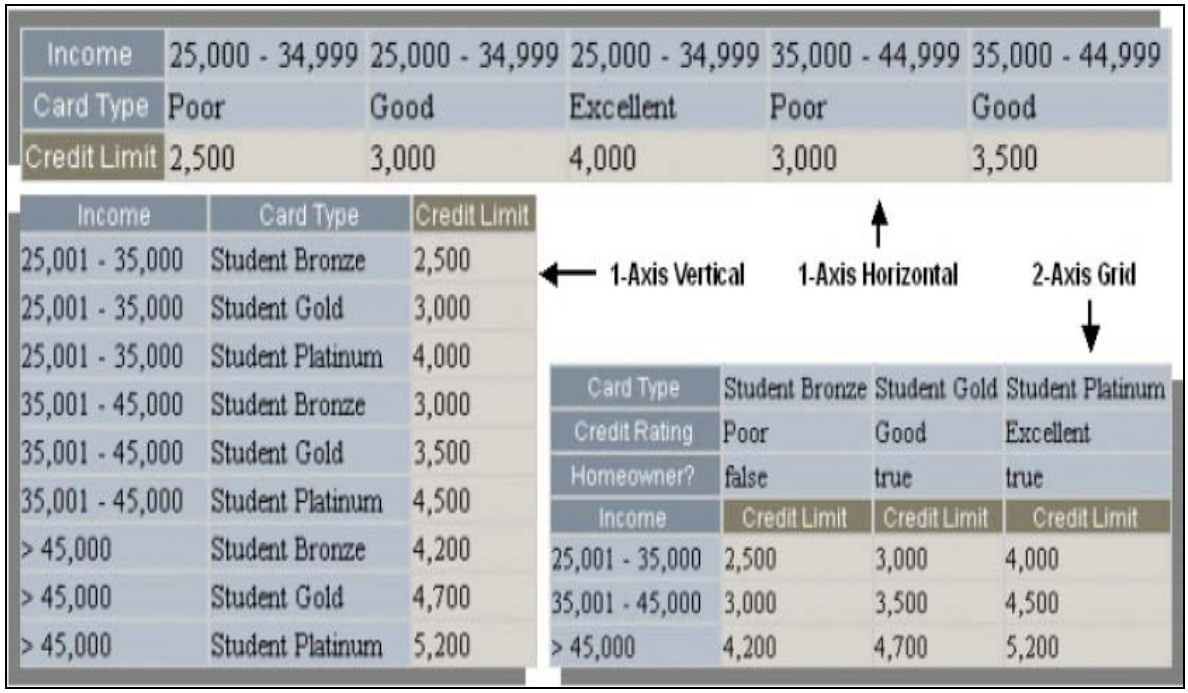


Quelle: „Fair Isaac Blaze Advisor: How it works“, April 2006.

Rulesets sind Regelgruppen, die für eine höhere Klarheit der Beschreibung sorgen. *Entscheidungstabellen* werden zur Beschreibung des Bedingungsteils einer Regel benutzt. Entscheidungstabellen (siehe Abbildung 7) können auch aus externen Quellen importiert werden.

Entscheidungsbäume (siehe Abbildung 8) stellen Ketten von Abhängigkeiten, die zu einer Aktion führen, grafisch dar. Bäume können aus dem Model Builder importiert werden. Wie Entscheidungstabellen können Aktionsknoten im Baum einen einzelnen Rückholwert oder ein Regelsprach-Konstrukt enthalten.

Abbildung 7: Entscheidungtabellen von Blaze Advisor



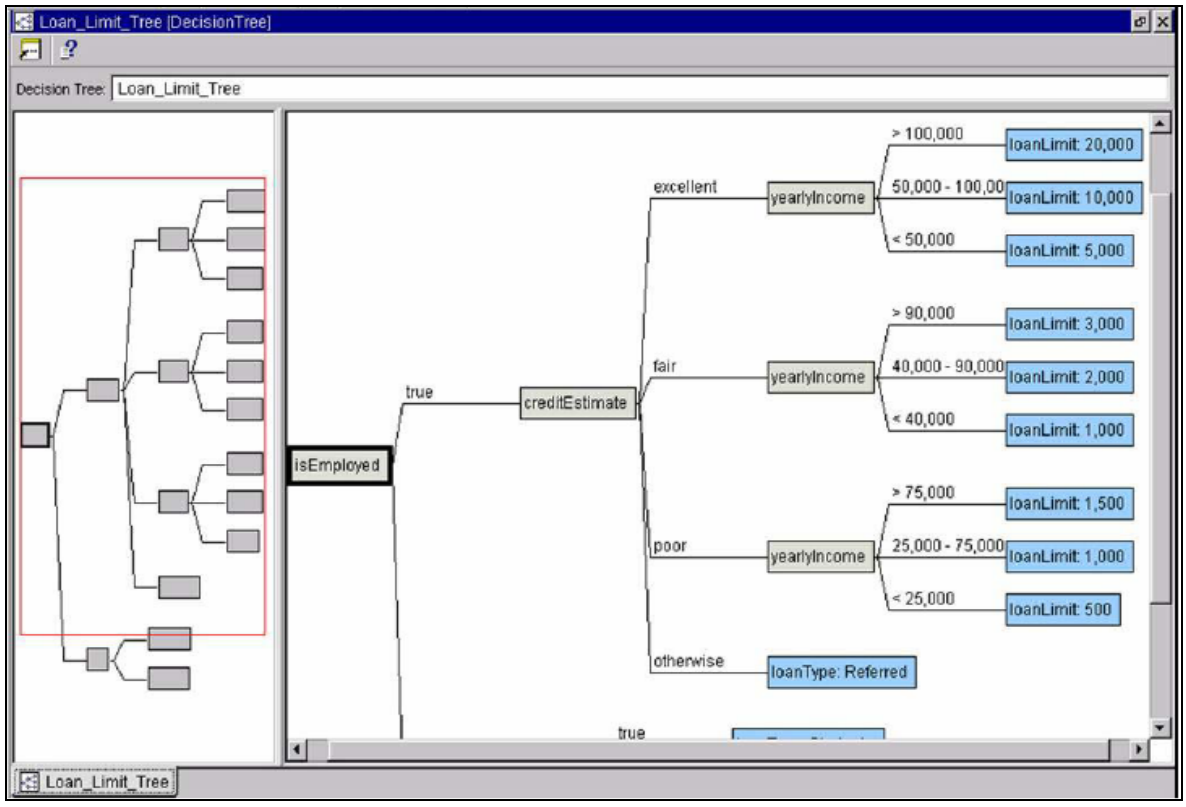
Income	25,000 - 34,999	25,000 - 34,999	25,000 - 34,999	35,000 - 44,999	35,000 - 44,999
Card Type	Poor	Good	Excellent	Poor	Good
Credit Limit	2,500	3,000	4,000	3,000	3,500

Income	Card Type	Credit Limit
25,001 - 35,000	Student Bronze	2,500
25,001 - 35,000	Student Gold	3,000
25,001 - 35,000	Student Platinum	4,000
35,001 - 45,000	Student Bronze	3,000
35,001 - 45,000	Student Gold	3,500
35,001 - 45,000	Student Platinum	4,500
> 45,000	Student Bronze	4,200
> 45,000	Student Gold	4,700
> 45,000	Student Platinum	5,200

Card Type	Student Bronze	Student Gold	Student Platinum
Credit Rating	Poor	Good	Excellent
Homeowner?	false	true	true
Income	Credit Limit	Credit Limit	Credit Limit
25,001 - 35,000	2,500	3,000	4,000
35,001 - 45,000	3,000	3,500	4,500
> 45,000	4,200	4,700	5,200

Quelle: „Fair Isaac Blaze Advisor: How it works“, April 2006.

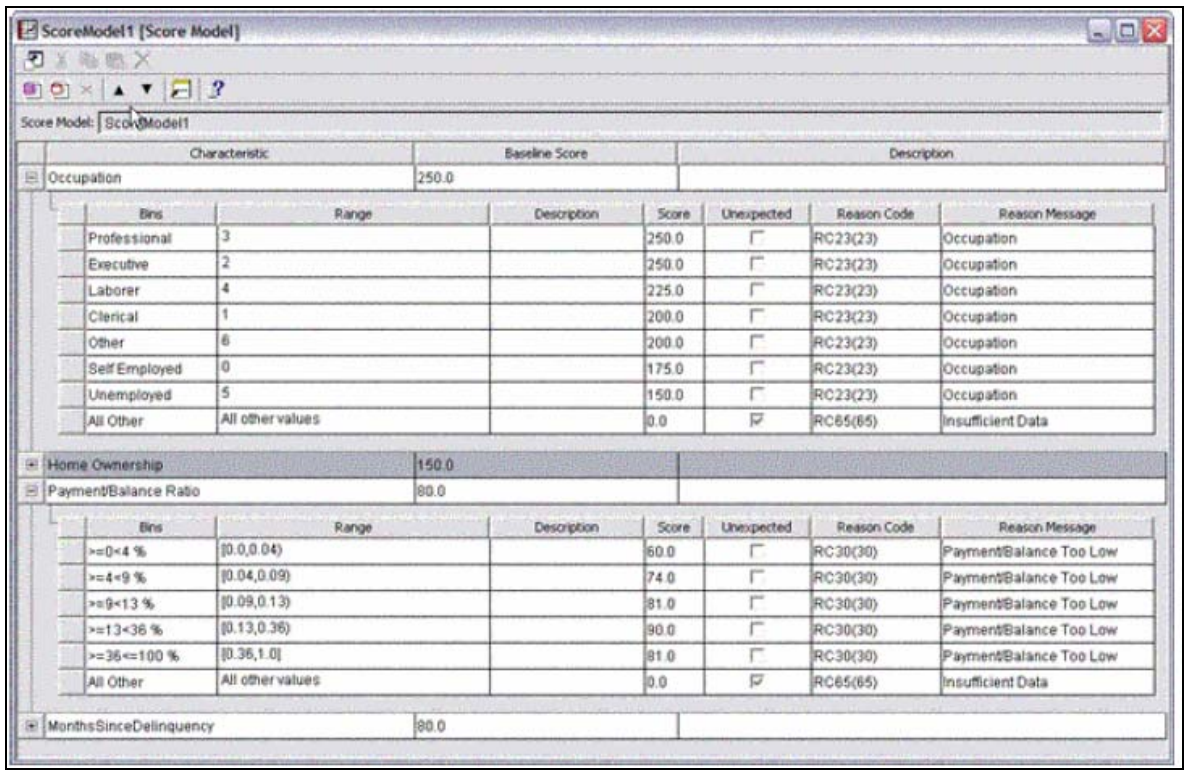
Abbildung 8: Entscheidungsbaum von Blaze Advisor



Quelle: „Fair Isaac Blaze Advisor: How it works“, April 2006.

Der grafische Baum-Editor erlaubt einen interaktiven Fokus auf jeden Knoten des Baums sowie eine dynamische Aktualisierung von Bedingungen und Aktionen. Scorecards (siehe Abbildung 9) können zur Bewertung von Ereignissen oder Eigenschaften herangezogen werden.

Abbildung 9: Scorecard von Blaze Advisor



Quelle: „Fair Isaac Blaze Advisor: How it works“, April 2006.

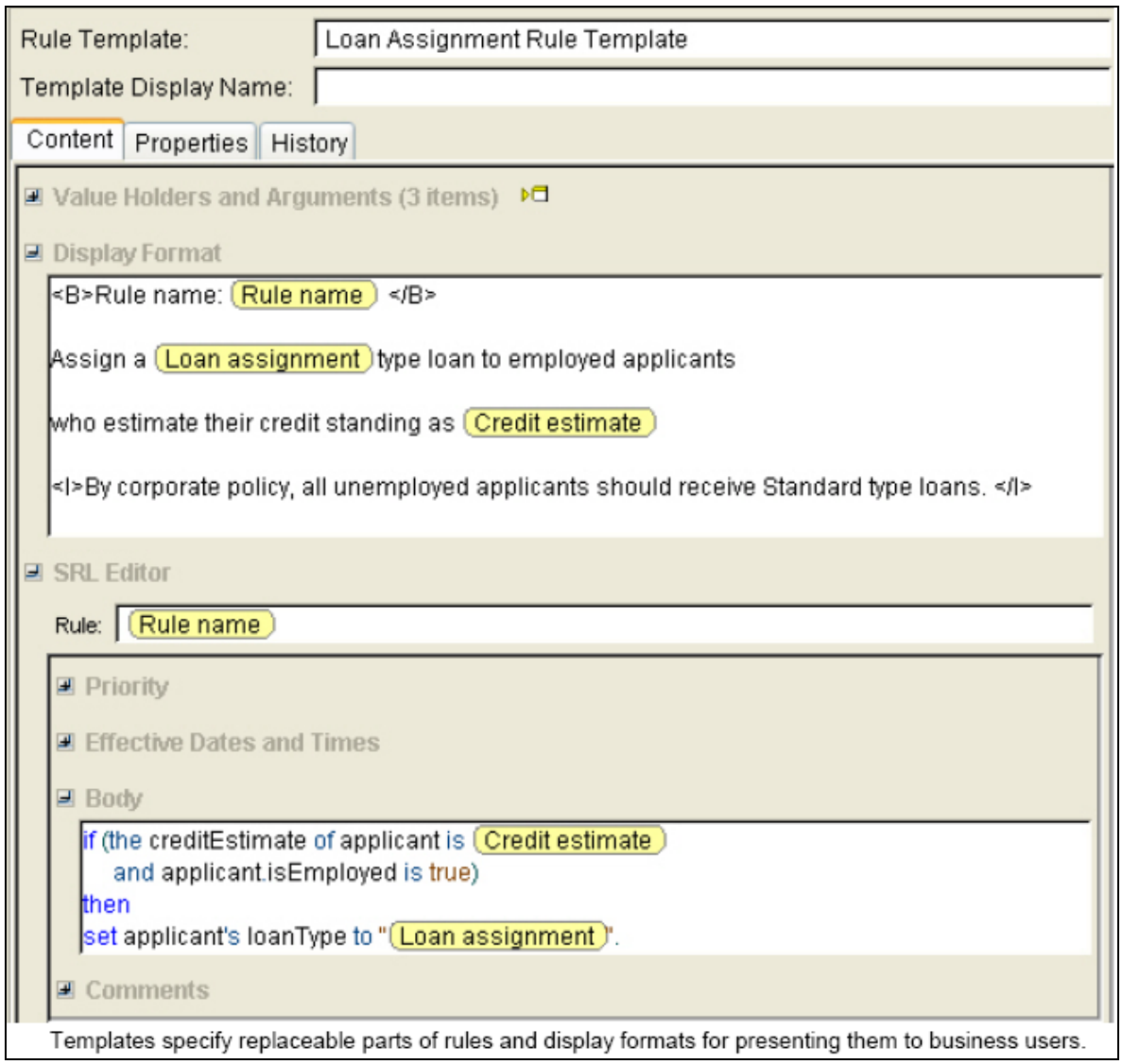
Die Rules-Liste wird typischerweise in Form von strukturierten Anweisungen in natürlicher Sprache aufgestellt, die die zu implementierenden Regeln, Fakten, Richtlinien und Verfahren ausdrücken. Der Geschäftsexperte arbeitet anschließend mit dem Objektmodell-Architekt an der Identifizierung der Objekte (Geschäftsdaten). Die Regeln werden anschließend mit Blaze Advisor bzw. RMA (Rule Maintenance Application) geschrieben.

Blaze-Advisor-Customised-Rules-Maintenance-Application ist eine Anwendung zur Pflege der Regeln auf Geschäftsebene.

Die Einbindung von bestehenden Datenbanken ist möglich, die Bewertungstabelle zeigt die unterstützten Standards. Assistenten lesen externe Datenstrukturen und erstellen Datenzugriffs- sowie -exportmethoden für jedes Format. Die Formate können mit dem Business-Object-Model-Adapter-Kit erweitert werden, um auf Strukturen in anderen firmeneigenen Systemen oder Datenbeständen unter Verwendung von Begrenzungszeichen- oder Feldgrößendefinitionen zuzugreifen und diese zu importieren. Außerdem unterstützt Blaze Advisor für COBOL Rule-Projekte, die auf einem COBOL-Daten-

modell in COBOL Copybooks basieren.

Abbildung 10: Rule-Template von Blaze Advisor



Quelle: „Fair Isaac Blaze Advisor: How it works“, April 2006.

5.1.3. Bewertung

Anwendungsunabhängige Kriterien:

Lizenzierung / Kosten	**0**
Entwicklungslizenz	
Produktionslizenz	Von 50.000 USD bis 1.000.000 USD, Im Normalfall ca. 250.000 USD
Zusatzprogramme	Hohe Kosten für z. B. Datenbanklizenz
Service & Support	ca. 18% für Wartung und vollen Support
Laufzeitumgebung	**5**
Unterstützte OS	Windows NT 4.0 / 2000 / 2003 / XP; Sun Solaris 2.6 bis 2.8; IBM AIX 4.3 bis 5.1; IBM OS/390 Version 2.6 bis 2.8; IBM z/OS 1.1 und 1.2; IBM OS/400 V4R4 bis V5RI; HP/Compaq NSK (Tandem); HP/Compaq Tru64 Unix; HP-UX 11.0 und Linux (Intel oder IBM E-Series
Schnittstellen	Zu Mono, .Net, Java2EE
Sprachanbindungen	C/C++, Perl, Java,
Applikationsserver	Web Services und SOA, Java 2 Enterprise Edition (J2EE) Plattformen, Microsoft .NET und COBOL für z/OS Mainframes
Datenbanken	JDBC kompatible Datenbanken, wie Oracle, MySQL, MSSQL, DB2, MS Access
Performance	**5**
Messbarkeit	Benchmarks vorhanden; eigene Benchmarks möglich
Geschwindigkeit	650000 Rule-Objekt- Bewertungen pro Sekunde (mit 500 komplexen Business Rules und mehr als 1 Mio. Klassifizierungsobjekten)
Skalierbarkeit	Skalierung erfolgt linear gemäß der zunehmenden Anzahl von CPUs)
Ressourcenbedarf	?
	Keine Angaben
Usability	**4**
Administration	Aufwand: relativ einfach. Kann von Endbenutzer selbst oder dessen Support-Mitarbeiter durchgeführt werden Installation; Werkzeuge: RuleAuthor um Regeln zu bearbeiten und JDeveloper um Java-Anwendungen zu erstellen Grafisch/textuell: Grafischer RuleAuthor; Regeln können auch in Textform erstellt werden
Betrieb / Anwendung	Einarbeitungszeit: schnell; der Applikationsserver benötigt etwas mehr Einarbeitungszeit Dokumentation/Hilfe: Dokumentation ist sehr umfangreich, es gibt viele Beispiele und auch ausführliche Dokumentationen und Viewlets
Kompatibilität/ Austauschbarkeit	**0**

Anwendungsbezogene Kriterien:

Regel-Darstellung	**4**
	Natürliche Sprache: English-like Structured Rule Language (SRL) Sprachstandard: SRL
Regel-Verarbeitung	**4**
	RETE III-Execution Engine
Verkettung	Vorwärtsverkettung
Regel-Auswahl	Keine Angaben
Suchverfahren	Keine Angaben
Faktenbasis	**5**
	Anbindung bzw. direkte Einbindung in PL/SQL/Oracle Database: ja
Entwicklungsunterstützung	**?**
Regel-Editor	Textorientiert
Code-Prüfung	Keine Angaben
Validierung	Ja
Identifizierung von Fakten	Ja
Deployment von Regeln	**5**
	Deployment von Regeln läuft automatisch, so dass alle Änderungen an Regeln bei laufendem Betrieb möglich
Sicherheit	**5**
Fehlerbehandlung	Ja
Persistenz	Ja
Nachvollziehbarkeit	Ja
Revisionssicherheit / Abwärtskompatibilität	Ja
Versionierung (der Regeln)	Ja
Zugriffssicherheit	Authentifizierung, Autorisierung und Verschlüsselung: ja

5.2. ILOG

ILOG JRules ist ein vollständiges Business-Rule-Management-System (BRMS) für die Java-Plattform. Die erweiterten Funktionen von ILOG-JRules ermöglichen das Generieren, Bereitstellen und Verwalten von Geschäftsregeln. Die Regelsprache ist Englisch , bzw. eine Sprache mit einer an das Englische angelehnten Syntax. Mit JBoss wird eine Rule Engine zur Verarbeitung von Regeln bereitgestellt.

ILOG-JRules bietet große Anpassungs- und Erweiterungsmöglichkeiten. Fast alle mitgelieferten Funktionen lassen sich an die speziellen Bedürfnisse eines Benutzers anpassen.

Eine Installation unter Linux (Suse Linux 10, ubuntu) war trotz vieler Versuche nicht erfolgreich. Auch die Installation unter Windows ist mit Problemen verbunden und konnte nicht vollständig durchgeführt werden, eine Be-

wertung des Systems ist dennoch möglich.

5.2.1. Technische Daten

JRules ist ein Produkt der Firma ILOG. Die Kosten für eine Entwicklungs- und Anwendungslizenz betragen ca. 50.000 Euro. JRules basiert auf Java. Eine Version für die .NET Umgebung von Microsoft wird ebenfalls angeboten. Es können verschiedene Datenbanken verwendet werden (siehe Bewertungstabelle). Für die Regelerarbeitung ist Eclipse ab Version 3.0 notwendig. Dieses und der Application-Server JBoss werden für Windows mitgeliefert.

JRules ist sowohl für Unix/Linux-Umgebungen als auch für Microsoft Windows verfügbar. Um den Rule Execution Server zu verwenden ist einer der folgenden Application Server nötig:

- BEA WebLogic 8.1 oder 9.1
- IBM WebSphere 5.1 oder 6.0
- JBoss 4.0
- Oracle 10g
- AS 10.1.3.0
- Tomcat 5.

5.2.2. Regel-Editor

Als Regeleditor kommt Eclipse ab Version 3.0 zum Einsatz. Eine entsprechende Version ist bei der Windows-Installation im Lieferumfang enthalten und wird bei Bedarf installiert. Weiterhin werden eine Vielzahl von vorgefertigten Methoden, Hilfen, und Tutorials bereitgestellt.

Eine ILOG-Regel in der der englischen Sprache angepassten Notation ist nachfolgend angegeben. Abbildung 11 zeigt ein weiteres Regel-Beispiel.

```
set 'minAge' to 0;
set 'maxAge' to 150;

if
  it is not true that the age of 'the borrower' is between minAge and maxAge
then
  in 'the loan report' , reject the data with the message "The borrower's age is not
valid.";
```

Abbildung 11: ILOG-Team-Server und Regel-Beispiel

Quelle: ILOG, www.ilog.de/products/jrules/ruleteam.cfm, (Zugriff 22.05.2007).

5.2.3. Bewertung

Anwendungsunabhängige Kriterien:

Lizensierung / Kosten	
Entwicklungslizenz	Ab 12.000 Dollar (ein Arbeitsplatz)
Produktionslizenz	Rule-Anwendung ab 40.000 USD (2 Prozessoren); Rule-Management ab 20.000 USD (1 Prozessor oder 10 Nutzer)
Zusatzprogramme	keine Angabe
Service & Support	Service und Support vorhanden, in verschiedenen Abstufungen (Premium, VIP), aber keine Angaben zum Preis
Laufzeitumgebung	
Unterstützte OS	Linux, Windows, Apple, (alle Javafähigen Systeme)
Schnittstellen	.NET: ILOG Rules for .NET Java2EE: ILOG JRules andere: ILOG Rules
Language Bindings	C/C++, Java
Applikationsserver	JBoss 4.0, Oracle 10g, Tomcat 5, Weblogic 8.1/9.1., Websphere 5/6
Datenbanken	Cloudscape, DB2, derby, hsqldb, MySQL, Oracle, Pointbase, Microsoft SQL-Server, Sybase
Performance	
Messbarkeit	Performance-Testanwendungen werden als Beispiele mitgeliefert
Geschwindigkeit	Bis zu 384.000 Regeln/s (nach eigenen Aussagen)
Skalierbarkeit	Ja
Ressourcenbedarf	Arbeitsspeicher: 512 MB, CPU: x86 Festplattenplatz: Development 150 MB, Deployment 30 MB
Usability	
Administration	Die verfügbare Installation ist nur unter Windows installierbar. Es war nicht möglich, den Server in Betrieb zu nehmen
Betrieb / Anwendung	Ausführliches Tutorial, Testen leider nicht möglich
Kompatibilität/ Austauschbarkeit	**4**

Anwendungsbezogene Kriterien:

Regel-Darstellung	**4**
	Natürliche Sprache, IRL Ilog Rule Language (japanisch wird unterstützt), eigene Sprachbausteine können definiert werden, BAL (Business Action Language)
Regel-Verarbeitung	**4**
	RETE
Verkettung	Vorwärtsverkettung
Regel-Auswahl	keine Angabe
Suchverfahren	keine Angabe
Faktenbasis	**5**
	Fakten aus Datenbanken können eingebunden werden
Entwicklungsunterstützung	**5**
Regel-Editor	Textorientiert: Eclipse bietet eine Unterstützung für die Regelbasis.
Code-Prüfung	Java-Compiler prüft Code, weitere Überprüfung der Module ist nicht bekannt
Validierung	Ja
Identifizierung von Fakten	keine Angabe
Deployment von Regeln	
	Deployment von Regeln läuft automatisch, so dass alle Änderungen an Regeln bei laufendem Betrieb möglich
Sicherheit	**5**
Fehlerbehandlung	keine Angabe
Persistenz	keine Angabe
Nachvollziehbarkeit	keine Angabe
Revisionssicherheit / Abwärtskompatibilität	keine Angabe
Versionierung (der Regeln)	keine Angabe
Zugriffssicherheit	Authentifizierung, Autorisierung, Verschlüsselung

5.3. JBoss

JBoss Rules ist eine Open-Source-Lösung und bietet eine Java-basierte Business Rules Engine, die auf einfache Weise einen Zugang zur regelbasierten Verwaltung von Business-Strategien ermöglicht.

JBoss Rules ist eine schnelle und effiziente Rules Engine, die es Analytikern und Prüfern ermöglicht, die in der Applikation implementierten Regeln nachzuvollziehen und die Übereinstimmung mit den dokumentierten Geschäftsregeln zu prüfen. Weiterhin unterstützt JBoss Rules eine Fülle an Sprach- und Entscheidungstabelleneingaben, wodurch Änderungen an der Geschäftsstrategie leicht zu realisieren sind.

JBoss Rules ist nur eine Komponente zur Regelverarbeitung und stellt kein vollständiges Business-Rules-Management-System dar.

5.3.1. Technische Daten:

Hersteller: Red Hat (JBoss)
Kosten: keine, außer Support)
Java: Voraussetzung JDK 1.4
Datenbank: Alle JDBC-Kompatiblen Datenbanken, wie Oracle, MySQL, MSSQL, DB2, Postgres, MS Access
Umfang**:**
1. JBoss Rules Workbench (Plugin für Eclipse),
2. Entwicklungsoberfläche ist Eclipse,
3. Regel-Editor,
4. JBoss Application Server.

JBoss Rules basiert auf Java und der Rule-Engine-Sprache DROOLS. Die Programmieroberfläche ist der Java-Editor Eclipse, in dem das Drools-Plugin integriert ist. Funktionen werden somit in Java programmiert. Die Abfragen bestimmter Werte und die Bestimmung der resultierenden Aktion erfolgen in der Regelverarbeitung.

Die Abfrage einer Variablen eines Objekts wird durch die DROOLS-Sprache ausgedrückt. Ebenfalls werden hier auch Vergleiche mit vorgegebenen Werten durchgeführt. Wenn eine Regel zutrifft, wird als Aktion Java-Code ausgeführt, der sowohl die Daten des Objekts verändern, als auch eine einfache Ausgabe auf dem Monitor erzeugen kann.

5.3.2. Regel-Editor

Der Regel-Editor ist sehr stark an die Programmiersprache Java angelehnt. Die nachfolgenden Abbildungen illustrieren verschiedene Entwicklungsetappen eines einfachen Beispiels für die Kreditvergabe: Personen unter 18 Jahren können keinen Kredit bekommen. Abbildung 12 zeigt eine Java-typische Definition von Personen als Klasse mit Merkmalen und Zugriffsmethoden.

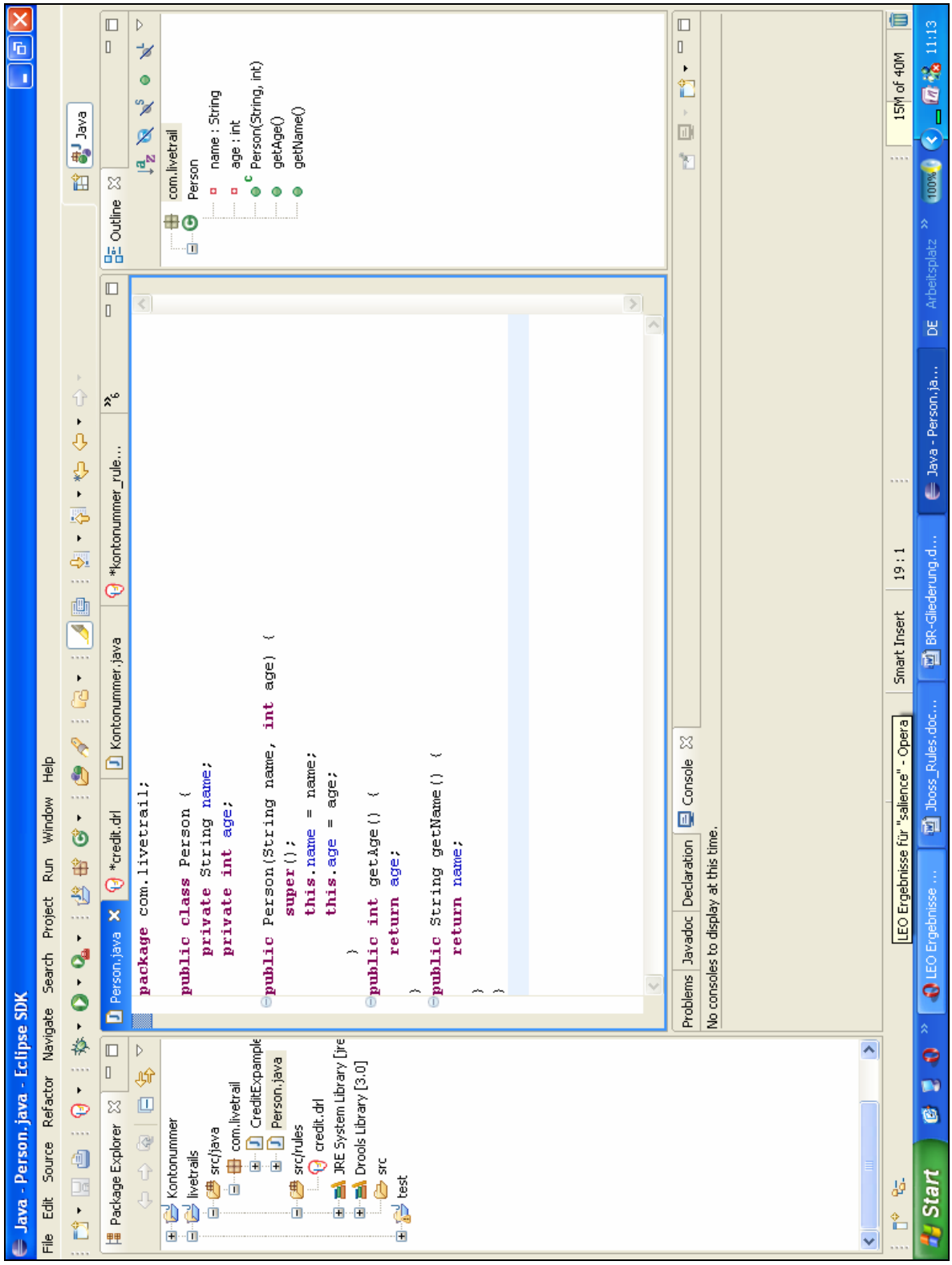
Danach kann in den Regeln auf diese Definitionen Bezug genommen werden. Die Formulierung von Regeln in JBoss ist der Abbildung 13 zu entnehmen und entspricht einer zusätzlichen Java-Sprachkonstruktion *rule-when-then*. Der Aktionsteil einer Regel enthält Methodenaufrufe.

Java typisch ist eine main-Methode zu definieren, die in diesem Beispiel die konkret beteiligten Personen Mr. Trout Sen sowie Mr. Trout jun. als Objekte erzeugt und die Abarbeitung der Regeln initiiert. Durch die Programmausführung wird die folgende Programmausgabe erzeugt:

```
Credit denied for MR Trout Jnr
Credit allowed for MR Trout Snr
```

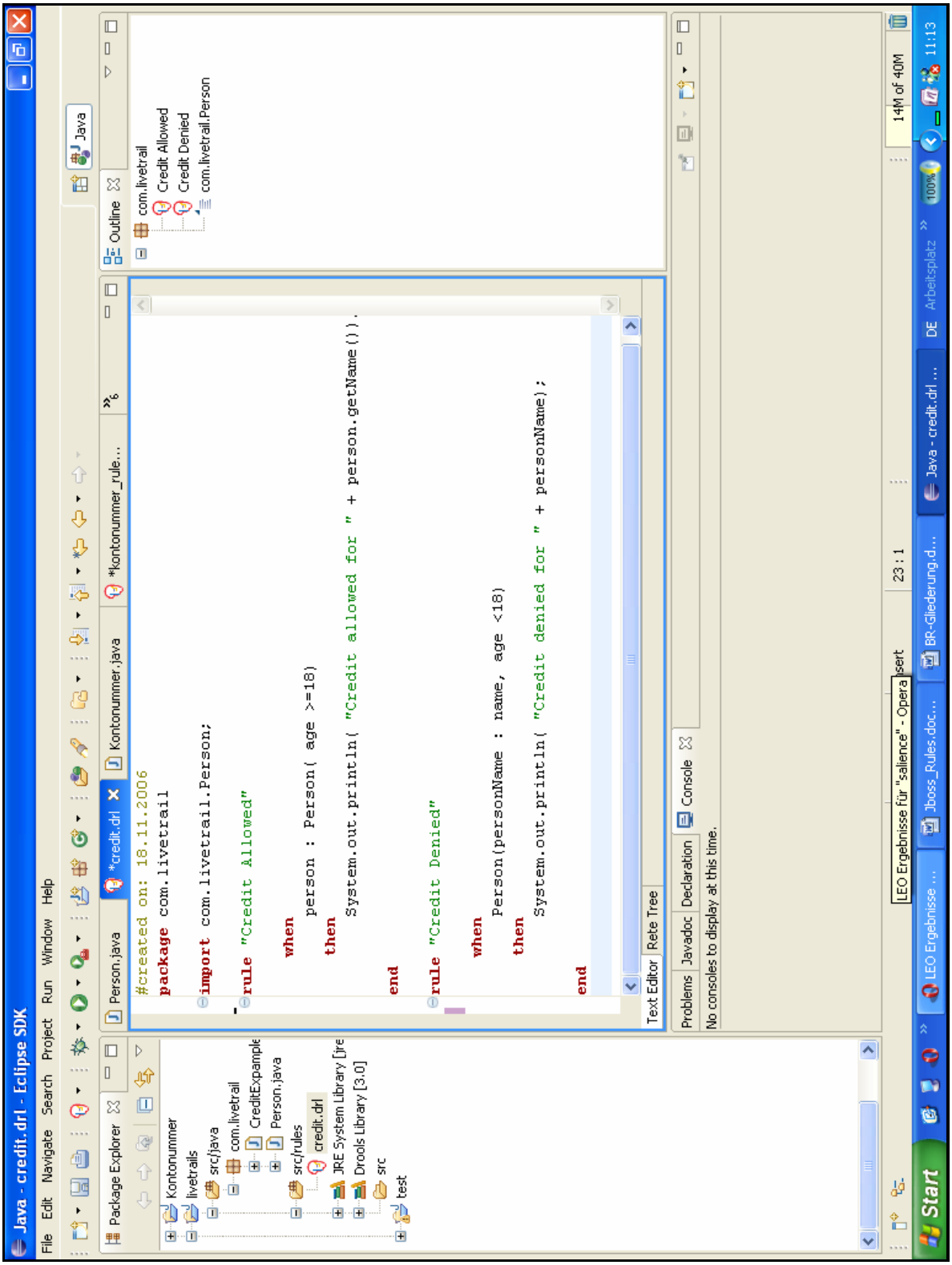
Das Beispiel geht noch nicht über eine einfache IF-Anweisung hinaus, zeigt jedoch die Möglichkeiten der Regeldarstellung in JBoss.

Abbildung 12: Beispiel: Klasse Person mit Variablen und Methoden



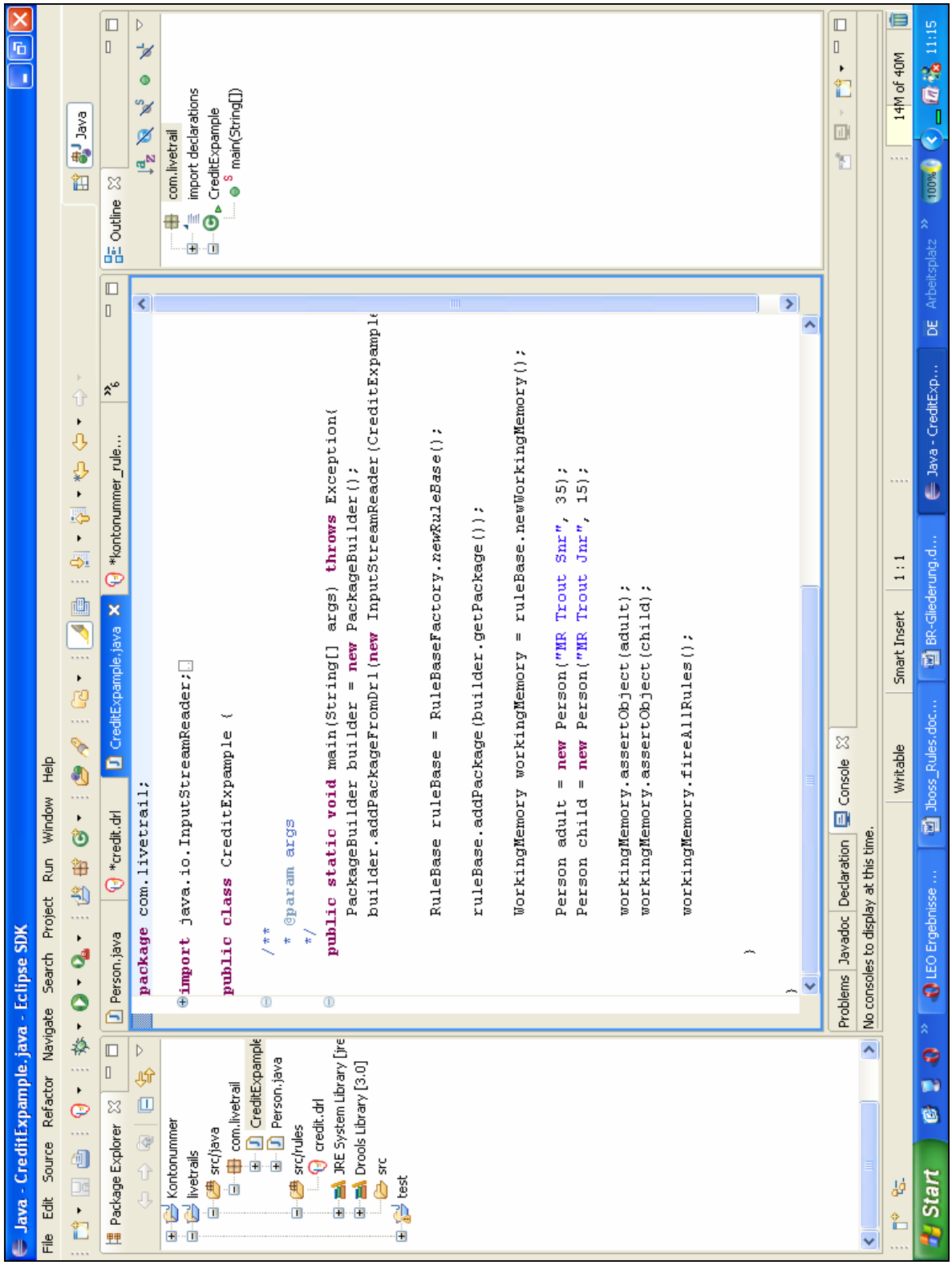
Quelle: JBoss.

Abbildung 13: Beispiel: Regeln für die Klasse Person



Quelle: JBoss.

Abbildung 14: Beispiel: Main-Klasse mit WorkingMemory



Quelle: JBoss.

Eine Regel in JBoss Rules hat immer folgenden Aufbau:

```
rule „Name“
  when
    Abfrage von Daten: positives Ergebnis in der Regel, damit die Regel zutrifft
  then
    Aktion:  Beschreibung der auszuführenden Aktion,
             wenn die „when“-Klausel zugetroffen hat (in Java)
end
```

Beispiel für eine „when“-Klausel:

Variablenzweisung	Objektname	Variable	Vergleich mit vorgegebenen Wert
kontonummer:	Kontonummer	(nummer	<=999999)

Beispiel für eine „then“-Aktion:

Java-Code (Ausgabe)	Variable aus „when“-Klausel
System.out.println(„Die Kontonummer „ +	kontonummer.getNummer() + „ liefert den Fehlercode 2“);

nummer ist eine Variable (int) in der Klasse „Kontonummer“
getNummer() ist eine Methode (int) in der Klasse „Kontonumme“

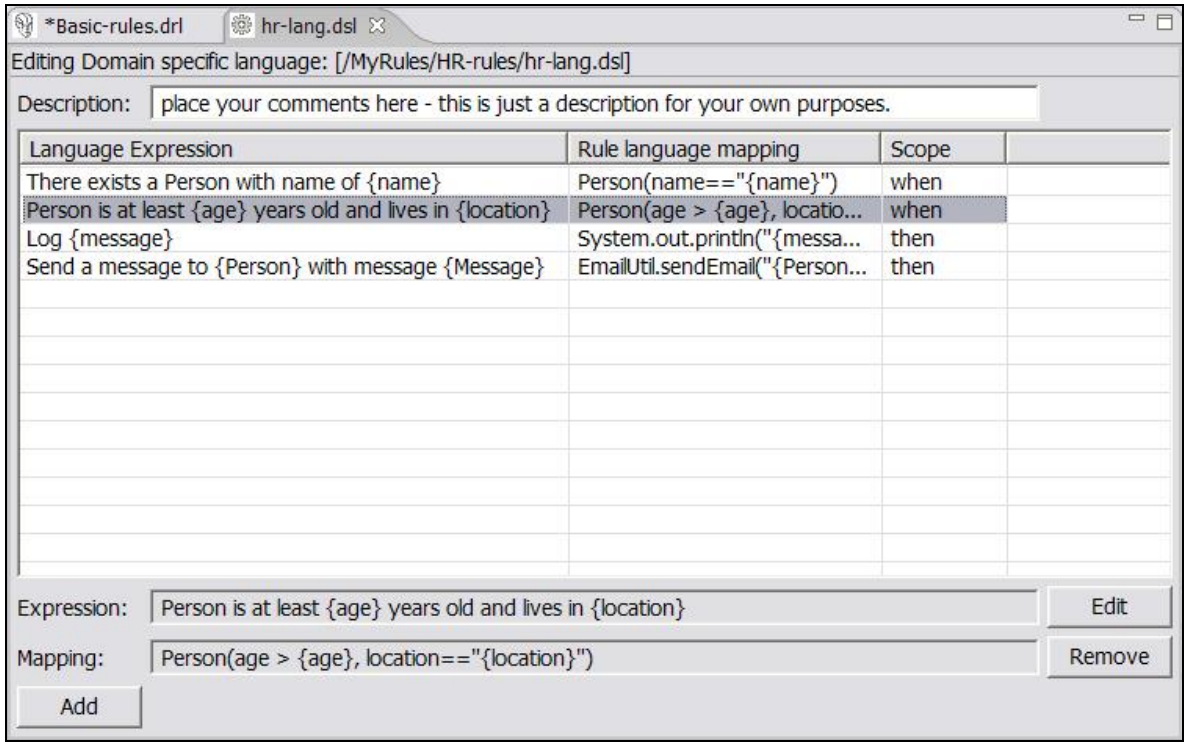
Für eine bessere Lesbarkeit kann die DROOLS-Sprache mit der Domain Specific Language (dsl) erweitert werden. Diese Sprache kann natürlichen Text darstellen, wenn sie vorher definiert wurde:

Abbildung 15: Beispiel für die anwendungsspezifische Sprache

Domain Specific Language	**Java**
„when“-Klausel:	
Die Kontonummer ist größer als {wert}	Kontonummer:Kontonummer(nummer>{wert})
„then“-Aktion	
Ausgabetext: {text}	System.out.println(„{text}“);

Quelle: Eigene Darstellung.

Abbildung 16: Beispiel: DROOLS



Quelle: JBoss.

5.3.1. Bewertung

Anwendungsunabhängige Kriterien:

Lizensierung / Kosten	**5**
Kosten Entwicklungslizenz	Keine
Kosten Produktionslizenz	Keine
Kosten Zusatzprogramme	Je nach Bedarf und gewähltem Server-Programm; Entwicklungsoberfläche Eclipse kostenlos
Kosten Service & Support	Keine Angaben
Laufzeitumgebung	**5**
Unterstützte OS	Linux, Windows ab 2000, Apple
Schnittstellen	.Net, Java2EE (JDK 1.4)
Language Bindings	C/C++ (mit DROOLS.NET), Java, DROOLS (Dynamic Rule Object-oriented Language Systeme)
Applikationsserver	
Datenbanken	Alle JDBC-Kompatiblen Datenbanken, wie Oracle, MySQL, MSSQL, DB2, Postgres, MS Access
Performance	**5**
Messbarkeit	Keine Angaben
Geschwindigkeit	Keine Angaben
Skalierbarkeit	Ja
Ressourcenbedarf	>128MB RAM, 512 MB empfohlen, 1GHz oder mehr, 1,6GHz empfohlen, 250 MB Festplattenspeicher
Usability	**2-3**
Administration	relativ geringer Aufwand; einfache Installation; Bearbeiten der BR über Workbench in Eclipse möglich; Workbench baut auf Eclipse auf; Erstellen von BR's und testen sowie entwickeln von Scenarios möglich; Erstellung von Verbindungen zwischen natürlicher und Programmiersprache möglich; Workbench erlaubt textuelles Arbeiten (mit Highlighting).
Betrieb / Anwendung	2 Stunden für die Einarbeitung mit ersten sichtbaren Ergebnissen; generell aber Java-Kenntnisse sehr hilfreich bei der Einarbeitung; Einarbeitung in Rule-Sprache (DROOLS) relativ schnell möglich; Dokumentation/ Dokumentation ist vorhanden; sehr ausführlich mit Beispielen und bebildert; Video-Tutorial und Wiki im Internet und auf der JBOSS-Webseite
Kompatibilität/ Austauschbarkeit	**2**

Anwendungsbezogene Kriterien:

Regel-Darstellung	**3**
	Natürliche Sprache nahezu natürlicher Sprache nach Spezifizierung (Verbindung zwischen natürlicher Sprache und Programmsprache); DSL (Domain Specific Language) Sprachstandard DROOLS Schnittstelle Im-/Export keine Angabe
Regel-Verarbeitung	**4**
Verkettung	Rete, Leaps ,Vorwärtsverkettung
Regel-Auswahl	keine Angabe
Suchverfahren	keine Angabe
Faktenbasis	**5**
	Anbindung bzw. direkte Einbindung von Datenbanken
Entwicklungsunterstützung	**3**
Regel-Editor	textorientiert
Code-Prüfung	Test auf unerreichbare Codeabschnitte: nein Plausibilitätsprüfung: nein Konsistenzprüfun: nein
Validierung	ja
Identifizierung von Fakten	ja
Deployment von Regeln	**2**
	DRL-File
Sicherheit	**2**
Fehlerbehandlung	ja
Persistenz	Nein
Nachvollziehbarkeit	Ja
Revisionssicherheit / Abwärtskompatibilität	keine Angabe
Versionierung (der Regeln)	keine Angabe
Zugriffssicherheit	Authentifizierung: nein Autorisierung: nein Verschlüsselung: nein

5.4. Mandarax

Mandarax ist eine Open-Source Java-Klassenbibliothek für regelbasierte Systeme. Es basiert auf Algorithmen der logischen Programmiersprache PROLOG und unterstützt daher Rückwärtsverkettung von Regeln. Die Abbildung von Klauseln und Fakten im Programm erfolgt über Java-Klassen. Das erlaubt

eine einfache Einbindung externer Datenquellen, wie z. B. Datenbanken oder Dateien.

Mandarax kann unter http://mandarax.sourceforge.net kostenlos heruntergeladen werden. Im Lieferumfang des Paketes sind mehrere Beispielanwendungen inklusive Quellcode enthalten.

Mandarax ist kein Business-Rules-management-System sondern stellt lediglich eine Java-Klassen-Bibliothek für die Behandlung von Regeln, hier Geschäftsregeln, bereit.

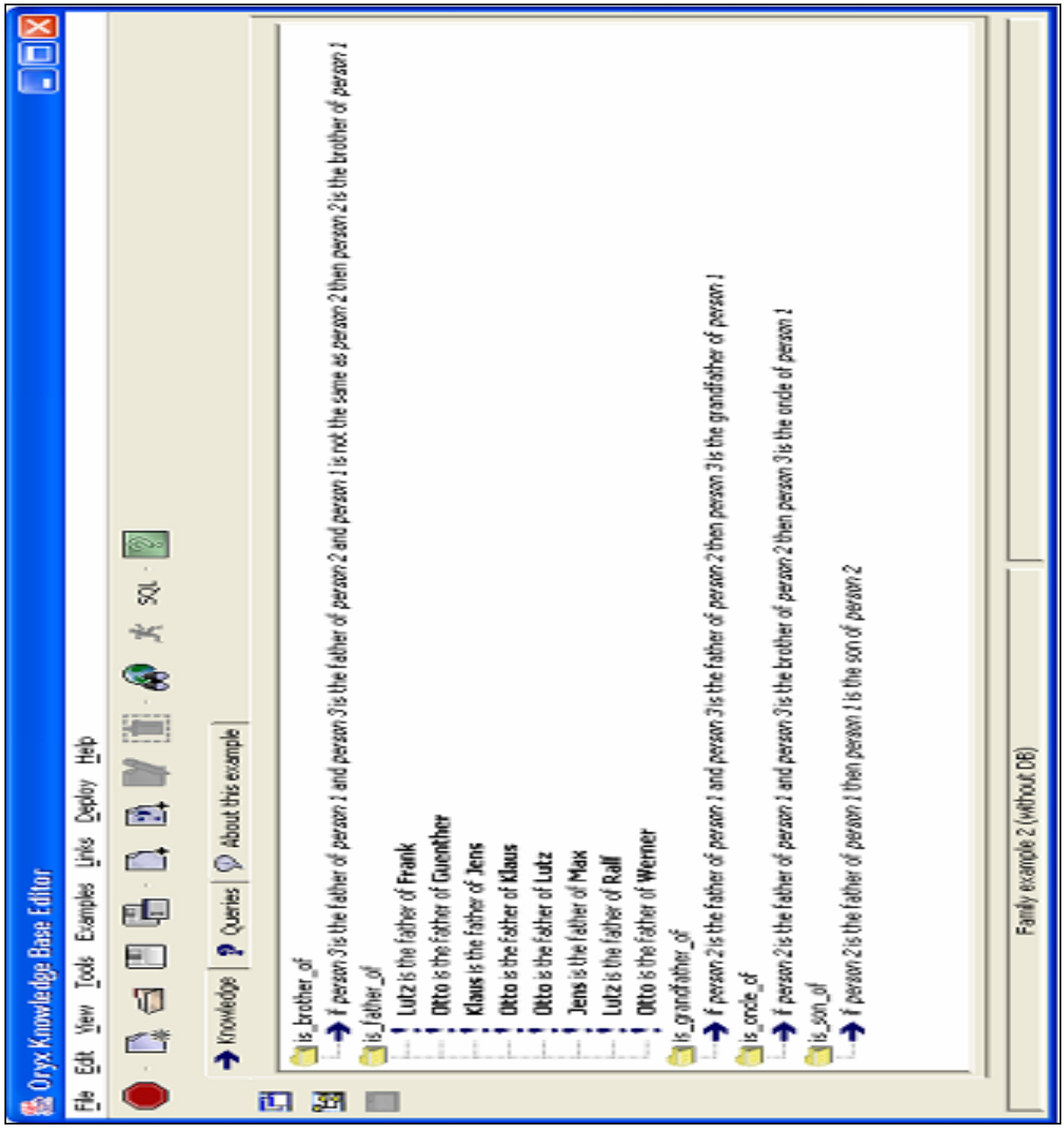
5.4.1. Technische Daten

Hersteller: Open Source, www.mandarax.org, mandarax.sourceforge.net
Kosten: kostenfrei
Lizenz: Mandarax: GNU LGPL, Oryx: GPL / Commercial Licence
Java: JDK 1.4 (Mandarax 3.4)
Datenbank: JDBC-kompatible Datenbanken
Umfang: Klassenbibliothek

5.4.2. Regel-Editor

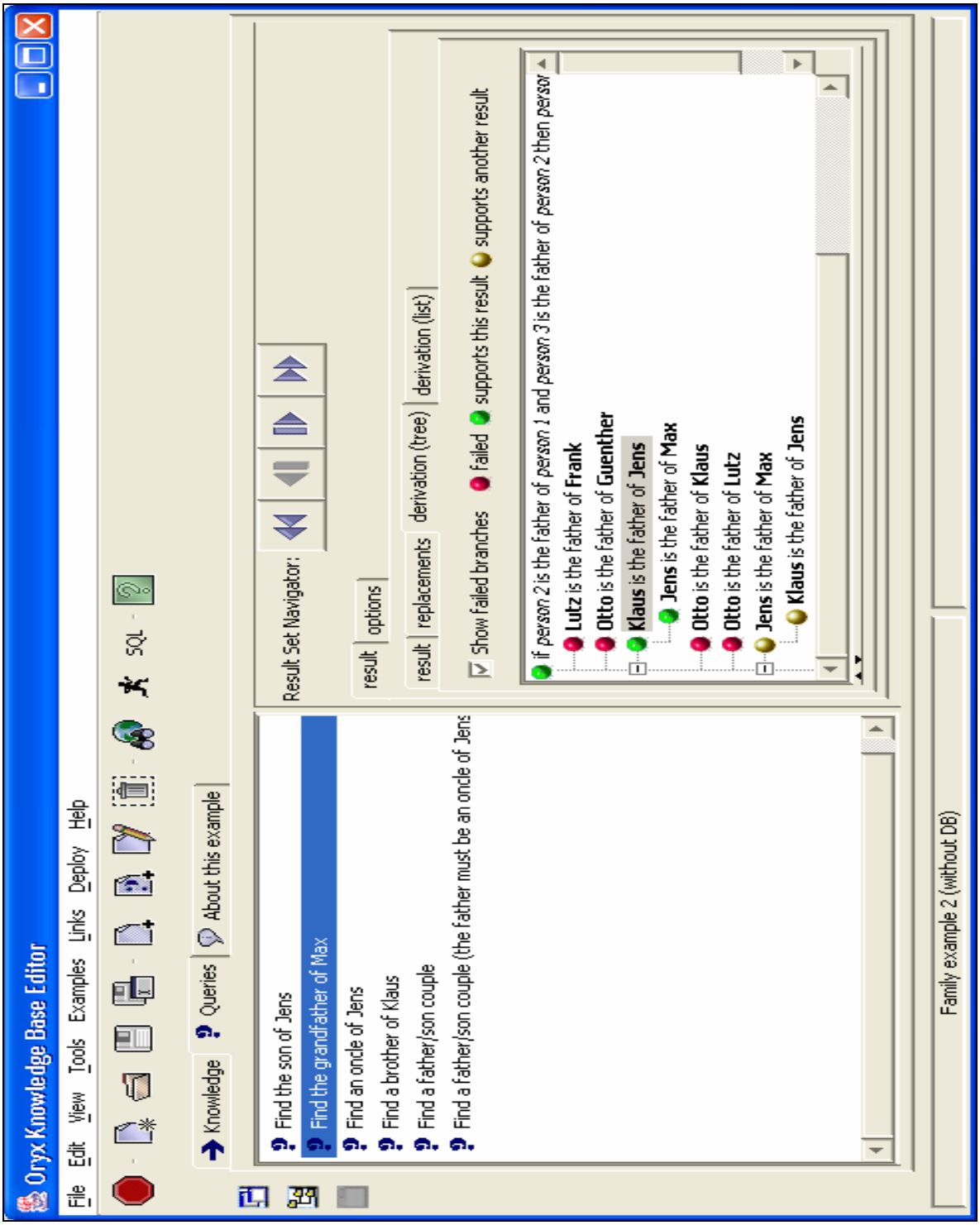
Das Mandarax-Paket bietet selbst keinen Regeleditor. Regeln und Fakten können unter direkter Nutzung der Mandarax-API, mittels eines Texteditors als RuleML oder unter Nutzung des semi-grafischen Regel-Editors „Oryx“ erstellt werden. Der „Oryx“-Editor bietet hierbei den besten Komfort für die Entwicklung von Regeln, siehe Abbildung 17.

Abbildung 17: Regeleditor Oryx



Quelle: Oryx mit eigener Darstellung.

Abbildung 18: Anfrage und Ableitungsbaum



Quelle: Oryx mit eigener Darstellung.

Aufgrund der Nähe zur Programmiersprache PROLOG, wird an dieser Stelle das klassische PROLOG-Beispiel der Verwandtschaftsbeziehungen aufgegriffen und in Mandarax abgebildet.

Abbildung 17 zeigt die Vater-Beziehung zwischen den Personen im Regel-Editor Oryx.

Eine Regelverarbeitung ist erforderlich, wenn man nun nach einem Großvater fragt. In Abbildung 18 wird die Bearbeitung einer solchen Anfrage illustriert.

5.4.3. Bewertung

Anwendungsunabhängige Kriterien:

Lizenzierung / Kosten	**5**
Kosten Entwicklungslizenz	Mandarax: LGPL Oryx: GPL oder Commercial Licence für Nutzung in Nicht-GPL Produkten
Kosten Produktionslizenz	Mandarax: kostenfrei Oryx: GPL kostenfrei, Commercial Licence unbekannt
Kosten Zusatzprogramme	OS, Server-Hardware, Datenbank
Kosten Service & Support	-
Laufzeitumgebung	**5**
Unterstützte OS	OS-Unabhängig, benötigt JDK 1.4
Schnittstellen	-
Language Bindings	Java
Applikationsserver	-
Datenbanken	JDBC-Kompatible Datenbanken
Performance	**3**
Messbarkeit	Performance-Testanwendungen werden mitgeliefert
Geschwindigkeit	-
Skalierbarkeit	-
Ressourcenbedarf	**4**
	Installation: 50 MB Festplattenplatz Hauptspeicher: anwendungsabhängig
Usability	**3**
Administration	Einfache Installation. Mandarax ist sowohl kompiliert als auch als Quellcode verfügbar. Eine spezielle Installation ist nicht erforderlich. Oryx ist kompiliert und als Quellcode verfügbar. Eine spezielle Installation ist nicht erforderlich.
Betrieb / Anwendung	Fortgeschrittene Kenntnisse in der Programmierung mittels Hochsprachen sind zwingend notwendig. Speziell Java-Kenntnisse sind vorteilhaft.
Kompatibilität/ Austauschbarkeit	**3**
	Mandarax ist eine Klassenbibliothek. Die Regelbasis kann über Standardformate (RuleML) exportiert werden.

Anwendungsbezogene Kriterien:

Regel-Darstellung	**4**
	RuleML, XML, Java Serialized Objects
Regel-Verarbeitung	**4**
Verkettung	Rückwärtsverkettung
Regel-Auswahl	Linear; Sortiert Sortierung nach (absteigend): - Fakten und SQL-Klauseln vor Regeln - Fakten mit weniger Variablen - Regeln mit mehr Vorbedingungen - Regeln mit mehr negierten Vorbedingungen Eigene Sortierung ist möglich.
Suchverfahren	keine Angabe
Faktenbasis	Java-Code, Repository, Datenbank
Entwicklungsunterstützung	**2**
Regel-Editor	Grafisch: Oryx
Code-Prüfung	-
Validierung	Ja
Identifizierung von Fakten	Ja
Deployment von Regeln	**1**
Sicherheit	**1**
Fehlerbehandlung	-
Persistenz	-
Nachvollziehbarkeit	Ja
Revisionssicherheit / Abwärts-kompatibilität	Ja
Versionierung (der Regeln)	nur durch eigene Implementierung
Zugriffssicherheit	nur durch eigene Implementierung

5.5. Oracle Business Rules

Oracle Business Rules ist eine Business Rules Engine, die es auf einfach Weise erlaubt, Business Rules in Java-Applikationen oder als ein Service in einer SOA-Umgebung zu verwenden.

Oracle bietet dabei die Möglichkeit, die Business Rules außerhalb der Anwendung mit Hilfe des RuleAuthors zu erstellen bzw. zu ändern. Dabei ist die Darstellung im RuleAuthor eine einfache Wenn-Dann-Darstellung. Der RuleAuthor wandelt die Regeln dann in eine eigene Sprache um, die Oracle RL Language nennt. Diese wird dann letztendlich in der Anwendung ausgeführt.

Die Regeln werden dann in einem Rule Repository gespeichert. Werden Änderungen gewünscht, dann ist es ausreichend, diese im Rule Repository vorzunehmen, Änderungen im Java Code sind nicht erforderlich. Innerhalb des Repositorys bietet Oracle die Möglichkeit, die Regeln in verschiedenen Versionen zu speichern, so dass stets auf eine Vorgängerversion zurückgegan-

gen werden kann.

Zur Auswertung der Regeln kann die Rule Engine auf Java-Klassen oder XML-Fakten zugreifen.

Neben dem RuleAuthor steht als Entwicklungswerkzeug von Oracle auch noch der JDeveloper zur Verfügung, der ein umfangreicher Editor ist.

5.5.1. Technische Daten:

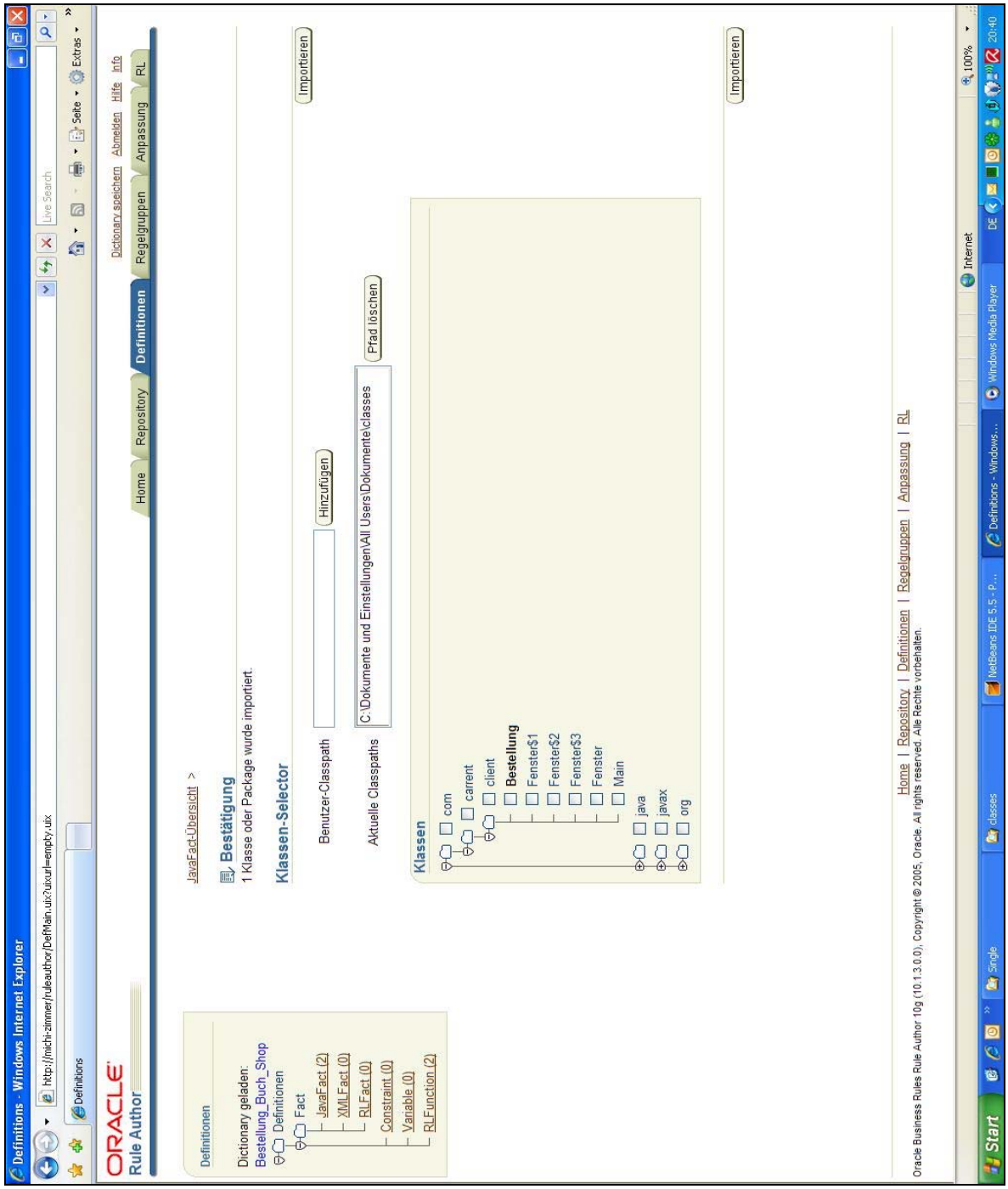
Hersteller: Oracle
Kosten: unklar, Oracle Business Rules ist Bestandteil des Applikationsserver
Java: Ja, Voraussetzung JDK 1.4
Datenbank: Alle JDBC-Kompatiblen Datenbanken, wie Oracle, MySQL, MSSQL, DB2, Postgres, MS Access
Umfang**:** Oracle Applikationsserver, Editor Jdeveloper

5.5.2. Regel-Editor

Der Rule Author ist ein grafischer Regel-Editor, mit dem sehr komfortabel Regeln bearbeitet werden können. Die folgenden Abbildungen zeigen die Vorgehensweise beim Aufbau einer Regel beginnend mit der Auswahl eines Fakts über die Festlegung von Bedingungen hin zur Definition des Aktionsteils einer Regel.

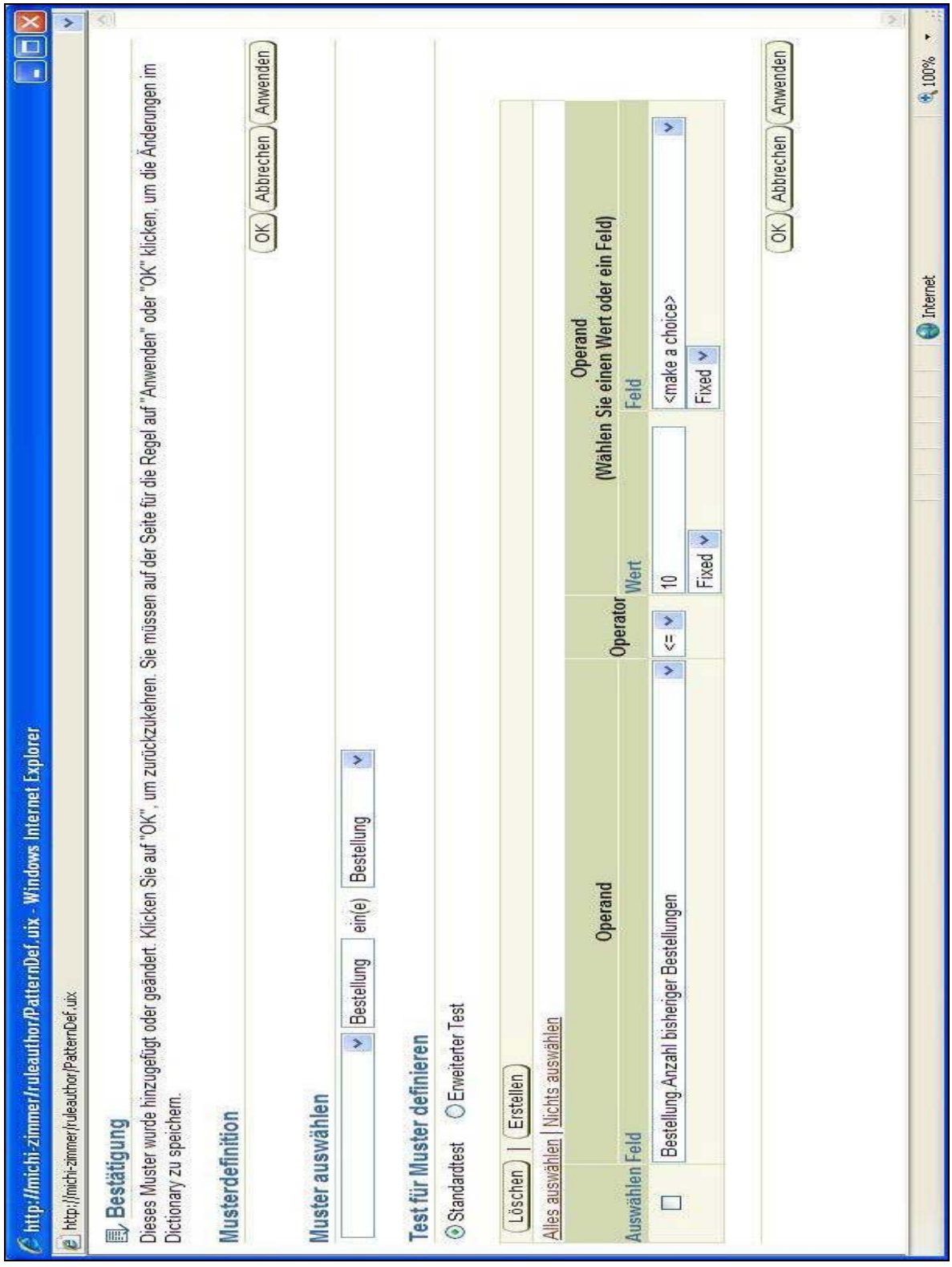
Abbildung 22 zeigt dann die vollständige Regel, die einen Kunden, der bisher nicht mehr als 10 Bestellungen vorgenommen hat, als Neukunden klassifiziert.

Abbildung 19: Auswahl einer Java-Klasse als Fakt



Quelle: Oracle.

Abbildung 20: Wenn-Bedingung

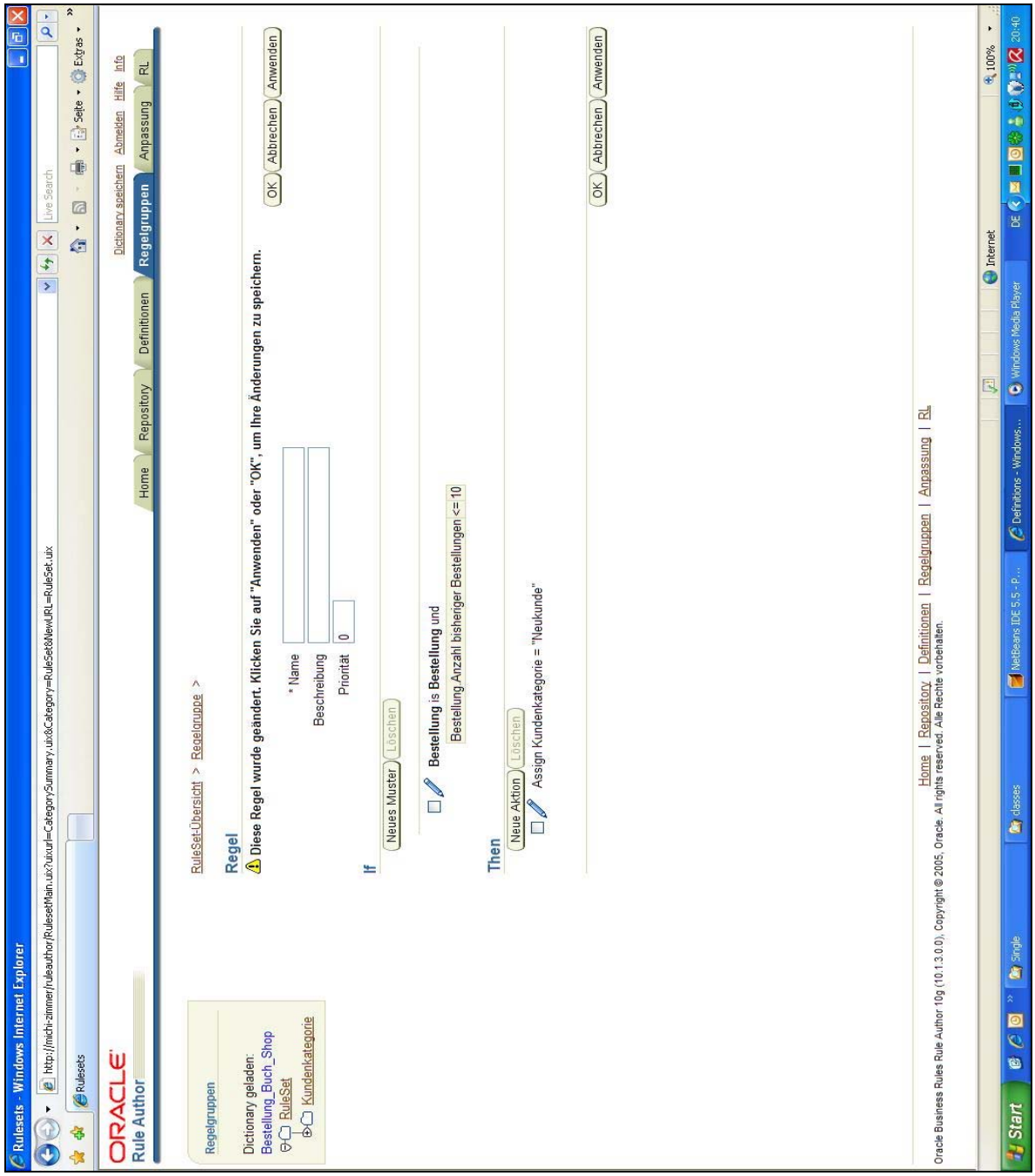


Quelle: Oracle.

Abbildung 21: Festlegen eines Dann-Teils

Quelle: Oracle.

Abbildung 22: Bearbeitung einer Regel



Quelle: Oracle.

5.5.3. Bewertung

Anwendungsunabhängige Kriterien:

Lizenzierung / Kosten	**2**
Entwicklungslizenz	Kosten für Application Server
Produktionslizenz	
Zusatzprogramme	OS, Server (je nach OS) Oracle Application Server dabei JDeveloper (kostenlos)
Service & Support	Nicht angegeben
Laufzeitumgebung	**4**
Unterstützte OS	Linux, Windows ab Windows 2000
Schnittstellen	Java2EE, SOA
Language Bindings	C/C++ über SOA, Java
Applikationsserver	Batchbetrieb möglich, Plattform unabhängig
Datenbanken	Alle JDBC-kompatiblen Datenbanken, wie Oracle, MySQL, MSSQL , DB2, Postgres, MS Access
Performance	**4**
Messbarkeit	Keine Benchmarks, Möglichkeit, eigene Benchmarks zu erstellen, vorhanden
Geschwindigkeit	Keine Angabe
Skalierbarkeit	Ja
Ressourcenbedarf	**4**
	512 MB Arbeitsspeicher 1 Ghz CPU 700MB Festplattenspeicher
Usability	**4**
Administration	relativ geringer Aufwand; einfache Installation; Bearbeiten der Regeln über den RuleAuthor oder direkt als Text; RuleAuthor zur Regel-Bearbeitung, grafisch und textuell JDeveloper für Java-Anwendungen,
Betrieb / Anwendung	relativ schnelle Einarbeitung möglich, Applikationsserver benötigt etwas mehr Einarbeitungszeit; Dokumentation sehr umfangreich, viele Beispiele und Viewlets
Kompatibilität/ Austauschbarkeit	**2**
	Java-Klassen sind einfach zu ex- und importieren in anderes BRMS, sofern Java unterstützt wird; Regeln nicht exportierbar in andere Anwendungen; Austauschbar der Anwendung sehr leicht, wenn die Struktur als SOA aufgebaut ist

Anwendungsbezogene Kriterien:

Regel-Darstellung	**4**
	Natürliche Sprache Wenn – Dann – Darstellung; Variablen können benannt werden ; Sprachstandard: RL Language
Regel-Verarbeitung	**4**
	RETE
Verkettung	Vorwärtsverkettung
Regel-Auswahl	Keine Angabe
Suchverfahren	Keine Angabe
Faktenbasis	**5**
	Anbindung bzw. direkte Einbindung in PL/SQL / Oracle - Database
Entwicklungsunterstützung	**3**
Regel-Editor	Textorientiert
Code-Prüfung	Keine Angaben
Validierung	Ja
Identifizierung von Fakten	Ja
Deployment von Regeln	**3**
	Austausch des Repository
Sicherheit	**3**
Fehlerbehandlung	Ja
Persistenz	Nein
Nachvollziehbarkeit	Ja
Revisionssicherheit / Abwärts-kompatibilität	Keine Angabe
Versionierung (der Regeln)	In einem Repository können mehrere Versionen gespeichert werden
Zugriffssicherheit	Authentifizierung, Autorisierung

5.6. QuickRules

Yasu Technologies entwickelt mit Quickrules hochwertige Komponenten zur Arbeit mit Business Rules. Hierbei sind sowohl eine .Net-Version sowie eine Version, die auf der J2EE-Technologie aufsetzt, verfügbar. Im Folgenden wird die J2EE-Version betrachtet.

Quickrules besteht aus drei Komponenten: Die QuickRule-Engine stellt die Laufzeitumgebung dar, die in der Lage ist, auch mehrere Business Rules parallel auszuführen. Durch die Nutzung der JavaBean-Technologie kann die Engine leicht in andere (Java-)Anwendungen integriert werden.

Der QuickRule-Builder bildet eine Eclipse-basierte Umgebung, in der regelbasierte Anwendungen in unterschiedlicher Art und Weise entwickelt und verwaltet werden können. Die Regeln werden als XML-Anweisungen spezifiziert und können dann entweder im Dateisystem oder in jeder JDBC-kom-

patiblen Datenbank abgelegt werden.

Der Quickrule-Webeditor ist ein webbasierter Geschäftsregeleditor, der es ermöglicht, Geschäftsregeln zur Laufzeit anzuzeigen und zu ändern. Der Webeditor setzt auf den mit dem Builder konzeptionierten Objektmodellen auf.

Quickrules kann mit einem hohen Angebot an unterstützten Datenbanken sowie Applikationservern aufwarten.

5.6.1. Technische Daten

Hersteller: YASU Technologies (Indien)
Kosten: auch auf Nachfrage keine konkrete Angabe
Java: Voraussetzung JRE 1.3.1 (für die Java-Version von Quickrules)
Datenbank: Alle JDBC-Kompatiblen Datenbanken, wie Oracle, MySQL, MSSQL, DB2, Postgres, MS Access
Umfang**:**

1. QuickRule Builder – Umgebung für Entwicklung regelbasierter Anwendungen: Regelmanagement,
2. QuickRule Webeditor – Webbasierter Geschäftsregeleditor: Anzeige und Änderung von Geschäftsregeln,
3. QuickRule Engine – Laufzeitumgebung: Durch JavaBean-Technologie leicht in andere Anwendungen integrierbar.

5.6.2. Regel-Editor

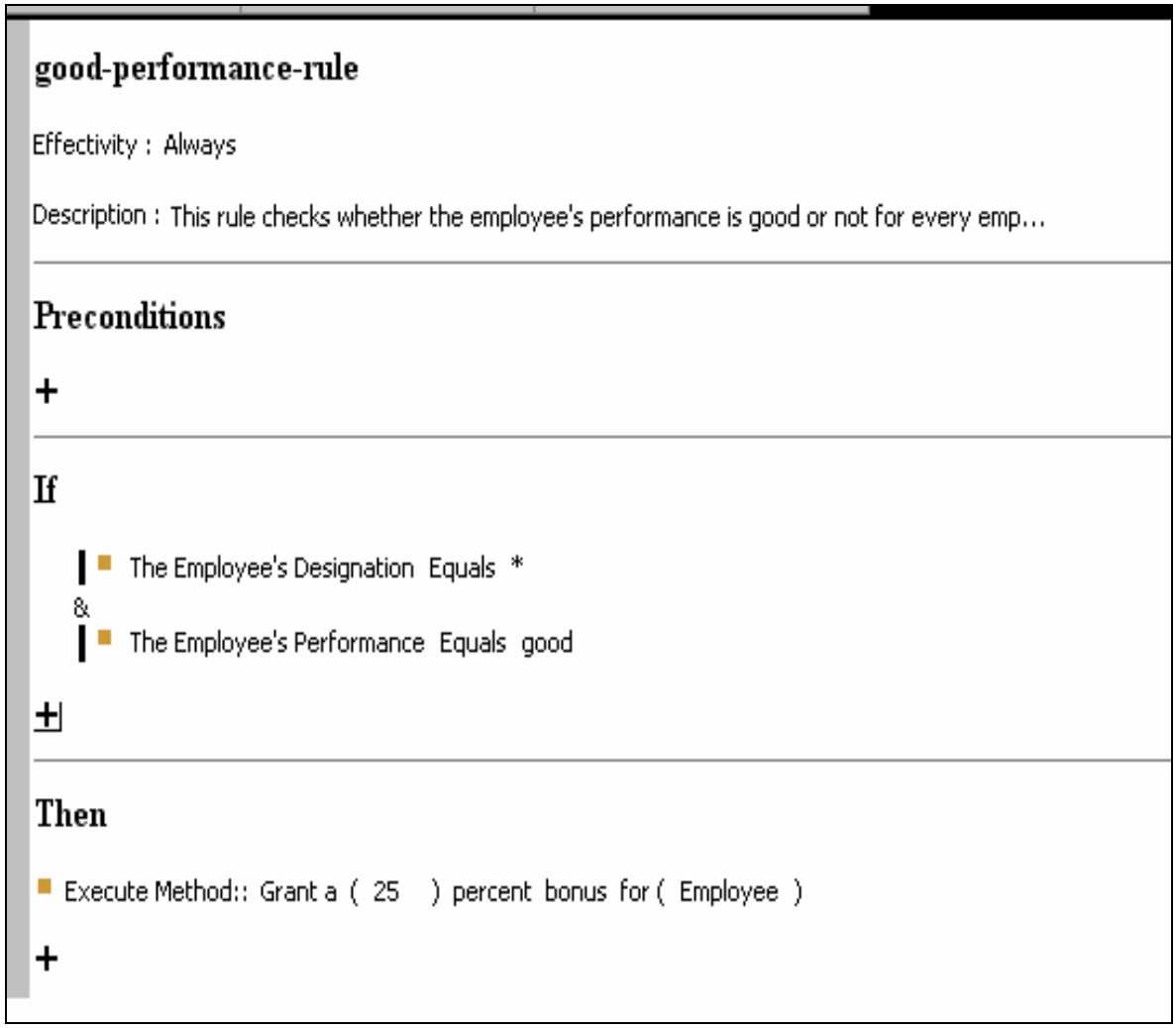
QuickRules beinhaltet zwei verschiedene Regel-Editoren. Zum einen fungiert der QuickRules-Builder in seiner Erscheinung als Eclipse-Anwendung als vollständiger Editor. Hiermit können sowohl neue Projekte geplant, entworfen, angelegt als auch geändert werden.

Hierfür dienen If-Then-Klauseln oder Entscheidungstabellen sowie, in begrenztem Umfang, grafische Werkzeuge.

Der zweite Editor ist der Quickrules-Webeditor: ein „Business-User-Interface“. Mit seiner Hilfe kann man zwar keine Projekte anlegen, dafür aber bestehende Regeln und Geschäftsobjekte über das Internet ändern und sogar löschen. Auch Tests und Versionisierung sind möglich. Alle Aufgaben können mit dem Webeditor zur Laufzeit vorgenommen werden.

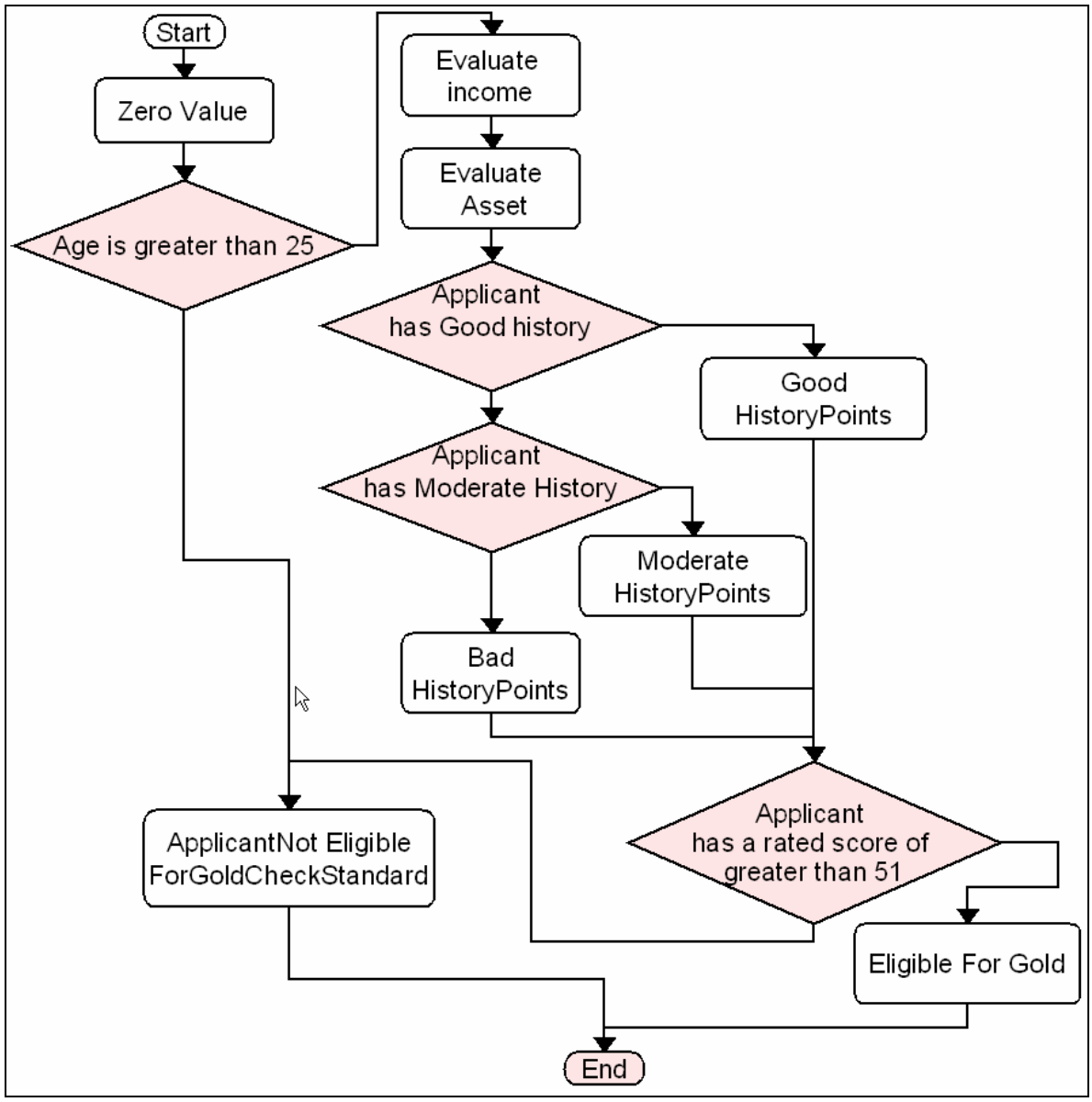
Abbildung 23 zeigt eine Geschäftsregel, die die Bedingungen für eine Bonus-Zahlung festlegt.

Abbildung 23: Beispiel: Entwicklung einer Geschäftsregel im Builder



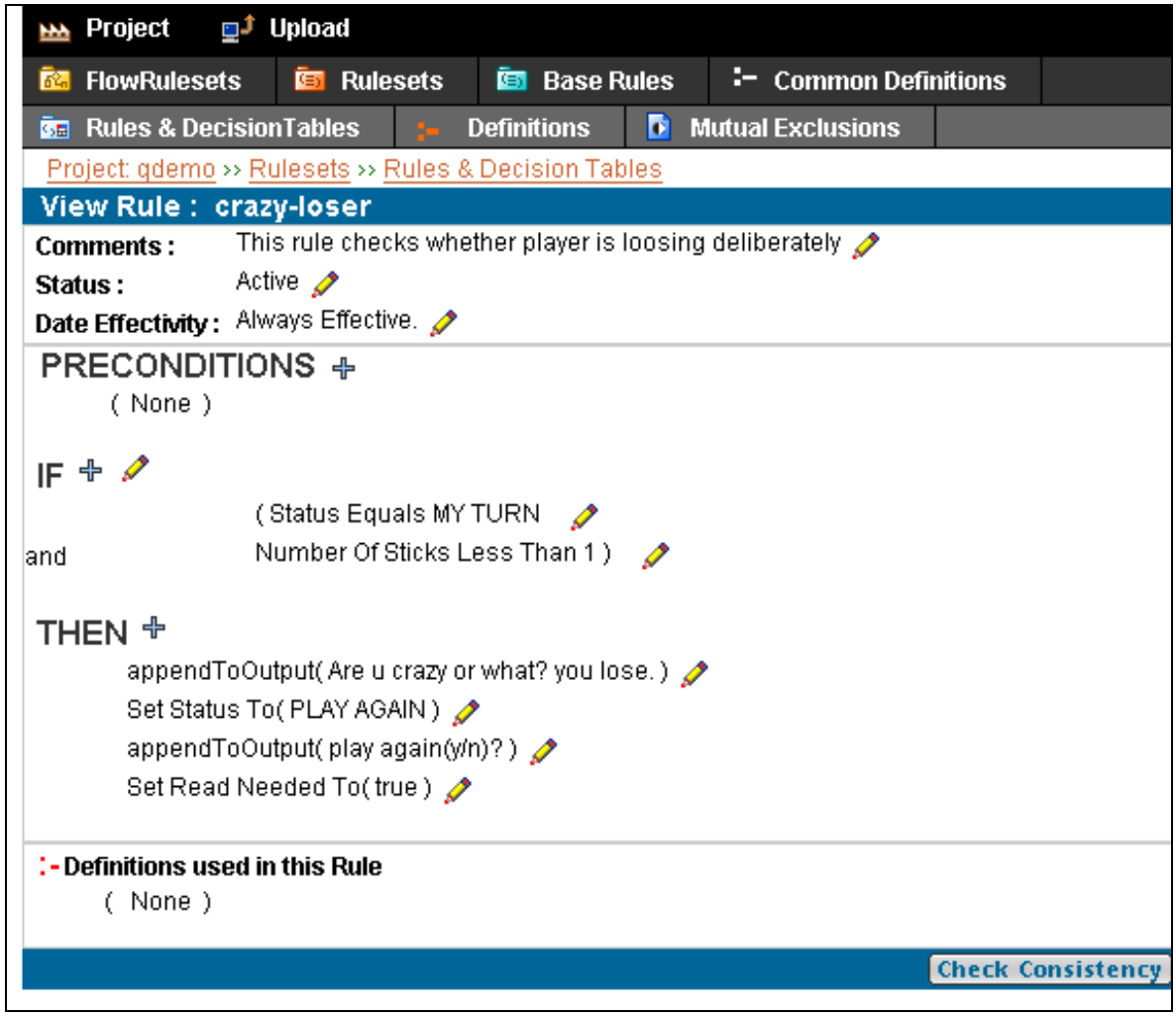
Quelle: Quickrules.

Abbildung 24: Darstellung von „Flow-Rulesets“ im Builder via „Tasks“ and Decisions



Quelle: Eigene Darstellung nach Quickrules.

Abbildung 25: Beispiel: Bearbeitung einer Regel im Webeditor



Quelle: Quickrules.

Regeln werden immer in einer IF-THEN-Notation dargestellt. Hierbei können mehrere Bedingungen mit den Operatoren AND sowie OR verknüpft werden und bei Eintreffen wiederum mehrere Aktionen zur Folge haben, siehe Abbildung 25.

5.6.3. Bewertung

Anwendungsunabhängige Kriterien:

Lizensierung / Kosten	**1**
Kosten Entwicklungslizenz	keine Angabe
Kosten Produktionslizenz	keine Angabe
Kosten Zusatzprogramme	keine Angabe
Kosten Service & Support	keine Angabe
Laufzeitumgebung	**5**
Unterstützte OS	Linux (nur Java-StandardEditon) Windows (.NET nur unter Windows lauffähig)
Schnittstellen	.NET (QuickRules .NET) Java2EE (QuickRules Java-StandardEditon)
Language Bindings	C# (nur .NET-Version), VB.NET (nur .NET-Version) Java
Applikationsserver	IBM Websphere Advanced 5.1.1.x, 6.0.2.3; IBM WSAD 5.x oder höher; BEA Weblogic 7.0, 8.1, 9.0; JBoss 3.2.x and 4.0.x; Oracle 10g AS oder höher; SunOne Application Server 7.0; Pramati 4.1 oder höher
Datenbanken	Alle JDBC-kompatiblen Datenbanken, wie Oracle, MSSQL, DB2, Postgres, MS Access, MySQL
Performance	**4**
Messbarkeit	Laut Hersteller existieren interne Benchmarks
Geschwindigkeit	Laut Hersteller eine der schnellsten Engines auf dem Markt
Skalierbarkeit	Ja
Ressourcenbedarf	**5**
	Arbeitsspeicher: min 256MB empfohlen: 512 MB CPU: keine Angabe; Festplattenplatz: 300MB
Usability	**4**
Administration	Problemlose Installation; QuickRule Builder (Eclipse) und WebEditor als Werkzeuge zum Erstellen/Bearbeiten/Löschen von Geschäftsregeln; Webeditor erlaubt nur textuelles Arbeiten – im Builder teilweise auch grafisches Arbeiten möglich (FlowRulesets)
Betrieb / Anwendung	Anfangs recht kompliziert zu erlernen – Hilfe beim Einstieg durch Hersteller scheint erforderlich; Demos für den Einstieg machen Zusammenhänge sichtbar; Dokumentation ist vorhanden; API's für WebEditor und RuleEngine; Integrationsmanuals für viele Applikationsserver vorhanden.
Kompatibilität/ Austauschbarkeit	**2**
	Durch JavaBean-Technologie kann die Engine in jede Java-Applikation integriert werden.

Anwendungsbezogene Kriterien:

Regel-Darstellung	**3**
	Natürliche Sprache (IF – THEN) JavaVersion erzeugt Plain Java-Code Ablaufdiagramme (FlowRulesets)
Regel-Verarbeitung	**4**
Verkettung	Vorwärtsverkettung -> RETE, auch andere Algorithmen werden genutzt (wurden vom Hersteller aber nicht genannt)
Regel-Auswahl	keine Angabe
Suchverfahren	keine Angabe
Faktenbasis	**5**
	Anbindung von JDBC-Kompatiblen Datenbanken (Oracle, MySQL, MSSQL, DB2, Postgres, MS Access...), Dateisystem
Entwicklungsunterstützung	**3**
Regel-Editor	Vorwiegend textorientiert
Code-Prüfung	Konsistenzprüfung
Validierung	keine Angabe.
Identifizierung von Fakten	ja
Deployment von Regeln	**4**
	Deployment in Dateisystem und JDBC- oder OLEDB-kompatible Datenbanken
Sicherheit	**4**
Fehlerbehandlung	Keine Angaben
Persistenz	Keine Angaben
Nachvollziehbarkeit	ja
Revisionssicherheit / Abwärtskompatibilität	ja
Versionierung (der Regeln)	Ja – Rollbacks möglich
Zugriffssicherheit	Benutzerrollen mit Authentifizierung

5.7. Versata Logic Suite

Versata 6 Business Rules Engine basiert auf der OEM-Version von Quickrules. Es kann sowohl auf Java, auf XML oder auf Eclipse basierende Rulesets importieren/exportieren und benutzen. Regeln werden mit Hilfe der grafischen Oberfläche von Eclipse erstellt.

5.7.1. Technische Daten

Hersteller: Versata GmbH (Postfach 14 11, D-61214 Bad Nauheim)
Standards: 100% Java Implementierung,
Java Rule Engine API (JSR94), EJB 2.0, Service Data Objects (JSR235)

Web services,
Feinkörnige Sicherheitskontrollen benutzen externe Sicherheitsprovider
Entwickelt auf J2EE-Grundlagen,
Datenbanken: IBM WebSphere®, BEA WebLogic Server®, JBoss, IBM DB2®, Oracle, Sybase, MySQL®, Microsoft SQL Server® Datenbanken

5.7.2. Regel-Editor

Versata 6 BRE Workbench wird beeinflusst von der offenen Eclipse IDE für die Definition und das Editieren der Regeln. Die Benutzung von Eclipse bietet die Zusammenarbeit mit Java-basierten Entwicklungsumgebungen und die Interoparabilität mit anderen Eclipse basierten Plugins.

Mit Versata 6 Workbench macht es möglich, Regelausdrücke zu entwerfen, Entscheidungstabellen zu konstruieren und Regelsysteme zu definieren. Für die allgemeine Erleichterung der Bearbeitung durch Geschäftsleute können Entscheidungsregelelemente mit Aliasausdrücken versehen werden. Ein Web-Interface macht das Editieren, Anschauen und Einrichten von Entscheidungsregeln für Entwickler und Geschäftsleute einfach.

Das Eclipse-Graphical-Editor-Framework wird zur Implementierung von Prozessdiagrammen benutzt. Mittels Drag-and-Drop können diese Diagramme erstellt werden. Metadaten und Beschreibungen werden mit Hilfe von flow elements assoziiert.

Versata 6 erlaubt die Erstellung von Abfragemodellen zur Abbildung von einem oder mehreren Daten- oder Prozessmodellen. Diese Abfragen können auch in SQL dargestellt werden. Ein Servicemodell kann ausgehend von einem Abfragemodell erstellt werden und die Benutzung von Abfragedaten in einem Serviceelement oder einem Web-Service ermöglichen.

5.7.3. Bewertung

Es liegen zu wenige Informationen über das Business-Rules-Management-System Versata 6 vor, um eine fundierte Beurteilung und Bewertung vornehmen zu können. Versata wird deshalb nicht mit in die tabellarische Bewertung aufgenommen.

5.8. Visual Rules

Das Business-Rule-Management-System *Visual Rules* ist ein Produkt der deutschen Innovations Softwaretechnologie GmbH. Es ermöglicht das Verwalten, Erstellen, Dokumentieren und Bearbeiten von Geschäftsregeln und deren Einbindung in den betrieblichen Produktionsprozess. Zu den wichtigsten Leistungsmerkmalen von *Visual Rules* gehören:

1. Grafischer Regeleditor,
2. Simulations-, Test und Debugging-Möglichkeiten von Regel mit grafischer Hervorhebung von Fehlern,
3. Monitoring- und Statistik-Komponenten zur Überwachung und Analyse von verwendeten Regeln,
4. Automatisch generierte Dokumentation des Regelmodells in verschiedenen Ausgabeformaten,
5. Codegeneratoren für unterschiedliche Programmiersprachen und Plattformen,
6. Unterstützung aller gängigen J2EETM Application Server,
7. Database Connectivity mit Aktionen und Funktionen für Datenselektion und Bearbeitung in RDBMS.

5.8.1. Technische Daten

Das Produkt ist als Plugin für Eclipse, SAP NetWeaverTM oder IBM WebSphere® Studio Workbench verfügbar.

5.8.2. Regel-Editor

Das Erstellen, Verwalten und Bearbeiten von Regel, lässt sich mit dem grafischen Regel-Editor gut realisieren. Eine Geschäftsregel wird erstellt, indem per Drag & Drop diverse Regelobjekte und Anweisungen auf der Arbeitsfläche platziert und miteinander verknüpft werden. Nötige Informationen für verschiedene Objekte können problemlos kommentiert werden und sind jederzeit verfügbar.

Visual Rules bietet umfangreiche Möglichkeiten, um Regeln zu testen und deren Ablauf zu überwachen. Der Editor ist übersichtlich, sehr umfangreich und gut zu bedienen. Als Anwendungsbeispiel wird eine Regel für Validierung von Daten betrachtet: Eine Kontonummer wird überprüft hinsichtlich:

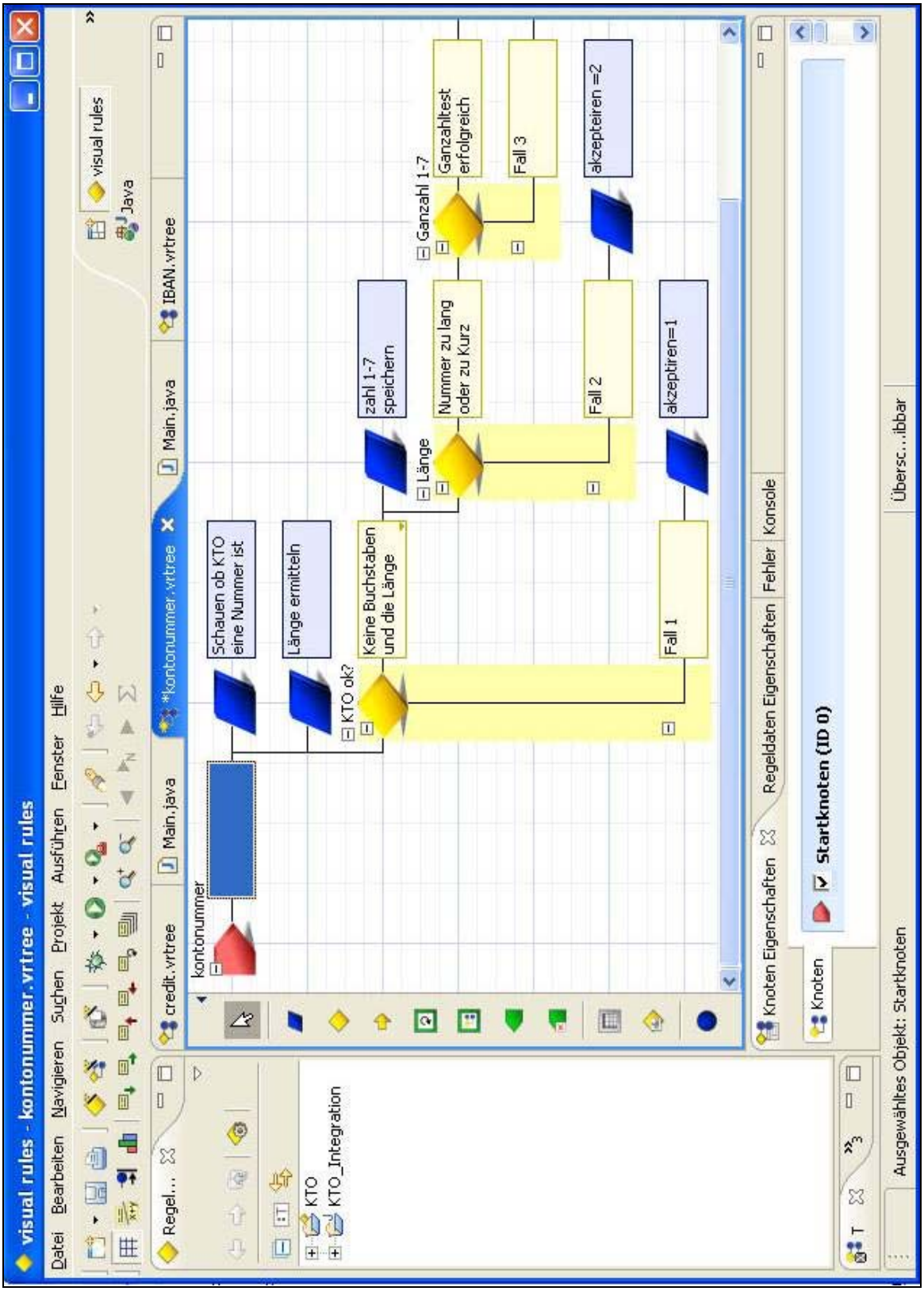
- Kontonummer besteht nur aus Ziffern.
- Kontonummer ist lang genug.
- Kontonummer ist gültig.

Das Ergebnis ist:

- 0, falls alle Prüfungen erfolgreich waren
- 1, falls Prüfung 1 fehlschlägt
- 2, falls Prüfung 2 fehlschlägt
- 3, falls Prüfung 3 fehlschlägt

Um diese Regel zu erstellen zieht man per Drag-and-Drop die benötigten Regelobjekte (Zuweisung, Entscheidung, Fall, Schleife, …) auf die Arbeitsfläche und passt diese seinen Bedingungen an.

Abbildung 26: Grafische Darstellung von Regeln im Editor



Quelle: Eigene Darstellung.

Mit dem Java-Codegenerator wird ein Java-Archiv (jar-Datei) erzeugt. Dies enthält den generierten Regelcode und kann in die Anwendung integriert werden. Das Einbinden eines Archivs in eine Java-Anwedung zeigt der folgende Quelltext:

```
// Einbinden der Archive
package kto.check.pak;
import KTOpk.*;

public class Main {

  public static void main(String[ ] args) {
    // Names des verwendeten Regelbaumes
    final String KTOrule = „kontonummer“;
    // Erzeugen eines Neuen Regelobjektes
    KTOcl rulet = new KTOcl();
    Parameters param = new Parameters();
    String KTO= new String(„143011692“);
    param.setKontonummer(KTO);
    // Initialisierung der Verarbeitung
    rulet.newData();
    rulet.setParameters(param);
    rulet.execute(KTOrule);
    rulet.finish();
    System.out.println(rulet.getVariables().getErgebnis());
  }//main-Methode
}//Main-Klasse
```

5.8.3 Bewertung

Anwendungsunabhängige Kriterien:

Lizenzierung / Kosten	**3**
Entwicklungslizenz	keine Angaben
Produktionslizenz	keine Angaben
Kosten Zusatzprogramme	keine Angaben
Kosten Service & Support	keine Angaben
Laufzeitumgebung	**5**
Unterstützte OS	Windows, Linux (Linux Standalone Version auf Anfrage)
Schnittstellen	Java2EE, Java2E
Language Bindings	JAVA
Applikationsserver	Alle bekannten J2EE Application Server
Datenbanken	Zugriff auf Datenbanken per JDBC, daher alle die von Java Database Connectivity API unterstützt werden
Performance	**5**
Messbarkeit	keine Angaben
Geschwindigkeit	keine Angaben
Skalierbarkeit	keine Angaben
Ressourcenbedarf	**4**
	Keine Angaben vom Hersteller. Standalone Version benötigt ca. 300MB Speicherplatz zzgl. benötigter Java-Komponenten
Usability	**4**
Administration	Administration erfolgt über einen Grafische Benutzeroberfläche
Betrieb / Anwendung	Einarbeitungszeit kann als gering eingeschätzt werden
Kompatibilität/ Austauschbarkeit	**0**

Anwendungsbezogene Kriterien:

Regel-Darstellung	**5**
	Natürliche Sprache, Sprachstandard-Schnittstelle Im-/Export
Regel-Verarbeitung	**4**
	keine genauen Angaben. Regelverarbeitung findet nicht mit dem RETE-Algorithmus statt.
Verkettung	keine Angaben
Regel-Auswahl	keine Angaben
Suchverfahren	keine Angaben
Faktenbasis	**3**
	Die Möglichkeit zur Anbindung einer Oracle DB ist vorhanden. Jedoch sind keine Informationen über Integrationsfähigkeit in PL/SQL gefunden worden. Es ist aber davon auszugehen, dass diese nicht vorhanden sind.
Entwicklungsunterstützung	**4**
Regel-Editor	Grafischer Regeleditor
Code-Prüfung	Regeln können umfangreich überprüft werden, jedoch nicht der aus den Regeln erzeugte Code.
Validierung	Datentypen müssen beim erstellen der Regeln definiert werden.
Identifizierung von Fakten	
Deployment von Regeln	**4**
Sicherheit	**4**
Fehlerbehandlung	Auf Fehlern in den Regeln wird hingewiesen
Persistenz	Regeln können direkt in eine Anwendung integriert, oder auf einen Rule Server abgelegt werden.
Nachvollziehbarkeit	Regeln werden können Umfangreich dokumentiert und überwacht werde
Revisionssicherheit / Abwärtskompatibilität	keine Angaben
Versionierung	Alle Elemente eines Regelprojekts lassen sich über ein Versionierungssystem verwalten. Der grafische Vergleich unterschiedlicher Stände ermöglicht einen schnellen Überblick über vorgenommene Änderungen.
Zugriffssicherheit	keine Angaben

5.9. Weitere nicht-kommerzielle, auf Java basierende Systeme

Die folgende Tabelle enthält kurze Charakterisierungen der auf der Webseite www.manageability.org/blog/stuff/rule_engines/view (Zugriff 26.01.2007) gelisteten, nicht-kommerziellen Systeme.

Tabelle 4: Nicht-kommerzielle Systeme mit Bezug zur Regelverarbeitung

Software	**Kurzbeschreibung**
OFBiz	Open for Business; Keine Rule Engine, System für Datenverwaltung, genutzt als Programm für Onlineshops, CMS, verwendet für Warehouse-Management, ermöglicht Katalog Verwaltung
Algernon	Keine Rule Engine, wird verwendet zum Regel basierten Programmieren in JAVA
TyRuBa	Höhere logische Programmiersprache
JTP	„Java Theorem Prover“, keine Rule Engine, „Object-oriented Modular Reasoning System“
JEOPS	„Java Embedded Object Production System“; Ermöglicht das Erstellen von Regel-Objekten, Projekt ist aber wahrscheinlich eingestellt, letztes Aktualisierung der Website war im September 2000
Info Sapient	Rule Engine welche fuzzy logic verwendet, Regel werden umgangsprachig formuliert, in Java implementiert, arbeitet mit Rückwärtsverkettung, Vorwärtsverkettung soll implementiert werden
RDF Expert	Link defekt: keine Informationen
Jena 2	„Sematic Web Framework for Java” „ Jena 2 includes a generic rule based inference engine”
JLisa	Keine relevanten Informationen auf der Projektseite, nur ein Entwickler, wahrscheinlich inaktiv
Euler	„Inference Engine”
JLog	In Java entwickelter Prolog-Interpreter
Prova	Logische Programmiersprache: „derived from Mandarax Java-based inference system“
Open Rules	Business Rule Engine; Vollwertiges BRMS, GPL License, jedoch wird für den kommerziellen Gebrauch eine Lizenz benötigt, Regeln werden mittels MS Excel erstellt und mit Hilfe von Eclipse verarbeitet.
Sweet Rules	„Open Source Platform for Semantic web business rules“, Noch im Alpha Stadium, Website wenig informativ
Isotop 2	Auftrags Planungssystem
Open Lexicon	Laut Hersteller eine Rule Engine Für Informationen ist eine Anmeldung erforderlich
Hammurapi Rules	JSR-94 konforme Rule Engine; Regeln werden nicht in einer Regelsprache beschrieben, sondern direkt in Java implementiert. Bereitstellung als API
MINS Reasoner	Keine Rule Engine
Zilonis	Website noch im Aubau, daher nur wenig Informationen verfügbar Multithreading Rule Engine, Vorwärtsverkettung
JCHR	Keine Rule Engine – Logik-Sytem
Esper	Keine Rule Engine: „Event Stream Intelligence
mProlog	Prolog für „Agenten“
OpenLTablets	Ermöglicht das Erstellen von Regeln, versteht sich aber selber nicht als Rule Engine. Regeln werden in MS Excel mit Tabellen erstellt und als XML exportiert. Mit Hilfe eines Eclipse Plugins können diese Regeln dann eingebunden werden

Quelle: www.manageability.org/blog/stuff/rule_engines/view (Zugriff 26.01.2007)

6. Schlussfolgerungen

Das Thema der Geschäftsregeln ist hochaktuell. Es ist mit einer weiteren Zunahme der Anwendungen von Business Rules, sprich Geschäftsregeln, zu rechnen. Sowohl IT-Entwickler als auch Anwender müssen in der Lage sein, Vorgänge und Situationen der realen Welt in Regel-Form abzubilden. Die Fähigkeit zur Wissensdarstellung wird eine entscheidende Rolle spielen. Entscheidend hierfür sind auch die von den Systemen angebotenen Unterstützungen hinsichtlich einer nutzerfreundlichen Wissenseingabe und Wissensdarstellung.

Die vorliegende Arbeit gibt einen Überblick über derzeit angebotene Systeme für die Verwaltung von Geschäftsregeln. Es zeigt sich, dass alle Systeme einen Einsatz in verschiedenen Betriebssystemen ermöglichen und allgemein gebräuchliche Datenbanken eingebunden werden können. Durch die Basis Java wird eine große Kompatiblität erreicht. Wünschenswert ist ein stärkerer Einsatz einer Standard-Beschreibungssprache, wie beispielsweise RuleML, um den Wissensaustausch zwischen verschiedenen Systemen zu ermöglichen.

Eine eindeutige Empfehlung für ein konkretes System kann an dieser Stelle nicht gegeben werden. Mit entscheidend für eine System-Auswahl wird sicher die Kostenfrage sein, die hier nur teilweise bzw. auch nur recht grob behandelt werden konnte.

Literatur

Boley, Harold: The RuleML Family of Web Rule Languages, http://2006.ruleml.org/slides/RuleML-Family-PPSWR06-talk-up.pdf, (letzter Zugriff 19.01.2007).

Das Business Rule Portal, http://www.brportal.org , Zugriff 11.12.2006.

Endl, Rainer: Regelbasierte Entwicklung betrieblicher Informationssysteme - Gestaltung flexibler Informationssysteme durch explizite Modellierung der Geschäftslogik, Dissertation, Wirtschafts- und Sozialwissenschaftlichen Fakultät, Universität Bern, 2004.

Haas, Matthias: Methoden der künstlichen Intelligenz in betriebswirtschaftlichen Anwendungen, Diplomarbeit, Hochschule Wismar, 2006. auch: Wismarer Schriften zu Management und Recht, Band 7, [CT Salzwasser] Bremen 2007.

Lämmel, Uwe/**Cleve**, Jürgen: Lehr- und Übungsbuch Künstliche Intelligenz, 2. Auflage, Leipzig, 2004.

Schacher, M./**Grässler**, P.: Agile Unternehmen durch Business Rules, Berlin Heidelberg 2006.

Sinur, J.: Rules: Adding Intelligence to the Enterprise Architecture, Gartner-Group 2002, www.gartner.com/reprints/fairisaac/108391.html, Zugriff 26.01.2007.

The Business Rules Group, http://www.businessrulesgroup.org , Zugriff 10.12.2006.

Wikipedia, wikipedia.de, Stichworte: Geschäftsregel-Managementsystem, Geschäftsregel, Zugriff 11.12.2006.

Autorenangaben

Prof. Dr.-Ing. Uwe Lämmel
Grundlagen der Informatik / Künstliche Intelligenz
Hochschule Wismar, Fachbereich Wirtschaft
Philipp-Müller-Straße
Postfach 12 10
D - 23952 Wismar
Telefon: ++49 / (0)3841 / 753 617
Fax: ++ 49 / (0)3841 / 753 131
E-mail: u.laemmel@wi.hs-wismar.de

Studenten des Masterstudiengangs Wirtschaftsinformatik
Dipl.-Wirtsch.Inf. (FH) Anatoli Beifert
a.beifert@wi.hs-wismar.de
Dipl.-Wirtsch.Inf. Stefan Wissuwa
s.wissuwa@wi.hs-wismar.de
Dipl.-Wirtsch.Inf. (FH) Marcel Brätz
m.braetz@stud.hs-wismar.de
Matthias Buse
m.buse@stud.hs-wismar.de
Christian Höhn
c.hoehn@stud.hs-wismar.de

Studenten des Diplomstudiengangs Wirtschaftsinformatik
Stefan Brandenburg
stefan.brandenburg@gmx.de
Gert Mannheimer
g.mannheimer@stud.hs-wismar.de
Michael Rehfeld
m.rehfeld@stud.hs-wismar.de
Alexander Richter
alexander.richter@stud.hs-wismar.de

WDP - Wismarer Diskussionspapiere / Wismar Discussion Papers

Heft 13/2006:	Kristine Sue Ankenman: Austrian Neutrality: Setting the Agenda
Heft 14/2006:	Jost W. Kramer: Co-operative Development and Corporate Governance Structures in German Co-operatives – Problems and Perspectives
Heft 15/2006:	Andreas Wyborny: Die Ziele des Neuen Kommunalen Rechnungswesens (Doppik) und ihre Einführung in die öffentliche Haushaltswirtschaft
Heft 16/2006:	Katrin Heduschka: Qualitätsmanagement als Instrument des Risikomanagements am Beispiel des Krankenhauses
Heft 17/2006:	Martina Nadansky: Architekturvermittlung an Kinder und Jugendliche
Heft 18/2006:	Herbert Neunteufel/Gottfried Rössel/Uwe Sassenberg/Michael Laske/Janine Kipura/Andreas Brüning: Überwindung betriebswirtschaftlicher Defizite im Innoregio-Netzwerk Kunststoffzentrum Westmecklenburg
Heft 19/2006:	Uwe Lämmel/Andreas Scher: Datenschutz in der Informationstechnik. Eine Umfrage zum Datenschutzsiegel in Mecklenburg-Vorpommern
Heft 20/2006:	Jost W. Kramer/Monika Passmann: Gutachten zur Bewertung der Struktur-, Prozess- und Ergebnisqualität der allgemeinen Sozialberatung in Mecklenburg-Vorpommern
Heft 21/2006:	Marion Wilken: Risikoidentifikation am Beispiel von Kindertageseinrichtungen der Landeshauptstadt Kiel
Heft 22/2006:	Herbert Müller: Zahlen und Zahlenzusammenhänge - Neuere Einsichten zum Wirken und Gebrauch der Zahlen in Natur und Gesellschaft
Heft 01/2007:	Günther Ringle: Genossenschaftliche Prinzipien im Spannungsfeld zwischen Tradition und Modernität
Heft 02/2007:	Uwe Lämmel/Eberhard Vilkner: Die ersten Tage im Studium der Wirtschaftsinformatik
Heft 03/2007:	Jost W. Kramer: Existenzgründung in Kleingruppen nach der Novellierung des Genossenschaftsgesetzes
Heft 04/2007:	Beate Stirtz: Hybride Finanzierungsformen als Finanzierungsinstrumente mittelständischer Unternehmen
Heft 05/2007:	Uwe Lämmel/Anatoli Beifert/Marcel Brätz/Stefan Brandenburg/Matthias Buse/Christian Höhn/Gert Mannheimer/Michael Rehfeld/Alexander Richter/Stefan Wissuwa: Business Rules – Die Wissensverarbeitung erreicht die Betriebswirtschaft. Einsatzmöglichkeiten und Marktübersicht